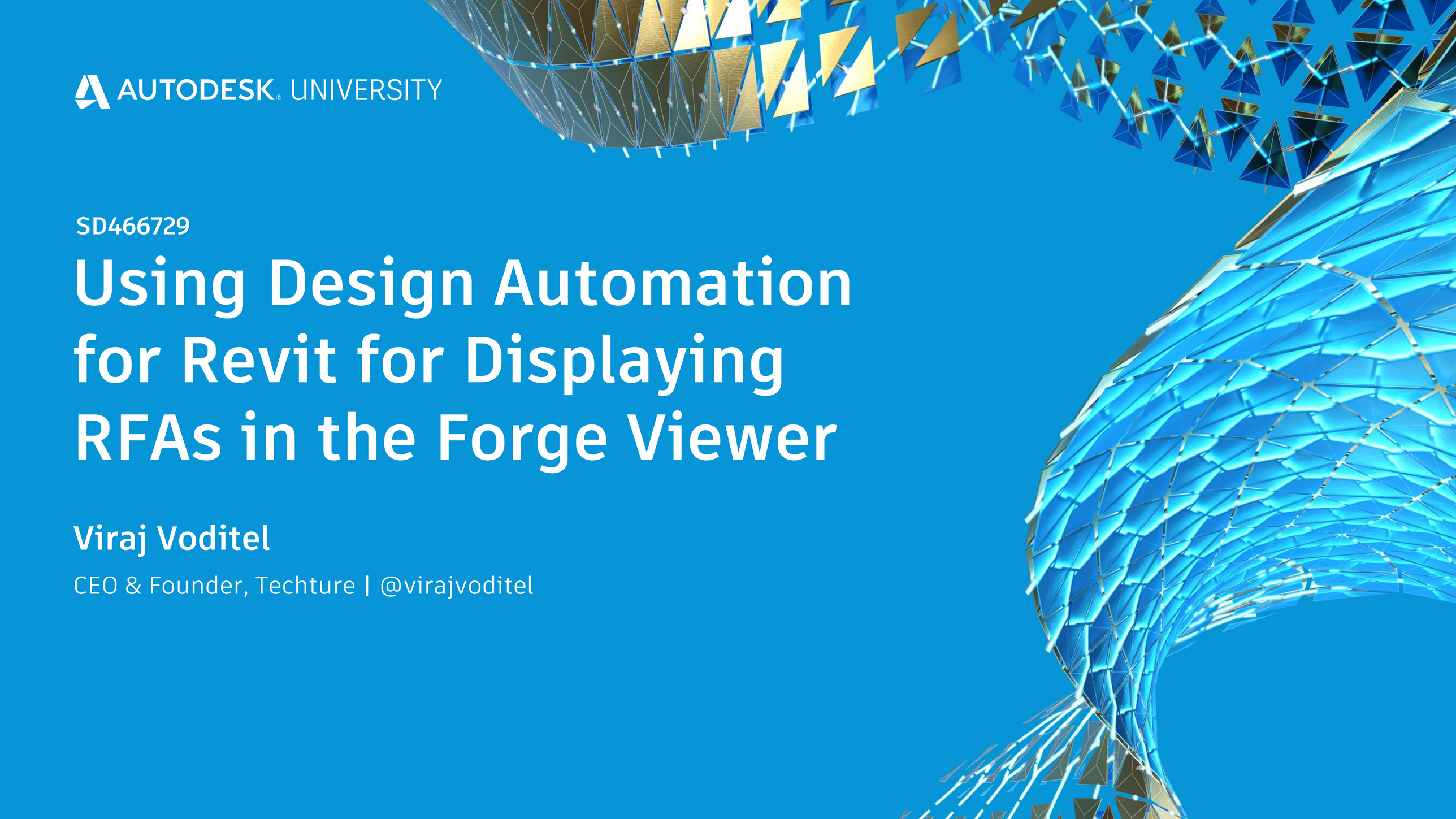
AUTODESK. UNIVERSITY

SD466729

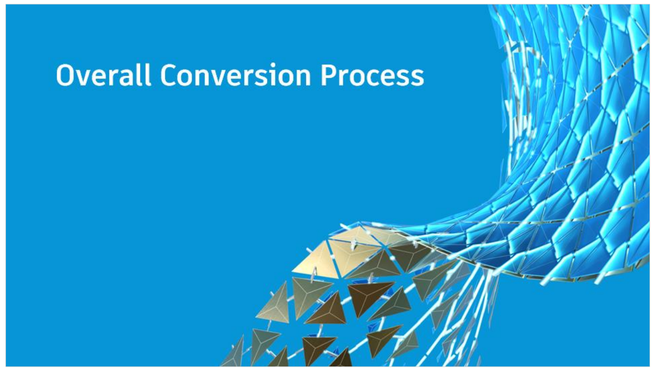
Using Design Automation for Revit for Displaying RFAs in the Forge Viewer

Viraj Voditel

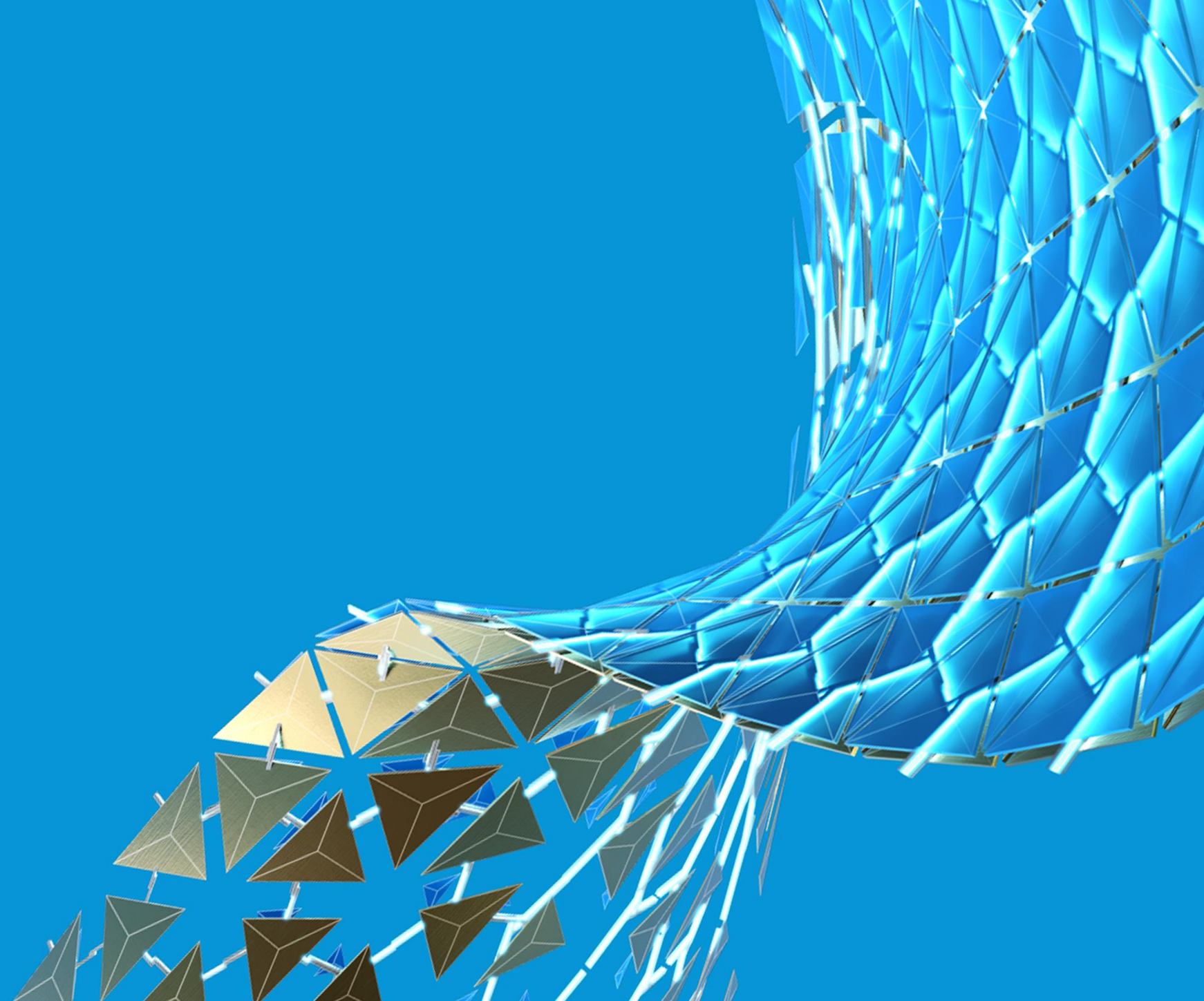
CEO & Founder, Techture | @virajvoditel



What we’re going to cover



Prologue



TECHTURE

About the speaker

Viraj Voditel

- Viraj is the CEO and Founder of Techture, a global BIM consulting firm having offices in UK, UAE, India and Singapore.
- He started out as a Student Expert for Autodesk while pursuing his Civil Engineering degree and currently is an Autodesk Expert Elite and a Certified Professional for various software.
- He is a BIM evangelist and frequently talks about BIM at various platforms. He has delivered technical lectures at the national and international level and is actively involved in championing newest technologies in the AEC space.
- At Techture, he leads multidisciplinary teams on developing newer and more efficient workflows. He strives towards ensuring that they always keep up with the latest technologies and diversify into broader segments.
- Viraj has been able to amass a rich experience on BIM Implementation for various large scale projects includes hospitals, hotels, airports, hydropower projects and smart cities.

