MICHAEL HUDSON: How many other people have already been to a Dynamo class so far? Good. We're not going to give you the Dynamo 101 or do a great deal of live on-screen stuff here. We're going to really quite try and show you showcases of what we've done over the previous 12 months in our practice specifically.

So we're not necessarily saying what we've done is the best way to do it. It's just the way we have done it, so that's our quick caveat. But we've got a fair amount to get through, so if we could leave the questions at the end, if possible, as well.

So this is obviously the standard information, but really what we're focusing on is how we've used Dynamo to accelerate the BIM processes that we've got in our practice using both our primary offering platform, which is Revit, but also our coordination platform, which is Navisworks. We'll give you a really, really quick demonstration of why PythonScript is quite easy to get into and how we've used it to facilitate some API access. So hopefully by the end of it, you'll have a confidence that you can go back to your offices and approach some repetitive tasks using Dynamo to resolve them, do them quicker, have a concept of the visual programming language. Hopefully, you probably already do, because of all the other classes you've been to, have an overview of some of the design script and PythonScript stuff that we are doing and feel that you can also apply this to things outside of Revit, so specifically Navisworks.

So about us, if you didn't come to the class last year, nice to see you. I'm an associate director and chartered architect for a company in the UK. I'm also a BIM consultant and a University lecturer.

I've been using parametric and BIM software for well over 10 years now. I'm nondenominational. I try and learn everything. I'm just one of those individuals.

HORSTEN STRATHAUS: Hi, my name is Thorsten Strathaus, German architect. I have a background in architect and engineering and being in between these provisions, I usually had to deal to get information from A to B and back. So I started very early to trade some parametric workflows to keep information exchange, [INAUDIBLE], free and some other things which interest me is like, for example, multi-objective optimisation for solutions for structural engineering and architecture.

MICHAEL So a little bit about our practice, Flanagan Lawrence. We've only been around for about three

HUDSON: years or so, and we're based in London. We've got about 90 employees.

We only do BIM work. We've been very lucky to receive a number of RIBA awards and World Architecture Festival awards for our design projects, but that doesn't mean that we only do exotic shapes. And I think that's part of the kind of aim from this class is that actually we've applied these processes to all of our work.

To give you a feel for some of the ranges of what that we do, we do masterplans, but we do small theatre projects. We do residential/mixed use. We do offices. And we're also part of a wider design group that offers interior design BIM consultancy, and we started a visualization company as well and also site expertise. So we're really kind of a multi-disciplinary practice, and part of that is related to just realizing that we can use our tools much better because we just like using them.

So previously, it's called Dynamo Hero 2. There's a reason because we did Dynamo Hero last year. And like all movies, at the beginning of any movie, you kind of a step backwards and talk about what happened previously.

So back in 2013, we started playing around with Dynamo. And the main motivation for this-- and some people forget this-- is that it had a direct link, and it was bi-directional to Revit. It meant that all of our other processes were very lossy, which meant that had to interchange data between various platforms in order to migrate some conceptual design into Revit. It was gone. Suddenly you actually had a bidirectional process, and that was fantastic for us, so we really threw ourselves in on it.

And like basically everybody, we just tried to do some crazy curves, of course, like everybody else does. Make some geometry. And we started to realize that it wasn't just geometry that you could make in this software. You could also use it for acoustic simulation and optimisations of forms and structures.

And we use that to develop on a really, really tight schedule a project which was built on the South Coast and has won quite a few awards for us. And I talk about this in great detail in last year's presentation, so I encourage you to go onto the online resource and download this presentation too if you want to know about this specific project. But the great thing was that we learned so much in this project, that we have tried to apply it to everything else now.

So Dynamo Hero 2, a sequel, so normally in sequel movies, what happens is the hero has some new challenges coming along. They also have some new friends.

[LAUGHTER]

And there's a lot more classes on Dynamo, and that's fantastic because the software now is developing at such a pace. There are new packages being released to access new programs. I encourage all of you just have a play and find out what you can do with it. It might not do what you want at the moment, but the development cycle is so incredibly fast that it probably will do very soon.

I was in a class earlier today with automotive designers, and they were really excited about it too. So it just continues to move ahead at quite a pace. But the challenges is the data, migrating things. As our jobs become more ambitious and we work with more and more design team members that are using BIM themselves, it means that there's just so much more data now, even 12 months since the last time as I spoke here.

So how do we start to use this work? Because the thing is this is actually what we find ourselves in in the majority of the scenarios, just chaos, just huge amounts of data that isn't very organized or isn't in the form that you need it to be. So we started thinking to ourselves, what superpowers should Dynamo possess? Or, indeed, how can we make Dynamo a superhero for us really? And it came down to four tropes, accessibility, connectivity, teleportation and time travel. And hopefully I'll explain all of those to you, and you may or may not believe that that's what they are, but they very much are kind of things that have worked well for us.

