TAG 2 SESSIONRUNDE 3

Rationale BIM-Programmierung mit Forge und Vergleich mit der Revit API

Jeremy Tammik
The Building Coder
Autodesk

Learning Objectives

- Envision the most efficient, productive, optimal BIM workflow for your specific task
- Split your software portfolio accordingly between desktop add-ins and cloud apps
- Make optimal use of Revit, Forge and other programming environments
- Understand the pros and cons of commercial, open source and DIY code
- Foster ubiquitous connectivity between components

Description

Overview of Forge and Revit API programming options in the BIM domain. Introduction to Forge and comparison of cloud based apps with desktop add-ins. Detailed discussion of comparable applications in both environments.

Überblick der Programmierungsmöglichkeiten mit der Revit API und Forge im BIM-Umfeld. Einführung in Forge und Vergleich der Cloud-basierten Webapplikationen mit Desktop-APIs. Besprechung der Implementation vergleichbarer Applikationen für Revit und Forge.

Please note that prior web, .NET and Revit add-in development experience is not required for this class.

Speaker

Jeremy works with the Forge Partner Development team on Autodesk APIs and web services, providing developer support, training, conferences, presentations, and blogging on the Revit API and cloud and mobile technologies. Jeremy is a prolific author and passionate about cooperation and sharing. Jeremy graduated with degrees in mathematics and physics, worked as a teacher and translator of both computer and human languages and as a C++ programmer on early GUI and multitasking projects. Jeremy is fluent in six European languages, vegetarian, has four kids, three grandchildren, loves cooking, climbing, hiking, sports, nature, literature, adventure, survival, problem solving and challenges of all kinds.

Table of Contents

Overview

- Envision the most efficient and productive BIM workflow for your chosen task
  - Who needs what where, when, and how?
  - Do your users engage in hard-core BIM design?
  - Or do they primarily reuse and enhance exiting BIMs?
- Partition your software development portfolio accordingly
  - Desktop versus cloud
  - Make optimal use of Revit, Forge and other environments
  - Pros and cons of commercial, open source, DIY code
- Programming BIM using web based components
- Connecting web based access with desktop BIM design
- Design Automation API for server-side RVT manipulation
- BIM related Forge and Revit code samples

Introduction

We discuss the desktop versus web based programming paradigms and explore the opportunities and advantages they offer for the BIM realm, including optimised, workflow-specific transparent connections between the two environments.

The desktop side is represented by Revit and custom add-ins.

The web based part is implemented by simple custom web servers, NoSQL databases and Autodesk Forge components.

To summarise further still, the focus is on two main topics:

- Generic BIM programming using web technologies versus Revit-specific add-ins
- BIM connections bridging desktop and cloud to make best use of both worlds

Desktop versus Cloud-Based BIM Programming

Takeaway

- Split wisely between Revit API and cloud based programming
- Revit is for BIM design
- Many BIM related workflows are read-only
- Many BIM related programming tasks do not require Revit
- Design Automation API for Revit is coming soon

Web-based BIM Programming versus Connecting Desktop BIM and Cloud

Your BIM data increases significantly in value the more you make use of it. Cloud-based technologies make it extremely easy to optimise access, use and reuse of data. We explore how to connect the desktop and the cloud to make any BIM data you choose available to anyone you choose, including optional round-trip real-time editing of selected properties and features. This class discusses various ways to enhance access and interaction with Revit models making use of the Forge platform, open source web development components and the Revit API. Forge enables easy translation of design files from and to different formats and extracting data from them for transparent usage almost anywhere. We explore suitable sample code and optimal starting points for your own apps and show how these programming tools can easily and vastly enhance your BIM workflows. This class addresses both beginners and experienced programmers.

Free Your BIM Data!

How can connecting the desktop and the cloud help free your BIM data?

Well, your BIM data will be criminally underutilised if the people that can make good use of it – or much worse, urgently require it – have no access to it.

What kind of data might you be interested in sharing or interacting with?

- Properties versus graphics?
- Simplified schematics, 2D, 3D or full-fledged renderings?
- One-way for information purposes, two-way interaction, full round-trip?

You cannot provide each potential user with Revit installation, or even assume they have access to a desktop computer.

The number of desktop computers is shrinking. The number of mobile devices is growing.

In general, growing numbers of potential users are already or will soon be using a mobile device.

The Internet and the cloud provide an easy and effective communication path today to anyone in the world of architecture, construction and maintenance of buildings.

HTML5 provides an easy and effective programming environment to interact with users across all devices and platforms, display properties, retrieve input, etc.

SVG enables you to easily create your own simplified 2D and 3D graphics for presenting data and potentially interacting graphically with it.

WebGL enables full-fledged high-performance graphics and renderings.

The Forge platform provides a complete collection of mature web services for sharing, displaying and querying CAD designs. It supports a one-way communication path from the CAD design to the cloud.

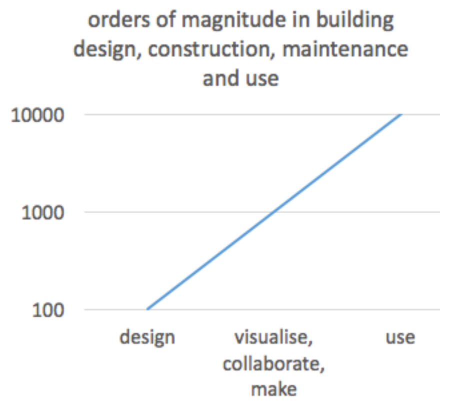
You can optionally add your own functionality on top of Forge to interact dynamically with specific aspects of the design and implement your own links to communicate updates back to the BIM.