AUTODESK® EXPERT ELITE

AUTODESK CERTIFIED PROFESSIONAL

AUTODESK CERTIFIED PROFESSIONAL

AU SPEAKER 2017

Learning Objectives

LEARNING OBJECTIVE 1

Discover the different formats the Forge Viewer supports, and why the RFA format is a limitation and currently is not supported.

LEARNING OBJECTIVE 2

Learn how to design a workflow for visualizing Revit families and their family types and accessing parameters in the Forge Viewer.

LEARNING OBJECTIVE 3

Learn how to render hundreds of Revit families in the Forge Viewer and optimize for loading time and Forge credits usage.

LEARNING OBJECTIVE 4

Learn how to create headless Revit plug-ins to enable a cloud-connected workflow using the Design Automation for Revit API.

Description

Using Design Automation for Revit for Displaying RFAs in the Forge Viewer

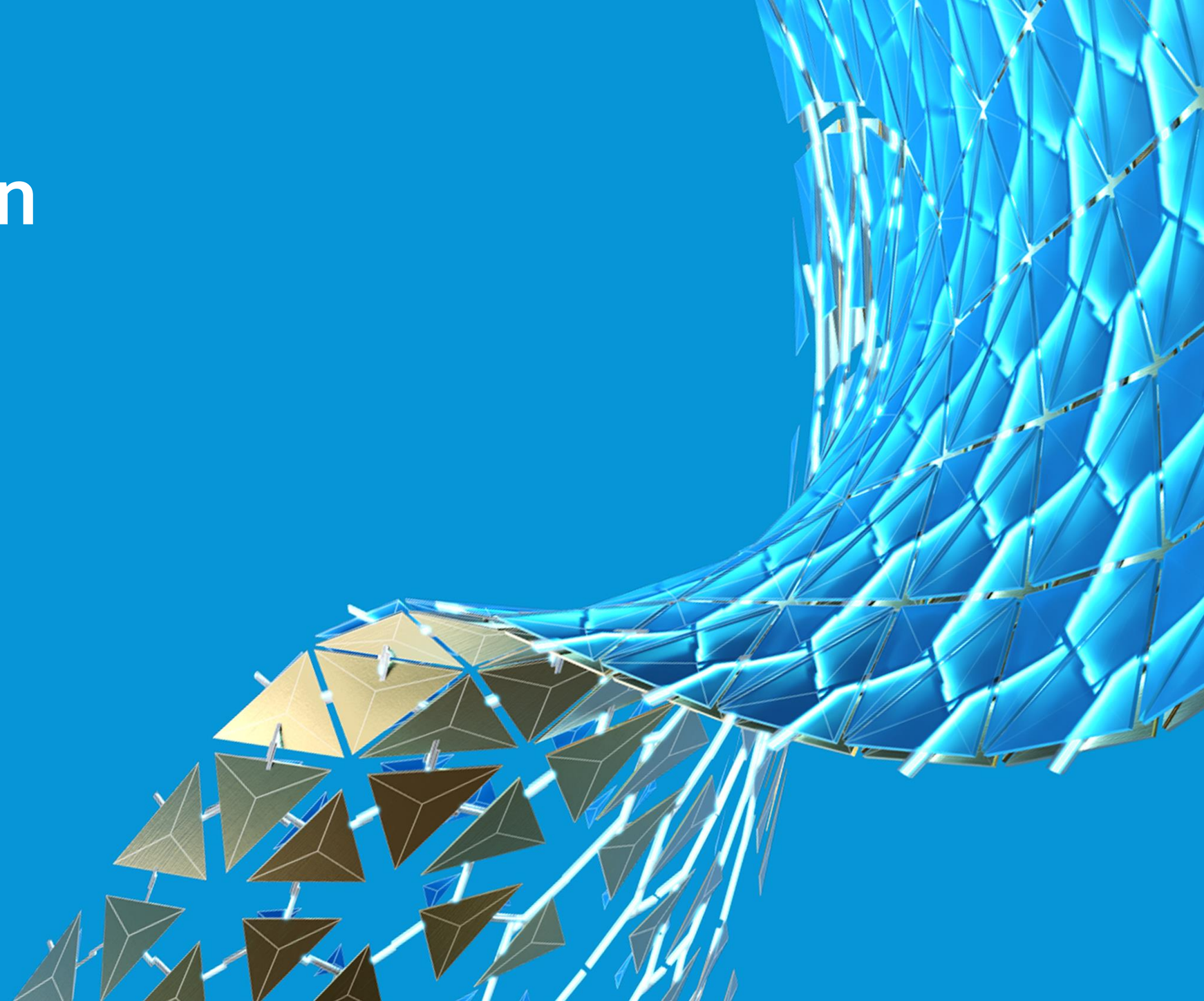
There are a variety of formats that the Forge viewer supports. The Model Derivative API enables translation of more than 70 different types of source file formats. Yet, there is one format ubiquitous in the architecture, engineering, and construction (AEC) industry that does not find its place in the list—RFA. There could be various ways to visualize RFA files in the viewer; the most common is: conversion to a format that's supported by Forge, though it's not the fastest or most effective. In this class, we'll look at a workflow that can help developers optimize up to 90% of their Forge credits usage. Accomplishing this requires building a headless Revit plug-in using the Revit API, preparing it for the Design Automation API, processing the outputs using the Model Derivative API, and finally rendering individual Revit families in the viewer. We'll also look at how we can give end users the ability to switch between multiple family types, as well as access the Instance and Type parameters directly in the Forge Viewer.

Important Points

Some general points about this session

- As you're aware, this time AU 2020 is happening online, and thus, this is a prerecorded instructional demo.
- If you have any questions or comments, please drop them on the class page on the AU website. You're also recommended to attend my dedicated Q&A session during AU 2020 where some of the questions will be answered live.
- If you're watching this after AU 2020 is over, feel free to drop me an email with your questions, comments or constructive feedback on viraj.voditel@techture.global
- A Handout & this Presentation are also available for ready reference on the AU website.

Introduction



What is Forge?

Authentication

Generate tokens based on the OAuth 2.0 standard to authenticate requests made to Forge APIs and SDKs.

- Developer's guide
- API reference

BIM 360

Integrate with the Autodesk BIM 360 platform to extend its capabilities to reach segments of the construction ecosystem that don't have direct access to BIM data.

- Intro
- Developer's guide
- API reference

Data Management

Access data across BIM 360 team, Fusion Team, BIM 360 Docs, and the Object Storage Service to build apps to display and extend your data in ways that add value to your users.

- Intro
- Developer's guide
- API reference

Design Automation

Automate repetitive tasks by leveraging on the scale of the Forge Platform and running scripts on your design files in the cloud.

- Intro
- Developer's guide
- API reference

Model Derivative

Derive outputs viewable by the Forge Viewer from more than 60 CAD file formats, and extract metadata about the models as well as the individual objects within the model.

- Intro
- Developer's guide
- API reference

Reality Capture

Convert digital images into high resolution textured meshes, dense point clouds and orthophotos.

- Intro
- Developer's guide
- API reference

Token Flex

Access Autodesk Token Flex Usage Data platform to generate reports on consumption, usage, and contract details.

- Intro
- Developer's guide
- API reference

Viewer

Render 3D and 2D model data within a browser. The models can come from a wide range of applications such as AutoCAD, Fusion 360, Revit, and many more.

- Intro
- Developer's guide
- API reference

Webhooks

Subscribe to and receive notifications of the occurrence of events within the Forge eco system.