So the first one is accessibility. And 12 months ago, Dynamo only worked inside Revit pretty much, and now it is a standalone operation. There is Dynamo Sandbox. There's also Dynamo Studio.

And it means that it opens up a whole bunch of opportunities for, well, number one, for you to learn without the risk of mucking up your Revit files because it works independently now. It also means it runs a lot faster because it's standalone. It has it's own kind of power to work by itself.

Also, grouping. If anyone's ever used any kind of visual programming language, they get messy very fast. You know, you're throwing your thoughts together, and you're throwing them down on a canvas. The grouping functionality allows you to actually remind yourself when you come back from that coffee break what you were doing. It's incredibly useful to us. It seems like a relatively small thing, but the UI is moving at quite a place, and that's really helpful, particularly when explaining to your colleagues what you're trying to achieve. And also campus operations, basically if you right-click on any space inside the canvas, you can now jump to a small window, which gives you the main functionality that you want to have. All these things that just make you work foster and allow you to think about what the task in hand is rather than struggling with a user interface.

And also, there's PythonScript, which runs inside IronPython. I'm going to now attempt a potentially suicidal live demo of this, so you'll have to bear with me. So obviously, this is the main canvas that we're working on. And if I just right-click and there I immediately get this interface to go in here.

AUDIENCE: [INAUDIBLE].

MICHAEL HUDSON: It's not showing. It's not showing. And the tech support is not in here, amazing. OK.

HORSTEN STRATHAUS: Can we kick the presentation for a second?

MICHAEL HUDSON: Yeah.

HORSTEN STRATHAUS: [INAUDIBLE]. but thanks.

[INTERPOSING VOICES].

MICHAEL HUDSON: Yeah, so if we just press Escape.

[SIDE CONVERSATION].

OK. I need tech support.

AUDIENCE: [INAUDIBLE]?

MICHAEL Yeah, well, that's live demos for you. They always do this kind of thing. That's even better.

HUDSON:

[INAUDIBLE].

HORSTEN STRATHAUS: I don't see anything [INAUDIBLE].

MICHAEL HUDSON: Oh, we can't see on the screen. OK.

[LAUGHTER]

Yeah, apologies for this, guys. We'll just have to change our screen settings briefly. I'm obviously stalling for time here. Essentially, the interface that we've got going on here is not visible anymore.

[LAUGHTER]

HORSTEN STRATHAUS: Invisibility is one of the key--

MICHAEL HUDSON: Yeah, so this is a new one.

[LAUGHTER]

HORSTEN STRATHAUS: Every hero at that, come on.

MICHAEL HUDSON: OK, right. OK, so I'm going to right-click. So I can't obviously use the traditional dropdown search ar here, but there's this section here which also does the same thing. So I can type in point, and then it will also give me that tool. And the same functionality is still there, so I can still control C, control V it, and I've got two points.

And I want to input some values in here, and I'm probably just going to use a slider because that's going to-- slider. And I'm going to slide her like that. I'm just going to go in there, and I'm going to [INAUDIBLE].

Now everyone has their own way of doing things, but what I do encourage to anyone who's using this is that you can change these headings in here. And it's not often said, but if you know what this definite input value is going to be, I tend to give them a name or something like that so that it's really apparent to anyone else who's picking up the graph from you that they can understand what you're trying to achieve. So I'm just going to leave it like that for the minute. It's just so that people understand that what's happening there.

And then you can select both of these and right click in Create-a-Group. And then give that a name, Input like that. And then you can make it a massive font as well if you want to. But when the graphs get really big, honestly, this is an enormous time saver. It seems totally banal on this kind of example, but that's actually quite helpful.

So I'm just going to move these sliders across slightly, so they're now on top of each other like that. And then just add another line. Start point, end point. Anybody can do this. This is quite preposterously easy to do.

So you can say, when you're generally sketching things out-- and we both do this too-- when you just trying to form an idea, this is the way to do it. It always is. You just drop all of the existing nodes onto the canvas, start developing an idea.

But say, for example, it's actually closed off, and it works really well for. And you don't really want to be able to muck it up yourselves or forget what you were trying to do. One thing to do is actually just Select these three, and then by clicking in the canvas space, you can click on a function called Node to Code.

And what that does, it just compresses this into a code block, which is really, really useful when you've got some more complex definitions happening. Obviously, this is a really simplistic one which I have to use for this example in the time that we've got, but you can see what it does here.

Now one thing that is a bit weird about it at the moment because it's still obviously a beta is that every single time I've done this example in practice, these names here have been different. So sometimes this T1, T5. Today it's 0.1, 0.2, and 9.1. So there's still a little bit of coding that those guys need to do, but it is incredibly powerful.

Now what this also does is it gives you an avenue into the design script syntax and language. And it's inherently very similar to C Sharp, but it also has a lot of similarities with the syntax or language being used inside PythonScript. And PythonScript is a node that can also run inside

Dynamo.