To reiterate: making effective use of the web based BIM programming possibilities today can help provide global access, generic solutions and transparent connectivity between desktop and web based BIM apps. It might be crucial to the future of your company as well.

BIM Roles Collaboration

Participant counts grow by orders of magnitude

- design - architect, engineer – Revit
- visualise - client, everybody – Viewer
- collaborate - management - Glue + Plan
- make - construction - Field + Layout
- use - inhabit, maintain, FM - Building Ops



What Cloud? How Secure?

- Locally installed, totally private – 2D room editor uses private CouchDB web server
- Global, open source, with standard security measures – FireRatingCloud uses node.js web server on Heroku and MongoDB database on mongolab
- Forge OAuth2 user access authorisation – Roomedit3dv3

Access Restrictions and Security Issues

When I say ‘cloud’, what cloud am I actually talking about?

Please note that you yourself define exactly what the cloud is, where it is located and who can access it.

Most of what I demonstrate here can be installed locally on your own machine – no strings attached – only you can access it – on an intranet, extranet, or published on the public Internet to the entire world.

You decide exactly and have complete control over who can access what.

Obviously, the more widely accessible the BIM data, the more widely useful it can potentially become.

In case you publish it openly, you can restrict access rights as you wish and get involved with security issues.

Security in the public Internet is a pretty difficult topic. Happily, there are numerous libraries and services that can help handle it as easily and reliably as possible.

Keep It Simple!

If your solution is complex, you are almost certainly doing something wrong. Use the huge number of existing mature solutions, both open source and Forge based. Implement small, simple, custom add-ins, extensions and glue code. If it gets too complex, rethink, reanalyse, redesign, refactor and restart.

Simplify your data:

- Graphics? 2D? 3D? Properties?
- Customise, optimal workflow, minimal complexity
- Based on 'need to know'

Use existing components:

- Minimise add-ins, custom components, glue code
- Open source
- Forge

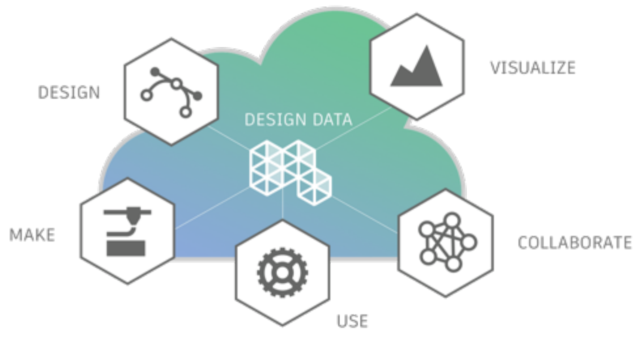
Autodesk Forge and the Design Automation API for Revit

Forge is a platform.

It empowers developers.

They can in turn empower their customers, clients, end users in all the main goals of Forge:

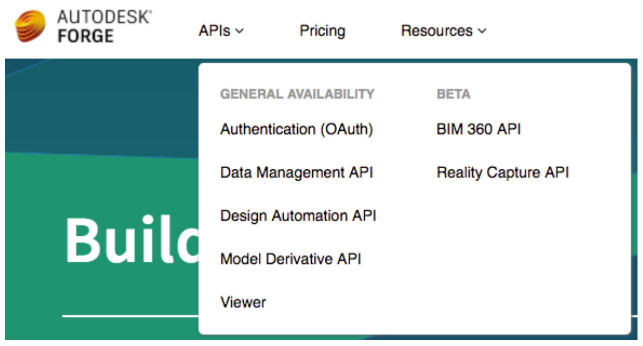
- Design
- Visualise
- Collaborate
- Make
- Use



This is exactly the kind of functionality developers can make good use of for making optimal use of existing BIM data.

Autodesk uses Forge to build its products.

Your application has access to the exact same APIs.



GENERAL AVAILABILITY

- Authentication (OAuth)
- Data Management API
- Design Automation API
- Model Derivative API
- Viewer

BETA

- BIM 360 API
- Reality Capture API

Forge Components

The following fundamental Forge components have been launched and are running in full production mode:

- Authentication
- Data Management API
- Design Automation API
- Model Derivative API
- Viewer

The first four of these are of interest to us right here and now and used by the following Forge room editor sample. The design automation currently applies to AutoCAD DWG and is being expanded to cover Revit RVT and other formats as we speak. By the way, your input on that topic would be very much appreciated!

Many more web services are coming soon. Some are not visible yet to the general public. As you can see, a few are currently in public beta:

- BIM 360 API
- Reality Capture API

Check it out for yourself at https://autodesk-forge.github.io > APIs.

Manipulate RVT via the Forge Design Automation API

No misunderstandings, please!

When I talk here about transparent connectivity between a BIM designed on the desktop in Revit and various web based apps that support access, view and manipulation of it, I am not suggesting that you can edit a Revit model in the browser, or make use of Revit as a service or on-line in any way at all.

If you are interested in serious web based interaction with RVT files, you will be glad to hear that the Forge platform includes the Design Automation API. At the time of writing, it enables manipulation and creating of AutoCAD DWG files. We are currently working on adding support for Revit RVT, Inventor IPT and other formats. In case of interest, please read about our thoughts on Revit I/O and get in touch to discuss your specific needs with us, cf. links below.

The advantage I underline here lies in leveraging your BIM data and sharing the absolute minimum of information required to complete a streamlined workflow in an optimal manner. Limited data sets can be passed around and used to drive an extremely efficient globally connected process.

