

FlightGear Architectural Enhancement

CISC 322 W24 – Group: Chicken AI

<https://youtu.be/OVMbq25nKvI>

Team Members

Team Lead: Derek Youngman

- Current State of System, Effects on NFRs, SEI SAAM Analysis

Group Presenter: Marion Anglin

- Presentation, Conclusion, Lessons Learned

Group Presenter: Shrinidhi Thatahngudi Sampath Krishnan

- Presentation, Risks, SEI SAAM analysis

Team Member: Akash Singh

- Abstract, Introduction, Testing Plan, Impacted Files/Directories

Team Member: Abbey Cameron

- Enhancement Proposal, Low-Level and High-Level Architecture

Team Member: Ximing Yu

- Use Cases



Our Proposed Enhancement & Motivation

FlightGear users feel there is a lack of realistic emergency landings & realistic crash simulation.



Plane Crash Simulation Feature

- Visual effects and animation of the plane crash (parts of plane breaking off, more detailed rendering of environment, etc.)
- Fully realizing water landing, as it currently isn't modelled differently from land landing
- Model instrument failure
- Crashes would be more dynamic & varied, effects would depend on the nature of the crash
- User is provided with feedback on their flight, allowing them to reflect and learn



Presentation of Two Approaches to Realize Our Feature & SEI SAAM Analysis

Two Approaches

Connected Approach:

- **Focus on flight physics & flight control**
- Increasing connections between already existing components to ensure accurate behaviour of aircraft components in critical situations (impacts, structural failures)
- Interactions with other components, such as aircraft, UI, sound & visual effects subsystems

Component Approach:

- **Focus on the graphics of crashing**
- Addition of a new component that takes in basic info about the crash, then uses Pipe & Filter processing to produce extraordinary graphics
- Focus on visual effects of the impact of the crash & the visual partial destruction of the plane, with less focus on how the plane destruction impacts actual flight

Most Important NFRs for Stakeholders

Stakeholders	Primary NFRs
Gamers	<p>Performance: Gamers would want the crash feature to be fast and responsive with satisfactory visual effects.</p> <p>Accuracy/Realism: Secondary concern, some gamers appreciate realism.</p>
Pilot	<p>Accuracy/Realism: The feature needs to be as accurate as possible to be useful and realistic to the pilot.</p> <p>Modifiability: Pilots would like to be able to modify the behaviour of the aircraft depending on the type of plane they are flying or the scenario they would like to practice.</p> <p>Performance: Pilots need their inputs to the system to be responded to quickly to maintain a realistic experience.</p>
FlightGear Developers	<p>Maintainability: Developers want the software to be as easy to maintain as possible.</p> <p>Testability: Ease of testing is positive for developers, to produce bug-free code.</p> <p>Evolvability: Developers would like to maintain the ability to add new features.</p>

The NFRs in Each Approach

NFR	Connected Approach	Component Approach
Performance	Possibly lower in terms of graphics, but reasonable in terms of flight control.	High speed rendering.
Accuracy/Realism	High. The constant interaction with all of the already realistic components of FlightGear can provide a realistic crash experience.	Lower. In this implementation, the “wow factor” of the graphics is prioritized over realistic crash simulation.
Maintainability	Little change. Since FlightGear is open-source, developers who are interested in this feature can freely focus on this feature while other developers focus on other features.	Little change. Since FlightGear is open-source, developers who are interested in this feature can freely focus on this feature while other developers focus on other features.
Testability	Lower. The large amount of new interactions and scenarios present many things to test.	Higher. Graphics present less new different complex interactions and scenarios.
Evolvability	Slightly lower. More interactions hinders evolvability in the long run.	Slightly higher. Less interactions encourages more evolution.

Pros/Cons & Conclusion of SEI SAAM Analysis

Connected Approach

Better accuracy / realism.

- More suited to pilots as accuracy / realism is their most important NFR

Component Approach

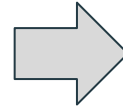
Better performance, testability, evolvability.

