

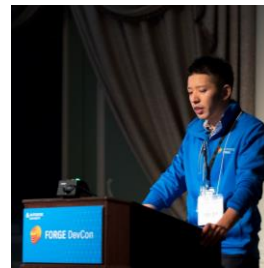
Autodesk Construction Cloud と Design Automation for Revit でつながる維持管理プロセス

小笠原 龍司

Developer Advocate | @AutodeskForge

アジェンダ

- 建築物の維持管理に BIM データを活用するためのクラウドソリューション
 - 共通データ環境（CDE）としての Autodesk Construction Cloud
 - Forge Viewer と Extension
 - Design Automation for Revit
- FM システム連携アプリケーションの作成
 - 外部データベースとの連携
 - 問題の報告・履歴の閲覧
 - Revit ファミリを Forge Viewer 表示用にプロジェクトに変換する
 - プロジェクトのファミリインスタンスを別のファミリと交換する
 - Revit モデルの変更箇所を確認する



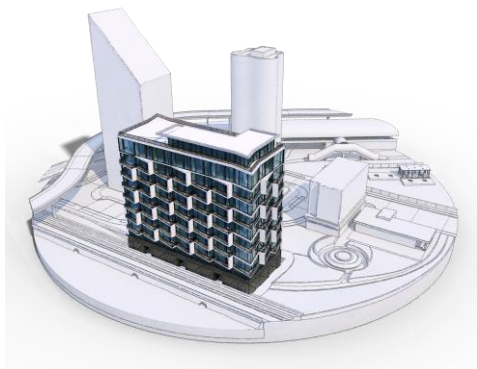
小笠原 龍司

2014年からオートデスクにて、Revit や Navisworks 等、デスクトップ製品 API ・アドイン開発・API のサポート、及び Forge プラットフォーム API のエバンジェリストとして活動中。前職では、Webアプリケーションエンジニアとして環境制御システムやリアルタイムコミュニケーションシステム、統合基幹業務システムの開発に従事。

建築物の維持管理に BIM データを
活用するためのクラウドソリューション

建築物の維持管理・運用の高度化・効率化

設計、施工・製造から維持管理まで、ライフサイクル全体を通じて BIM データを活用。



企画/基本/実施設計

設計BIM

(LOD 100-400)



施工/製造

施工BIM

(LOD 300-500)

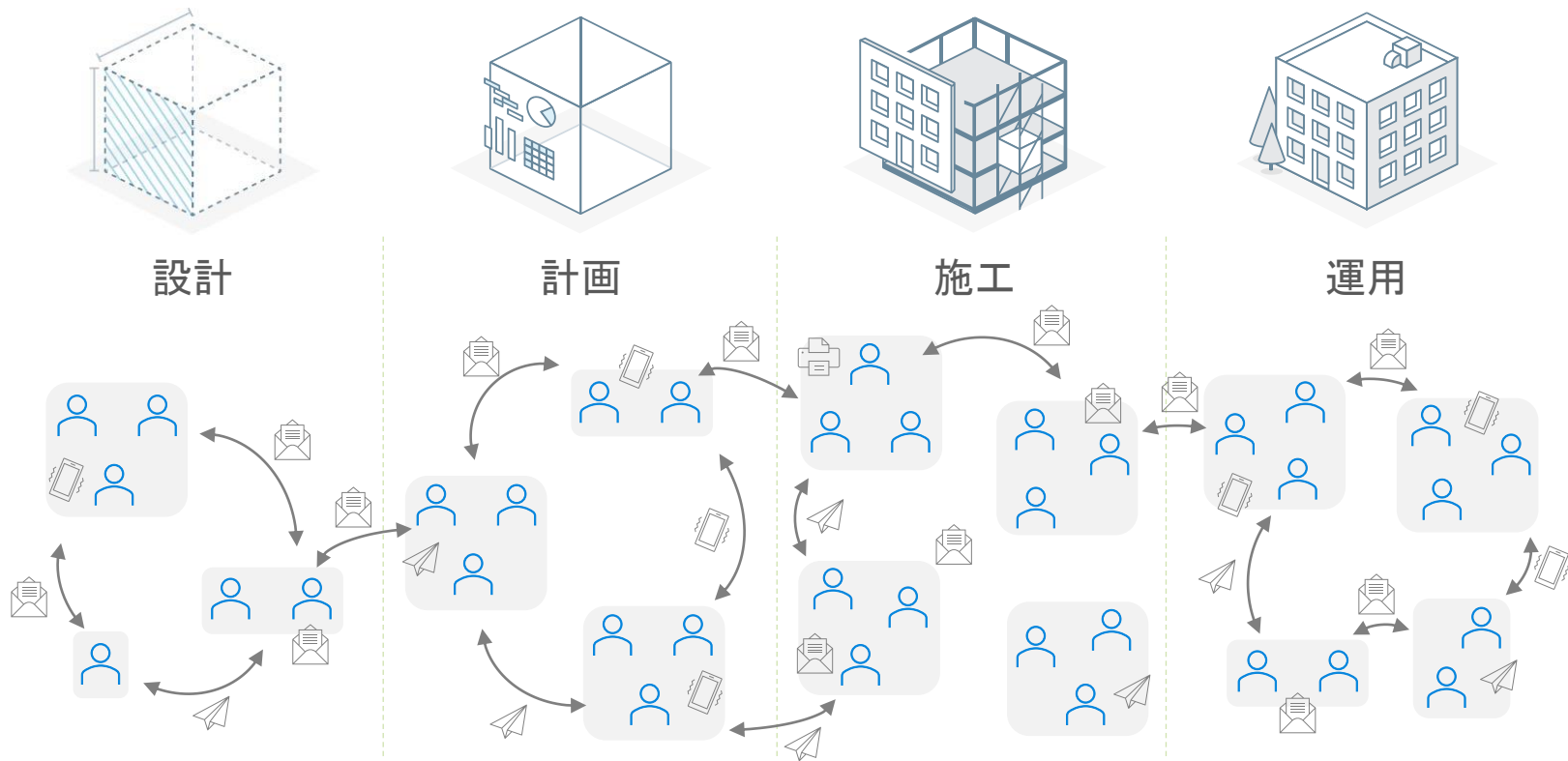


維持管理

維持管理BIM

(LOD 400-500)

データ共有の現状



共通データ環境（CDE）としての Autodesk Construction Cloud

国際規格 ISO 19650 に準拠した命名規則、バリデーション機能。

命名規則に準拠していないファイルを一時的に保持しておける「保持領域」機能。

機能	共同設計	調整 干渉チェック	数量拾い出し	プロジェクト管理	コスト管理	品質	安全	プロジェクト完了
ワークフロー	<ul style="list-style-type: none">構成レビュー変更箇所の視覚化Revit クラウドワークシェアリングC4C3DC4P3D	<ul style="list-style-type: none">モデル調整・干渉チェック構成レビュー安全計画指摘事項管理	<ul style="list-style-type: none">スコープ分析2D/3D 数量拾い出し	<ul style="list-style-type: none">RFIs提出ミーティング・日報進捗状況の追跡	<ul style="list-style-type: none">契約書作成/管理注釈変更支払い、精算予測	<ul style="list-style-type: none">品質計画品質チェックレビューリスト指摘事項/タスク管理	<ul style="list-style-type: none">安全計画安全教育安全巡視・監視安全検査	<ul style="list-style-type: none">コミッション引き渡し保証管理竣工現況調査
統合データ	ドキュメント管理・バージョン管理・ドキュメントワークフロー モデル・図面・ドキュメント・コスト情報・アセット・現況情報・写真・ビデオ							
Insights	ダッシュボード・レポート・Construction IQ・Data Connector							
管理	Admin コンソール・権限・プロジェクトセットアップ・テンプレート・通知							

導入障壁

様々なソフトウェアやサービス、システムを既に運用中。

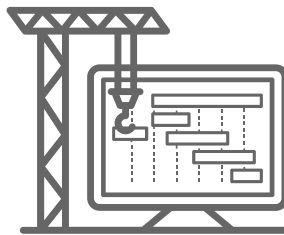
一度に置き換えることは困難。



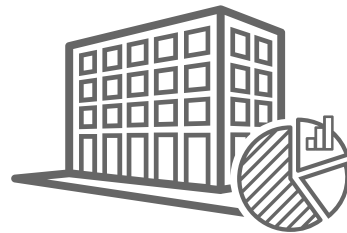
A ソフトウェア



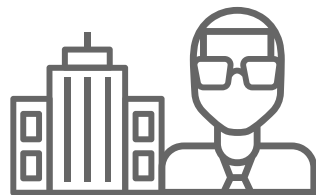
B サービス



C ソフトウェア



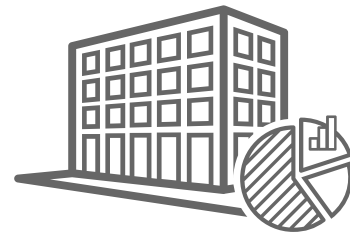
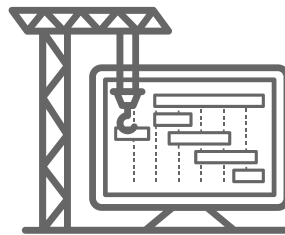
D システム



Autodesk Forge によって分断されたプロセスを接続する

Autodesk Construction Cloud を CDE として利用しつつ、部分的・段階的に導入。

複雑化したワークフローを Autodesk Forge によって統合。



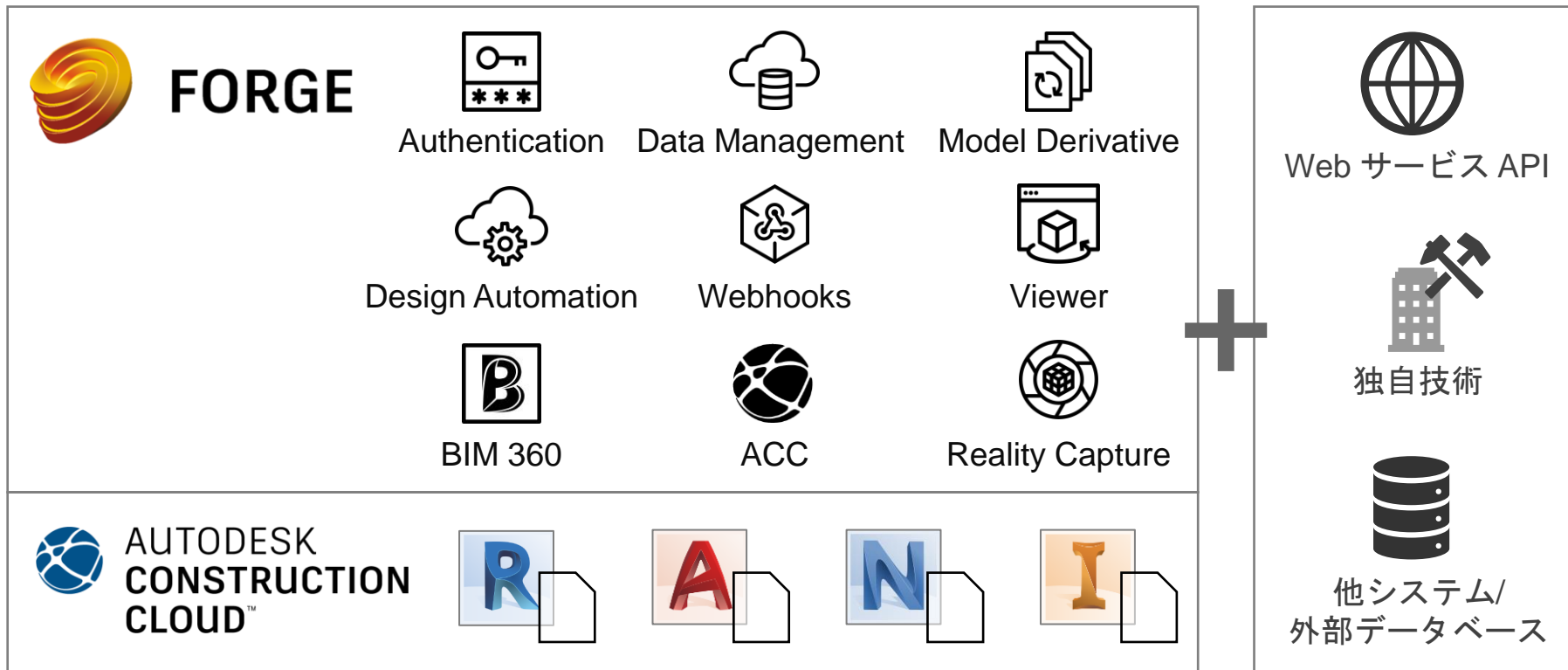
AUTODESK CONSTRUCTION CLOUD™



FORGE



Autodesk Construction Cloud + Autodesk Forge + ???