Forge API Evolution

3 years ago, for sharing:

- Translation API
- Viewer

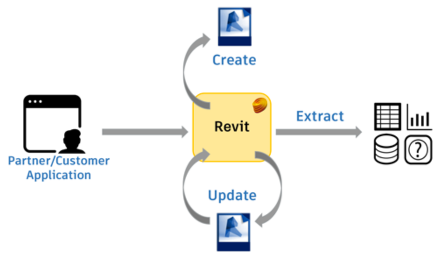
Today, 'Forge 2.0', for sharing, collaboration and DWG design:

- Model Derivative API
- BIM 360 HQ API
- Design Automation - AutoCAD
- Viewer
- Data Management API

Next year, 'Forge 3.0', covering all three areas more fully:

- Model Derivative API
- Data Management API
- Design Automation - AutoCAD
- Viewer
- Webhooks API
- Design Automation - Revit
- BIM APIs (HQ, RFIs, Issues, Checklists)
- Design Automation – Inventor
- Photo-to-3D API
- Forge IDX

Design Automation for Revit



What can it do?

- Full access to Revit DB API outside Revit via cloud services
- Execute .NET add-ins and Dynamo scripts
- Access custom functionality while operating on a Revit model
- Read data from anywhere, save data anywhere

What can't it do?

- No "sticky sessions" – only batch operations
- Access live Revit data in a RESTful way
- No Revit UI API access – there is no UI!

Using Web Technologies for BIM Programming

Revit provides a powerful tool for designing a BIM.

Many other downstream applications may require detailed BIM knowledge and serious programming and can still be achieved without limiting the application to the small universe of Revit users.

Revit API is BIM and end user product specific:

- Very powerful in certain areas
- Leverages Revit product functionality

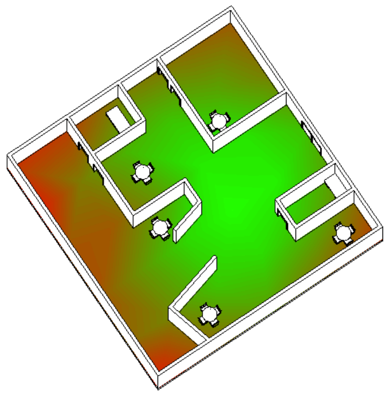
Therefore:

- Making use of it requires Revit product understanding
- It works only within a running Revit session
- It is completely event driven, hard to 'drive from outside'
- It is limited to the UI oriented Revit end user product
- Programming it requires special training
- Using a Revit add-in requires a Revit installation

Many BIM related tasks can be solved more generically.

RvtFader and ForgeFader demonstrate a compelling comparison solving the same non-trivial BIM analysis and visualisation task using the Revit API on one side versus JavaScript and three.js on the other.

Calculate and Display Signal Attenuation



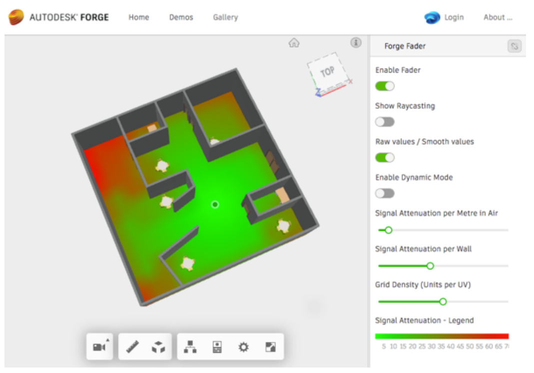
Both RvtFader and ForgeFader calculate and display signal attenuation within a BIM.

Given a source point, calculate the attenuation in a widening circle around it and display that as a heat map. Signal attenuation is caused by building elements and furniture. Their position and geometry is determined, and the resulting signal attenuation is calculated using ray tracing techniques.

RvtFader and ForgeFader

RvtFader and ForgeFader both implement the same task-specific functionality, calculating and displaying signal attenuation caused by obstacles such as walls and other building elements within the model. They provides a good example of the radically different approaches you can use to achieve the same task. In one case, it is limited to the Revit environment and a small set of expensive seats for exclusive users. In the other, you can make it available to every human being on the planet, if you wish.

The **RvtFader** Revit API add-in calculates and displays the signal attenuation using the analysis visualisation framework AVF and ReferenceIntersector ray tracing provided by the Revit API.



ForgeFader implements the same functionality within the Forge viewer using generic web programming techniques including JavaScript, three.js and a custom shader. This enables it to be used by anyone with access to the model and the app, and also enables use on models generated from other CAD seed files, not necessarily limited to Revit.

ForgeFader is a Forge viewer extension app that calculates and displays signal attenuation caused by distance and obstacles in a building model with a floor plan containing walls.

You can try it out for yourself in the forge-rcdb.autodesk.io Fader sample:

https://forge-rcdb.autodesk.io/configurator?id=59041f250007f5c0eef482f2

These two samples demonstrate that a non-trivial BIM programming task can be solved in a generic manner on the web. So once again: when analysing a given BIM programming task, think twice before implementing it as a Revit add-in. Do you want to limit its use to a small circle of specialists? Does it require or support hard-core BM design?

Connecting Desktop BIM and Cloud

Many other opportunities for providing BIM access and manipulation functionality to non-Revit users require modifying or adding small amounts of data to an existing BIM. For this purpose, implementing a transparent connection between the web-based app and the Revit BIM is useful. The following samples demonstrate such connectivity and specialised workflows.

Sample Overview

I demonstrate and discuss several working examples including (in chronological order):

- The original SVG based simplified 2D room editor
- The simplest connection sample, FireRating in the Cloud
- Roomedit3d, the Forge-based 2D + 3D simplified round-trip BIM editor
- The Forge meta property editor and RvtMetaProp Revit add-in

The first two are completely open source based, except for Revit.

