AECTECH: Easily build collaborative apps for VR and 3D with Alloverse

Abstract:

Working with three dimensional data is hard, especially in a computing world built for 2D graphics. Building software for working together is also really hard.

Learn how to build real three-dimensional user interfaces (for VR or flatscreen) on a platform for productivity software rather than game tech, where your data and controls can be seen and used in real time by multiple people at the same time, as if by magic. Nevyn and Tobias shows you their vision of the future of software.

Presentation:

210705 AECTECH Presentation	https://docs.google.com/presentation/d/1WjgcwH1qgPeyxresOhz_bOczZKDFDBTiW5G-hQbM6_A/edit?usp=drivesdk

Todo

- [x] ~~Write the script~~
- [ ] Make the slides
- [x] ~~Make sure C# SDK works well enough for demo~~
- [ ] Fix audio volume dropoff so we can be heard 😅
- [ ] Run a test to make sure we can actually present like this

Overview

- Practicals
  - Presentation taking place inside Alloverse itself

    - Screen shared from desktop
    - Tobi, me and PDF viewer running to do the presentation
    - Audio is computer audio + an extra microphone (which starts muted)
    - When doing live coding, resize Alloverse window and put VSCode and Terminal next to it and nevyn goes out of VR and uses headphones+mic instead.
- Content
    - Intro and dazzle (with a cool app or two showing the basic concept)
    - Introduce ourselves
    - Vision and magic
    - Live coding (in C#?)
    - Demo of apps and...
    - Demo of shapediver!!
    - Call to action

Presentation script

Nevyn

Tobi

Emil

Intro and dazzle

<slide 2>

Hey there! I'm Nevyn Bengtsson —

and I'm Tobias Kask. —

We really appreciate being invited to AECTECH to talk about our vision for the future of computer interfaces, and to show you what we're building here at Alloverse. It's a really different system from what you might be used to, so we figured the best way to explain it is to show you.

What you're looking at is Alloverse running on a flat screen PC being screen shared into the call, while Nevyn and I are inside Alloverse in VR, connected through our Quest 2 headsets. Note that that means we can't hear you, so we'll save questions to the end.

<slide 3>

Alloverse is:

- a collaborative workspace — which means we can be multiple people in this place, talking and working together;
- and an open source app platform — which means that software specifically written for Alloverse can run as 3D objects in the same space as us.

Let's try it out before we go deeper into what all this means. This place is pretty empty. Let's pull in an environment — I like this A frame house. Here's a clock on the wall. The slideshow behind me is also an app. And here's a whiteboard we can draw on together.

What makes this different from other platforms is that these are all separate apps running on the internet, and written in different languages and environments. Anyone can extend Alloverse with new functionality, which can be pulled into VR with us and used together with the other apps.

One really exciting such app is Emil Poulsen's ShapeDiver viewer app for Alloverse. Here's Emil to show us.

<slide 4>

Hey there! ShapeDiver Viewer presents parametric meshes in real time in 3D and VR in front of you, and parameters can be modified in real time. This was hacked together during a few morning hack sessions with surprisingly little effort given the complexity of what's going on.

My main takeaway is that Alloverse is a tool built for the coder; something that is easy to get started with, and very quick to iterate on. It's perfect for quick prototypes and experiments in 3D because you can see your app running in real time and collaboratively.

Introduce ourselves

<slide 5>

Before we go deeper, let us give you some background on who we are and why we're doing this. Alloverse is a team of six people. I'm Nevyn, and I've been an iOS developer for over a decade. I've worked at Spotify for four years, building large parts of their initial iPhone and iPad apps. I went on to found the UX research platform Lookback for another seven years.

<slide 6>

I believe in direct manipulation as an interaction method, but after having been disillusioned with products such as the iPad as a workstation platform, I'm fully on-board with VR and AR as the future of computing. I believe that the move to spatial human-computer interaction is as inevitable as the move to smartphones and multi-touch was. As we build new platforms for new software, we have the chance to correct some mistakes in the past decade of computing, and get back to the core values of the open web.

<slide 7>

I'm Tobias Kask. I've been working as a UI/UX engineer my whole career. Mainly for a US dating app, but also on a variety of other projects before and after. Outside of the 5 years I spent working from the company's offices in San Francisco and New York, most of my time has been spent working remote.

<slide 8>

When I began 15 years ago, "remote collaboration" meant receiving an email from "collaborators" in the morning, and sending one back as the work day ended in the afternoon.

Now, remote work is surging, and more sophisticated collaboration using tools like Zoom, Slack, Google docs, Miro, etc is a daily occurrence. This is way better, of course.

<slide 9>

But what's the next big thing? Surely, our kids aren't gonna bend their necks and squint their eyes at the phones and laptops of today, right?