Forge Viewer

- モデルやドキュメントの表示のみ。
- カスタマイズの機構を提供。

JavaScript API

- 属性の抽出や検索、モデルの断面解析や分解などの標準機能を制御。

Extension フレームワーク

- ビューアに独自処理を組み込んでカスタマイズするための拡張モジュール。
- JavaScript ファイル単位で読み込む。

three.js によるジオメトリ操作

- three.js の API を通じて、モデルのジオメトリやマテリアルを一時的に操作したり、ジオメトリを生成してメッシュを追加。



Extension による機能拡張

最新バージョン v7.50 2021/8/19

多数の Extension が公開済み

必要に応じてロード・アンロードして利用可能

AnimationExtension

BimWalkExtension

CrossFadeEffects

DocumentBrowser

Edit2DExtension

ExplodeExtension

FullScreenExtension

FusionOrbitExtension

GeolocationExtension

GestureDocumentNavigationExtension

gITF

GoHomeExtension

HyperlinkExtension

LayerManagerExtension

MeasureExtension

MinimapExtension

ModelBuilder

ModelStructureExtension

NavToolsExtension

PDFExtension

PopoutExtension

PropertiesManagerExtension

RollCameraExtension

SceneBuilder

SectionExtension

SnappingExtension

SplitScreenExtension

ViewCubeUi

ViewerSettingsExtension

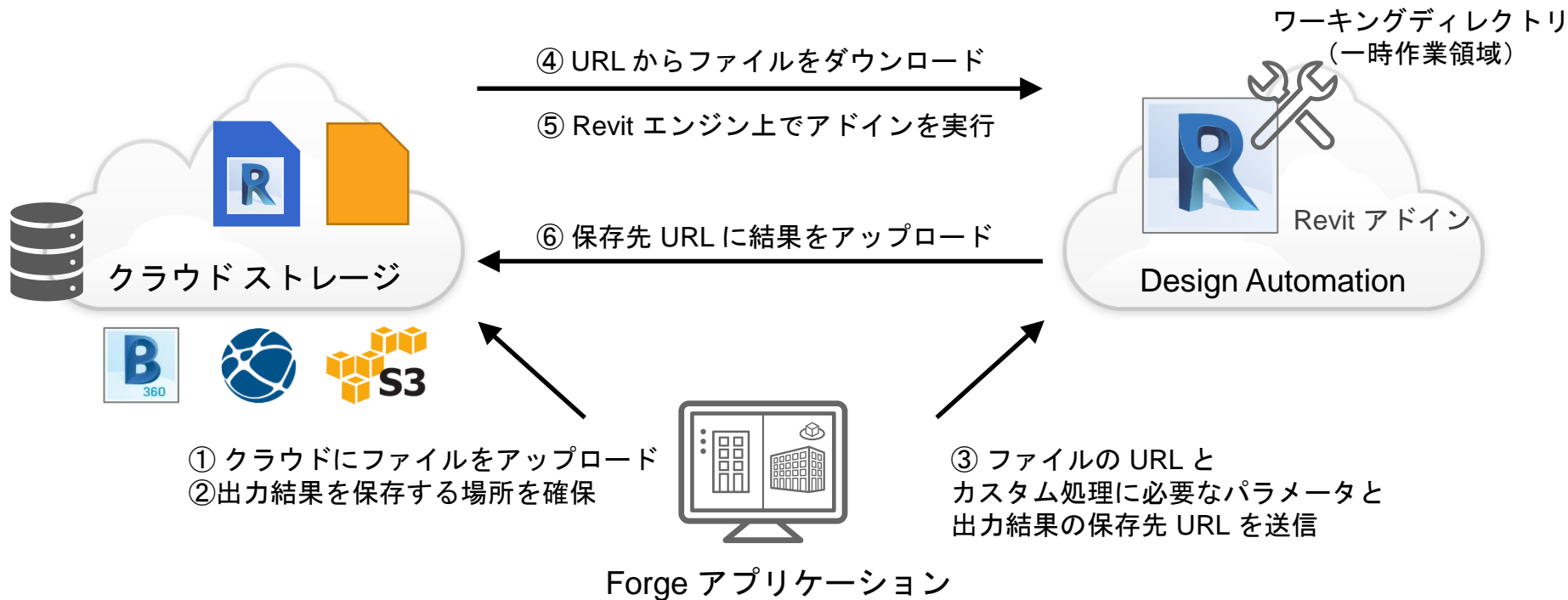
WireframesExtension

ZoomWindow

Design Automation for Revit による自動化

Design Automation API の一般的なワークフロー

Revit API を利用してカスタム処理を実装したアドインを通じて、Revit プロジェクトやファミリの作成、編集、属性情報へのアクセスをサポート。



Design Automation for Revit による自動化

Design Automation API の仕組み

AppBundle



アドインのバイナリパッケージ

Id : UpdateRVTParam

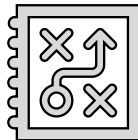
Engine : Autodesk.Revit+2021

Description: Test custom app

Package: Storage URL

Revit の .NET API で作成したアセンブリや関連ファイルを ZIP 圧縮してアップロードし、AppBundleとして登録する。

Activity



実行されるアクションの定義

Id: UpdateRVTParamActivity

Input Parameter : RVT, JSON

Output Parameter: RVT

AppBundle: UpdateRVTParam

カスタム処理の雛型を定義する。
.NET アセンブリ内でどんなデータを入力して、どんなデータを出力するか定義する。

WorkItem



指定のアクションを呼び出すジョブ

Id: 返却される文字列

Activity : UpdateRVTParamActivity

Input Parameter : File URL, JSON

Output Parameter: Storage URL

REST API でリクエストするジョブ。
対象のモデルやテキストデータ、出力先の URL と、実行する Activity を指定する。

デスクトップ版 Revit ≠ Revit エンジン

デスクトップ版 Revit のようなユーザーインターフェースはありません。

- 必要に応じて Forge Viewer の利用を検討できます。

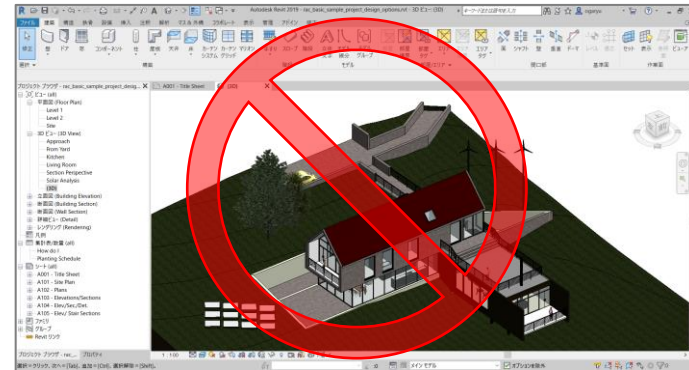
UI がないため、インタラクティブな操作はサポートされていません。

- 事前にパラメータとして必要な情報を渡す必要があります。

複雑なセッション管理は想定されていません。

WorkItem の処理は様々な影響を受けます。

- クラウドストレージとファイルを送受信するため。
- AWS と契約したコンピュータリソースを共有しているため。
 - パブリッククラウドの特性。
 - クラウドの自動伸張機能（elastic computing）の特性。
- インターネット回線などネットワークの負荷など。





FM システム連携 アプリケーションの作成

ワークフロー

1. Revit モデルの要素（ファミリインスタンス）を外部データベースと連携させる。
 - 今回は、外部データベースとして SQLite を使用し、アセットテーブルで管理。
2. 問題の発生している要素を報告する。履歴を閲覧する。
 - ACC は、現時点では指摘事項の API（Issues API）が未サポートのため、別途、SQLite で Maintenance テーブルで管理。
3. Revit ファミリから Forge Viewer 表示用に Revit プロジェクトを作成。
 - ACC, Model Derivative API, Forge Viewer は、Revit ファミリ（RFA ファイル）の表示は未サポート。
4. Revit モデルの要素（ファミリインスタンス）を別のファミリと交換する。
5. Revit モデルの変更箇所を確認する。

ACC プロジェクトのファイル構成

AUTODESK CONSTRUCTION CLOUD

Build ▾

Ryuji ACC Japan Office Building 🌐 ▾

🔍 ?

Ryuji Ogasawara ▾

ホーム

シート

ファイル

指摘事項

フォーム

写真

情報提供依頼

提出物

ミーティング

アセット

レポート

メンバー

設定

プロジェクト ファイル

フォルダ

保持領域 削除された項目 設定 ▾

📁 現場向け

📁 プロジェクト ファイル

📁 FM 連携

📁 ファミリタイ...

📁 Shared

📁 意匠

📁 構造

📁 設備

📁 ファミリタイ...

📄 ファイルをアップロード

🔍 検索

🔍

📁

☰

<input type="checkbox"/> 名前 ^	説明	バージョン	マーク	サイズ	最終更新日	⚙️
<input type="checkbox"/> 📁 ファミリタイプ選択用	--	--	--	--	2021年8月19日 11:...	⋮
<input type="checkbox"/> 📄 サンプル意匠.rvt		V1	🔗 👑	62.3 MB	2021年8月15日 23:...	⋮
<input type="checkbox"/> 📄 サンプル構造.rvt		V1	🔗 👑	48 MB	2021年8月15日 23:...	⋮
<input type="checkbox"/> 📄 サンプル設備.rvt		V1	🔗 👑	82.7 MB	2021年8月27日 09:...	⋮

4 個の項目を表示中

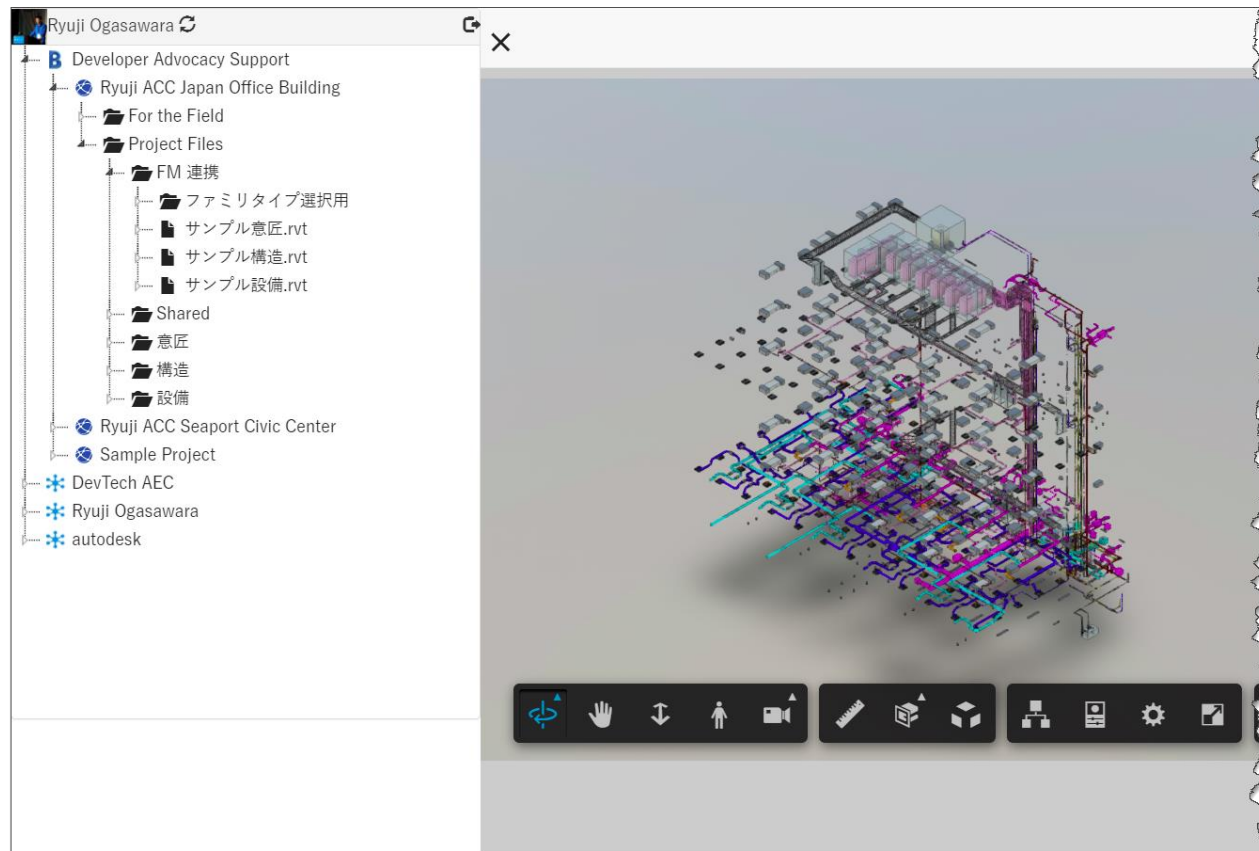
外部データベースとして SQLite を使用

Assets テーブル、Maintenance テーブル

名前	データ型	スキーマ
▼ テーブル (3)		
▼ assets		CREATE TABLE "assets" ("id" TEXT NOT NULL UNIQUE, "model_id" TEXT NOT NULL, "name" TEXT, "type_name" TEXT, "cate
id	TEXT	"id" TEXT NOT NULL UNIQUE
model_id	TEXT	"model_id" TEXT NOT NULL
name	TEXT	"name" TEXT
type_name	TEXT	"type_name" TEXT
category	TEXT	"category" TEXT
description	TEXT	"description" TEXT
location	TEXT	"location" TEXT
owner	TEXT	"owner" TEXT
serial_number	TEXT	"serial_number" TEXT
manufactured_at	TEXT	"manufactured_at" TEXT
installed_at	TEXT	"installed_at" TEXT
revit_guid	TEXT	"revit_guid" TEXT
▼ maintenance		CREATE TABLE "maintenance" ("report_id" INTEGER, "model_id" TEXT, "title" TEXT, "asset_name" TEXT, "asset_id" TEXT, "i
report_id	INTEGER	"report_id" INTEGER
model_id	TEXT	"model_id" TEXT
title	TEXT	"title" TEXT
asset_name	TEXT	"asset_name" TEXT
asset_id	TEXT	"asset_id" TEXT
revit_guid	TEXT	"revit_guid" TEXT
description	TEXT	"description" TEXT
priority	TEXT	"priority" TEXT
status	TEXT	"status" TEXT
reporter	TEXT	"reporter" TEXT
assignee	TEXT	"assignee" TEXT
created_at	TEXT	"created_at" TEXT
updated_at	TEXT	"updated_at" TEXT
> sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)
インデックス (0)		
ビュー (0)		