Roomedit3d makes use of Forge, for which the translation process is free up to a certain point, while the viewer is based on open source three.js.

The Forge meta property editor enables adding property data to a Forge model within the Forge viewer, a first step towards read-write capability within the Forge environment. The data is not really added to the CAD seed model – that is impossible in the current technology. Only the seed CAD file generator can do that. Except for DWG files, though, where the Forge Design Automation API provides this functionality. As said, design automation is supporting other formats as well in the near future, including RVT.

RvtMetaProp is a simple Revit add-in which reads the data added by the Forge meta property editor and integrates it into the seed CAD file by modifying exiting parameter values and creating and populating new shared parameters.

The 2D Cloud-Based Round-Trip Room Editor

The 2D Cloud-Based Round-Trip **Room Editor** was my first serious exploration of a desktop-cloud connection. I initially worked on it in 2013 and presented it at the Autodesk Tech Summit conferences 2013, 2014 and Autodesk University 2013.

I knew nothing at all about cloud development before starting and was totally enthused by the tools and possibilities I discovered and the incredibly useful functionality I was able to put together in a very short time.

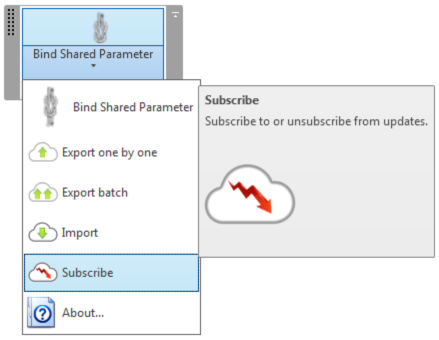
I learned about the NoSQL databases and used CouchDB for both data storage and web server.

I extracted a minimal subset of BIM data to store and display the element graphics, properties and relationships. The display and graphical user interaction is implemented using JavaScript and SVG.

It was immediately obvious that the exact same approach could be applied to address thousands of other tasks as well, both BIM related and totally generic.

It was also obvious how incredibly little effort was required to set up such a system.

FireRating in the Cloud



FireRating in the Cloud was implemented later, is much simpler, and intended to provide a perfect minimal starting point for you to base your own project on.

It does not deal with graphics or relationships, only a single value stored in a shared parameter, just like its predecessor, the Revit SDK FireRating sample. It demonstrates a simple modern architecture using a separate web server and database, node.js and mongodb.

FireRating in the Cloud commands:

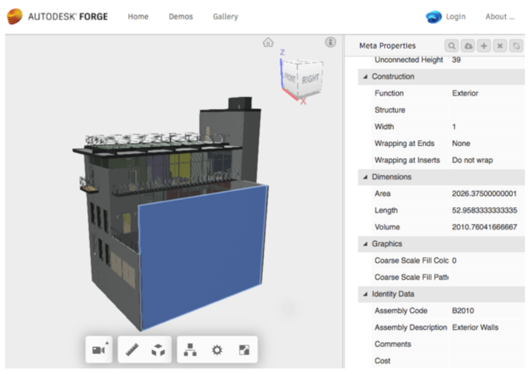
- Create the shared 'Fire Rating' parameter
- Export fire rating values for all doors
- Import the modified values back into BIM
- Store data for multiple projects
- Cloud database, Revit UniqueId
- Subscribe to changes

BIM in the Forge Viewer and Roomedit3d

Roomedit3d implements functionality superficially similar to the 2D room editor. However, the approach and approach is completely different. Instead of focussing on a streamlined workflow with an absolute minimum of data using completely open source components, It makes use of the Autodesk Forge platform. That saves a lot of effort and adds a lot of important functionality: full 3D model rendering, full set of BIM data, security,

The first release of roomedit3d uses a hard-coded model. **Roomedit3dv2** added the flexibility to process any model hosted on and selected from A360. It was implemented based on Augusto Goncalves' and Adam Nagy's Forge data management and model derivative samples for the DevCon developer conference in June 2016 and is no longer maintained. The current version **Roomedit3dv3** is based on a more robust sample and a viewer extension by Philippe Leefsma. It is well documented, totally suitable for reuse and demonstrates how easily you can adapt the existing Forge samples for your own purposes.

The Forge Meta Property Editor and RvtMetaProp Revit Add-In



Unconnected Height	39
◢ Construction	
Function	Exterior
Structure	
Width	1
Wrapping at Ends	None
Wrapping at Inserts	Do not wrap
◢ Dimensions	
Area	2026.37500000001
Length	52.9583333333335
Volume	2010.76041666667
◢ Graphics	
Coarse Scale Fill Colc	0
Coarse Scale Fill Patt	
◢ Identity Data	
Assembly Code	B2010
Assembly Description	Exterior Walls
Comments	
Cost	

The Forge meta property editor enables modifying and adding meta properties to any model in the Forge viewer. The resulting data can be downloaded in CSV or JSON format.

The RvtMetaProp Revit add-in imports and stores these properties in parameter values associated with individual BIM elements.

Handily enough, it can also be used as a stand-alone utility to automatically create shared parameters and populate their values on BIM elements from a spreadsheet, completely independently of the Forge app.

It reads the properties associated with individual BIM elements from a CSV or JSON file. If the property corresponds to an existing parameter on a BIM element, its value is updated accordingly. For a new property, a shared parameter is created.

