

Contents

1	Xplane Streaming project with Combination of my PhD	1
1.1	The short term plan:	2
1.2	Invested amount	3
1.3	Released to market	3
1.4	Invested projects	3
1.5	Release plan:	3
1.6	Business plan	3
1.7	Intermediate Release plan	4
1.7.1	P2PRC release	4
1.7.2	Flight sim work	4
1.7.3	Finance	5

1 Xplane Streaming project with Combination of my PhD

The Xplane WebRTC project takes inspiration from Google Stadia for a streaming based solution for playing video games. But our plan is to build one specifically for flight simulators. We target Xplane 11 for being cross platform and completely offline dependent on sceneries with reasonable documentation on usage of the SDK to use the game. The novelty of the project being all aspects of the project will be completely open source. There are already major segments of the project complete and have been tested which will be mentioned in the appropriate section for the necessary details. I am also investing in my PhD to familiarize myself to work with slim down kernels on HPC scenarios (This will help me extend the Xplane to run on a distributed scenario). This project is fun long term work and has no heavy deadlines but rather the art of optimizing the current paradigm of how we use heavy workload applications. The following above is a really high level abstraction description of the projects and barely covers the depth of what the project intends to push forward.

The big question is what parts have been rebuilt for the project and how they contribute to the end goal of the project. The first property of the project is that it should be able to run on a p2p network. P2PRC is a p2p orchestrator designed to run applications in a p2p network using Containers initially. There are plans to extend it to run on Uni-kernels and based on my PhD extend it to run on a Multi-kernel paradigm as well. This will be our custom alternative to Kubernetes which will be used to distribute and

run on workloads on p2p effectively. This would make our entire run on Anyone's machine which can reside behind NAT.

We also intend to make an open source solution to distribute a slim down version of X Plane so that nodes can quickly Spawn Xplane instantly with only the required scenery needed. This will be in contrast to running the full scenery of the game which is around 55 GB. Something novel that could be worked on here is a novel approach to only send the scenery of the flight path when distributing the game. This will mean building parsers for the Xplane scenery files and then finding techniques to only get a correct set of scenery files needed in other machines. The techniques are expected to open source but since the scenery files are proprietary they are expected to be public.

The streaming part of the project is expected to use the browser standard WebRTC sockets with the corresponding sockets. This is because there is already massive development of the chromium browser with GPU encoders and decoders for faster performance. We have already built a prototype which has been tested and seems to work as intended.

The PhD will be one of a long term novel approach which will support Multi-kernels with TAG based architecture support for running C++ programs more securely. This might mean most parts of the PhD might not be used. The Multi-kernel approach is definitely an interesting area to experiment on to figure out how the project would use such an approach and this open lot of areas of future research and hopefully better performant flight simulators with better purposed algorithm to offload tasks to devices such as FPGAs or potato machines in abstraction layer similar to speaking nodes in an network.

1.1 The short term plan:

- ☐ To do a full review of all the work currently complete.
- ☐ To set up clusters for testing Xplane 11.
- ☐ To set up nice documentation for setting up the Xplane 11 project.
- ☐ To allow players to play around with the Xplane 11 game as Beta testers.
- ☐ Plans for constant updates for the project broadcasted to the community.

- To sync with developers interested in the project and track how much potential contribution is possible from an external community.
- Start planning ways to incentivize developers interested in working on the project.

1.2 Invested amount

- Akilan & family (60,000 pounds) - Research and Development

1.3 Released to market

- Market release (March 2026)

1.4 Invested projects

- P2PRC (p2p orchestrator)
- CHERI based memory allocators (Research for fast C memory allocators to be injected to Xplane)

1.5 Release plan:

- Testers version for Xplane single instance to selected people (1st March 2025)
- Multiple instance of Xplane (1st April 2025)
- Xplane UDP communication protocol (1st June)

1.6 Business plan

The business plan for the flight sim project is initially self host versions of the game where a user can pay 4\$ a month to try running their game setup on our servers to start with. We plan to gain traction through our open source to create a framework to allow more developers to spawn companies on top of the project such as:

- (1.1) White labelling to Airline compaines with maintenance as apart of a formulated contract.
- (1.2) Adding servers to the network to generate income (With P2PRC)
- (1.3) Legal firms to follow FAA rules to be built on top of (1.1)

- (1.4) Flight schools on-prem setup with remote learning.
- (1.5) Researchers using the platform to test flight models on the simulator.