1. 外部データベースとの連携

Data Management API による ACC へのアクセス



ユーザ ストレージ アクセス許可と
Access Token の取得

Hub へのアクセス

Project へのアクセス

Folder へのアクセス

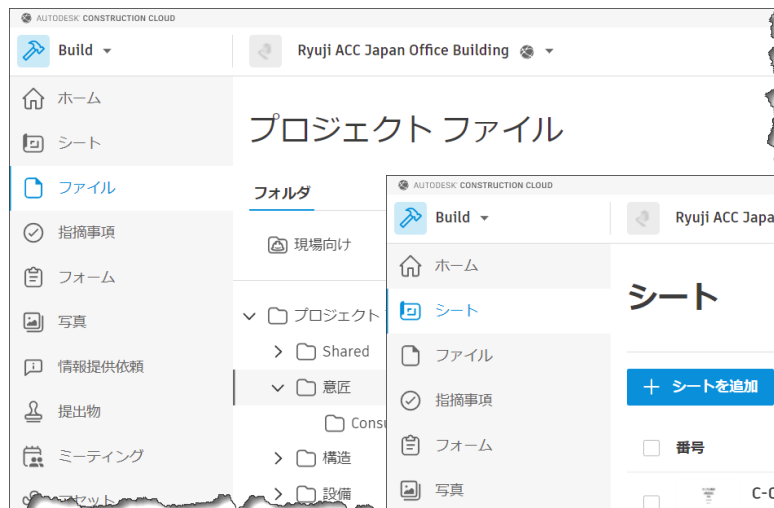
Item へのアクセス

Version へのアクセス

BIM 360 Docs と Autodesk Docs のフォルダ構成の違い

BIM 360 : 「Project Files」 フォルダと「設計図」フォルダの2つのルートフォルダ

ACC : 「ファイルツール」と「シートツール」



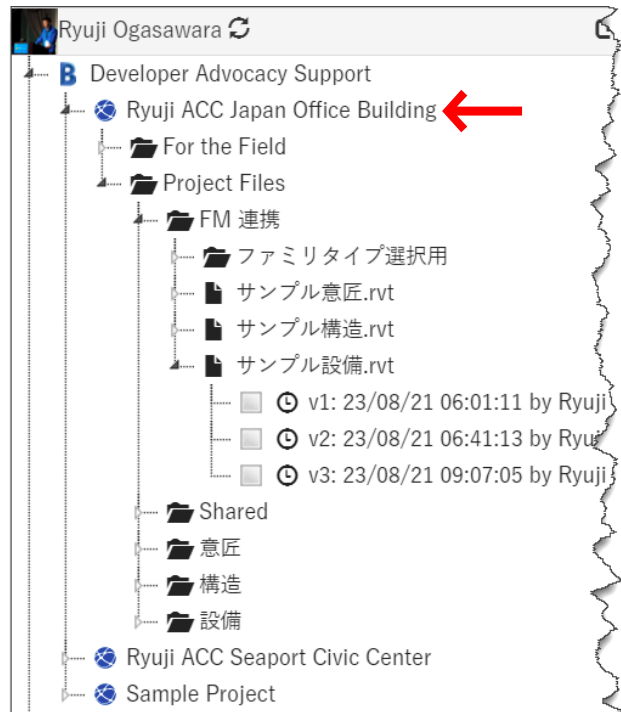
BIM 360 Docs と Autodesk Docs のフォルダ権限の違い

権限設定に変更があり、フォルダ権限のタイプが異なる。

BIM 360 Platform Pemption Type	Unified ACC Platfrom Permission Type	BIM 360 Platform API Permission Actions	Unified ACC Platform API Permission Actions
View Only	View (View files)	VIEW,COLLABORATE	VIEW,COLLABORATE
View/Download	View (View and Download files)	VIEW,DOWNLOAD,COLLABORATE	VIEW,DOWNLOAD,COLLABORATE
Upload Only	.	PUBLISH	.
.	Create (View+Download+Publish markups)	.	VIEW,DOWNLOAD,COLLABORATE,PUBLISH_MARKUP
View/Download+Upload	Create (View+Download+Publish markups+Upload)	PUBLISH,VIEW,DOWNLOAD,COLLABORATE	PUBLISH,VIEW,DOWNLOAD,COLLABORATE,PUBLISH_MARKUP
View/Download+Upload + Edit	Edit (View+Download+Publish markups+Upload+Edit)	PUBLISH,VIEW,DOWNLOAD,COLLABORATE,EDIT	PUBLISH,VIEW,DOWNLOAD,COLLABORATE,PUBLISH_MARKUP,EDIT
Full Control	Manage (Full Administrative Control)	PUBLISH,VIEW,DOWNLOAD,COLLABORATE,EDIT,CONTROL	PUBLISH,VIEW,DOWNLOAD,COLLABORATE,PUBLISH_MARKUP,EDIT,CONTROL

プロジェクトが BIM 360 か ACC か判別する方法

Data Management API のプロジェクトの “projectType” 属性を参照



GET `hubs/:hub_id/projects`

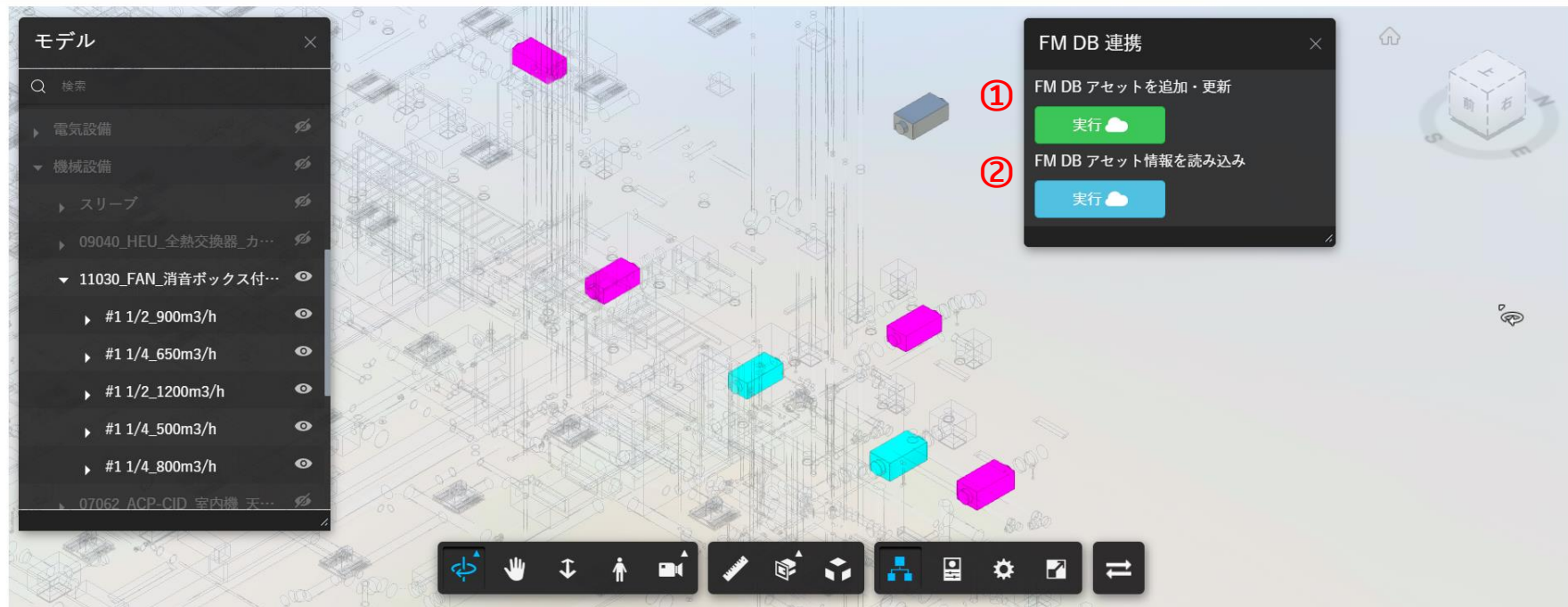
```
"data": [
  {
    "type": "projects",
    "id": "f6a1e3b5-abaa-4b01-b33a-5d55f36ba047",
    "attributes": {
      "name": "My First Project",
      "extension": {
        "type": "projects:autodesk.core:Project",
        "version": "1.0",
        "data": {
          "projectType": "ACC"
        }
      }
    }
  }
]
```

data.attributes.extension.data.projectType
ACC | BIM360

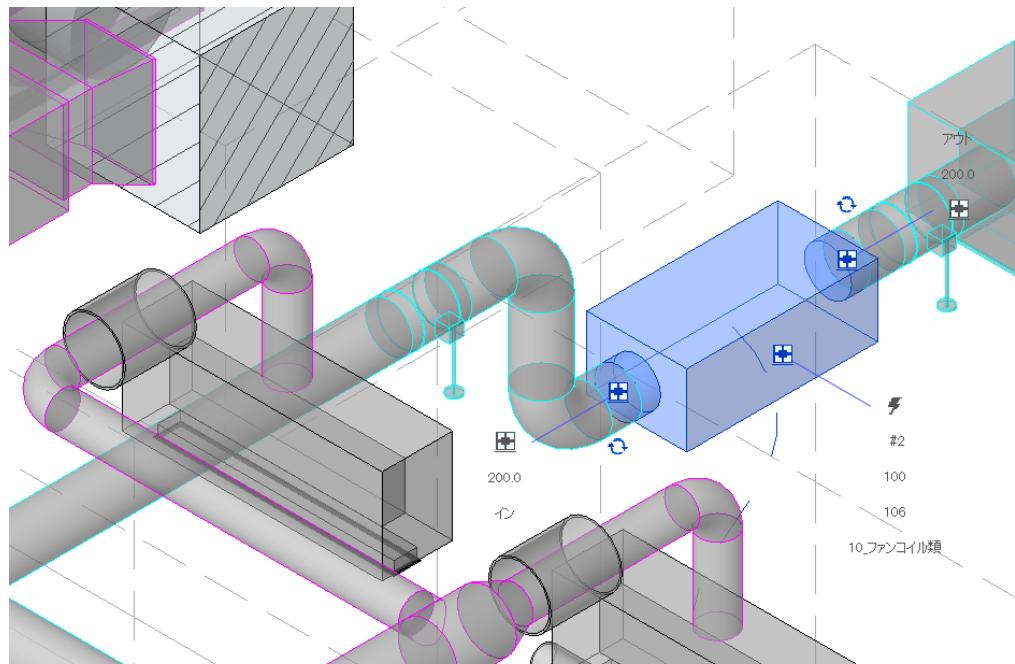
Revit モデルの要素と外部 DBを紐づける

選択されている要素の属性情報（一意の識別子）を取得して、外部 DB にアセットとして保存。

外部 DB からアセット情報を読み込んで、Revit モデルの要素にアセット ID を設定。



Revit 要素の識別子



ElementId は、**単一のプロジェクト内での一意の ID** であり、プロジェクト間では一意ではありません。

Uniqueld は、GUID と同様の機能を持ち、**個々のプロジェクト間でも一意**となります。

Field	Value
--- Element ---	
--- Properties ---	
AssemblyInstanceId	< null >
BoundingBox	< BoundingBoxXYZ >
Category	< Category 機械設備 >
CreatedPhaseId	< 新しい建設 86961 >
DemolishedPhaseId	< null >
DesignOption	< null >
Document	< Document >
Geometry	< Geometry.Element >
GroupId	< null >
Id	17299672
IsTransient	False
IsValidObject	True
LevelId	< 1F 17293146 >
Location	< LocationPoint >
Name	#1 1/4_650m3/h
OwnerViewId	< null >
Parameters	< ParameterSet >
ParametersMap	< ParameterMap >
Pinned	False
Uniqueld	706c1fe4-e098-4258-8c15-25674aab130c-0107f8d8
VersionGuid	b62152c4-271e-4a4d-a16a-0e38023f311d
ViewSpecific	False
WorksetId	< WorksetId >

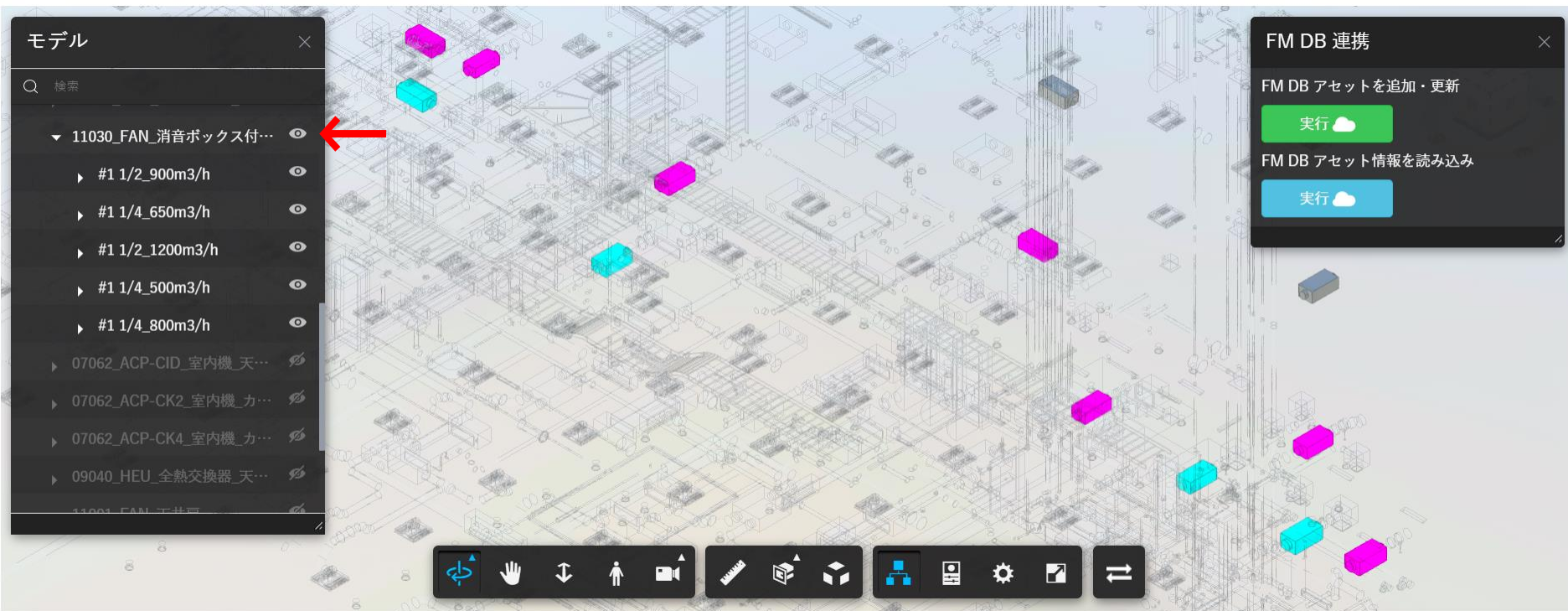
Forge Viewer の識別子

```
{  
  "objectid": 1706, = dbId  
  "name": "11030_FAN_消音ボックス付送風機 [17299672]",  
  "externalId": "706c1fe4-e098-4258-8c15-25674aab130c-0107f8d8", = Revit の Element.UniqueId  
  "properties": {  
    "拘束": {  
      "風量": "400.000 CMH",  
      "機外静圧": "118.000 Pa",  
      "基準レベル": "1F",  
      "基準レベルからの高さ": "3327.500 mm",  
      "ホスト": "レベル : 1F",  
      "ホストからのオフセット": "3327.500 mm",  
      "既定の高さ": "0.000 mm",  
      "番手": "#1 1/4"  
    },  
    "文字": {  
      "☑給気/☐排気": "Yes",  
      "特殊仕様": "",  
    },  
  },  
}
```

- objectid / dbId
Forge Viewer が要素を識別するための ID。
特定の要素のプロパティにアクセスする際などにも利用。
- externalId
Revit の UniqueId に保持されている GUID と同じ値。

Forge Viewer のモデルブラウザで選択されているノードの取得

```
var isolatedNode = viewer.impl.visibilityManager.getIsolatedNodes(viewer.impl.model);
```



多数の要素の externalId を取得する際の注意点

```
getChildLeafComponents(viewer, isolatedNode[0], function (dbIds) {  
  
    var promises = [];  
  
    for (var i = 0; i < dbIds.length; i++) {  
  
        var dbIdData = {};  
        dbIdData.Index = i;  
        dbIdData.Id = dbIds[i];  
  
        promises.push(getPropertiesAsync(dbIdData));  
  
    }  
};
```

```
Promise.all(promises)  
    .then(function (assets) {  
  
        var formData = new FormData();  
        formData.append('data', JSON.stringify({  
            assets: assets,  
            browserConnectionId: connectionId  
        }));  
  
        $.ajax({  
            url: 'api/forge/revit_fm/upsert_assets',  
            data: formData,  
            processData: false,  
            contentType: false,  
            type: 'POST',  
            success: function (res) {  
                console.log('Upserting assets started');  
                $upsertButton.button('reset');  
            }  
        });  
    });  
});
```