Technologies and Implementation Details

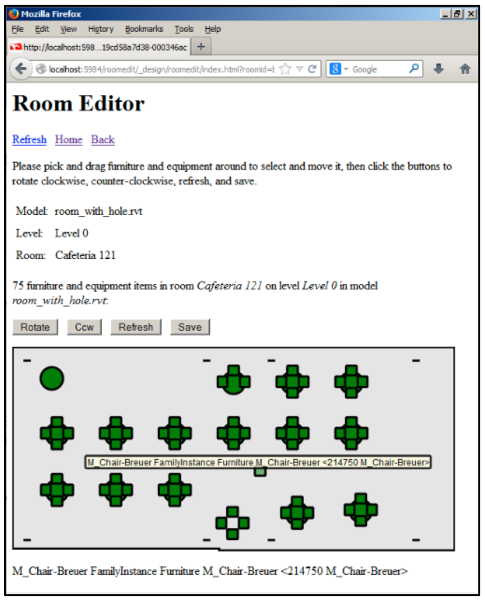
Skip this section if you will, unless you want nitty-gritty details and a closer look at some of the samples mentioned above, their implementation and the underlying terchnology used.

All of the samples include full documentation and pointers to numerous articles on the underlying research and analysis in their respecive GitHub repositories.

The 2D Cloud-Based Round-Trip Room Editor

The 2D room editor consists of two components:

- The RoomEditorApp Revit add-in
- The roomedit CouchDB web-based NoSQL database


Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:598...19cd58a7d38-000346ac
localhost:5984/roomedit/_design/roomedit/index.html?roomid=
Google
Refresh Home Back
Rotate
Ccw
Refresh
Save
M_Chair-Breuer FamilyInstance Furniture M_Chair-Breuer <214750 M_Chair-Breuer>

Room Editor

Please pick and drag furniture and equipment around to select and move it, then click the buttons to rotate clockwise, counter-clockwise, refresh, and save.

Model: room_with_hole.rvt

Level: Level 0

Room: Cafeteria 121

75 furniture and equipment items in room *Cafeteria 121* on level *Level 0* in model *room_with_hole.rvt*:

M_Chair-Breuer FamilyInstance Furniture M_Chair-Breuer <214750 M_Chair-Breuer>

It demonstrates bi-directional data exchange between the two, i.e., between a Revit BIM and a globally accessible cloud-based web database, usable on any device, in any browser.

The display is implemented based on SVG, scalable vector graphics. The interface supports position editing by 2D dragging.

The Revit add-in captures a simplified plan view of rooms and the furniture contained in them and exports that to the web database.

It captures the containment and relationship hierarchy including Project → Level → Room → Furniture = family instance → family symbol.

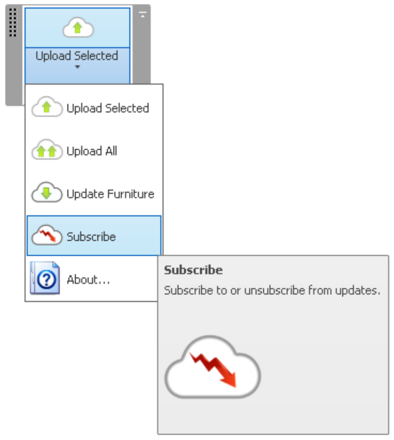
For the latter three, it also captures the 2D geometry:

- Room: 2D boundary loops.
- Furniture instance: 2D transform, i.e., location and rotation.
- Furniture family symbol: XY-projected 2D boundary loops.

The geometry is encoded in SVG strings, enabling easy visualisation and graphical interaction in any browser and on any device.

This data is stored in the database and displayed in the browser, supporting the following steps:

- Navigate through the Project → Level → Room containment hierarchy.
- Display a 2D view of a selected room with the furniture it contains.
- Click (or touch) and drag to modify the furniture rotation and location within the room.
- Edit any of the writeable furniture Revit parameters.



The main point is still to come, though:

Besides simply exporting the BIM data in the Upload Rooms and Upload All Rooms external commands, the Revit add-in implements two more:

- Update Furniture: reimport modified data from the database – this manual operation reads the browser-edited furniture rotation and location and updates the BIM accordingly.
- Subscribe: set up an external event to automatically poll for web database changes and immediately update the BIM in real time with no manual intervention at all.

Look at the recording of AU 2013 to see this in live action, and check out the implementation and complete source code in the RoomEditorApp and roomedit GitHub repositories (cf. links below).

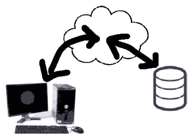
Software Architecture Connecting Desktop and Cloud

As said, the room editor presented above consists of two components: the Revit desktop add-in and a CouchDB database server.

We start out with a desktop data source, a cloud-based repository and a client on a mobile device:

- BIM – Building Information Model in Revit with add-in
- Cloud-based data repository in a CouchDB NoSQL database
- 2D rendering on mobile device using HTML, SVG and JavaScript

Real-time editing triggers database and BIM update:



- Graphical room editor on mobile device
- Update cloud database
- Reflect real-time changes in BIM

Due to the same origin policy, the JavaScript implementation uses server-side scripting. This is provided by the CouchDB database, which is provides a web server as well as the database functionality and reduces the number of components to two instead of three.

NoSQL Databases, CAP Theorem and ACID vs. BASE

NoSQL stands for *Not only SQL* and is the main modern database paradigm succeeding SQL and transactional, relational databases. It addresses their scalability issues and tends to be non-relational, distributed, open-source, highly scalable and capable of handling huge amounts of data. Frequent other characteristics include schema-free, easy replication support, simple API, eventually consistent.

Traditional transactional databases conform to the ACID paradigm, an acronym for Atomicity, Consistency, Isolation and Durability. These characteristics guarantee that all database transactions are processed reliably and that the database is in a consistent state every time a client accesses it.