So if I just go and look for the Python node, that's it. There's not a lot going on in there. And if you double click on it, then that shows you the IronPython Shell, which is running inside. So you can add the lines of code in here.

And the great thing is that they give you a helping hand to start off with. They actually put into it what library it should go and find, and it goes and finds the design script library. So it becomes very, very easy to actually creates a simplistic Python node using the design script node of the code block you've already made.

But you do have to do some crucial things. And this is just so if anyone's used Python before, apologies, this should be a bit boring for you. But if you haven't used Python, it's just to kind of explain it. Because at first off, you need to have some inputs.

So I need to make sure I've got inputs this time, so I need to have T1 equals N, which is the looking for an N function, and then in the square brackets 0. Just like any other coding language, the first value is 0 rather than 1. And T2 equals, in brackets, 1.

And then I'm going to make sure that my out is called, and I'm just looking at my design script language over here. I want it to be called Line 1 I'm hoping. Things could lock down and this could just go horribly wrong, so apologies in advance if it does.

So I'm just going to accept the change on this, and immediately it's going to say, this doesn't work because I need two inputs. [INAUDIBLE] have no functionality at the moment. So I'm going to go to my code block over here. I'm going to just copy it. Get back in here and then paste it.

Now one point is that there were no semicolons to execute lines in PythonScript, so you have to delete those. That's something you have to get rid of. I just want to double check through this. T1, T2, and then just to make sure it's really obvious to everyone, I'm just going a 0 set as well in there. I'm just going to accept the changes on there, and then hopefully if I connect this all up, it doesn't work.

[LAUGHTER]

Ah, what's going on?

HORSTEN STRATHAUS: Just kill the Autodesk. I didn't say that. We don't kill Autodesk. We just have to delete the variable.

MICHAEL HUDSON: Ah, yes. Well, said. Yes. I've done this a number of times.

In design scripts, the names only have these actual names in them. The Autodesk thing is a nuance that's hidden in there for some apparent reason. Thanks for reminding me of that one.

So that should work. Yeah, there we go. So that is it. Apologies for the slight delay there, but that's essentially how a PythonScript node can be assembled.

So there's the two different ones going on there, but you see that because this one only outputs the line rather than the point, so you can see only the line going on there. So obviously you use it for much more complex functionality than that. The real benefit of using Python is stability for creating recursive functions, and so I'd hit Looping and things like this. There are some Looping things in Code Block, so I wouldn't say don't use them. The while loop is available in there, but I just like Python because it's very stable.

Anyway, after that fun little sideshow, we should go back to our presentation. And, OK, we'll just have to do it this way, I think.

HORSTEN STRATHAUS: [INAUDIBLE]. Can we get some elevator music while we [INAUDIBLE].

MICHAEL HUDSON: Some elevator music. [LAUGHS] Actually, are there any questions whilst Horsten is getting that set up about what I just showed you there at all? Great, you're all experts now. Super.

HORSTEN STRATHAUS: Experts in Python. That's great. Nobody achieves this in this short time.

MICHAEL HUDSON: OK. Super, up and running. OK, so just to conclude that notion of accessibility. The fact that they've got these kind of really great UI functionalities in here that they continue to improve and they really listen to everybody when they give solid feedback about how they'd like the program to improve, above all, it makes you feel quite valued as a customer.

But also, the fact that it stably runs several textual-based coding languages inside it means that you've got a number of entry points in order to deliver the design that you want to do. And there's also some really great resources, so the main point of access is the DynamoBIM website, which I hope you've all had a look on. But there's also the DynamoPackages website with a number of people uploading things. One of the best uploaders has just arrived, Andreas to the back there. There's also GitHub because there's a lot of open source work that's happening on there and also Dynamoreach, which will go live very shortly I've been promised, which allows you to basically run Dynamo through your phone, which is pretty cool.

Anyway, let's move onto our next superpower, which is connectivity. So previously, this was kind of the limit of what we were really doing. We were doing extraction of schedules and things, pumping them through Dynamo and doing some manipulations going into Revit, round tripping, this kind of thing, in order to optimize our designs.

But over the last, say, 18 months, there's been a real proliferation of platforms that you can actually drive through Dynamo. It's amazing. Some packages to run Maya have just been released as well. Basically, you name it, you can do it. Because the background of it is all the same kernel.

So basically any software that you use or any of your partners use, you could potentially be manipulating, you can be working around, with Dynamo.

So we're going to focus on-- oh, sorry. And this is essentially a capture of a Python script. Essentially all it does is it brings in some extra libraries of that platform that you want to be able to use. And then you can basically use the functionality inside that too.

So we tend to use it that way. It's not the only way. But it is one way of doing things.