- More suited to gamers, because performance is very important to them
- More suited to developers as maintainability, testability, & evolvability are important to them
- Drawback of lower accuracy / realism of crash simulation feature doesn't detract from FlightGear's otherwise realistic simulation & any newly provided crash simulation capability is more realistic than current lack

- Complaints found online don't come from pilots who use FlightGear for pilot training, so makes sense to add crash simulation feature with goal of attracting gamers, especially given the benefits to the developers



Component Approach



will bring greatest benefit to largest number of stakeholders, while also not detracting FlightGear's value to other stakeholders.



Effects of Enhancement on the Architecture

High-Level Conceptual Architecture Effects

- Changes to the subsystems within the simulation subsystem (Flight Control (Aircraft, Cockpit & Instruments), Environment & Scenery, Flight Dynamics Model subsystems)
- Connected approach doesn't alter HLA, but Component approach does, however the system would still have a repository and object-oriented style
- Depending on which approach is chosen:

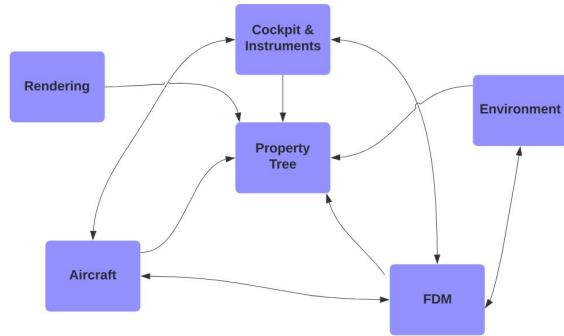
Connected Approach

- Use previously existing subsystems & add further connections and enhancements to these
- More detail in Low-Level Effects slide
- Wouldn't alter high-level architecture

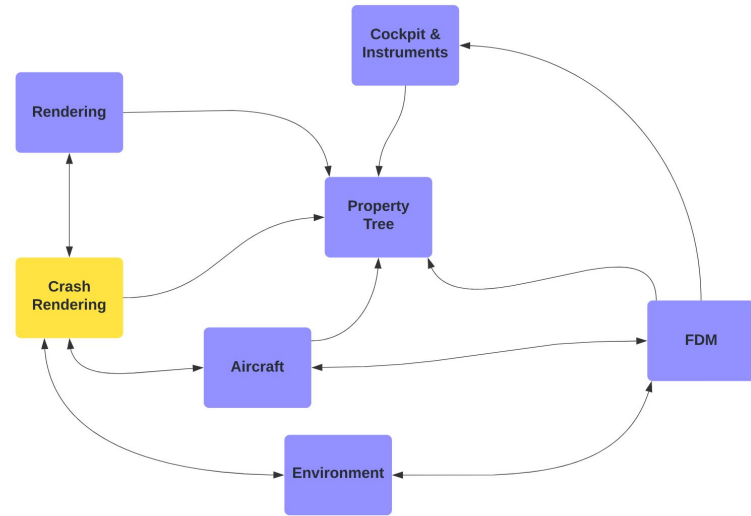
Component Approach

- Adding a new component to the simulation subsystem that entirely handles crash simulation
- New component would interact with Environment & Flight Control, Rendering subsystems
- Mild HLA alteration due to new component

Approach Diagrams



Connected Approach Interactions



Component Approach Interactions

Low-Level Conceptual Architecture Effects

Aircraft Component (both)

- Newly added damage model used on different aircraft features to consider how the damage taken affects the specific area, flight controls, and overall aircraft performance, realistic damage
- Ways for different elements on aircraft to break apart, both in sky, and when interacting with different features in environment

Cockpit Component (connected)

- Works with adapted aircraft features to create more realistic experience
- Example: if a part of the aircraft breaks due to collision, not all elements of plane in cockpit & instruments may be available for use anymore, & user would need to adapt

Flight Dynamics Model Component (connected)

- FDM calculations & controls affected by environment / weather collisions
- Any external force that will interact with aircraft to cause a crash will be sent through the FDM for a recalculation to alter the flying of the plane

Rendering Component (both)

- Enhances graphics in the simulation, which allows FlightGear to appear more realistic & further enhances user's FlightGear experience
- Creates a more realistic environment look, when user crashes into area, & in general

Environment Component (both)

- Enhanced detail on all scenery, allows user to get closer to obstacles & collide with them

Effects of Enhancement on...

... Maintainability? Minimal.

