Loosely coupled system of systems concept for the SpaceLiner real-time Human-in-the-Loop Space Flight Simulator

Frank Morlang
German Aerospace Center (DLR)
frank.morlang@dlr.de

ABSTRACT

DLR's advanced concept for a suborbital, hypersonic, winged passenger transport vehicle called SpaceLiner, revolutionizing ultra-long-distance travel, has been chosen as a use case for the testing and evaluation of procedures and functionalities for an improved handling of space vehicle operation. To adapt the real-time Human-in-the-Loop Space Flight Simulator to this use case, a SpaceLiner simulation model has been developed by geometry modeled flight dynamics for the commercial flight simulation software "X-Plane". Establishing it as a needed, centralized core will generate specific constraints and limit flexibility, especially if large distributed simulation scenarios in the future are concerned. The paper presents a system of systems concept to replace the current X-Plane focused approach by a very loose coupling alternative

Keywords
microservices, High Level Architecture, distributed simulation

Article Received: 10 August 2020, Revised: 25 October 2020, Accepted: 18 November 2020

Introduction

The actual SpaceLiner real-time Human-in-the-Loop Space Flight Simulator system is shown in Figure 1. It is based on the X-Plane flight simulator software with extensibility through its plugin system **Error! Reference source not found.** and user datagram protocol (UDP) interface **Error! Reference source not found.**.

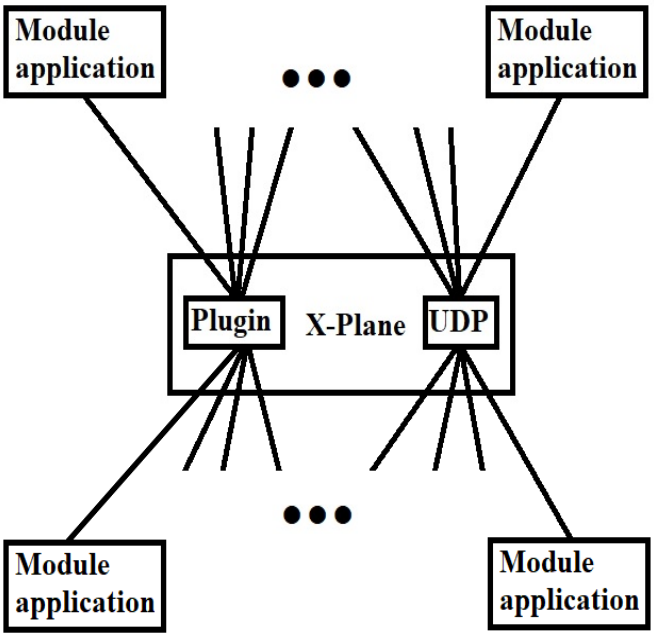

Figure 1. X-Plane based simulation system

Although the current setup covers a wide range of possibilities to be extended by own as well as foreign module application developments, the following constraints can be identified:

- Not all X-Plane internal data references are readable and writable, thus the read-only ones cannot be overwritten by external interfacing modules.
- The plugin and UDP interfaces are specific and of a proprietary format. Any direct data exchange can only be established by tight coupling without following any interoperability standard.
- Taking X-Plane as the core generates a dedicated vendor dependency.

These reasons drove the decision for a transformation to a system of systems approach like shown in Figure 2. Different system simulations will be connected via a middleware to form the whole target system.

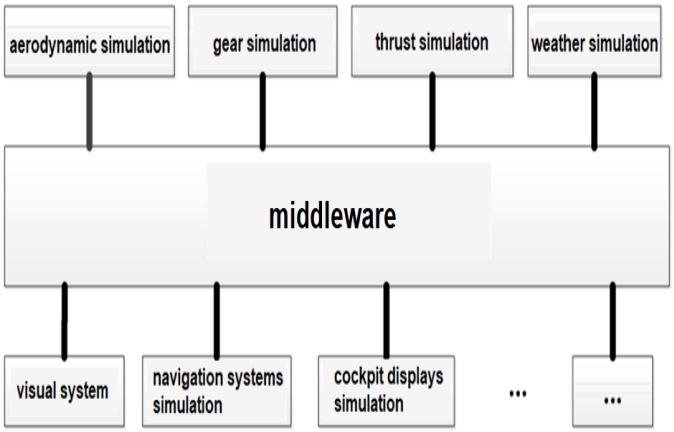

Figure 2. System of systems approach

Challenge

Especially regarding research flight simulators, modularity, and flexibility play an important role to cover a broad bandwidth of research questions. Generic aspects, e.g. cockpit handling human factors aspects as well as specific topics, like e.g. handling qualities of a specific aircraft type under icing conditions need to be addressed. Often, the focus of a question becomes more and more fine grained with time, requesting a sub-system interchangeability like shown in Figure 3. During a period of investigation an aerodynamic simulation system gets its change from a

solution programmed in Fortran to a version based on Matlab / Simulink.