Unfortunately for ACID, the CAP Theorem offers a strict mathematical proof that the ACID paradigm cannot simultaneously guarantee consistency, availability and partition tolerance, required for distributed systems.

The modern alternative BASE stands for Basic Availability, Soft-state, Eventual consistency. The system is not guaranteed to be in a consistent state at any given moment. Consistency is guaranteed, eventually.

Most large web sites today use NoSQL databases to handle huge numbers of transaction in a distributed and scalable manner.

FireRating in the Cloud

Next let us look at a simpler and more modern sample implemented in 2015.

FireRating in the Cloud is a re-implementation of the well-known FireRating Revit SDK sample. Its most important enhancement over the original sample is the multi-project support. It stores data from an unlimited number of RVT projects in one single NoSQL database. The original SDK sample implements three external commands:

- Create a shared 'Fire Rating' parameter and bind it to the Doors category.
- Export fire rating values for all doors in a project to an external spreadsheet.
- Import the modified values back into the project.

The enhancement is simple:

- Store data for multiple projects to one single cloud-hosted database.

This is achieved by storing the data in a MongoDB database instead of an Excel spreadsheet, using the Revit door element UniqueId instead of its element id for identification, and adding a field specifying the project it lives in.

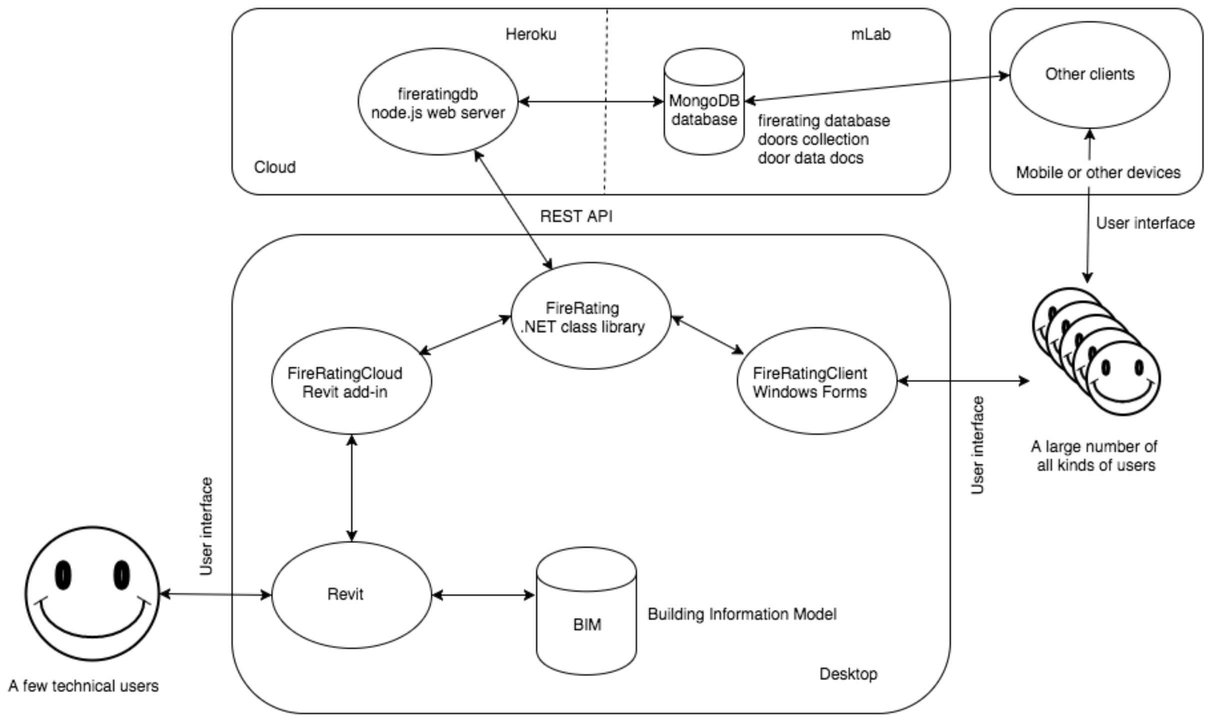
Just like the room editor, FireRating in the Cloud consists of two components, a web server firer atingdb and a Revit add-in FireRatingCloud:

- The web server is a very simple node.js REST server driving a MongoDB NoSQL web database. It has no user interface of its own.
- The Revit add-in implements the same three external commands as the original SDK sample and uses REST to read and write the fire rating values to the cloud-hosted database.

Again, bi-directional data exchange between the Revit BIM and a globally accessible cloud-based web database is enabled.

Furthermore, again like the room editor, a real-time BIM updating functionality is provided by the Subscribe command.

Although fireratingdb does not implement any user interface of its own, the mongo database can be hosted on a platform that does, e.g., mongolab, again making it usable and the data accessible on any device, in any browser.



I discussed the research and development for this in depth on The 3D Web Coder blog. The FireRatingCloud GitHub repository includes an overview of them all (cf. links at end).

I host the node.js web server on Heroku and the MongoDB web database on mongolab for free.

Deploying a GitHub project to Heroku can be totally automated.

Using Mongolab to host the database is more comfortable than keeping it locally.

I can switch back and forth each of these between remote or local deployment by simply setting two Boolean variables.

See for yourself in the live demo how utterly cool, easy and flexible it is to have these components communicating with each other and working together on their separate platforms: Heroku, Mongolab, and Revit.

We demonstrate this system up and running live, both locally on the desktop and globally accessible on the web using Heroku and mongolab.

In this case, we rely on Heroku and mongolab to handle security.