- Due to FG's open-sourced nature, developers of our enhancement would likely be those personally interested in it & would also be interested in maintaining it
- But developers could lose interest over time, resulting in poor maintenance

... Evolvability? Marginal.

- As FG is already very complex & interconnected, adding our enhancement wouldn't have any huge effect on evolvability
- But there is a limit to how much can be added without having to re-engineer the architecture

... Testability? Some.

- Adding our enhancement creates many new situations to test, difficult to provide exhaustive testing for all of these new & different scenarios
- But FG is open-sourced, so there are lots of people who could find new bugs over time & address them

... Performance? Some.

- Rendering and modelling crashes can be computationally expensive, so performance may be lacking, but only during crash simulation
- Newly added component in Component Approach could be engineered to ensure performance impact isn't too great



Risks & Plans for Testing

Potential Risks Due to Enhancement

1. **Performance & Stability:** increased computational demands & potential bugs could affect overall usability of FlightGear
2. **Development Complexity:** significant amount of developmental effort required, which could affect all resources and time devoted to other important aspects of FlightGear
3. **Compatibility & Integration:** an integration that is compatible across different platforms & hardware configurations is essential for a smooth user experience, but this seamless integration of crash simulation into already existing complex code frameworks & UI could present technical issues
4. **Documentation:** when new features are added, maintaining detailed documentation becomes crucial, so framework should be well-documented to facilitate knowledge transfer among the developers & ensure that future contributors to FlightGear can understand and modify the feature effectively if needed
5. **Evolvability and extensibility:** the code for the crash simulation should be able to evolve easily and extend if need be
6. **Scalability:** scalability issues may arise when simulating crashes with large aircraft or complex environments

Plans for Testing the Impact of Interactions of Enhancement with Other Components

Functional testing: unit testing (black & white box testing methodologies), ensure proper addition of new features throughout the development lifecycle & verifies that all interactions with other features occur as intended

1. Black Box Testing: testing a system with no prior knowledge of its internal workings, provide input & observe output
2. White Box Testing: checks internal structures/workings of an application, unit testing of subsystems, regression testing

Black Box Testing

Test a simulation within FG & compare the simulated crash outcome with expected behaviour.

Examples:

- Does the simulated crash accurately reflect real-world physics & aircraft behaviour or real-world visuals of a crash?
- Does the collision detection system accurately detect collisions with environment obstacles, like buildings, mountains, or bodies of water?
- Does the rendering system effectively render realistic environments & aircraft damage effects without causing any visual glitches or performance issues?