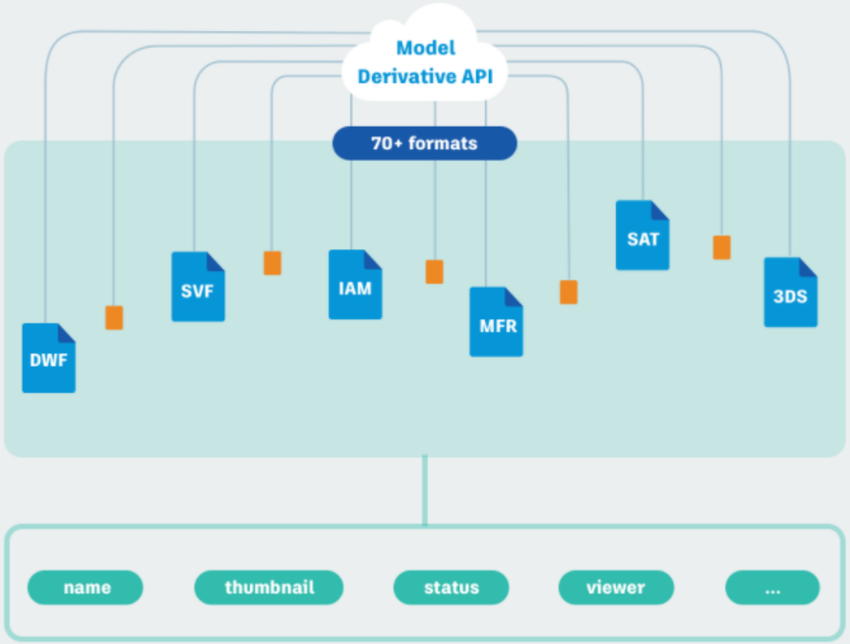
- Intro
- Developer's guide
- API reference

What is Forge?

Forge is a cloud-based developer platform from Autodesk. The Forge Platform offers APIs and services that help you access and use your design and engineering data via the cloud.

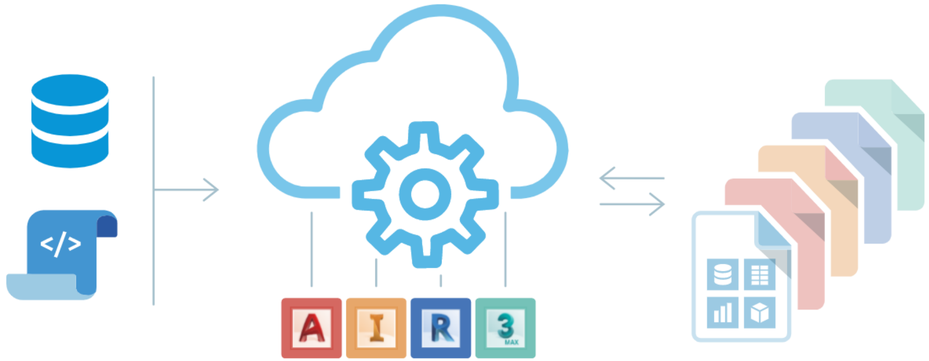
- BIM 360 API
- Data Management API*
- Model Derivative API*
- Design Automation API*
- Authentication API*
- Viewer API*
- Reality Capture API
- Token Flex API
- Webhooks

Model Derivative API



- This API can be used to prepare designs for rendering in the Viewer
- It can also be used to convert design files into other formats

Design Automation API



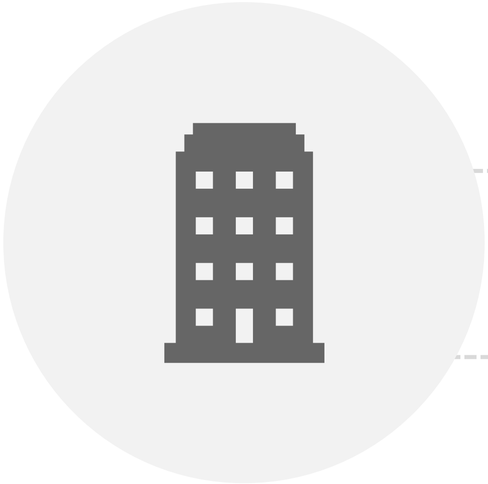
- Design Automation API for Revit is Revit's engine running in the cloud as a Forge service.
- It provides access to the full Revit DB API without a Revit desktop install, so that you can build cloud-native apps and services that create, extract, and modify Revit data

Different Formats supported by Forge

- Autodesk Forge viewer supports 70+ formats.
- Pertaining to the AEC industry, the popularly supported formats include .rvt, .dwg, .ifc, .dgn and more.
- However, .rfa, which is a format used for Revit families is not supported.

Different Revit formats

- There are 4 major formats in Revit
- When working in a project environment, the format is .rvt
- When working in a component/family environment the format is .rfa
- The other 2 formats are templates for the respective environments

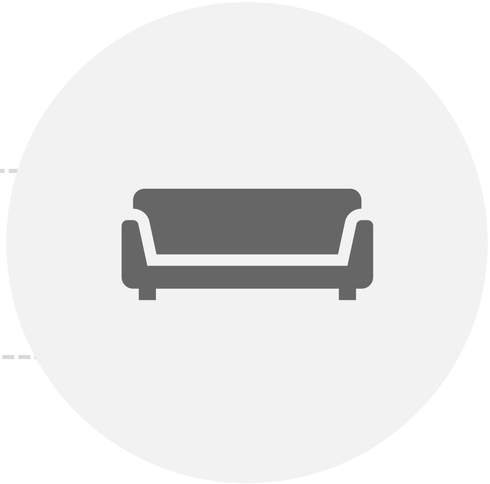


rvt

rte

rfa

rft



How to render RVT in Forge Viewer?

1

Convert and display
the processed file in
Forge Viewer.

How to render RFA in Forge Viewer?

1

Create Revit addin to place .rfa file(s) into a .rvt file.

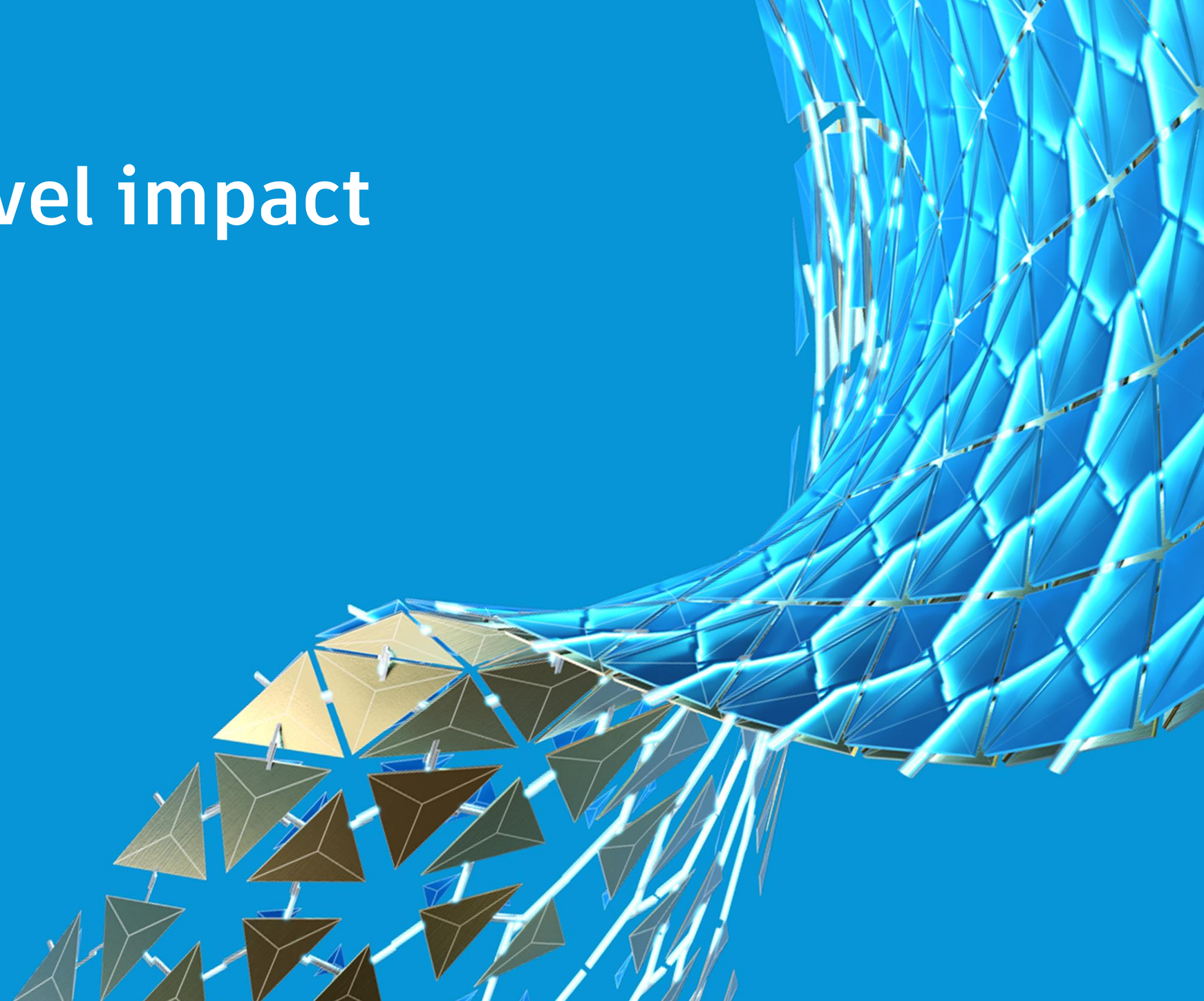
2

Execute the addin using Design Automation and process it to update the .rvt file

3

Convert and display the processed file in Forge Viewer.