So the one that we're going to talk about initially is a workflow where we've included Navisworks to streamline our repetitive tasks. Anyone who does clash coordinations, you know what I'm talking about here. But just a quick overview of the project.

So was is from our master plan. That's Wembley Stadium if anyone's been to the UK. Behind it we were delivering our 1-billion-pound master plan. And this was a residential phase, which was around about 100 million pounds. 360 residential units. Also an underground car park in Energy Center.

We're employed as the architect, the interior designer, and the BIM consultant on this job, so we were incredibly powerful to impose our rule upon the way that it was going to work. Which was great.

The client, however, was very well-informed. They wanted something called BIM Level 2. I don't know if anybody's aware of that, but essentially it's a government-mandated standard. There's no obligation for private companies to do this standard.

But essentially what it means is that you have certain levels of delivery gates where you have models that are assessed and qualified for a certain standard all the way through the project. There's a bit of similarity with some of the AIA E202 standard, but it's obviously written with their own English version of it.

All of the design consultants were delivering full BIM models. And there were biweekly model federations and there were about 60 models going through. And that was something we had to try and manage.

So the first thing that we did was we set up something called essentially a model responsibility matrix, which is very similar to the LOD system that you guys over here-- I think it's a BIMForum one. Essentially it's PAS1192 part two.

The crucial nuance is it forks and says geometry and information. And they can be two different levels at any point inside the project. So you have to be very granular in the way that you assess that the models are being delivered by consultants. And you also have to agree pretty rigorous resolution timetables.

Of course in reality, this is what I'm actually talking about. I'm talking about consultants designing pipes through their own designs and, you know, craziness. So this is the kind of challenge that we face.

So initially the traditional Navisworks workflow that we all know and love was, we get a few Revit files, we put them into Navisworks and we run some clash resolutions. And that's great, that's a fantastic process. It's very fast and you can get reports out-- which we did-- straight into Excel, great. So why would we do anything else?

Well, when you started to have 60 models and over 50,000 clashes every two weeks, it wasn't taking a few hours, it was taking the entire week. And if you've already agreed to very stringent timetables for exchanging information with all the consultants, and then as a reviewer you're unable to give them any results back because you don't have enough time to do it, it tends to make a mockery of the system.

So we needed to find a mechanism to get quicker at what we were doing. And it's a very arbitrary, repetitive process. Well, obviously Dynamo was the way to go on this one. And so I'll let Thorsten jump in and start to explain how that works.

THORSTEN STRATHAUS: Thanks. So we've got 60 models and only a few days to do something. What I didn't say at the beginning, I am reasonably lazy. So I came to the office and got new task and a new project, and everything was fine, then I jumped through that hoop. The second time I was a little bit stubborn, so we said we have to do something about that.

So for example, getting these 60 models together every second week, to have the same arrangement in the Navisworks world than you have in your authoring world-- in our case it's Revit-- then setting it up again and again, got a little-- yeah. We said, we can do this a little bit faster.

So we kind of only threw together a bunch of few components in Dynamo, reading the Revit file which the architects in-house were using. And they had already manually linked in these files, so I didn't have to do that. We're just reading the information off of these documents which are linked in there.

And we use a Dynaworks plugin from Adam Sheather, a very cool guy from Malaysia, unfortunately not here. So I want to really get him some beers if I meet him for the first time, because he really saved us some valuable time, creating a plugin where you can access the API of Dynaworks.

I think it started a year ago and it's really going well, putting stuff together. So I only have to do six or seven components and I can automate the workflow of setting up the Navis file and queueing all the architectural files which we pull from the server. And then I save that Navisworks file and the first step of the part is done.

Then running through the federation report itself. We set it already, 50,000 clashes. Where arranging and organizing the clashes and assigning these clashes to the consultant is still a kind of very much a manual task, where you have to use your intelligence.

Where other areas of actually, later on, getting the information of the number of clashes in the clash reports, and who is a little bit behind or in front of schedule in how our other consultants perform for our monthly report to the client.

This can be done automatically, again. And so we start with basically four groups. We have number one. I could go through that.

MICHAEL HUDSON: Yeah.

THORSTEN STRATHAUS: Number one, we find two files. The first one is the Excel file, where we want to write the data into it. And the lower one is the Navisworks file which we set up before.

So on the very lower bottom on the left side you see that there's a Boolean true/false. It's a very nice feature of Dynaworks to be able to access the Navisworks API, to start an instance of that file, of the process in the background.

It doesn't come up, it's not visible. It's more stable, it's a little bit faster, and I prefer to have it like this. But then you actually lose a little bit the overview of, is the process still running? Where am I in my process of running through 50,000 clashes? Which will take up six minutes or something.

So you have to open the Task Manager. You see the process on the very top roamer.exe which still has 6% of CPU use. That's your feedback when Dynamo is done. Then the roamer starts and you get your data back to Excel.