親ノードから再帰処理で子ノードの dbId を取得。

Viewer.getProperties() メソッドは、コールバックメソッドにプロパティ情報を返します。

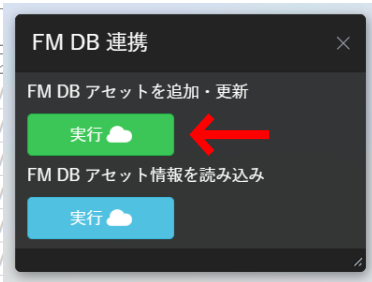
選択ノード配下の全ての子ノードのプロパティを繰り返し処理で取得する場合は、非同期関数 Promise の利用を推奨いたします。

```
function getChildLeafComponents(viewer, dbId, callback) {  
    var cbCount = 0; // count pending callbacks  
    var components = []; // store the results  
    var tree; // the instance tree  
  
    function getLeafComponentsRec(parent) {  
        cbCount++;  
        if (tree.getChildCount(parent) != 0) {  
            tree.enumNodeChildren(parent, function (children) {  
                getLeafComponentsRec(children);  
            }, false);  
        } else {  
            components.push(parent);  
        }  
        if (--cbCount == 0) callback(components);  
    }  
  
    viewer.getObjectTree(function (objectTree) {  
        tree = objectTree;  
        var allLeafComponents = getLeafComponentsRec(dbId);  
    });  
}
```

```
function getPropertiesAsync(dbIdData) {  
  
    return new Promise(function (resolve, reject) {  
  
        viewer.getProperties(dbIdData.Id, function (result) {  
  
            var objectProp = result;  
  
            var urn = viewer.impl.model.myData.urn;  
  
            var asset = {};  
  
            asset.name = objectProp.name;  
            asset.revit_guid = objectProp.externalId;  
  
            resolve(asset);  
        }, function(error) {  
  
            reject(error);  
        });  
    });  
}
```


Forge Viewer → externalId 取得 → SQLite DB レコード追加・更新

	id	model_id	revit_guid	name	type_name	category	
	フィルター	フィルター	フィルター	フィルター	フィルター	フィルター	
1	AU_0	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f7a5	11030_FAN_消音ボックス付送風機 [17299365]	#1 1/2_900m3/h	Revit 機械設備	
2	AU_1	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f7a6	11030_FAN_消音ボックス付送風機 [17299366]	#1 1/2_900m3/h	Revit 機械設備	
3	AU_2	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8d3	11030_FAN_消音ボックス付送風機 [17299667]	#1 1/4_650m3/h	Revit 機械設備	
4	AU_3	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8d4	11030_FAN_消音ボックス付送風機 [17299668]	#1 1/4_650m3/h	Revit 機械設備	
5	AU_4	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8d7	11030_FAN_消音ボックス付送風機 [17299671]	#1 1/4_650m3/h	Revit 機械設備	
6	AU_5	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8d8	11030_FAN_消音ボックス付送風機 [17299672]	#1 1/4_650m3/h	Revit 機械設備	
7	AU_6	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8dc	11030_FAN_消音ボックス付送風機 [17299676]	#1 1/4_650m3/h	Revit 機械設備	NULL
8	AU_7	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8e3	11030_FAN_消音ボックス付送風機 [17299683]	#1 1/4_650m3/h	Revit 機械設備	NULL
9	AU_8	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8ea	11030_FAN_消音ボックス付送風機 [17299690]	#1 1/4_650m3/h	Revit 機械設備	NULL
10	AU_9	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8f1	11030_FAN_消音ボックス付送風機 [17299697]	#1 1/4_650m3/h	Revit 機械設備	NULL
11	AU_10	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8f8	11030_FAN_消音ボックス付送風機 [17299704]	#1 1/4_650m3/h	Revit 機械設備	NULL
12	AU_11	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8ff	11030_FAN_消音ボックス付送風機 [17299711]	#1 1/4_650m3/h	Revit 機械設備	NULL
13	AU_12	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8d5	11030_FAN_消音ボックス付送風機 [17299669]	#1 1/2_1200m3/h	Revit 機械設備	NULL
14	AU_13	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8d6	11030_FAN_消音ボックス付送風機 [17299670]	#1 1/2_1200m3/h	Revit 機械設備	NULL
15	AU_14	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8d9	11030_FAN_消音ボックス付送風機 [17299673]	#1 1/4_500m3/h	Revit 機械設備	NULL
16	AU_15	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8e0	11030_FAN_消音ボックス付送風機 [17299680]	#1 1/4_800m3/h	Revit 機械設備	NULL
17	AU_16	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8e7	11030_FAN_消音ボックス付送風機 [17299687]	#1 1/4_800m3/h	Revit 機械設備	NULL
18	AU_17	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8ee	11030_FAN_消音ボックス付送風機 [17299694]	#1 1/4_800m3/h	Revit 機械設備	NULL
19	AU_18	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8f5	11030_FAN_消音ボックス付送風機 [17299701]	#1 1/4_800m3/h	Revit 機械設備	NULL
20	AU_19	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f8fc	11030_FAN_消音ボックス付送風機 [17299708]	#1 1/4_800m3/h	Revit 機械設備	NULL
21	AU_20	https://...	706c1fe4-e098-4258-8c15-25674aab130c-0107f903	11030_FAN_消音ボックス付送風機 [17299715]	#1 1/4_800m3/h	Revit 機械設備	NULL



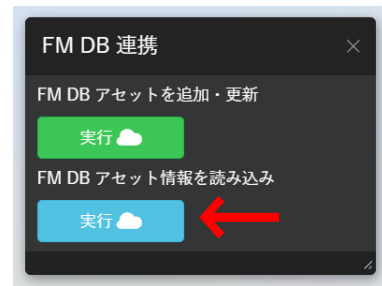
SQLite DB → Design Automation for Revit → Revit モデル編集

externalId を DA4R のアドインにパラメータとして渡す方法

Activity 定義

```
"id": "{{activityName}}",
"commandLine": [ "${engine.path}\\\\\\\\revitcoreconsole.exe /i \\\"${args[inputRvtFile].path}\\\" /al \\\"${appbundles[{
  {appbundleName}}.path}\\\" /l JPN" ],
"parameters": {
  "inputRvtFile": {
    "zip": false,
    "ondemand": false,
    "verb": "get",
    "description": "Input Revit model",
    "required": true
  },
  "inputJsonFile": {
    "zip": false,
    "ondemand": false,
    "verb": "get",
    "description": "Input param json",
    "required": true,
    "localName": "params.json"
  },
  "resultRvtFile": {
    "zip": false,
```

WorkItem のリクエスト時に実際のJSON データを送信すると、"localName"で指定されたファイル名で .json ファイルが作成され、Design Automation の一時作業領域に保存されます。



SQLite DB → Design Automation for Revit → Revit モデル編集

DA4R のアドインでパラメータを受け取る方法

一時作業領域に保存された JSON ファイルを読み込みます。

```
public void SyncFMAssets(DesignAutomationData data)
{
    Document doc = data.RevitDoc;
    Application app = data.RevitApp;

    List<AssetElement> assetElems = JsonConvert.DeserializeObject<List<AssetElement>>(File.ReadAllText("params.json"));

    foreach (AssetElement assetElem in assetElems)
    {
        Console.WriteLine("Asset GUID: " + assetElem.AssetRevitGuid.ToString());

        Element elem = doc.GetElement(assetElem.AssetRevitGuid);
    }
}
```

1. externalId (Uniqueid) からファミリインスタンスを取得し、パラメータ「AssetId」が存在するかチェック。
2. パラメータが存在しない場合、ファミリエディタでファミリドキュメントを開き、タイプパラメータ「AssetId」を追加して、Revit プロジェクトに再ロード。
3. ファミリインスタンスに「AssetId」パラメータが割り当てられているため、外部DB から取得した値を設定。
4. Revit プロジェクトを保存して WorkItem の処理を完了し、ACC - Autodesk Docs の指定した場所にアップロード。

Revit モデルのバージョン更新

WorkItem の処理が完了後、Data Management API でバージョンの更新を実行します。
成功すると、ACC 上で V2 と表示され、SVF 変換処理が自動的に始まります。

Data Management API v2

> Developer's Guide

▼ Step-by-Step Tutorials

Download a File

[Upload a File](#)

Create an Attachment

Create an App-Managed Bucket and Upload a File

Publish a Cloud Workshared Revit Model

Delete and Restore Files in BIM 360 Document Management

> API Reference

> Change History

Step 6: Update the version of a file

An uploaded resource can have multiple versions. To create a new version of an existing resource, Step 3 and 4 in this tutorial will need to be repeated. Make note of the new object ID in Step 3. Get the item ID of the resource from Step 5 or call the [GET projects/:project_id/folders/:folder_id/contents](#) endpoint.

The [POST projects/:project_id/versions](#) endpoint creates a new version of the uploaded file.

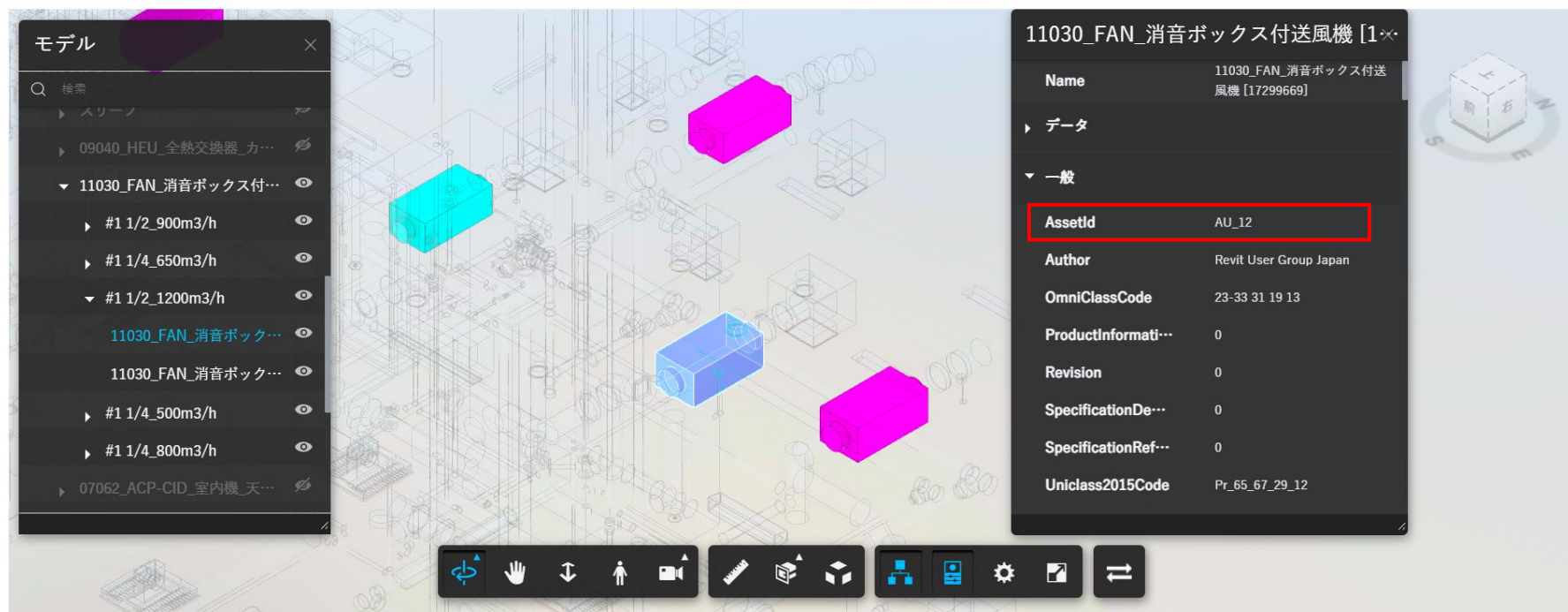
ファイルをアップロード

検索

<input type="checkbox"/>	名前 ^	説明	バージ...	マーク...	サイズ
<input type="checkbox"/>	📁 ファミリタイプ選択用	--	--	--	--
<input type="checkbox"/>	🔄 サンプル設備.rvt		V2	🔗 👑	82.7 MB
<input type="checkbox"/>	📐 サンプル意匠.rvt		V1	🔗 👑	62.3 MB
<input type="checkbox"/>	📐 サンプル構造.rvt		V1	🔗 👑	48 MB

パラメータ追加と値設定の確認

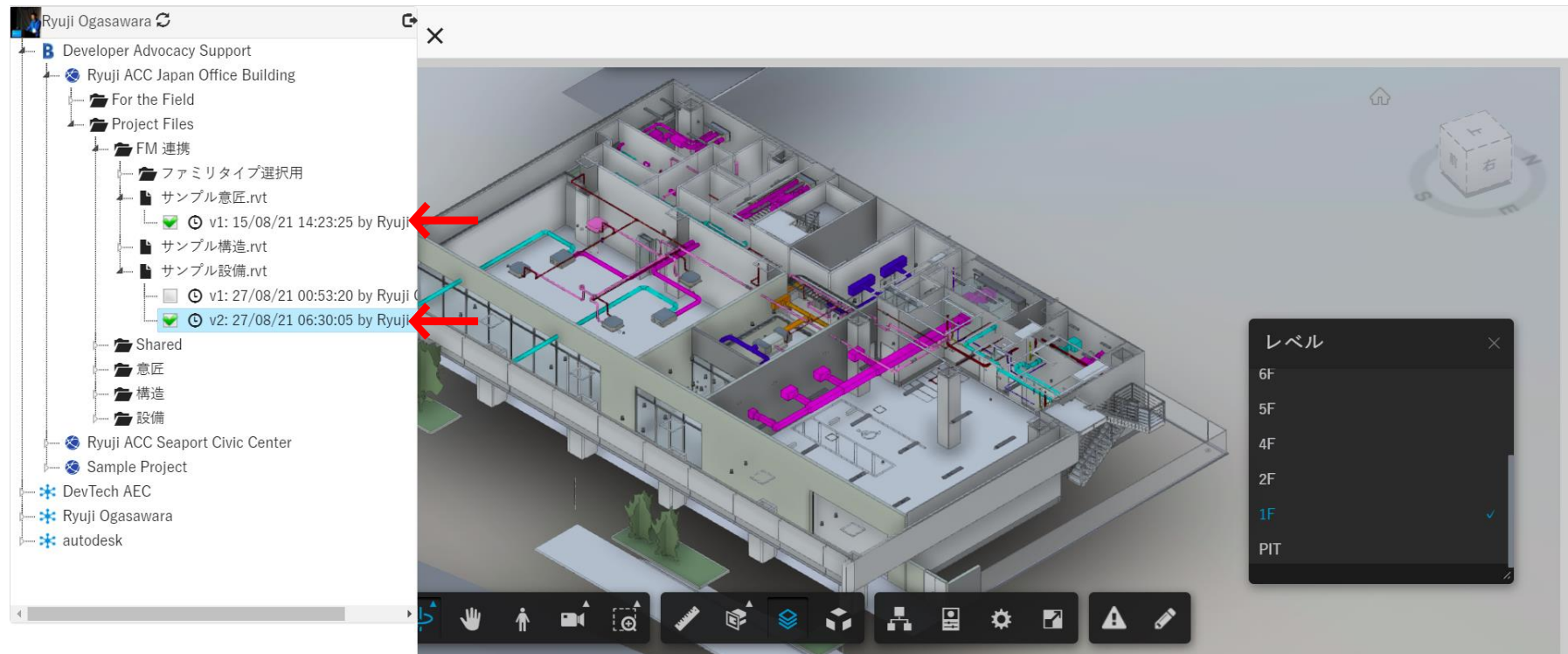
Forge Viewer で指定した要素に対応する AssetId を外部 DB から取得し、Revit モデルのファミリインスタンスに「AssetId」パラメータを追加・値を設定できました。