Business level impact



Applicable Forge Pricing

Standard Pricing

Design Automation API	6 credits per processing hour
Model Derivative API	1.5 credits per job

Applicable Forge Pricing

Example 200 Revit families to be displayed in viewer.

Workflow 1

Each .rfa placed in one .rvt file and processed using Model Derivative API	$ 1.5 x 200
Assuming Design Automation Engine runs for 1 hour	$ 6
Total	**$ 306**

Workflow 2

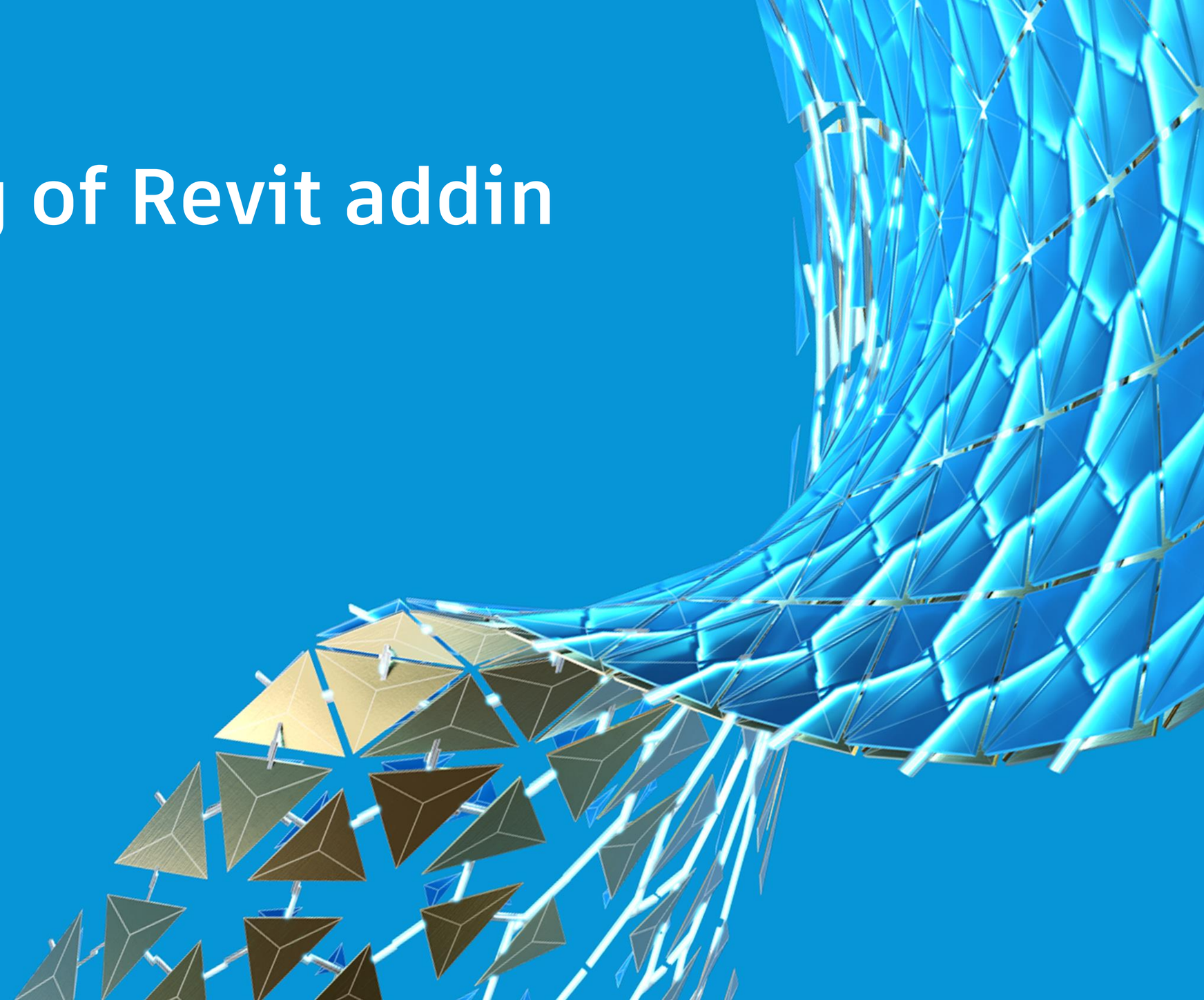
All .rfa files placed in one single .rvt file and processed using Model Derivative API	$ 1.5
Assuming Design Automation Engine runs for 10 mins	$ 1
Total	**$ 2.5**

*Assuming 1 credit = 1 USD

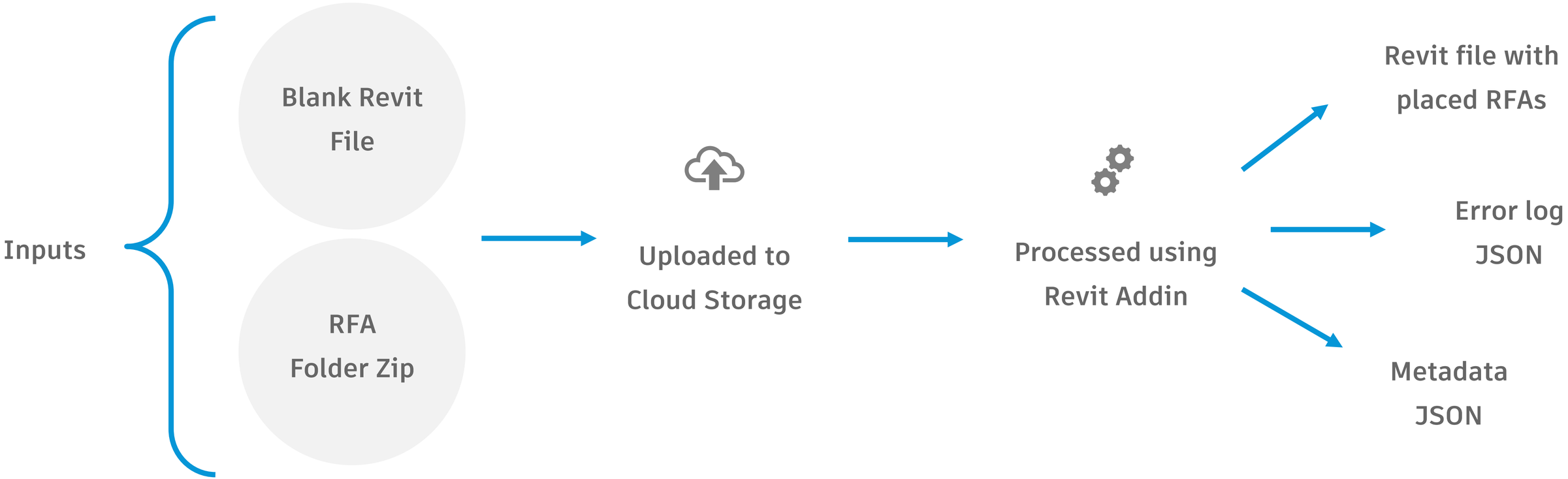
Potential Business Impact

- Savings of $306-$2.5 = $303.5 for a batch of 200 families
- A whopping 99% cost reduction
- Processing time reduction by almost 85%
- Exponential savings when batch of families increase to say 2000 instead of 200