Reading the data, basically you get the file, you get the clash detection internally, and then you have all the clash test. It's the first long list on the side. We've had about 42 different clash tests.

For example, the first one was architectural structure. And you then read out all the notes. Which are actually the clashes, it's just a different name. We want to keep it the same.

Having all the notes, we are filtering the notes for the status so we can actually know in which clash we have how many of resolved, or new, or active clashes.

And these are the same naming convention like you see on the right side. And the clash detector from Navisworks itself, what you are used to.

So you can actually see that, for example, the first clash test, which is architectural structure, we have 15,196 results clashes.

When you run through these 42 tasks, or 42 tests, analyze this and split that up and get the data a little bit remapped and reorganized to get it to the Excel, one might get carried away in doing it piece by piece for 42 things.

This is definitely not best practice. For sure, I only did this to show you how you shouldn't do it. I never did this myself in the beginning. And I never ran into massive problems with write data to Excel notes. Never happened.

[LAUGHTER]

So you can actually run through a little bit more clever regarding the data you are pulling and how you deal with list management. And the big thing to take away from this like before, list management is a little bit tricky. And there are options to use functions on lists where you wouldn't expect the stuff to happen because the naming is a little bit off.

If you come from other programs, like Grasshopper, you have a very different naming idea. And then you have to spend a few hours to figure out that there already is something, you don't have to write yourself.

And the upper area, the green one, it's the standard four components being used to create a range for this test file to show you here. We just have four elements in our list, 0 to 3. Not the 42. Running through, pulling all that data, getting it into separated lists.

If you want to reduce your code at some point, especially range is something very easy on the middle on the bottom you see in the red group. The design script syntax for range.

Super easy. So like the other one, you start from 1 to the count of the list, which is 4, minus 1. You have 4 elements.

You do it in the count and the design script syntax from 0 dot, dot is 2. Like direction, to count minus 1 and the steps is 1. It's one line, nice and-- yeah, sexy.

So you're already a expert in design script coding now. Then a little bit of remapping to get the data.

You might have heard of this before, like, a little bit of transpooling and splitting up lists to get it into the right format for Excel, because Excel is kind of only two-dimensional. And you have to split it up to rows and columns and then you're finally able to get the list of data elements over there.

You can see we have these three clash detections, clash tests from A versus S, A versus M. That's mechanical, architecturals, engineering-- electrical, sorry. And then you see the data pulling up there.

And that's how it looks in the Excel sheet. So sending this data to Excel is kind of only 10, 15 seconds. You have about 60, 80 variables which you put into the right cells. The sheet is a little bit prepared to see what we've got there. And you see that the results and the summed-up numbers actually are the same what we had before in the lists in Dynamo.

So we've done this quite a few times. The last one was federation number 17. But we are running over the time. Putting everything in Excel, having a look at it as a pivot table, quite helpful.

Where, at the same point, we are reading the data of these data table which we had before and created some automated tracker to actually see the results over time of the clashes. Like the last one compared to the one before. And can actually put a little bit of the finger on the consultant who has been a little bit lazy.

AUDIENCE: Are the [INAUDIBLE] from Navisworks or Dynamo [INAUDIBLE]?

THORSTEN STRATHAUS: That is just Excel graphs. So you pull over this data of one federation. Then you have several federations together. Not the [INAUDIBLE] table you see here, but just the same, just numbers.

And then automated graphs which always pull the last entries and give an overview of how the project is performing and where are we at. And sometimes you can see if, for example, we have raised, or we have decreased the tolerance for clash detection. It went down from 25 to 10 ml or something.

Or we had another consultant coming in, or when the MEP guys had the idea to copy around a few of their not-correctly-finished apartment layouts regarding for services.

MICHAEL HUDSON: OK, so the good thing about all of this was that we realized that we could automate and accelerate our processes so we could report information in a really concise way to the rest of the design team. And the design team went, that's not fair. Why can't we do that, too?

And we realized that actually, it was really quite easy to take the next step. So essentially, going back to that earlier diagram that I showed you is that, well, we've already got this kind of database that we're kind of manipulating and reporting into Excel.

Well, we've got all these IDs. So we could actually be commuting that information back into a Revit file. So I call this teleportation, because essentially the process, or at least the idea is, OK, well, we found the clashes. We've identified them, we've got reports and for resolutions.

Let's make it so that every consultant can find them in their projects rather than having to just go through reams of paper. So essentially, the first half of the process is the same as the earlier one where you're just manipulating the early clash information.

THORSTEN STRATHAUS: Reading the clash federation file, having the clashes in there. And in the background you already see something, some geometry. Looks like a wall. Might have a door opening.

The reason why we see that is because we read-- you can read these two elements which belong to a clash in Navisworks. And since we know we have the architectural central file in front of us, you can be sure that we can actually pull every information possible out of the Revit database of this one element which is from the file we are in.