2. 問題の報告と履歴の閲覧

Forge Viewer – 複数モデルの統合表示

複数モデルを重ねて統合表示することで、問題の発生している要素がみつけやすくなります。



Forge Viewer – AggregatedView クラス

機能説明

- 複数ドキュメントの BubbleNode (2D または 3D の Viewable コンテナ) を重ねてロード。
- それぞれのビューを切り替えて表示、またはモデルを動的にトグル表示 (ON/OFF) 可能。
- diffOptions の設定でビューの差分表示が可能。
- 次の Extension が自動的にロードされます。
 - Autodesk.CrossFadeEffects
 - Autodesk.AEC.LevelsExtension
 - Autodesk.ModelStructure
 - Autodesk.AEC.HyperlinkExtension
 - Autodesk.AEC.Minimap3DExtension
 - Autodesk.AEC.CanvasBookmarkExtension
 - Autodesk.AEC.DropMeExtension
 - 一人称視点モードも有効



Forge Viewer – AggregatedView クラス

サンプルコード

```
var view = new Autodesk.Viewing.AggregatedView();

function loadManifest(documentId) {
  return new Promise((resolve, reject) => {
    const onDocumentLoadSuccess = (doc) => {
      doc.downloadAecModelData() => resolve(doc);
    };
    Autodesk.Viewing.Document.load(documentId, onDocumentLoadSuccess, reject);
  });
}

Autodesk.Viewing.Initializer(options, function () {

  const viewerDiv = document.getElementById('forgeViewer');

  view.init(viewerDiv);

  const tasks = [];

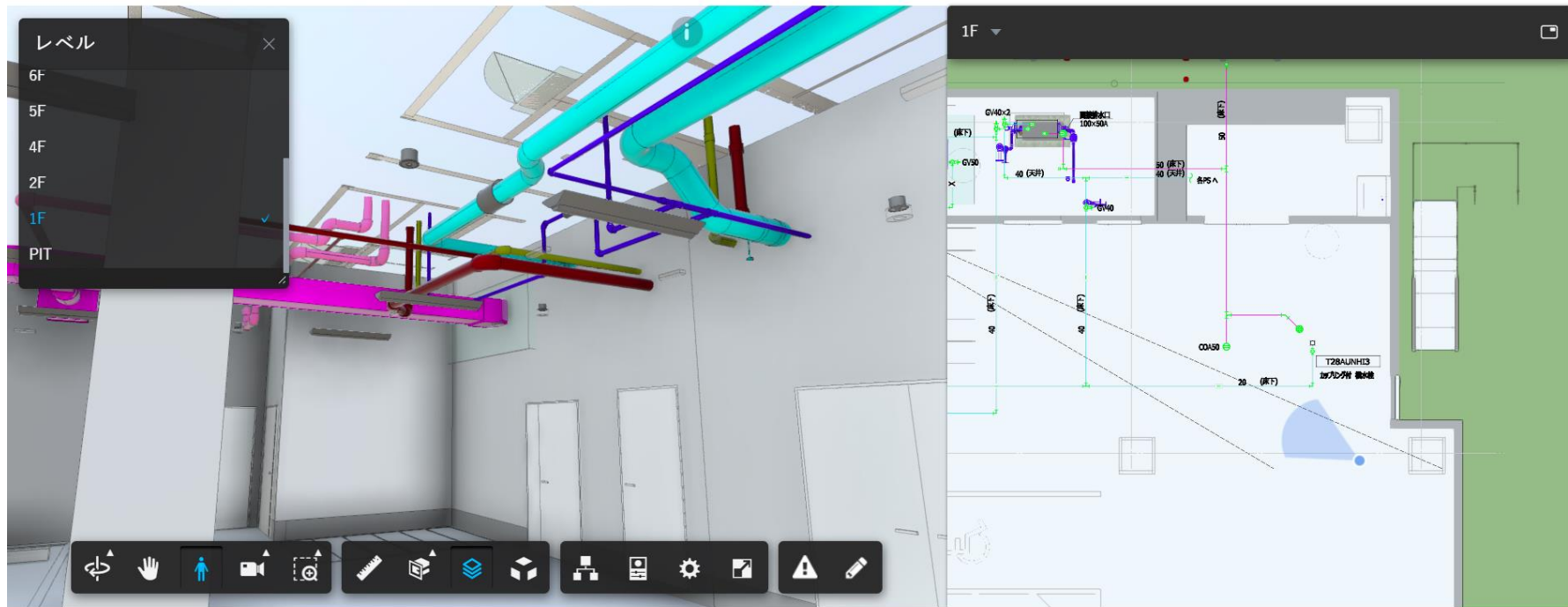
  treeNodes.forEach(treeNode => tasks.push(loadManifest('urn:' + treeNode.id.split('/')[0])));

  Promise.all(tasks)
    .then(docs => Promise.resolve(docs.map(doc => {
      const bubbles = doc.getRoot().search({ type: 'geometry', role: '3d' });
      const bubble = bubbles[0];
      if (!bubble) return null;

      return bubble;
    })))
    .then(bubbles => view.setNodes(bubbles));
});
```

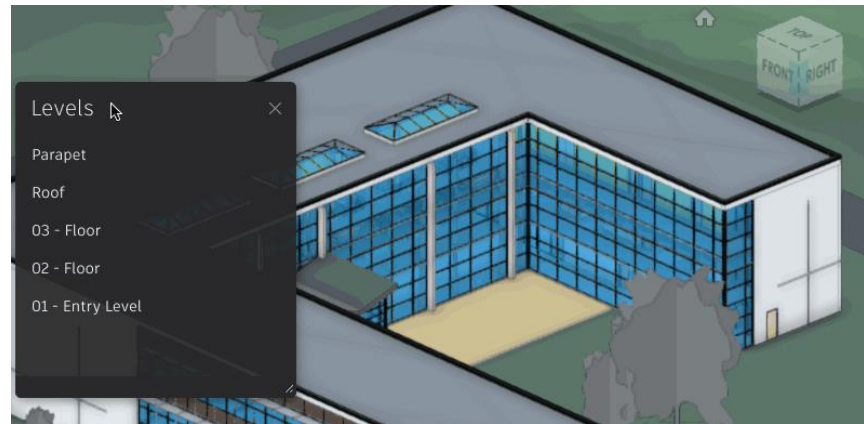
Forge Viewer – 3D View + 2D Minimap + Levels Extension

Levels Extension で 3D モデルを指定のレベルのみ表示し、Minimap3D Extension で、2D 図面ビューと連動して視点の移動ができます。



Forge Viewer – 3D View + 2D Minimap + Levels Extension

1. SVF ファイルを Forge Viewer にロードします。
2. AECModelData.json を Forge Viewer にロードします。このファイルがレベルの情報を保持しています。
 - IFC ファイルの場合は、ifcLevelsEnabled: true を設定すると、レベルの情報が抽出されます。
 - AECModelData は、他に通芯、フェーズ、ビューポートなどの情報を保持しています。
3. “Autodesk.AEC.LevelsExtension” と “Autodesk.AEC.Minimap3DExtension” をロードします。
4. 一人称視点モードを起動します。



※ AggregatedView クラスを使用する場合は、自動的にこの機能を利用できます。

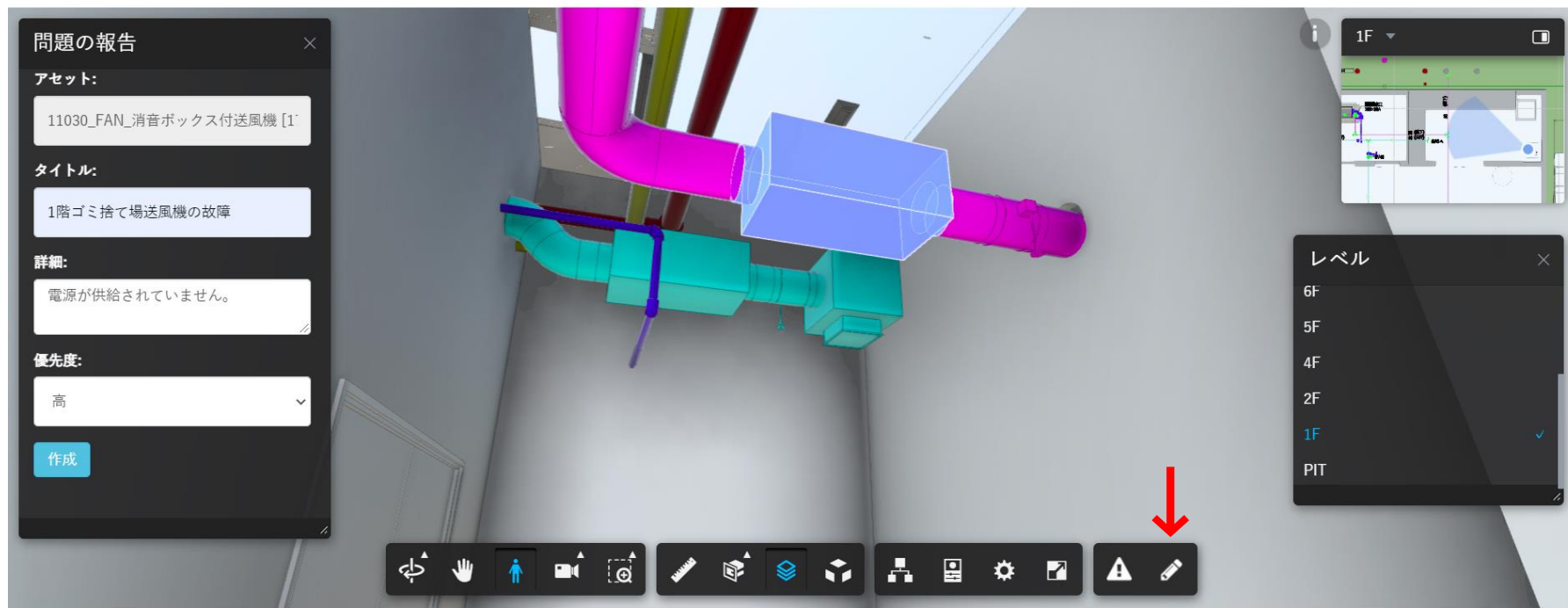
Forge Viewer – 3D View + 2D Minimap + Levels Extension

サンプルコード

```
Autodesk.Viewing.Initializer(options, () => {  
  
    const div = document.getElementById('forgeViewer');  
  
    const config = { extensions:[  
        "Autodesk.AEC.LevelsExtension",  
        "Autodesk.AEC.Minimap3DExtension"  
    ]};  
  
    viewer = new Autodesk.Viewing.Private.GuiViewer3D(div, config);  
    viewer.start();  
    Autodesk.Viewing.Document.load(`urn:${urn}`, (doc) => {  
        var viewables = doc.getRoot().getDefaultGeometry();  
        doc.downloadAecModelData();  
        viewer.loadDocumentNode(doc, viewables).then( onLoadFinished );  
  
        viewer.addEventListener( Autodesk.Viewing.TEXTURES_LOADED_EVENT, ()=>{  
            viewer.getExtension("Autodesk.BimWalk").activate();  
        });  
    });  
});
```

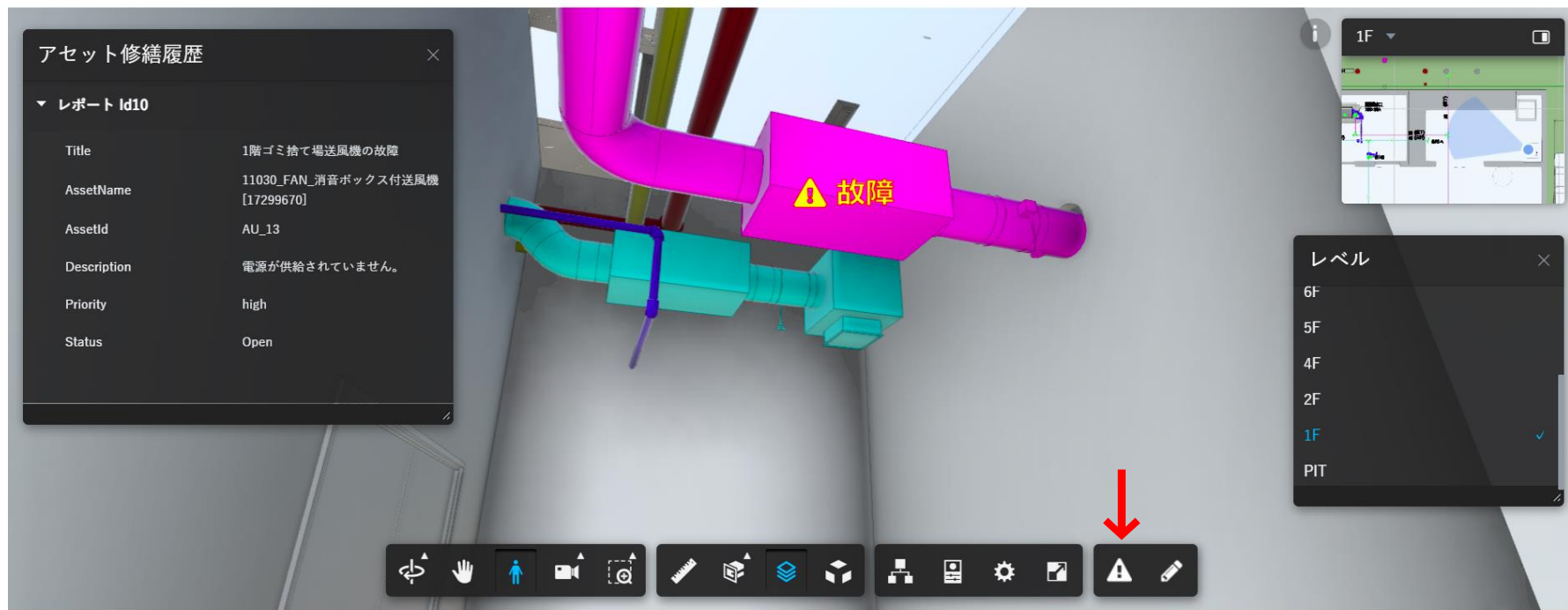
問題の報告

ACC は、現時点では指摘事項の API (Issues API) が未サポートのため、別途、SQLite の Maintenance テーブルで管理。



Forge Viewer - カスタム PropertyPanel

外部 DB からデータを取得し、Forge Viewer のカスタムプロパティパネルに表示します。



Forge Viewer - カスタム PropertyPanel

サンプルコード

PropertyPanel. addProperty(name,value,category,options) でプロパティ追加

```
function LoadRecordPanel(viewer, container, id, title, options) {
  this.viewer = viewer;
  Autodesk.Viewing.UI.PropertyPanel.call(this, container, id, title, options);
}
LoadRecordPanel.prototype = Object.create(Autodesk.Viewing.UI.PropertyPanel.prototype);
LoadRecordPanel.prototype.constructor = LoadRecordPanel;

AssetMaintenanceExtension.prototype.showRecords = function (maintenanceRecords) {
  var _this = this;

  console.log(maintenanceRecords);

  maintenanceRecords.forEach(function (record) {
    if (_this.loadRecordPanel) {
      _this.loadRecordPanel.addProperty('Title', record.title, "レポート Id" + record.reportId);
      _this.loadRecordPanel.addProperty('AssetName', record.assetName, "レポート Id" + record.reportId);
      _this.loadRecordPanel.addProperty('AssetId', record.assetId, "レポート Id" + record.reportId);
      _this.loadRecordPanel.addProperty('Description', record.description, "レポート Id" + record.reportId);
      _this.loadRecordPanel.addProperty('Priority', record.priority, "レポート Id" + record.reportId);
      _this.loadRecordPanel.addProperty('Status', record.status, "レポート Id" + record.reportId);
    }
  })
}
```