Like Nevyn, I'm convinced the move to 3D and spatial interfaces for collaboration isn't just inevitable - it's also coming soon™.

<slide 10>

- VR is really taking off. Social VR is really really taking off. Working in VR is really, really, *really* taking off. It feels like every week a new competitor pops up.

<slide 11>

- Tech is rapidly getting cheaper and better.
- There's a reason 2D interfaces are heavily influenced by emulating real-world 3D interactions, from "radio buttons" to skeuomorphism to modern haptic feedback. Interacting with things in 3D "just makes sense" for us - for obvious reasons.
- But the really exciting future is when real AR headsets start coming out and becoming mainstream, and start replacing the smartphone as the de facto computing platform.

<slide 12>

VR is as a great place to explore spatial computing and building the infrastructure for how we will use computers in the next twenty years.

With Alloverse, we're given the opportunity to invent a completely new way to interact with data - and each other. And with it comes new opportunities and future standards for communication and collaboration.

So what does Alloverse want the future of remote collaboration to be like?

Vision and Magic

With Alloverse, we can make sure the core values of the open web gets translated to spatial computing: open standards that anyone can interoperate with; and self-publishing instead of being locked into commercial and walled gardens. Anyone can build your own Alloverse app in any language, and it can be running anywhere on the internet.

<slide 13>

Our goals for Alloverse over the next couple of years is:

- Work inside Alloverse (kinda like we're doing right now!) using screen sharing and other coder- and designer companion tools)
- Build a marketplace with subscription payments, hosting etc
- Work with domain experts to build useful apps (architecture, data visualization, etc)
- Become the open and free standard for XR collaboration, preventing Facebook and Apple from owning a centralized walled garden ecosystem for app development

All of us who get a start in VR now, will be ready to transfer all of our skills and software into this AR future, and be the pioneers of human-computer interaction. That, to me, is incredibly exciting.

Now, enough talk - let's write some code!

Live Coding

<slide 14>

Our main app building language is Lua. However, Alloverse is usable from any language, and we've built UI libraries for a few languages we really like. We've understood that AEC favors C#, so we have a beta of a C# library which was used to build Emil's ShapeDiver viewer. I'm going to jump out of VR now and join you in 2D to do some live coding to show you how easy it is to get started.

This guide is available at docs.alloverse.com/csharp, so if you want to run through it at your own pace, you can do that.

I'm a Linux nerd, so I'm going to use the dotnet core command line tools. If you're more of a Windows person, Emil tells me you can do all these steps through the regular Visual Studio user interface.

Let's start off by creating a new console project, and then add AlloUI as a package dependency.

Now we can just open up Program.cs and code away.

There are three initial steps before this project will run:

1. Create a client and and an app, so we can maintain a connection to an alloplace server.
2. Setup some sort of initial user interface
3. and finally connect and run our app.

Writing the UI should feel somewhat familiar. It's a tree of view objects, like a DOM tree or an iOS UIKit view hierarchy. The difference is that it's 3D, and you suddenly have one additional dimension to account for. This is new for most people, but I imagine this crowd has an extra good grasp on their third dimension.

Let's do something simple. Let's make a cube that the rest of our UI can rest atop. All UI components are created with a Bounds object, which represents the position and the size of the object. This one will be a meter wide and tall, and a decimeter deep. Then we'll move it a meter and a half up from the ground, and two meters into the scene, so that when it launches, it launches right in front of our eyes.

On top of the cube, we'll add a button. addSubview lets us layer it on top, so that the button's coordinates will be relative to the cube. The origin of all objects in Alloverse is in the middle in all dimensions, x is horizontal, y is vertical, and z is depthwise, with positive Z facing towards the user. We'll need to move the button depthwise so that it's on top of the cube instead of inside it.

We'll add a label, and a callback for when the button is pressed. In this simple delegate, we just print to the console, or... actually, let's randomize the button's own color: `button.Cube.Color = Color.Random();`.

Finally, let's make the cube grabbable so the user can move it around.

And there we are! Let's run it...

```
dotnet run alloplace://sandbox.places.alloverse.com
```

and go back to VR to have a look at it.

OMG, it's a working UI! I can now make adjustments in real time, and both me and Tobi can see them happening in front of us.

This project could be importing heavy-weight libraries for AEC, or talk to other servers to do complex calculations and then visualize them to the user. In neither case will the user's computer or headset be burdened by the computation, and a lightweight visualizer can visualize really complex data.

Finale

Thanks for that run-through, Nevyn! That's all the content we have for you today. If you have any questions, we're now heading out of VR so we can answer them.

<slide 15>

— qa —

<slide 16>

Thanks from all of us at Alloverse for tuning in!

If you're interested in learning more about Alloverse, please join our Discord channel, or check out alloverse.com to download the desktop client.