There are actually already a few nicely programmed Python scripts, again. From people who are able now to access linked files. So you will never be able to write something back to a link file. But now you are already able to pull data from linked files for elements.

In this case we separated the two elements of the clash note, selected one of the interesting clashes in this situation. I can't read it. Clash 11,000-something-94. And on the upper right side you see that we have a Python note again, which was written by Dimitar Ivankov. It's a note I found on the package manager.

Actually, not package manager. It wasn't the plugin. In the forum. That's another great source of stuff, where experts actually from the developer team and experts of the profession all over the world post questions and answer your stuff. You can get solutions for your issues if you don't find it yourself.

Then we want the Python code to actually create a section box around the element because if you have 60 models in your Revit file open, and you see everything and you see nothing. So you have to focus on something. But you still want to have all information around that object. We are talking about the wall with the door opening.

So I was offsetting a little bit and illustrating an offset of a section box. And on the lower right, we just overwrite the element property of the color for that wall with the door. So one second later we created a new 3D view with a turned-on section box, with having a nice offset of 1, where 1 was feet.

Revit internally still calculates in feet. So I was thinking for millimeters, but no. Foot is good. Gives a little bit of idea what's happening around.

And the element of your interest is turned red. So we took out all these different file formats and PDFs and Excel sheets which somebody might have to look at. So our designers in the architectural aisle actually can just run through the clashes and turn on their complete focus on the element which has created the clash which was assigned to them to clear.

And they can use this and they're done with that.

MICHAEL HUDSON: OK, so after we did that for our architecture team, they said, can you anything else useful?

[LAUGHTER]

THORSTEN STRATHAUS: We weren't sure.

MICHAEL HUDSON: It took us a while, but we found one. And I'm sure you've seen this as well.

When you're developing projects-- particularly residential projects which have a lot of partitions and space calculations that need to be continuously updated-- you find yourselves often in an awkward scenario, where you either make a choice of having many, many models so that you can report room data in a siloed way, or you use rooms on phases in order to try and take advantage of, say, party walls moving around and recalculating internal spaces.

That's what we did on this project to try and make that work, but what that also meant was that there was no direct link of data between the rooms. Even though they're in the same file as each other, rooms are totally phase-dependent.

So the room on the left hand side, which is essentially one apartment, has no way of transmitting this information to the rooms that are actually contained within it. So the challenge for us was, OK, well, can we do this data manipulation in Dynamo?

And of course we could.

THORSTEN STRATHAUS: Yeah. So we have again a few small groups together there. In the visualization on the right, you see the apartment rooms as the geometry and a little bit pushed to the side manually. In the original file they are located where they should be. The much more smaller and differentiated rooms inside these apartments.

The first step was to read the file which we have open. And have a little bit of an idea, which was don't create it for-- You have to separate or differentiate between the apartments and the rooms itself. We did it by worksets. This was already set up in the project, we didn't need to push them to the worksets.

So I get information about the parameter, about the workset of a room. I pick one example room to be sure I picked an apartment. And then I can compare the different worksets these elements are on.

And I selected all rooms of one level to run through that. You'll see the list of the worksets which are like basically numbers. I didn't have the names. Revit hosts this data as a number. So I could compare that.

Since I already had separated the rooms from the apartments, I used a very interesting tool, a very interesting component which is called Room Is Point Inside.

So Dynamo can ask Revit if a point I give to Dynamo is inside a room. So I don't have a boundary box or something, I have the real shape of the room. And since I have the geometry of the rooms already inside the apartments I can pull the centroid, which is definitely d always in the apartment.

And I can super-easily filter this. It's a very fast function. And find out which rooms are in which apartment. Then a little bit of number crunching again, and who doesn't like that? That's the part where we get gray hair.

MICHAEL HUDSON: Or no hair.

THORSTEN List management-- Yeah, just pull that.

STRATHAUS:

List management, I can only stress it out once more. You have to look into that this stuff that's named differently and works a little bit differently.

I had a colleague from another company before that, who wrote like a tiny definition with 25 components to transpose data like we know from Excel. And then somebody told him that it's already there. Yeah. Have a look what you're looking for, ask somebody. Perhaps somebody might know already where the stuff is to find.

We rearrange the numbers and then it's super easy to pull a parameter, or a value of a parameter, from the apartment. The parameter is the apartment number and the value would be like 206 or 601 and write that stuff back to the chosen parameter of the room in the apartment.

This was before. This was after. Both plans in different phases used the same color scheme. And here we go. So you can only do this with rooms or whatever. You can actually get data from every object which is hosted in something, or on a specific level, and add that automatically by not having to do this manually.

If you have lots of rooms, or lots of doors or lots of equipment in spaces, for example, you don't need to do this manually. Let the computer do that.