Forge Viewer – マークアップ

dbId に紐づくオブジェクトの 3D ジオメトリの中心にマークアップ（Label 要素）を表示



Forge Viewer – マークアップ

仕組み

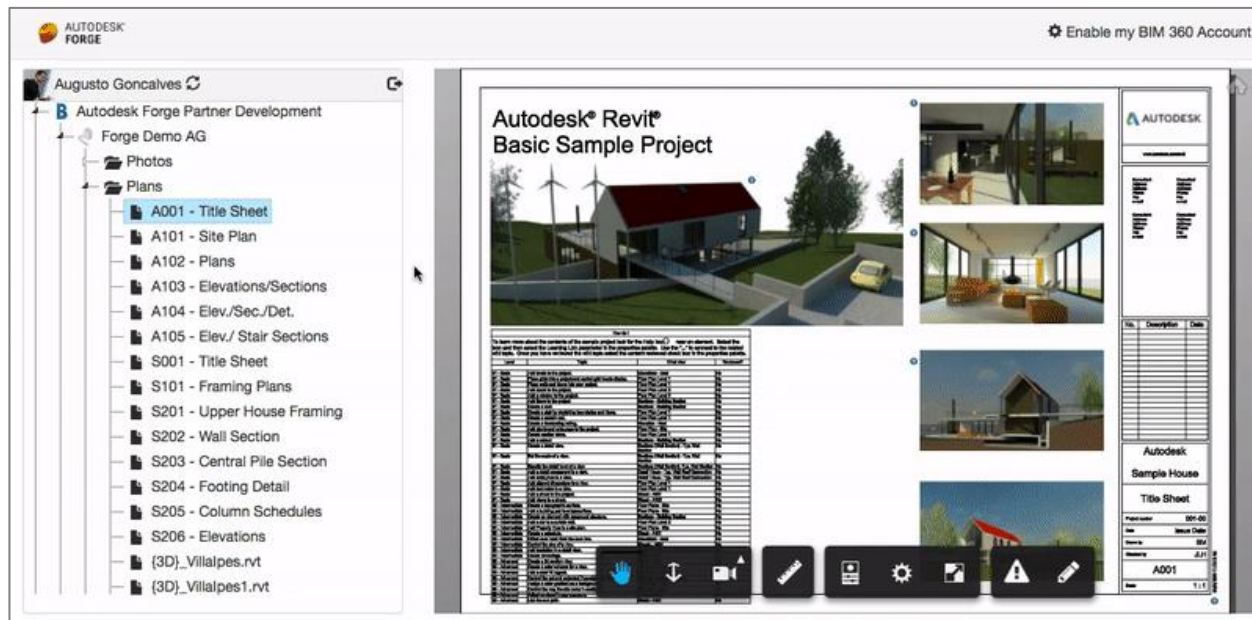
1. 指定した dbid のオブジェクトを構成するフラグメントを取得。
 - フラグメント : Transform マトリクス、マテリアル、バウンディングボックスなどのプロパティを保持する個々のメッシュ
2. 各フラグメントのバウンディングボックスを結合して中心座標（ワールド座標）を取得。
 - `THREE.Box3.union()`
3. ワールド座標をクライアントのスクリーン座標に変換。
 - `Viewer3D.worldToClient()`
4. Label 要素の位置を CSS で設定。
5. オービットやパン・ズームなどのカメラ更新イベントで Label の位置も更新。
 - `Autodesk.Viewing.CAMERA_CHANGE_EVENT`

※ アイコンは Font Awesome を使用。

BIM 360 Issues API と Pushpin Extension サンプル

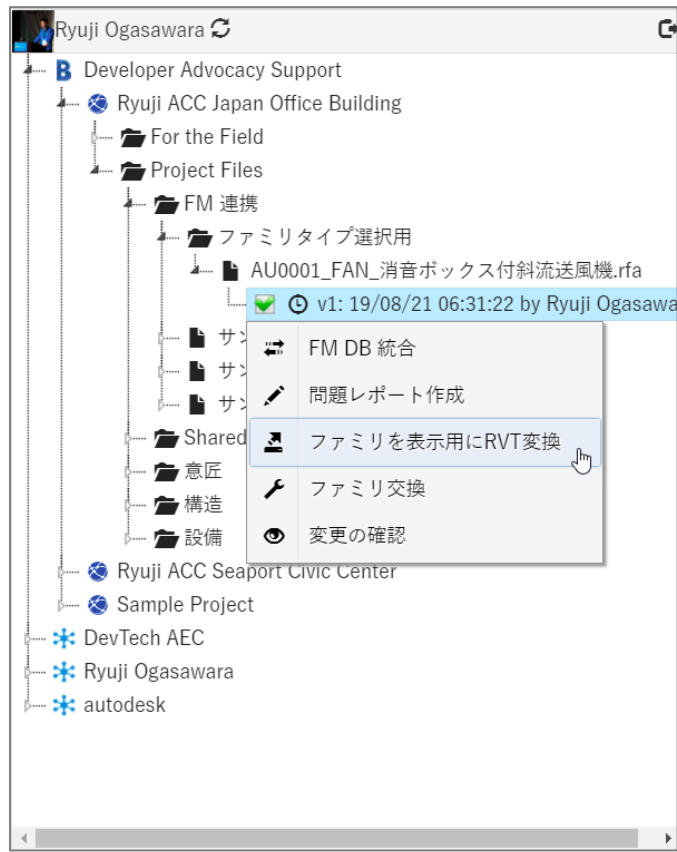
ドキュメントの指摘事項を Forge Viewer に表示

Autodesk Construction Cloud – Issues API のリリースまで今しばらくお待ちください。



3. Revit ファミリを Forge Viewer 表示用に プロジェクトに変換する

Model Derivative API / Forge Viewer は RFA 未サポート



Model Derivative API, Forge Viewer は Revit ファミリ (RFA ファイル) は未サポート。



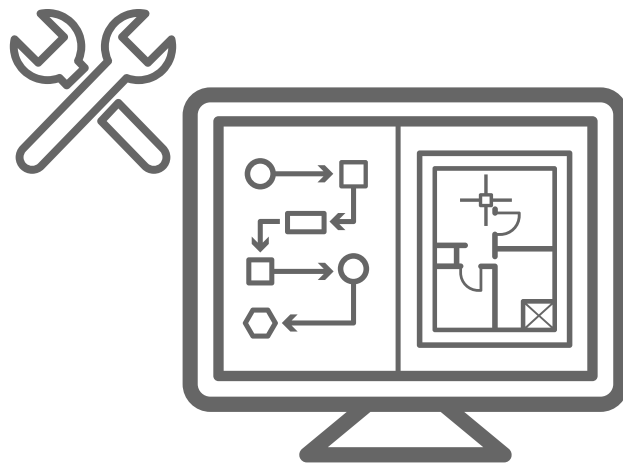
Forge Viewer で表示するためには、一旦、ファミリをプロジェクトに配置する必要があります。

RFA ファイル → Design Automation for Revit → RVT ファイル

Revit 上での操作手順とルールが決まっている場合、アドインを作成して自動化できます。

※ただし、ユーザインタラクションが必要な操作は DA4R では実行できません。

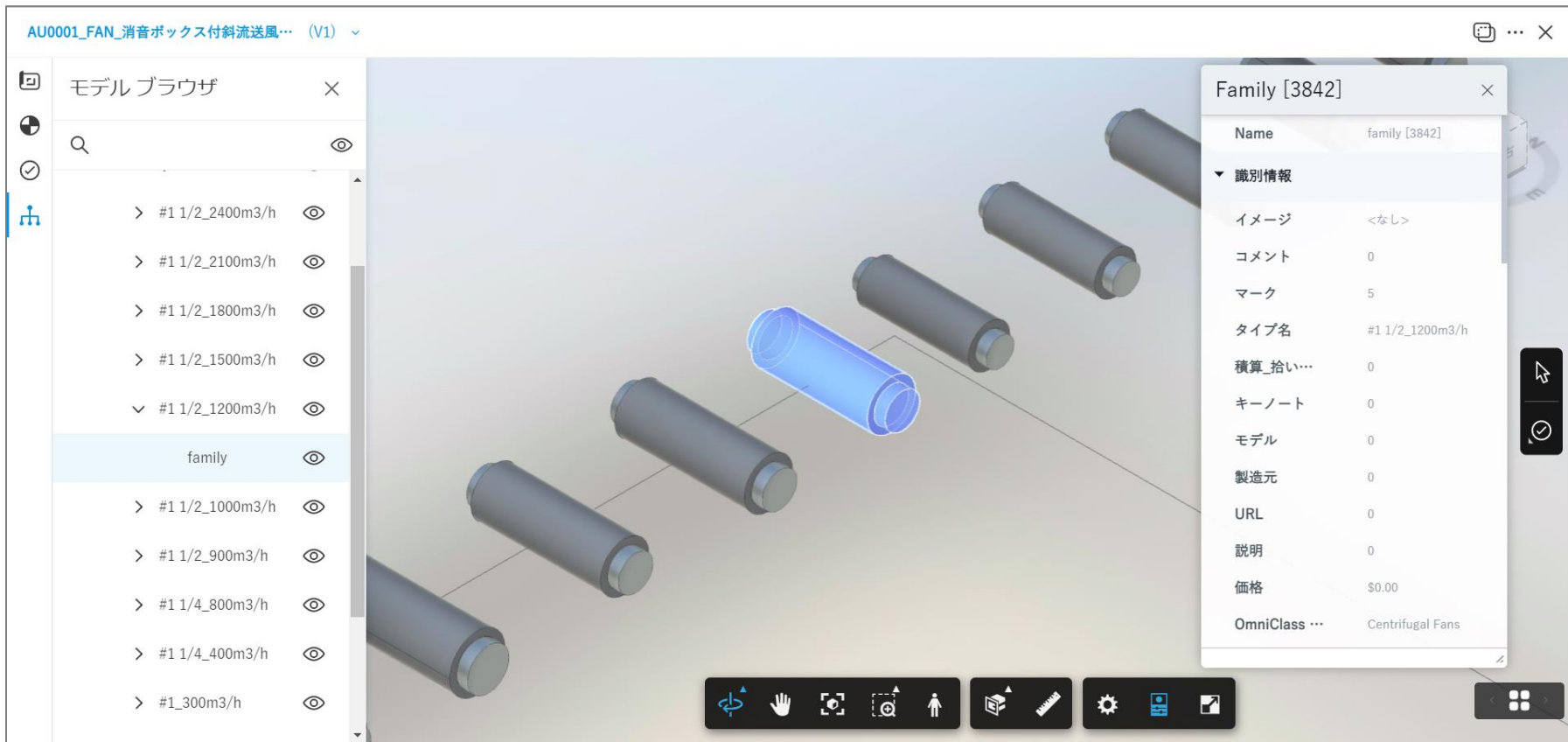
1. 新規プロジェクトを作成。
2. ファミリをプロジェクトにロード。
3. 平面図ビューを開く。
4. ファミリタイプを選択。
5. ファミリインスタンスを平面図ビューに配置。
6. ファミリタイプの数だけ、4-5を繰り返す。
7. 3D ビューを作成。
8. ビューの詳細レベルを変更。
9. プロジェクトを保存する。



Webhooks API と組み合わせれば、指定のフォルダに RFA ファイルがアップロードされたら自動的に DA4R を起動して RVT ファイルを生成するといった仕組みも作成可能です。