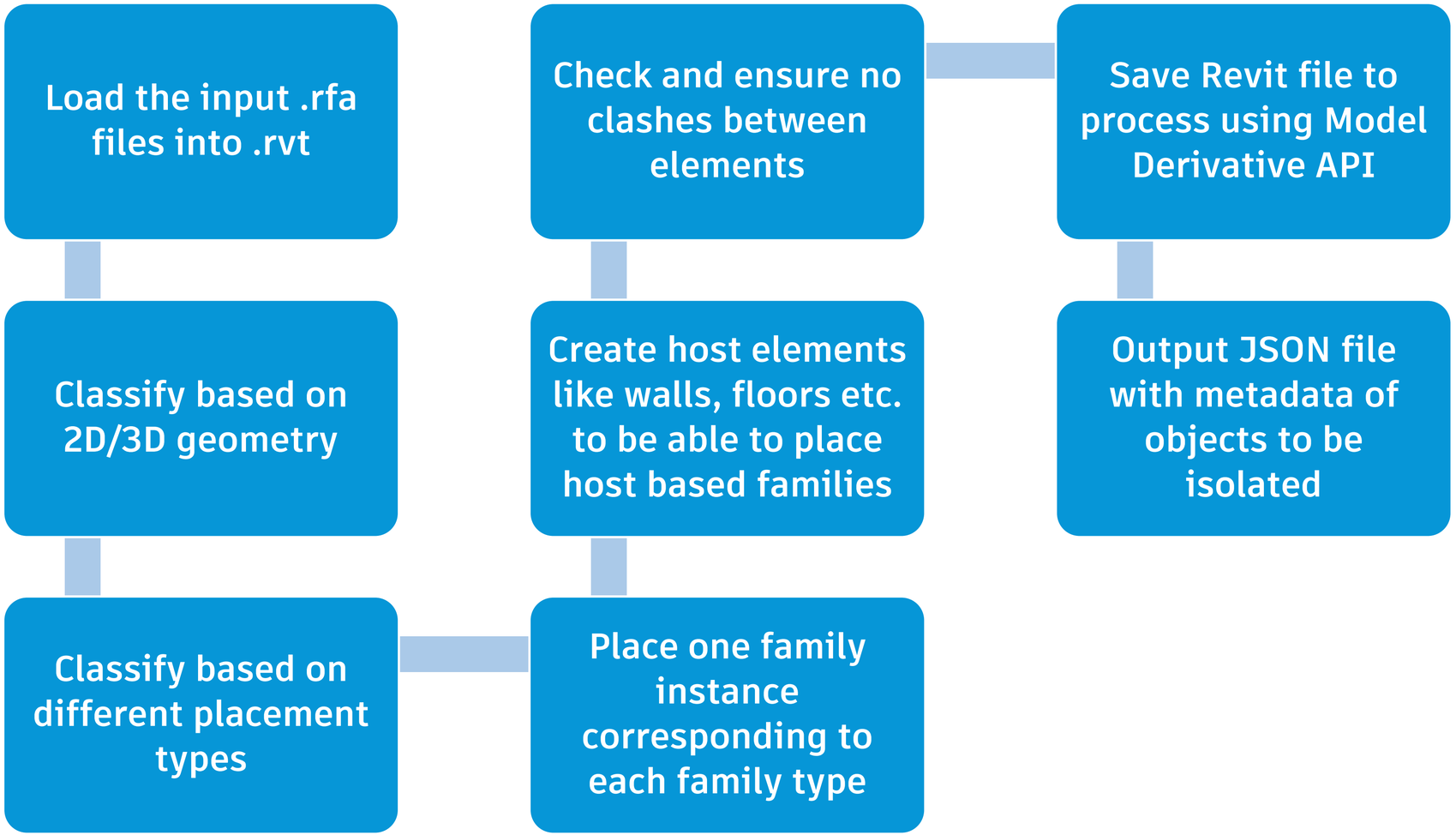
Functioning of Revit addin



Functioning of Revit addin



Revit Addin Logic



Types of Families

There are various kinds of templates on which the families can be based upon. The addin should be able to handle all these cases.

- Host Based
- Line Based
- Face Based
- Non-Host Based
- Adaptive Component

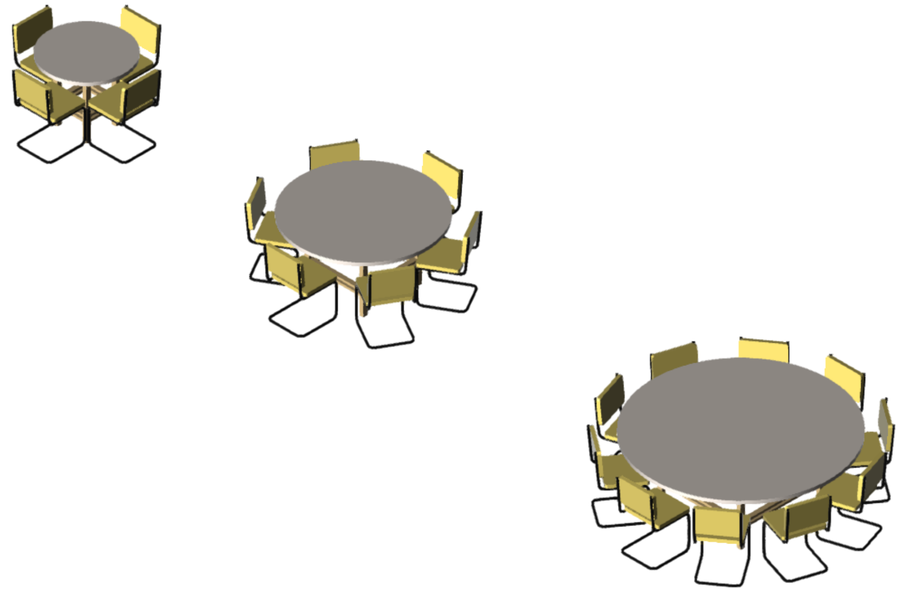
Extracting Family Types and Parameters

- We need to place instances of all the types of each RFA. During placement, we are storing a JSON file which will store the metadata which will be like this:

  Family name > Type name > Element ID of that type instance

  This is done to facilitate the frontend development to be able to show family types and their parameters.

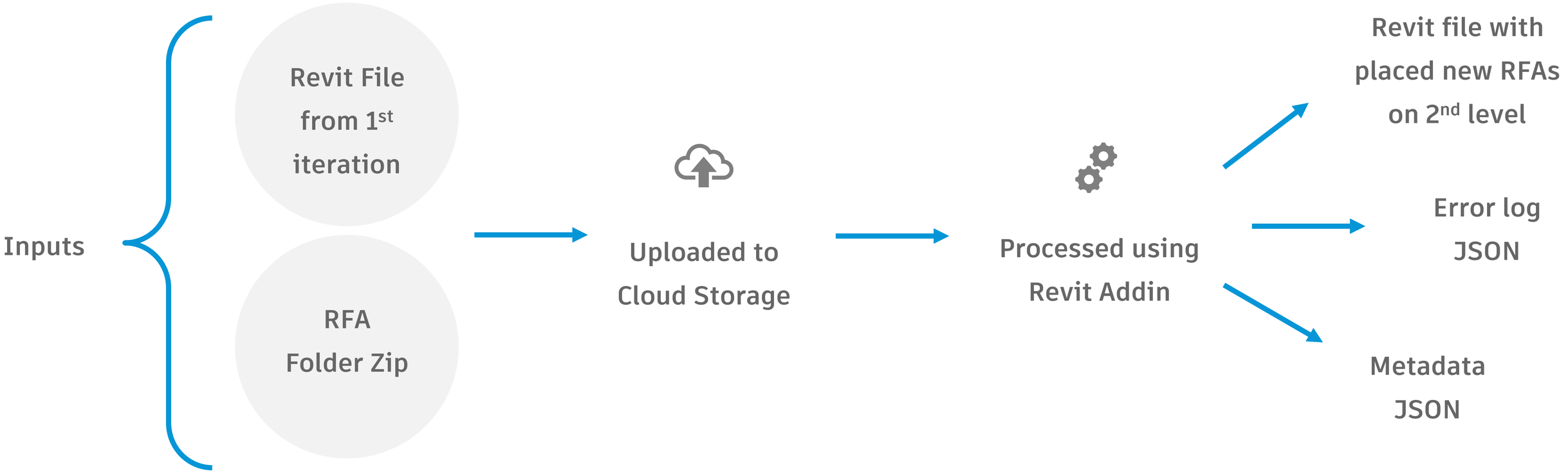
- We store it in our cloud storage in a JSON file.
- In case of any errors we also log the errors in a separate file which is also on cloud storage.



Forge API limitations and overcoming them

- There is a soft limit on Forge Design Automation API. It can only process 200 objects at a time.
- This being a limiting factor, we need to design our addin to place these instances on one level in the Revit file, and subsequent ones, on other levels. (Level refers to the Levels category in Revit)
- If for instance, we have to devise a solution in which we have 400 RFAs.
- So in this case, we will first work on 200 RFAs as it is the current limit.
- We call it the 1st iteration and the rest of the process would be similar as above.
- In the 2nd iteration, plugin expects an input file which is an output from the first iteration.