We can also dive into the source code at this point. Let’s take a look and see how simple the implementation really is.

Revit Add-In C# REST Client

Retrieve all doors through a filtered element collector, extract their data and PUT it to web server:

```
foreach( Element e in collector )
{
  Debug.Print( e.Id.IntegerValue.ToString() );

  doorData = new DoorData( e,
    project_id, paramGuid );

  jsonResponse = Util.Put(
    "doors/" + e.UniqueId, doorData );

  Debug.Print( jsonResponse );
}
```

PUT helper method using RestSharp:

```
public static string Put(
  string collection_name_and_id,
  DoorData doorData )
{
  var client = new RestClient( RestApiBaseUrl );

  var request = new RestRequest( _api_version + "/"
    + collection_name_and_id, Method.PUT );

  request.RequestFormat = DataFormat.Json;

  request.AddBody( doorData ); // uses JsonSerializer

  IRestResponse response = client.Execute( request );

  var content = response.Content; // raw content as string

  return content;
}
```

Node.js MongoDB Web Server

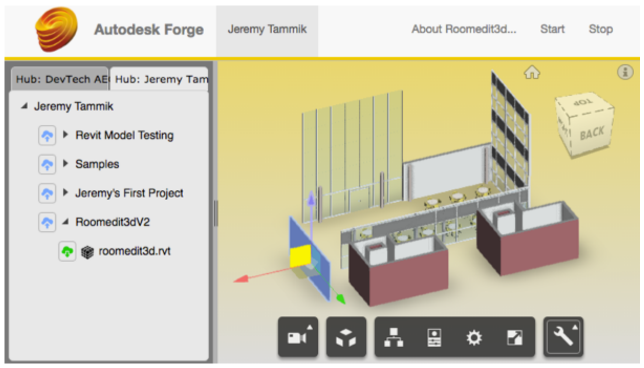
The entire mainline server implementation:

```
var pkg = require( './package.json' );
var express = require('express');
var mongoose = require( 'mongoose' );

var localMongo = false;

if(localMongo) {
  // local database
  var mongo_uri = 'mongodb://localhost/firerating';
} else {
  // mongolab hosted
  var mongo_uri = 'mongodb://revit:revit@ds047742.mongolab.com:47742/firerating';
}

mongoose.connect( mongo_uri );
var db = mongoose.connection;
db.on( 'error', function () {
  var msg = 'unable to connect to database at ';
  throw new Error( msg + mongo_uri );
});

var app = express();

var bodyParser = require( 'body-parser' );
app.use( bodyParser.json({ limit: '1mb' }) );
app.use( bodyParser.urlencoded({ extended: true, limit: '1mb' }) );

require( './model/door' );
require( './routes' )( app );

app.get( '/', function( request, response ) {
  response.send( 'Hello from the cloud-based fire rating '
                + 'database ' + pkg.version + '.\n' );
});

app.set( 'port', process.env.PORT || 3001 );


var server = app.listen(
  app.get( 'port' ),
  function() {
    console.log( 'Firerating server '
                + pkg.version
                + ' listening at port '
                + server.address().port + ' with '
                + (localMongo?'locally ':'mongolab-')
                + 'hosted mongo db.'); }
);
```

That is the simplest complete desktop and cloud connection sample that I am aware of.

What if you need a more powerful and realistic visualisation, need to handle larger models, or require other CAD related functionality that is not available as public open source? Enter Autodesk Forge.

Forge-Based 2D + 3D Round-Trip BIM Editor



The newest incarnation of the ancient 2D room editor is the Forge-based roomedit3dv3.

It uses Forge authentication to download the model from A360 and display the BIM data in the viewer, so there is no need for creating our own simplified custom view.

This entails several huge advantages over the custom visualisation:

- Realistic model rendering in both 2D and 3D, optionally linked
- Complete access to all BIM data, incl. geometry, structure and properties
- Not bound to any specific model
- Secure authenticated access
- Embedded in a full ecosystem of mature CAD related web services
- Minimal amount of coding based on boilerplate sample code

The model display and user interaction can be implemented in either 2D or 3D, depending on your specific requirements. Please refer to the LmvNav sample to see how the two views can be displayed and synchronised side by side.

The Forge authentication and data management functionality is used to access any sample hosted on A360, providing flexibility and enabling you to test-run it right out of the box on your own models.

A custom viewer add-in can be optionally activated. It supports selection and interactive dragging of an individual element on the screen. The final translation is broadcast to the world via socket.io. With

Revit up and running in the corresponding BIM and the Revit add-in loaded and subscribed to the broadcast, the translation applied in the viewer is used to update the BIM in real time.

- GitHub: https://github.com/jeremytammik/roomedit3d
- Live sample URL: https://roomedit3dv3.herokuapp.com
- Discussion and demo recording:

http://thebuildingcoder.typepad.com/blog/2016/10/
roomedit3dv3-up-and-running-with-demo-recording.html

Conclusion

For DIY, I love the NoSQL database concept and implementations. CouchDB is mean and lean, but harder to work with than more mainstream and globally portable tools like node.js and MongoDB.

Forge provides access to all required data in any CAD model, whether from Autodesk or other platforms, and enables new powerful global approaches and solutions built using simple independent custom web based components.

All I can really say is repeating the main message:

- This is simple.
- This is scalable. I can put specific data for all my projects into one single container for global search, analysis and further interaction.
- This is powerful. Properties, simplified graphics, full-fledged renderings, all models working interactively in 2D or 3D models in the browser on any device, according to your optimised workflows and specific requirements.
- This is useful. I can share certain data globally in the browser, with anybody I choose, with zero installation, on any device.