出力した RVT ファイル

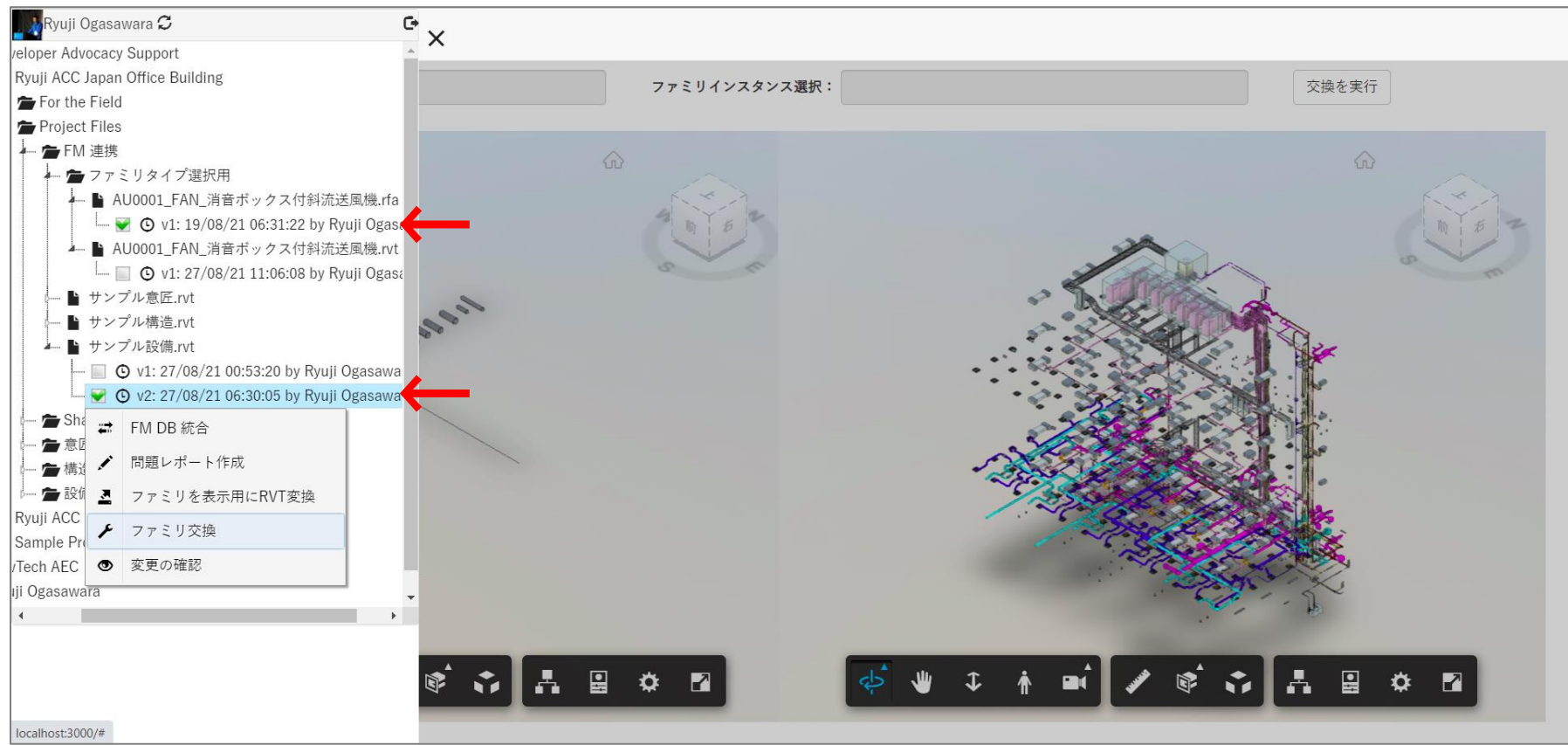
タイプ毎に1つのインスタンスを位置をずらして配置しました。



4. プロジェクトのファミリインスタンスを別のファミリと交換する

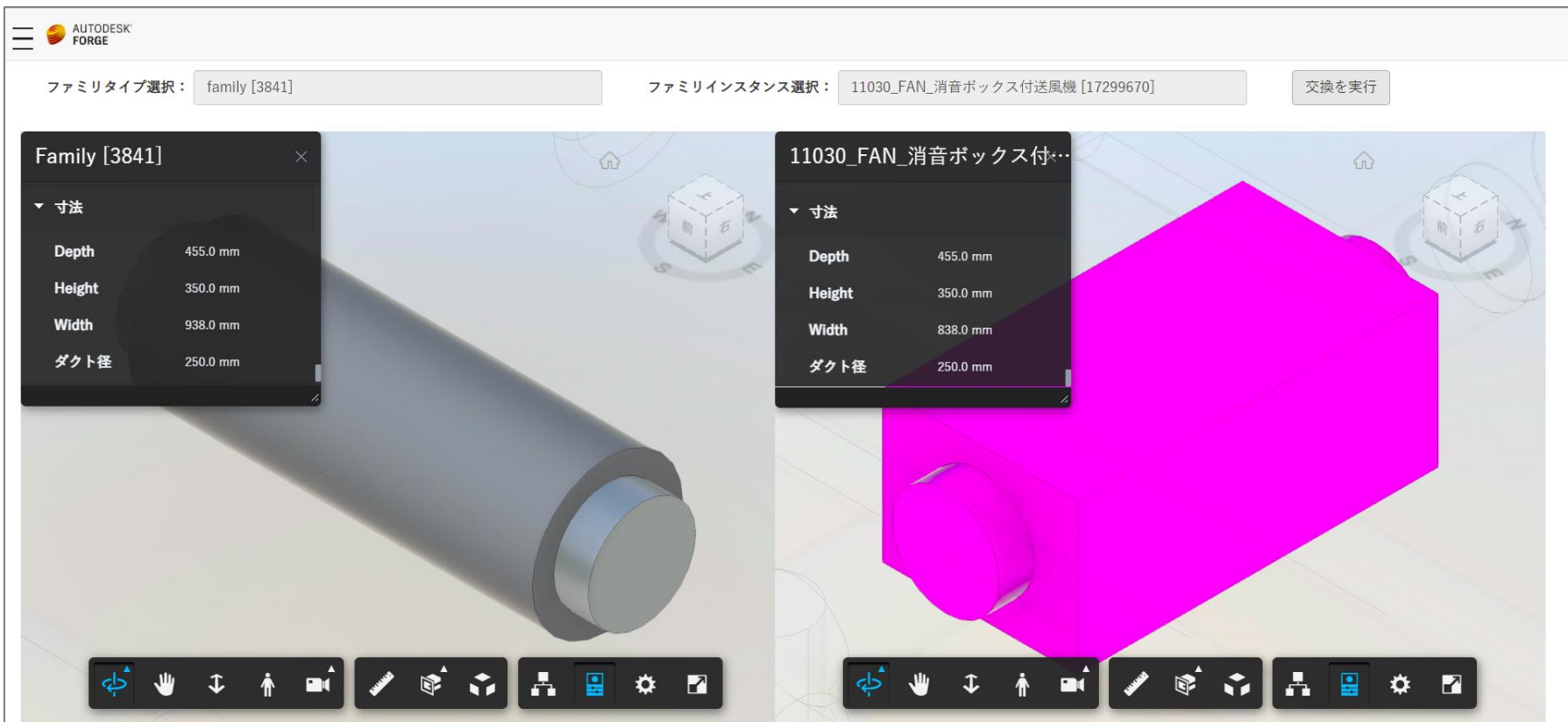
複数の Forge Viewer を同時に表示する

左の Viewer にファミリタイプ選択用プロジェクトを表示しています。



ファミリタイプを指定して交換を実行

Autodesk.Viewing.SELECTION_CHANGED_EVENT で DOM 要素を更新



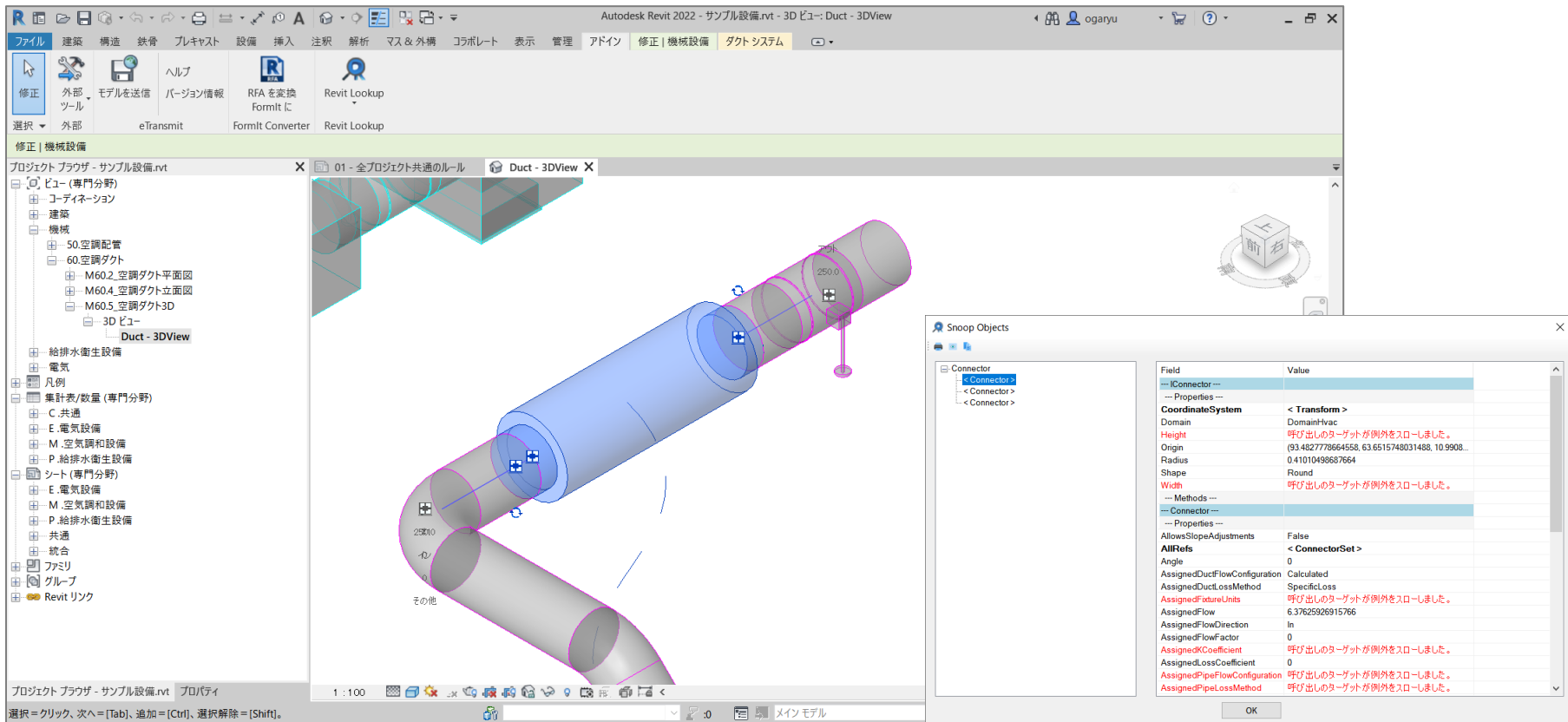
RVT/RFA/JSON → Design Automation for Revit → RVT ファイル

アドインの処理フロー

1. Revit プロジェクトを開く。
2. JSON データをデシリアライズ。
 - 交換対象のファミリインスタンスの UniqueId
 - 交換するファミリのファミリタイプ名
3. ファミリをプロジェクトにロード。
4. 既存のファミリインスタンスの位置情報、Transform、MEP コネクタの接続状態を一時保存。
5. ファミリタイプを取得しアクティベート。
6. 既存のファミリインスタンスを削除。
7. 新規ファミリインスタンスを作成。
8. Transform の適用。
9. MEP コネクタの接続状態を復元。
10. プロジェクトを保存する。

出力した RVT ファイル

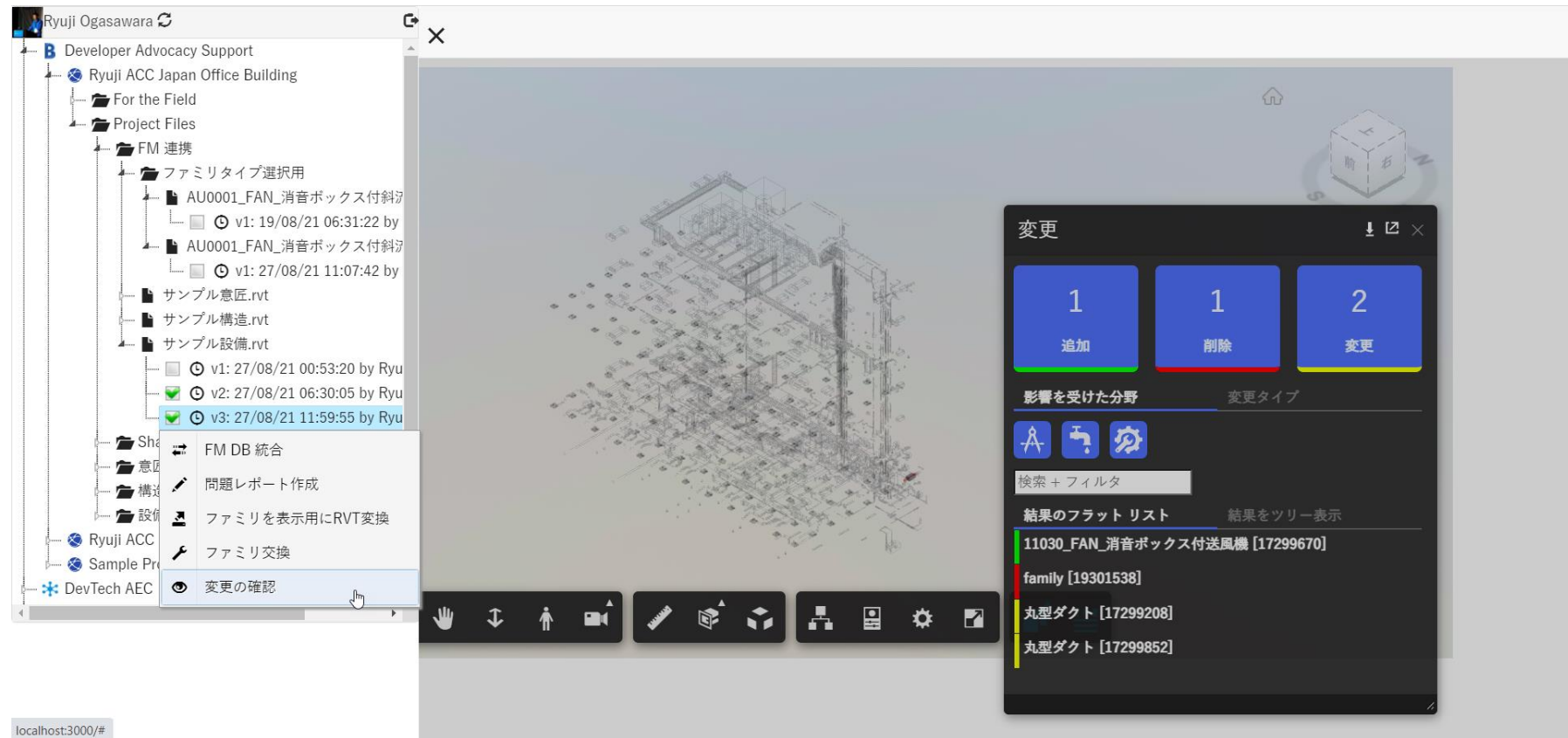
Revit API を使用してファミリインスタンス（MEP 要素）を交換しました。