Functioning of Revit addin



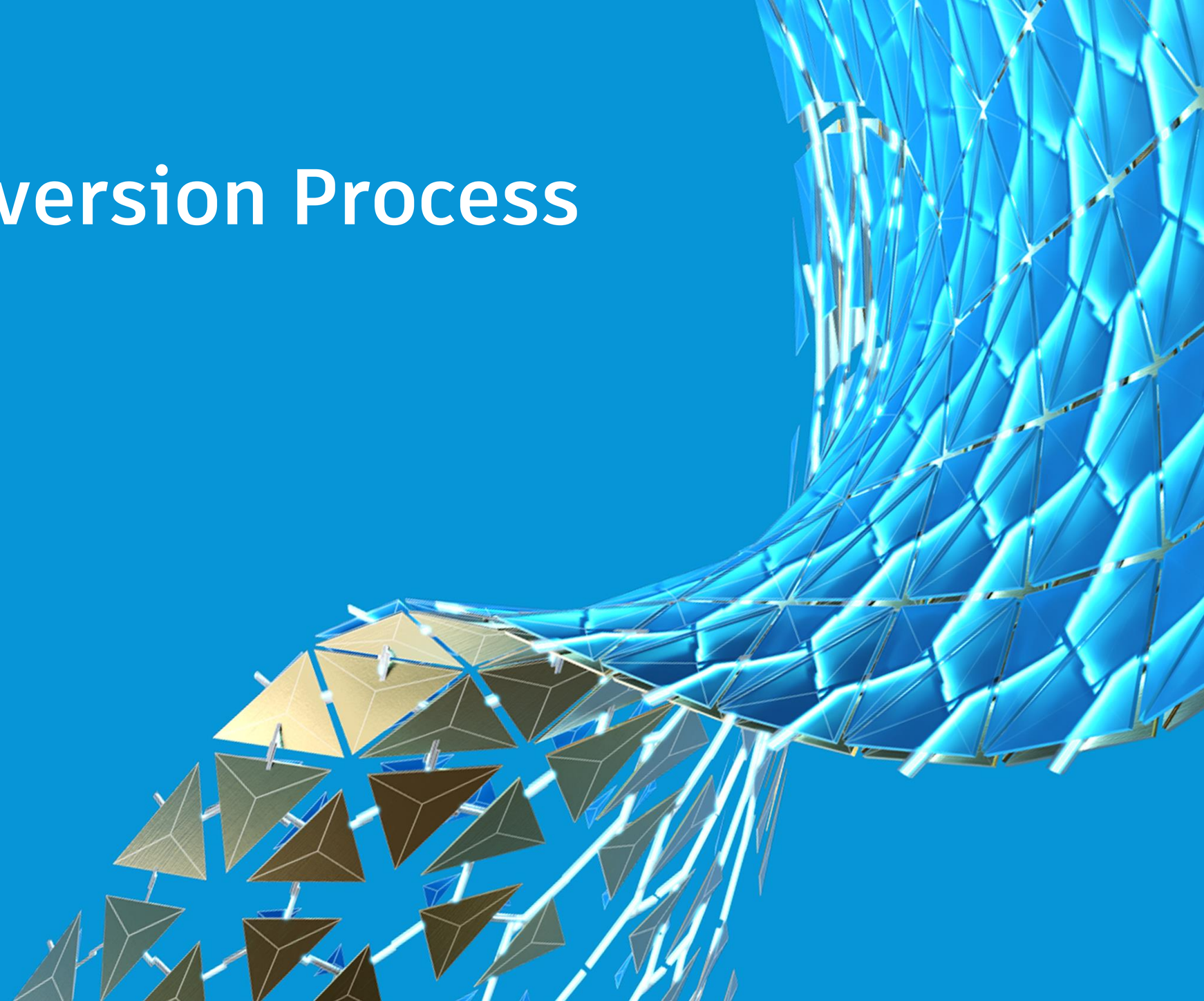
Running Revit addin in the cloud

Once this “headless” Revit addin is ready, you need to process it using Design Automation which consists of the below steps.

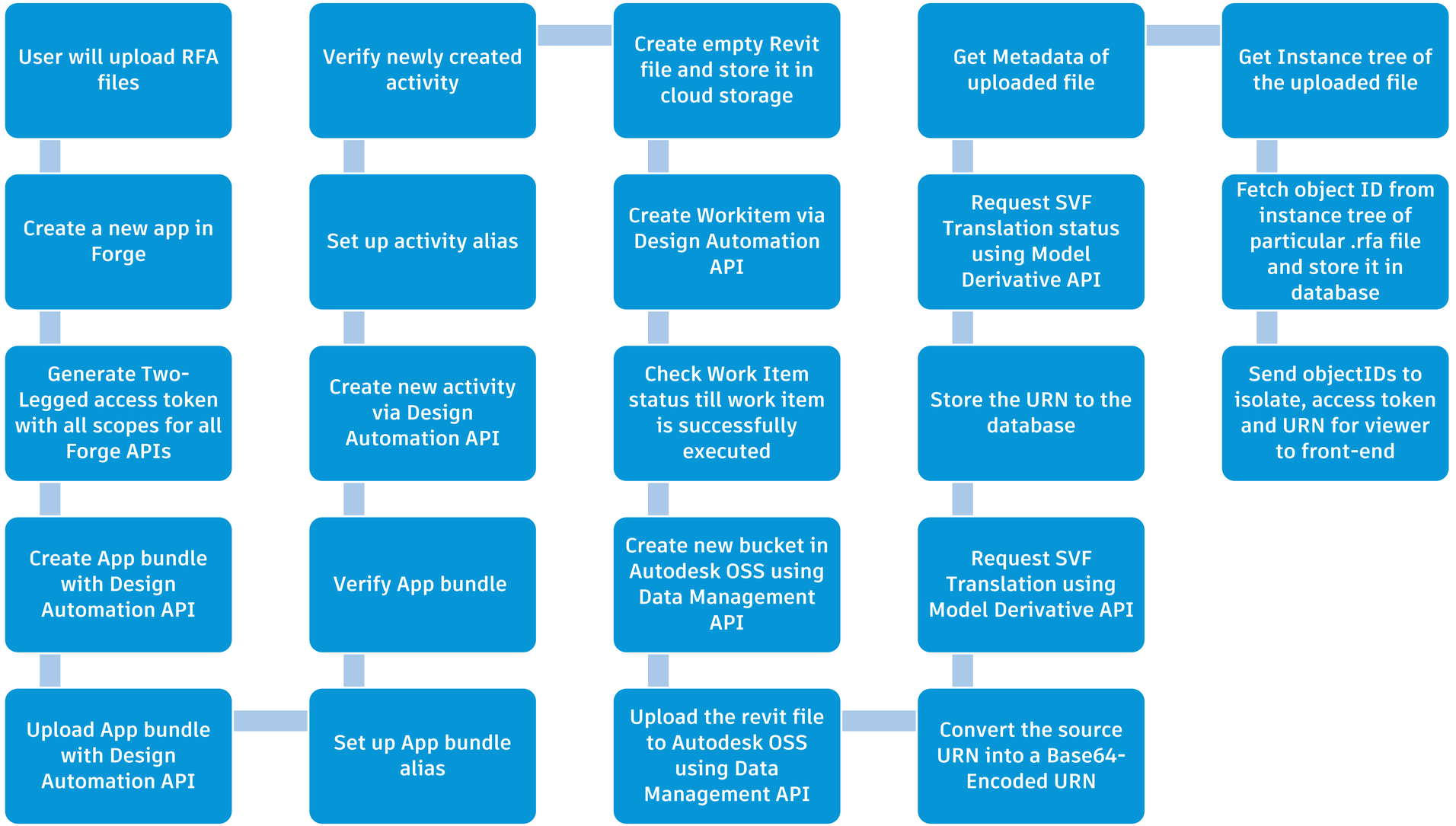
- Create nickname for app
- Create AppBundle for Design Automation
- Create activity for the app bundle
- Create work item
- Check work item status

You can refer Design automation guide provided by Autodesk here:
https://forge.autodesk.com/en/docs/design-automation/v3/tutorials/revit/

Overall Conversion Process



Overall Conversion Logic



STEP 1: Get your authentication token

This code snippet illustrates getting a simple 2-legged Oauth token from Forge APIs.