The BIM domain offers innumerable opportunities to make productive use of connecting the desktop with the cloud, or just providing cloud-based access to query and manipulate the BIM.

If you don't jump into making use of them, watch out for those that do...

Resources

Sample Repositories on GitHub

Each of the samples lives in its own GitHub repository with its own documentation pointing to more detailed underlying research and implementation steps discussed in sequences of blog posts.

The first two demonstrate BIM programming techniques to solve the same task using either built-in Revit API functionality or generic web technologies:

- **RvtFader**
  https://github.com/jeremytammik/RvtFader
- **ForgeFader**
  https://github.com/jeremytammik/forgefader
- **Forge RCDB Fader demo**
  https://forge-rcdb.autodesk.io/configurator?id=59041f250007f5c0eef482f2

The rest demonstrate connecting a globally accessible cloud component with a BIM on the desktop. Each therefore consists of two components, a C# .NET Revit API desktop add-in and a web server:

- **RoomEditorApp**
  https://github.com/jeremytammik/RoomEditorApp
  and the
- roomeditdb
  https://github.com/jeremytammik/roomedit
  CouchDB database and web server demonstrating real-time round-trip graphical editing of furniture family instance location and rotation plus textual editing of element properties in a simplified 2D SVG representation of the 3D BIM.
- **FireRatingCloud**
  https://github.com/jeremytammik/FireRatingCloud
  and the
- fireratingdb
  https://github.com/jeremytammik/firerating
  node.js MongoDB web server demonstrating real-time round-trip editing of Revit element shared parameter values stored in a globally accessible mongolab-hosted db.
- **Roomedit3dApp**
  https://github.com/jeremytammik/Roomedit3dApp
  and the
- roomedit3d
  https://github.com/jeremytammik/roomedit3d
  Forge Viewer extension demonstrating translation of BIM elements in the viewer and updating the Revit model in real time via a socket.io broadcast.
  An updated Forge-based version adds the option to select any Revit model hosted on A360, again using the Roomedit3dApp Revit add-in working with the
- roomedit3dv3
  https://github.com/Autodesk-Forge/forge-boilers.nodejs/tree/roomedit3d
  Autodesk Forge Viewer extension to demonstrate translation of BIM element instances in the viewer and updating the Revit model in real time via a `socket.io` broadcast.
- **RvtMetaProp**
  https://github.com/jeremytammik/rvtmetaprop
- **Forge RCDB meta property editor**
  https://forge-rcdb.autodesk.io/configurator?id=59780eec17d671029c53420e

Access and Reuse of the Sample Material

You can fork and clone all samples discussed directly from their GitHub repositories to get started implementing your own ideas really fast.

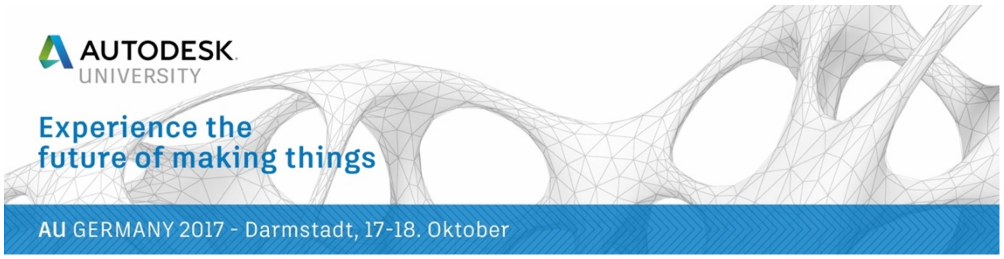
My underlying research is extensively documented in blog posts, so it should help you circumvent the numerous snags I hit and resolved. Pointers to the blog posts are listed in the GitHub repository documentation.

Recordings

An earlier incarnation of this document, the accompanying slide deck and a live 75-minute recording of the RTC presentation are available from The Building Coder blog at http://thebuildingcoder.typepad.com/blog/2016/10/connecting-desktop-and-cloud-at-rtc-material.html

Learning More

- Revit Developer Centre: DevTV and My First Plugin Introduction, SDK, Samples, API Help
  http://www.autodesk.com/developrevit
- Developer Guide and Online Help
  http://www.autodesk.com/revitapi-help
- Autodesk Community Revit API Discussion Group
  http://forums.autodesk.com > Revit Architecture > Revit API
- ADN AEC DevBlog
  http://adndevblog.typepad.com/aec
- The Building Coder Revit API Blog
  http://thebuildingcoder.typepad.com
- ADN, The Autodesk Developer Network
  http://www.autodesk.com/joinadn and http://www.autodesk.com/adnopen
- DevHelp Online for ADN members
  http://adn.autodesk.com
- NoSQL: http://nosql-database.org, https://en.wikipedia.org/wiki/NoSQL,
  http://www.mongodb.com/nosql-explained
- Autodesk University 2013 session *DV1736 – Cloud-Based, Real-Time, Round-Trip, 2D Revit Model Editing on Any Mobile Device*
- Connecting desktop and cloud presentation at RTC Europe 2015
  http://thebuildingcoder.typepad.com/blog/2015/11/connecting-desktop-and-cloud-room-editor-update.html
- Connecting desktop and cloud recording at the AU 2015 AEC booth
  http://thebuildingcoder.typepad.com/blog/2015/11/connecting-desktop-and-cloud-at-au-and-devdays.html
- LmvNav sample synchronizing 2D and 3 https://calm-inlet-4387.herokuapp.com/
- Autodesk Forge: https://forge.autodesk.com
- Forge API overview, documentation, and your own apps: https://developer.autodesk.com



AU GERMANY IN DARMSTADT