MICHAEL HUDSON: Yeah, and exactly that. So we did the same thing with the doors. So this project, it has 300 apartments. It has almost 2,000 doors. It has 4,000 rooms. People have to renumber that, or at least they used to.

Sure, we're not going to lie, it did take a while to write the script and get it stable. But now every single residential project or office project we do, we just press Run and then it's done.

So there's a huge amount of time saving that can actually be implemented by putting these scripts in place. Sure, it's a bit of a challenge. But the thing is, I go back to my original point about the deliverables on our scheme.

This is a program called Solibri Model Checker. It checks IFCs.

One of the standard deliverables on information gates that you have to pass on British Level 2 BIM projects is that you have to have completely resolved data inside an IFC file. And for those of you who don't know what an IFC file is, it's a completely open platform BIM software. Software platform, sorry.

And thus we were confident that all of our rooms had all the correct data inside them because we'd taken, actually, the human element out of it. There was no manual entry anymore, it was all just automated.

OK, so let's move on to the final battle. Like in any superhero story, there's a final battle. And with us it's always acoustic design. The amount of data in acoustic design is considerably more than anything else.

And the project that we have been working on this year is, we've got a lot of work in Wembley. And this is the Wembley Theatre.

And the interesting challenge for this was they wanted to create a completely open foyer which would provide multiple functions. So there would be large gatherings of people entering the actual performances, but also they wanted it to be a nice restaurant and also a gathering place for the local residents.

That meant that the acoustic performance of the actual space had to respond to a number of different criteria. Which is different from any of the previous workflows that we've done, where we've purely been focusing on the transmission of a performer.

In fact, we had a number of challenges. Because the other challenge was, it's a traditional kind of ceiling space that we had. So unlike traditional geometric form finding kind of uses that we'd done previously, we were limited in what we could actually do geometrically.

So we have to start thinking more creatively about how we optimized the acoustics of the space. And just to give you an idea of the spaces, there were quiet dining areas and in the same space there would be large congregations by a bar.

So obviously, Dynamo was our place to go to start trying to think about how we could optimize a solution with essentially a standardized ceiling.

THORSTEN STRATHAUS: Yeah. So we see the geometry. We are not going to bore you with geometry anymore, we are going to invisible stuff. The sound rays.

We have 3x8 modules which usually have 16 panels. This Dynamo definition can actually create that, but that's not the part you're looking into.

The office created a plugin called Acustamo, which is available on the package manager, which basically consists of three components. The first one is to create a number of random sourced rays, like a straight line which will be considered as a part of the wave.

At the source when it makes the sound, then we've got the directed rays. So we can set up a number of sources which have a number of rays in both directions, x and y. x and y are u and v. And directed a surface.

For example, an acoustic panel which could be hanging here instead of that thing. It might matter to get this dust off.

And the sub1 is actually to solve the raytrace using the vectors we created before, and running the analyzers and having the sound bouncing back. Like if I speak over here, it's straight. If I would speak over there without the microphone the sound would be bounced back from the side to the back, and the person over here hears me twice.

So we had a very simplistic setup. First five sound sources randomly put in this big space and 25 rays per sound source. On the top you see the panels. And actually you already see the created rays and bouncing rays on the lower left side. It didn't take too long, but wasn't really what we were looking for.

So we said, we will at least put in like 10 sources, which could be considered as flocks of people, each having a talk in a restaurant area or dining table or standing at the bar. And we set that up to have 250 sound rays per source, and then actually the computation time went up a little bit. So we arrived at six hours.

So do not leave your Dynamo graphs on automatic execution at that point. There was a day where I'd forgotten about that and had bad luck. And I had to restart like 15 times in 15 minutes. Something crashed, and--

So we have these 10 sound sources, multiplied by 250 rays, which have to be directed to 8x3 panels with each 16 pieces. So you have roughly about 960,000 calculations which have to be done, and it took these six hours.

So it sounds a lot. It's not. Because these three components of Acustamo actually have been developed properly in a C# environment using multithreading.

So what you see on the bottom actually is a very stretched out Task Manager, where I had like an overkill demo graph running, where you see that all of the other seven of your eight CPUs is hanging around and not doing anything while you still pay a lot of money for it.

It's multithreading, and since you have a very stable task in this scenario because you can run all of these functions, like creating a ray from a point being reflected from a surface. Nothing does change. So it's a very stable one.

You could, for example, measure a distance of 1 million of points of a point cloud to a specific point in the room. It's very easy but computational-wise intensive function. But it's stable.

If you were to take on the other side, I need to punch 1,000 holes into one surface, you couldn't multithread this. Because the surface will always be recreated, so the first process has to run through, second process has to start.

Where if you try to do this manually in the multithreading way, you would most probably fail and your surface wouldn't have all your holes necessary.

The same stuff, like multithreading in C# is called parallel computing in Python. Since you're already experts in Python you can use the parallel computing function of IronPython and put that into your Python code in Dynamo. It works in Grasshopper, it works in Dynamo.