REQUEST

```
POST /authentication/v1/authenticate HTTP/1.1

Host: developer.api.autodesk.com

Content-Type: application/x-www-form-urlencoded

client_id=<CLIENTID>&client_secret=<CLIENTSECRET>&gr
ant_type=client_credentials&scope=data:read
```

RESPONSE

```
{

  "access_token":
"eyJhbGciOiJIUzI1NiIsI...mtpZCI6Im
1tZXRyaWNfa2V5InO.eyJzY29wZg....UTJO-LD2WSC",

  "token_type": "Bearer",

  "expires_in": 3599

}
```

STEP 2: Upload file to Autodesk Storage for conversion

To enable the file to be processed by Forge APIs, we need to upload it to Autodesk's OSS. For that, we need to create a bucket.

REQUEST

```
POST /oss/v2/buckets HTTP/1.1

Host: developer.api.autodesk.com

Authorization: Bearer eyJhbGciOiJI-LITD2WSCo....
Content-Type: application/json
```

RESPONSE

```
{

  "bucketKey": "yourbucketname",
  "policyKey": "persistent"

}
```

STEP 2: Upload file to Autodesk Storage for conversion

Once the bucket is created, we need to upload the RVT file to that bucket. This RVT file is the one that was created after processing by the “headless” addin.

REQUEST

```
PUT
/oss/v2/buckets/mybucket/objects/FILENAME
HTTP/1.1

Host: developer.api.autodesk.com

Authorization: Bearer eyJhbGciOiJIUzI1NiIs...
Content-Type: application/octet-stream

Cookie: PF=kQMtLhxRMMFS5ZLUXw9K9Q

"<file contents here>"
```

RESPONSE

```
{
  "bucketKey": "mybucket",
  "objectId": "urn:adsk.objects:os.object:mybucket/ABC.rvt",
  "objectKey": "ABC.rvt",
  "sha1": "1ad934ea67a012bd557f3c24b7ba929cf7f95aaa",
  "size": 28807168,
  "contentType": "application/octet-stream",
  "location":
"https://developer.api.autodesk.com/oss/v2/buckets/mybucket/objects/ABC.rvt
}
```

STEP 3: Convert the uploaded file

We will use the Model Derivative API to convert the file to SVF Format so that it is viewable in the Forge Viewer.

REQUEST

```
POST /modelderivative/v2/designdata/job
HTTP/1.1

Host: developer.api.autodesk.com

Authorization: Bearer eyJhbGciOiJ...

Content-Type: application/json
```

RESPONSE


```
{
 "input": {

  "urn":
"dXJuOmFkc2sub2JqZWNOczpvcy5vYmplY3Q6dGVzdGFuZGRldjIyMDkyMDIwL0hWQUNfQ
UhVLENoaWxsZXIlMjAmJTIwQ29vbGluZyUyMFRvd2VyMzEwNy5ydnQ"

 },

 "output": {

  "formats": [
   {
    "type": "svf",

    "views": [

     "2d",

     "3d"

    ]
   }
  ]
 }
}
```

STEP 4: View the converted file in Forge Viewer

To be able to view the files in the viewer, you need to set up a basic frontend, done here using HTML and JS.

Setup HTML File

Setup Javascript file and initialize viewer

View the file in viewer

STEP 4: View the converted file in Forge Viewer

Setup HTML File

Preparing for viewer is very simple, create a folder in your desired directory, and create index.html file inside it and copy below code into it

```
<head>

      <meta name="viewport" content="width=device-width, minimum-scale=1.0,
initial-scale=1, user-scalable=no" />

            <meta charset="utf-8">
                  <link rel="stylesheet"
href="https://developer.api.autodesk.com/modelderivative/v2/viewers/7.*/style
.min.css" type="text/css">

      <script
src="https://developer.api.autodesk.com/modelderivative/v2/viewers/7.*/viewer
3D.min.js"></script>

      <style>

            body {

            margin: 0;

            }

            #forgeViewer {

            width: 100%;
            height: 100%;
            margin: 0;
            background-color: #F0F8FF;
            }

      </style>

</head>
<body>

 <div id="forgeViewer"></div>

</body>
```

STEP 4: View the converted file in Forge Viewer

Setup Javascript file and initialize viewer

Create a main.js file in the same directory where you have created index.html file and copy paste below code into it

Please paste the access token that you get from your forge app in place of ‘YOUR_ACCESS_TOKEN

```
var viewer;
var options = {
  env: 'AutodeskProduction',
  api: 'derivativeV2',  // for models uploaded to EMEA change this option to 'derivativeV2_EU'
  getAccessToken: function(onTokenReady) {
    var token = 'YOUR_ACCESS_TOKEN';
    var timeInSeconds = 3600; // Use value provided by Forge Authentication (OAuth) API
    onTokenReady(token, timeInSeconds);
  }
};
Autodesk.Viewing.Initializer(options, function() {
  var htmlDiv = document.getElementById('forgeViewer');
  viewer = new Autodesk.Viewing.GuiViewer3D(htmlDiv);
  var startedCode = viewer.start();
  if (startedCode > 0) {
    console.error('Failed to create a Viewer: WebGL not supported.');
    return;
  }
  console.log('Initialization complete, loading a model next...');
});
```

STEP 4: View the converted file in Forge Viewer

View the file in Viewer

Let's complete the final step and view the file in viewer, paste the below code into your main.js file

```
var documentId = 'urn:<YOUR URN HERE>';

Autodesk.Viewing.Document.load(documentId, onDocumentLoadSuccess,
onDocumentLoadFailure);

function onDocumentLoadSuccess(viewerDocument) {

  var defaultModel = viewerDocument.getRoot().getDefaultGeometry();

  viewer.loadDocumentNode(viewerDocument, defaultModel);

}

function onDocumentLoadFailure() {

  console.error('Failed fetching Forge manifest');

}
```

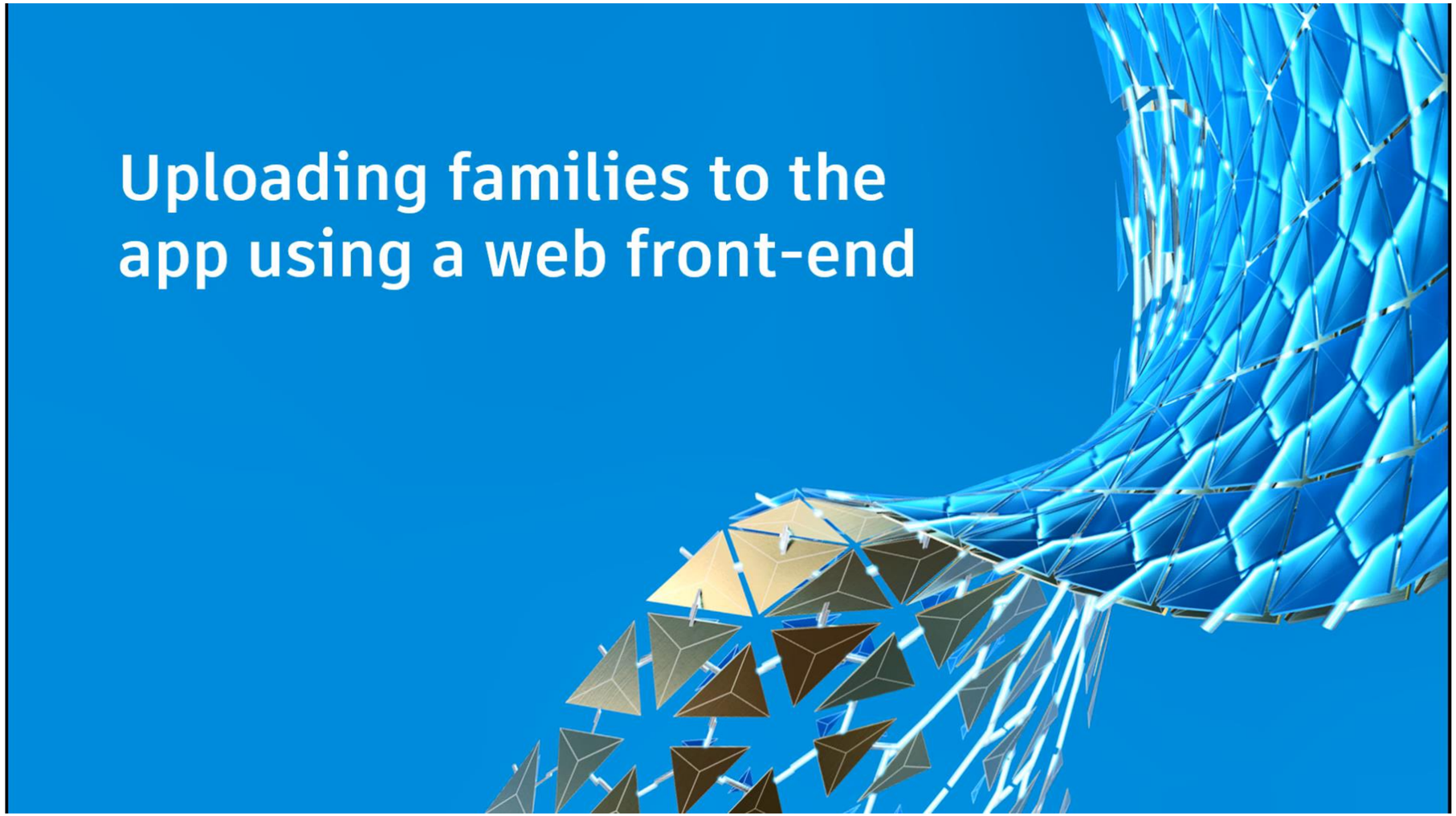
Paste the urn that you have saved from the file conversion, don't worry if you haven't saved or copied it, just hit that api again with all the same parameters and you will get the URN.