5. Revit モデルの変更箇所を確認する

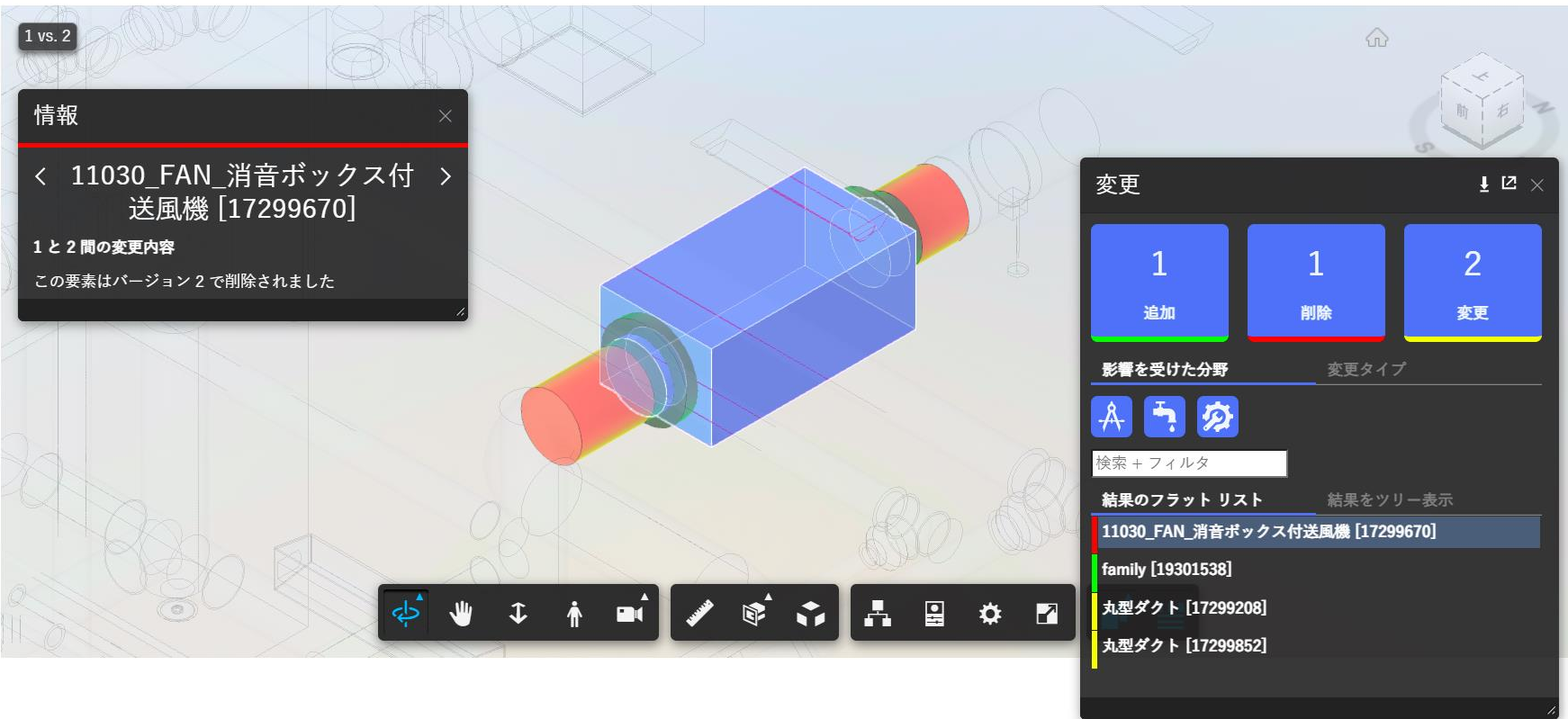
複数バージョンの派生ファイルの差分を表示する

Autodesk.DiffTool Extension をロード



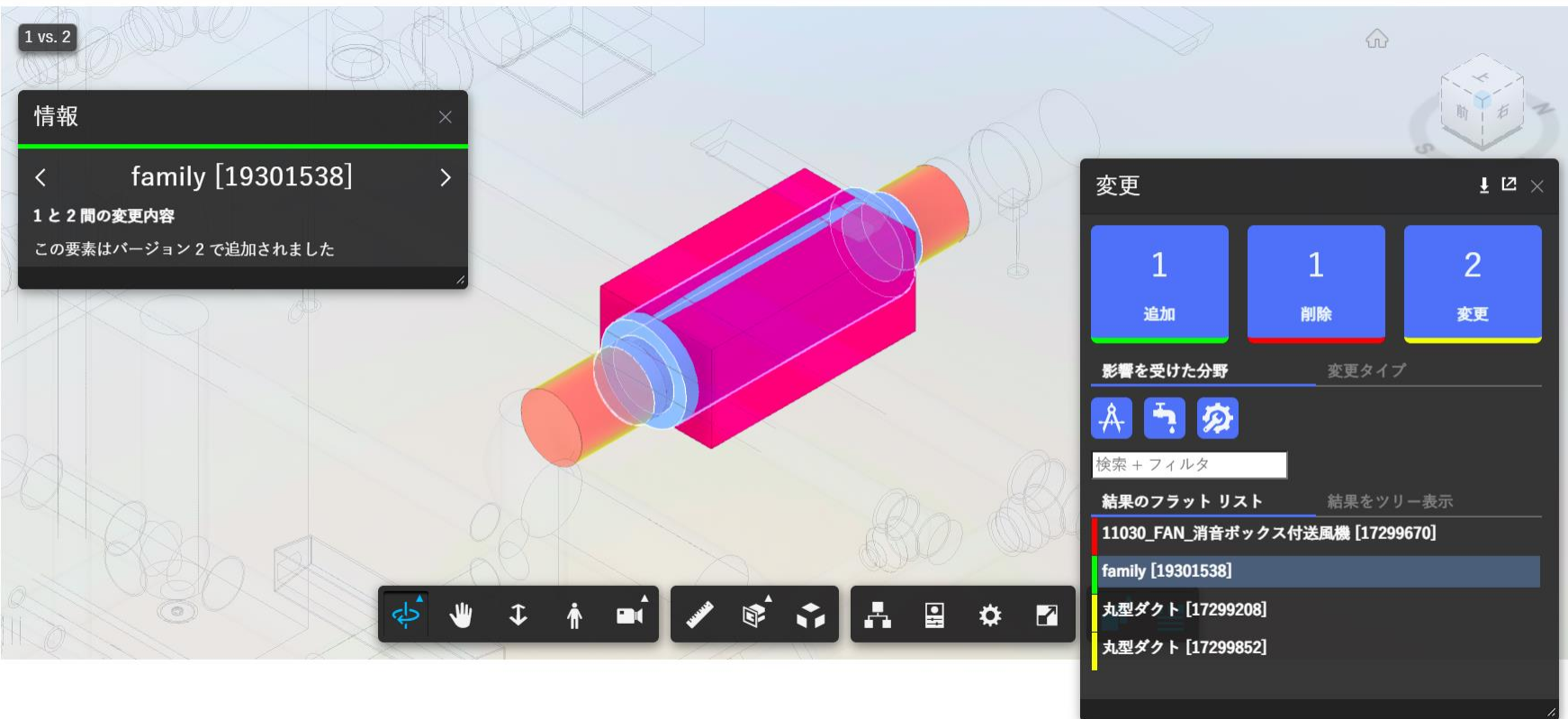
Autodesk.DiffTool Extension

削除された要素をチェックする



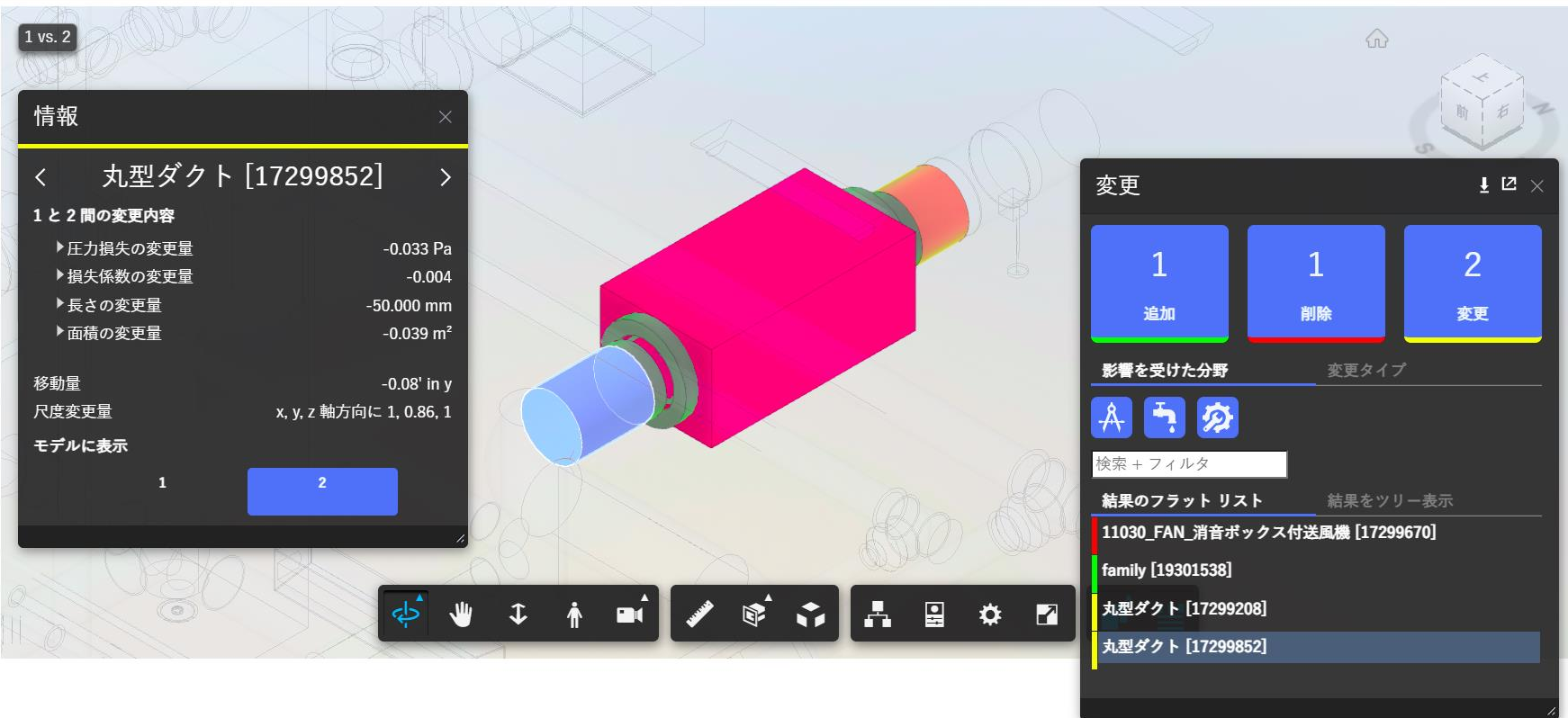
Autodesk.DiffTool Extension

追加された要素をチェックする



Autodesk.DiffTool Extension

変更のあった要素をチェックする



Autodesk.DiffTool Extension

実装手順

1. 2つの派生ファイルを Forge Viewer にロードします。
2. ロード完了後に 'Autodesk.DiffTool' Extension をロードします。
3. ロード時にオプション（すべて必須）を指定します。

primaryModels : ロード済みモデルの配列でプライマリです。

diffModels : ロード済みモデルの配列で差分比較対象のモデルです。

versionA : プライマリモデルのバージョン識別子です。

versionB : 差分比較対象モデルのバージョン識別子です。

contentType : タイプに応じてロジックが最適化されます。

- 'application/vnd.autodesk.revit': Revit
- 'application/vnd.autodesk.r360': Revit
- 'application/vnd.autodesk.fusion360': For Fusion 360
- 'application/vnd.autodesk.f3d': For Fusion 360
- 'application/vnd.autodesk.inventor.assembly': For Inventor (IAM)
- 'application/vnd.autodesk.navisworks': For Navisworks (NWD)
- 'application/vnd.autodesk.cad': For IFC
- 'application/vnd.autodesk.dxf': For DXF
- 'application/vnd.autodesk.autocad.dwg': For DWG

Autodesk.DiffTool Extension

サンプルコード

```
var models = [];  
  
treeNodes.forEach(treeNode => {  
  models.push({ name: treeNode.text, urn: "urn:" + treeNode.id.split('/')[0]});  
});  
  
const options = {  
  env: 'AutodeskProduction',  
  getAccessToken: getForgeToken  
};  
  
Autodesk.Viewing.Initializer(options, function () {  
  
  const viewerDiv = document.getElementById('forgeViewer');  
  
  forgeViewer = new Autodesk.Viewing.Private.GuiViewer3D(viewerDiv);  
  
  const util = new MultipleModelUtil(forgeViewer);  
  util.processModels(models).then(() => {  
  
    var extensionConfig = {}  
    extensionConfig.mimeType = 'application/vnd.autodesk.revit'  
    extensionConfig.primaryModels = [forgeViewer.getVisibleModels()[0]]  
    extensionConfig.diffModels = [firstViewer.getVisibleModels()[1]]  
    extensionConfig.diffMode = 'overlay'  
    extensionConfig.versionA = '2'  
    extensionConfig.versionB = '1'  
    forgeViewer.loadExtension('Autodesk.DiffTool', extensionConfig);  
  
  });  
});
```



まとめ

Tips まとめ

FM システム連携アプリケーション

- 外部データベースとの連携
 - ✓ Autodesk Construction Cloud を CDE として利用し、Forge からアクセスする方法
 - ✓ 識別子の取得方法
 - ✓ 外部 DB → Design Automation for Revit → RVT ファイル（パラメータの作成、属性編集）
- 問題の報告と履歴の閲覧
 - ✓ 複数モデルの統合表示 – AggregatedView
 - ✓ 3D View + 2D Minimap + Levels Extension
 - ✓ カスタム PropertyPanel
 - ✓ マークアップ
- Revit ファミリを Forge Viewer 表示用にプロジェクトに変換する
 - ✓ RFA ファイル → Design Automation for Revit → RVT ファイル（プロジェクト作成）
- プロジェクトのファミリインスタンスを別のファミリと交換する
 - ✓ RVT/RFA/JSON → Design Automation for Revit → RVT ファイル（モデル編集）
- Revit モデルの変更箇所を確認する
 - ✓ 複数バージョンの派生ファイルの差分を表示する – Autodesk.DiffTool Extension

組み合わせて、つなげる

Autodesk Construction Cloud + Autodesk Forge + ???



FORGE



Authentication



Data Management



Model Derivative



Design Automation



Webhooks



Viewer



BIM 360



ACC



Reality Capture



Web サービス API



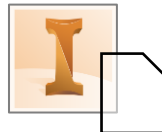
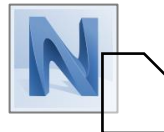
独自技術



他システム/
外部データベース



**AUTODESK
CONSTRUCTION
CLOUD™**



The background features several large, dark, metallic-looking geometric shapes, possibly representing computer components or architectural elements, arranged in a way that frames the central text. These shapes have sharp edges and some reflective surfaces, creating a high-tech, modern aesthetic.

AUTODESK UNIVERSITY

Autodesk およびオートデスクのロゴは、米国およびその他の国々における Autodesk, Inc. およびその子会社または関連会社の登録商標または商標です。その他のすべてのブランド名、製品名、または商標は、それぞれの所有者に帰属します。オートデスクは、通知を行うことなくいつでも該当製品およびサービスの提供、機能および価格を変更する権利を留保し、本書中の誤植または図表の誤りについて責任を負いません。

© 2021 Autodesk. All rights reserved.