And then you can actually, finally, once a week have the benefit of having eight CPUs in your computer.

So analysis result after six hours. Standard legend from green over yellow to red, showing the panels categorized for amount of absorption which we actually would like to get in this ceiling. To make sure that there wouldn't be areas like, for example, around the bar where everything is too loud and you just can't hear yourself, like in an English pub. I would love to get this into the pubs.

[LAUGHTER]

So a different graph just for these holes. We discussed that. We're not going into detail for this stuff. Just for you to have a look. There's a layout and we actually do not only have the one layer-- let me re-setup again.

So there's a definition for all these modules, which is creating analysis and putting these values into a database. There's a package called Slingshot where you can create data rays and store data and this stuff.

It's a kind of a lookup table, if you're familiar with this terminology. So we're not going to look into that stuff, not too sexy. So we are going back to the general idea.

We had an overlay of, let's say a nice pattern. Because you still want to see something nice if you're standing at the bar. A generally nice pattern plus the overlay of the extra analysis data which we've got back regarding the necessary category of absorption that the panel would qualify for.

And the attractor field which we used to create the general pattern, once again, is a Python code by--

[? MICHAEL HUDSON: Knight. ?]

THORSTEN STRATHAUS: By [? Knight ?] Miller, who was here just before us. So you can grab these codes, tweak them a little bit if you need, or just use them as they are and be happy with that.

So then we used adaptive panels in Revit which pull data alike from a lookup table to conform the holes about the size they need to be. For one panel on the left and 16 panels together for the model on the right.

And then we had laser-cut scale mockups of, like, 25x25 centimeters. And it was, like, 2,000 holes done in about one hour. And the whole manufacturing process of these mockups took like half a day in total. And this was something to be discussed with the client very fast.

MICHAEL HUDSON: OK I'll rattle on, I know we're quite tight for time.

Like every sequel, they always shamelessly plug the third movie, so I'm going to do that very quickly. There's always an exotic location. It's Poland.

Basically, continuing on with our acoustic optimization, we tried to really look at enclosed space and actually provide for 1,000 people under one tensile roof structure. It's actually a ring beam.

But once again, we went back to our old friend Optimo, which is a brilliant piece of coding. Actually, it implements a non-elitist genetic algorithm for multi-objective optimizations, which sounds like a very complicated idea. And it is, essentially.

What it allows Dynamo to do is, once we're doing the stable processes ins instead of doing multithreading, you also need to have some mechanism to essentially do random evolutions to get an optimal result.

It's all well and good knowing your result doesn't work, or it's not very good, but then you have to guess. Well, why not automate the guesswork as well? Essentially that's what this does. Non-elitist essentially means that it tends to be a bit more random, which I think is--

AUDIENCE: [INAUDIBLE].

MICHAEL HUDSON: Andreas, can you answer that one?

ANDREAS: What was the question?

MICHAEL HUDSON: They said, what's the difference-- sorry to repeat your question-- what's the difference between non-elitist and elitist genetic algorithms?

ANDREAS: Yeah, that's above my--

[LAUGHTER]

MICHAEL HUDSON: Whatever [INAUDIBLE] afterwards actually [INAUDIBLE].

Essentially what happens is you have a number of iterations generated. And then you have a graph that's called a Pareto front. And essentially what it allows you to do is, instead of doing hundreds of thousands of essentially linear guesses, it does a lot of guesses on a wide array and then cross-evolves them to find the best result.

This particular algorithm is particularly effective. So that means that we got our first attempt good enough, best result, in actually five iterations, basically. And it means it's incredibly fast.

However, our challenge was to deliver actually high quality, opera-standard internal acoustic space. So what we're now looking to do is actually develop with another open source piece of software called Pachyderm. That's also on GitHub if anybody's interested.

Basically what that allows you to do is go one step further and start using real sound. So rather than just doing a generic sound source, what we do is we can then break that down into frequency ranges and apply genuine materials which have different levels of absorption for material, crucially.

Because certain frequencies travel less distance or more distance depending on their frequence. And then they tend to bounce better or worse off different materials. So that really is a game changer, but it also means that the calculation process is incredibly intensive as well. But it means that you could really, really fine-tune the quality of a space with a great deal of accuracy.

So that's where we're at, and that's what we hope to present next year. We'll probably need to buy some faster computers, though.

Thanks for your patience, everyone. I know we're slightly overrun here.

But we're hoping that just by seeing this-- and, frankly, all the other great classes-- that you've come to the conclusion that actually having some coding skills is going to be a huge thing for every company in the AC industry.

Being able to manipulate and accelerate the way that you're solving problems is huge. And essentially letting your computer to do what you paid for it to do. Rather than having some poor soul filling out stuff manually. And hopefully then you can be a Dynamo Hero too.

Thank you, everyone.