STEP 5: Isolate specific object(s) in Forge viewer

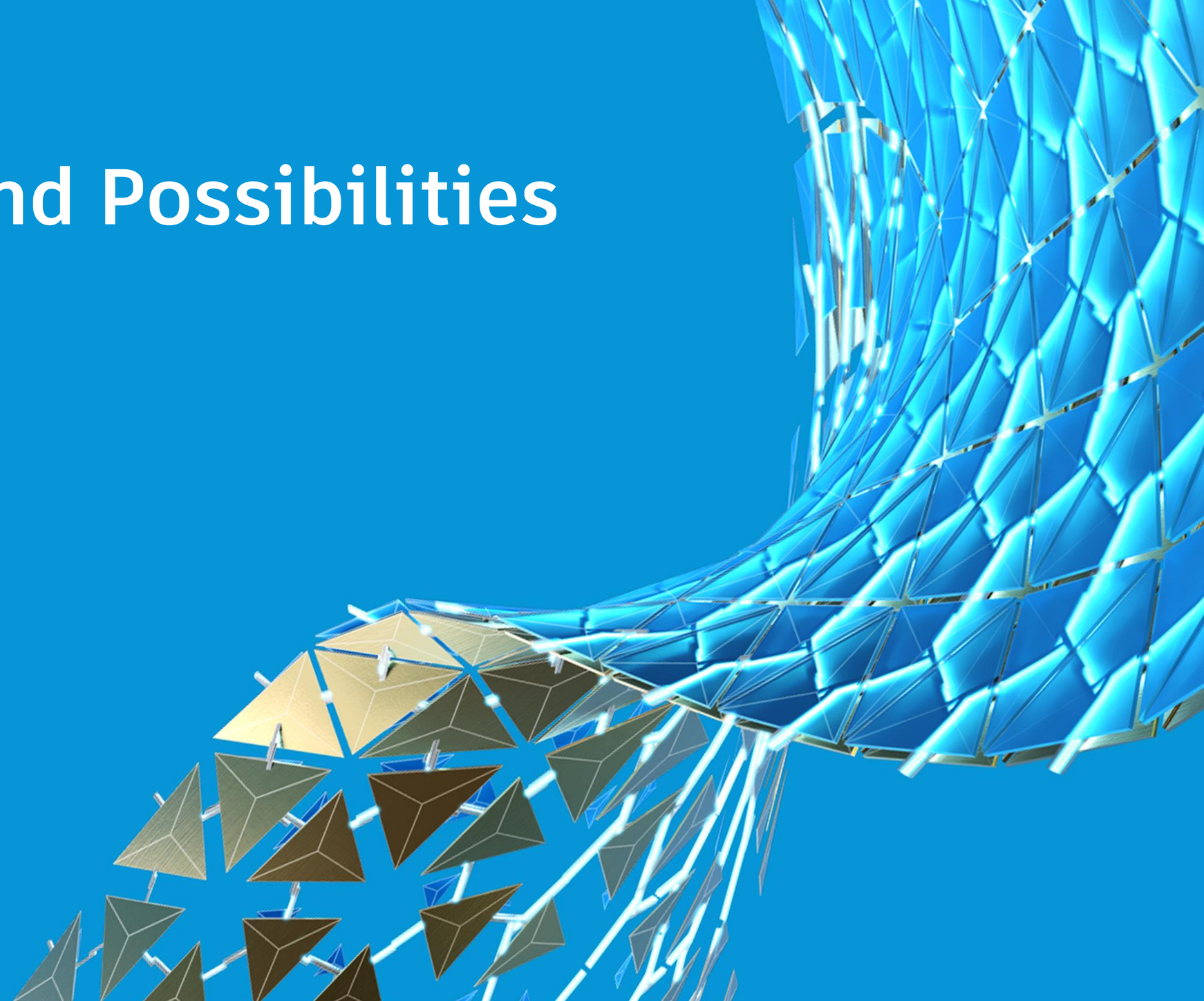
- Get DBID/ Object ID of the object you wish to isolate (this can be obtained from metadata)
- Call isolate function on the DBID/ Object ID
- We need to isolate a particular object because we are placing all the .rfa objects uploaded by user into a .rvt file, which can be further differentiated by isolating them.

```
viewer.addEventListener(Autodesk.Viewing.OBJECT_TREE_CREATED_EVENT, ev =>

{
      let dbids = [<YOUR DBIDS>]

      viewer.isolate(dbids)

      viewer.fitToView(dbids[0], 1, true);

    })

}
```

Sample demo of the process



Examples and Possibilities

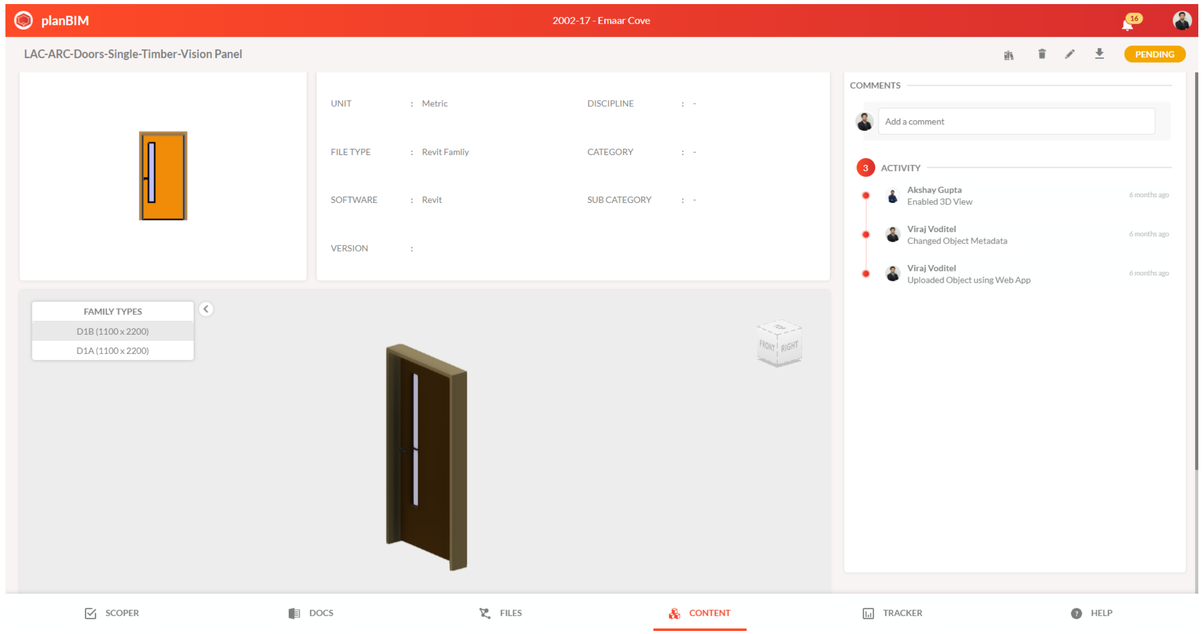


Integration of workflow in planBIM

- One of the live examples of where we've implemented this exact workflow is in a SaaS app www.planbim.io

- In the Content Module of the app, the user can upload RFA files and use it as a cloud repository of all the families to be used in their project/organization.

- The app provides a 3D preview of the families uploaded to the users as well as the family types and parameters information.



planBIM

Further Possibilities

This workflow will be useful for various use cases. Here are a few examples:

- An app for previewing and managing digital content like Revit families on the cloud e.g. planBIM
- A Manufacturer would like to host 3D models of all their Revit families on their website
- An app for communicating the design changes at a component level to someone who doesn't have Autodesk Revit installed on their computer.

Feel free to connect with me to discuss any further ideas around this. It can also be extended to showcase 2D families by making some changes in the workflow.