1.7 Intermediate Release plan

This release plan looks into 3 parallel segments of releases. The one which is P2PRC release plan, PhD work release and flight sim release plan. We intend to have this ready by the mid of march to present to researchers in Leeds and Andre from Lyon.

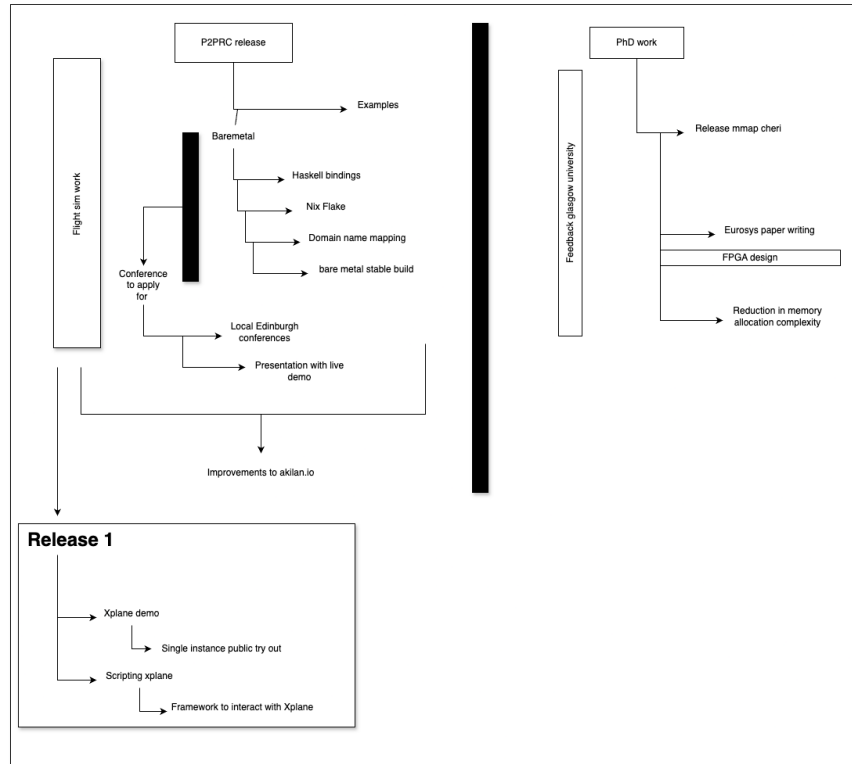
1.7.1 P2PRC release

The P2PRC release consists of work done by me (Akilan Selvacoumar) and Jose Fernandes. This will be official task orchestrator to run Xplane accross multiple machines. The main focus to get stable release out of the way which would focus on the Haskell bindings, Nix Flake, Automated domain mapping and a stable framework to deploy tasks on a baremetal build.

On the basis of this work we will apply to local conference allowing users to experience using the project on the fly as we talk through our demo giving a magic element to automation we have done.

1.7.2 Flight sim work

We will start resuming work on flight sim project by starting to host previous work done such as remote game play. We are also focusing on adding remote keyboard and mouse inputs embedded to the browser with a farmework to stream data to a database. We will build everything module by module which will be language agnostic since we will have to swap out a lot of modules such as the Go implementations to a inherently strongly typed language such as Haskell while keeping the performance sensitive parts in C++.



1.7.3 Finance

This is a expensive endeavour to fund but worth self financing due to freedom of control required in decisions. The industry is in shitshow and requires strong leadership from domain experts with a technically good track record.

This means in laymen terms I (Akilan Selvacoumar) will be fully finance intially this project on whatever I can give in (Time and required resource with the limited finance I have). As pointed out in figure 1.7.3 most of us savings from part time work and family funds will initially grow this project.

Finance