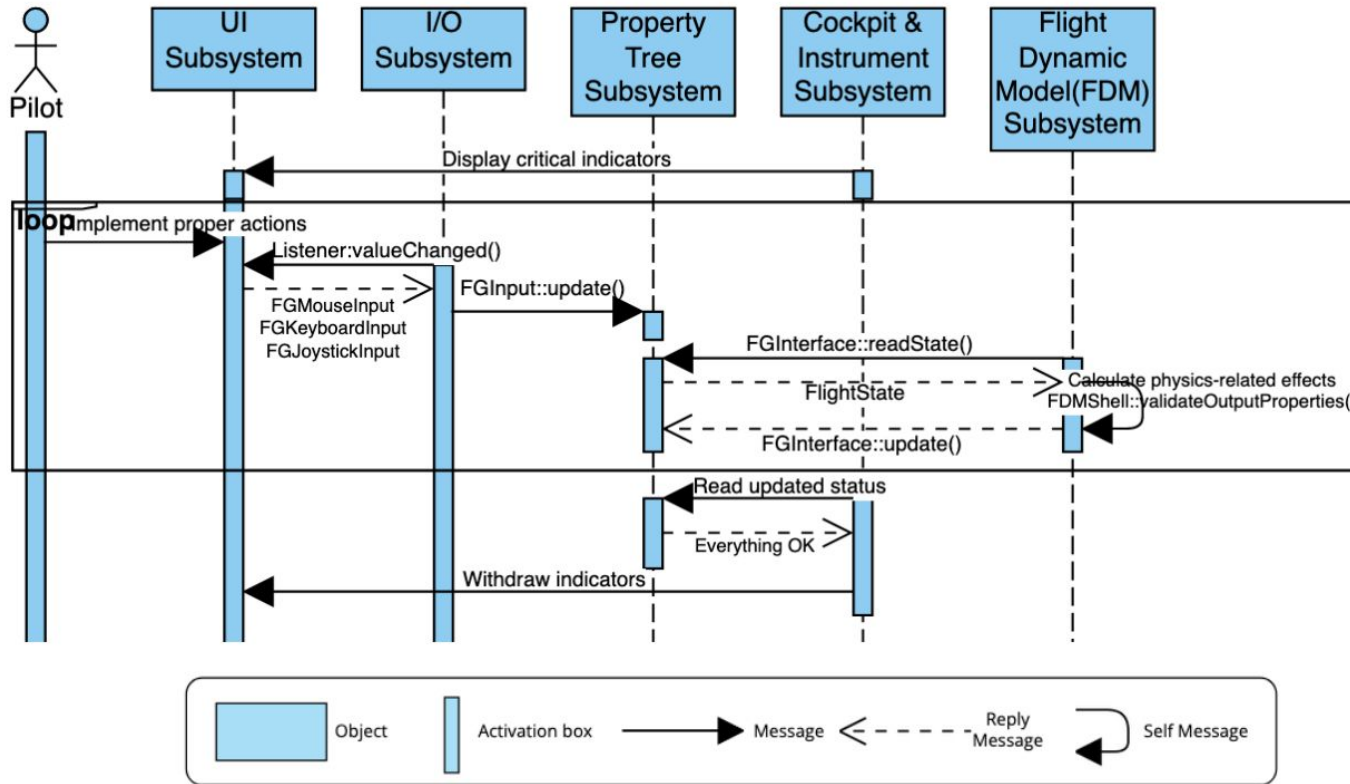
White Box Testing

Code coverage analysis. Verifying all relevant portions of the crash simulation algorithms ensures comprehensive testing of the underlying logic.



- Inspect codebase to verify that Environment, Aircraft, Cockpit, & Rendering components are appropriately modified to accommodate Plane Crash Simulation Feature
- Testing implementation of damage model within Aircraft component to ensure accurate simulation of damage effects on various aircraft parts & flight controls



Use Case



Sequence Diagram for **Pilot Experiences Mechanical Failure & Manages to Stabilize Aircraft**



Conclusion, Limitations, & Lessons Learned

Conclusion

- Due to FlightGear user demands, we proposed a **Plane Crash Simulation Feature** enhancement
- **Connected Approach** and **Component Approach** to implement this enhancement
- At a low-level view, various components would have to be adjusted to ensure functionality of our enhancement in both approaches
- Following a SEI SAAM analysis, we determined that the **Component Approach** would fit the stakeholders non-functional requirements best
- To ensure our enhancement functions properly after it is implemented we propose **functional, black box, and white box testing**
- Risks of implementing our enhancement in FlightGear

Limitations

- Using Component Approach to implement our enhancement may have an adverse effect on accuracy / realism of our enhancement
- Not satisfying all stakeholders

Lessons Learned

- Creative process can't always be performed from scratch, sometimes additional information, such as FlightGear's forums with user comments, is necessary
- It is almost impossible to find a solution that satisfies all stakeholders and their most important NFRs, prioritization and compromise is necessary in a real-world system

Resources

- [1] Crash is Not Realistic Enough - FlightGear Forum: <https://forum.flightgear.org/viewtopic.php?f=49&t=16330>
- [2] Emergency Landing in Water - FlightGear Forum: <https://forum.flightgear.org/viewtopic.php?f=18&t=10290>
- [3] Crash Simulation - FlightGear Forum: <https://forum.flightgear.org/viewtopic.php?f=4&t=30386>
- [4] FlightGear - Wikipedia: <https://en.wikipedia.org/wiki/FlightGear>
- [5] FlightGear - Source Code: <https://github.com/FlightGear/flightgear>
- [6] Agile: <https://www.atlassian.com/agile>
- [7] Black Box Testing: <https://www.imperva.com/learn/application-security/black-box-testing/>
- [8] White Box Testing: <https://www.geeksforgeeks.org/software-engineering-white-box-testing/>