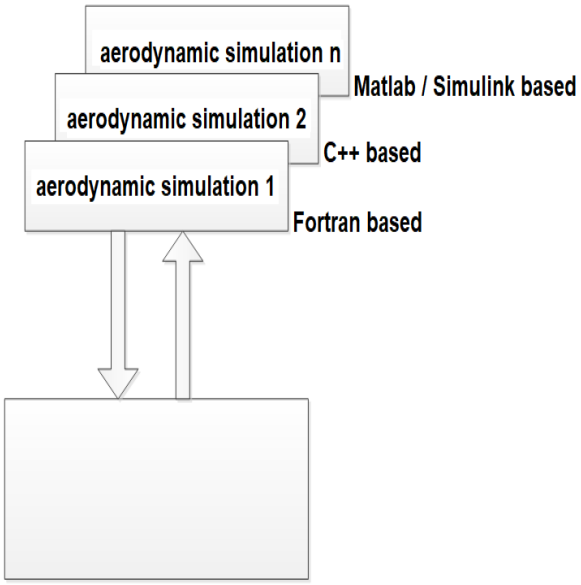

Figure 3. Sub-system interchangeability example

Keeping middleware interfacing development needs as transparent and low as possible can be solved by using a well-established standard. For distributed simulation, the High Level Architecture (HLA) is an Institute of Electrical and Electronics Engineers (IEEE) standard where the individual, participating simulation systems are called federates and the middleware is specified as run-time infrastructure (RTI). The RTI is responsible for standardized information exchange, management of the federates and synchronization. Since its version IEEE 1516-2010, also known as HLA Evolved, the standard provides a Web Service Application Programming Interface (API) and allows the development of web clients as federates. This generally establishes the possibility of building HLA compliant federations based on loosely coupled microservices. Taking into consideration that asynchronous communication can play in important role in human-in-the-loop flight simulation systems, the envisaged solution's general design target intersection can be illustrated as shown in Figure 4.

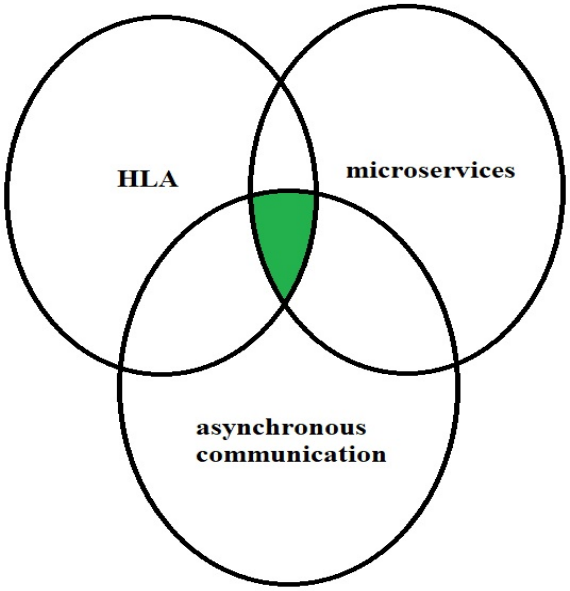

Figure 4. Design target intersection

Against the background that the current HLA web API only allows synchronous communication **Error! Reference source not found.**, the planned approach does not seem to be satisfiable. The question for an alternative solution arose, either to develop a non-web client bridge application or to reject HLA as part of the target design decision.

Solution

A new HLA version nick-named HLA 4 is currently under development **Error! Reference source not found.Error! Reference source not found.**. It is planned to support asynchronous behavior **Error! Reference source not found.**. This will open the door for a system of systems approach shown in Figure 5. The HLA 4 RTI as middleware will allow loose coupling of federates following the microservice architecture with asynchronous communication, thus being in line with the design target intersection of the planned approach.

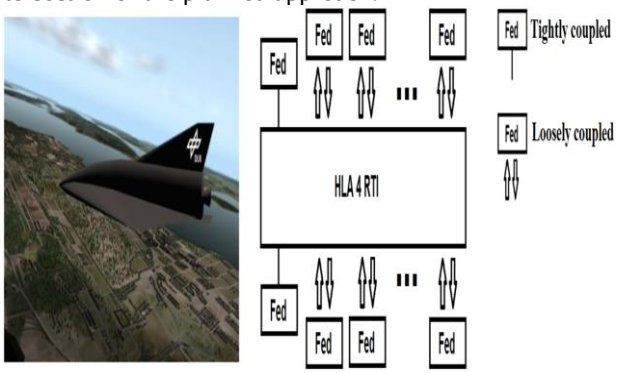

Figure 5. System of systems approach with HLA 4 RTI

Furthermore, HLA 4 is planned to be capable of mapping a declaration- and object management service to Message Oriented Middleware (MOM) product services/wire protocols with the Advanced Message Queuing Protocol (AMQP) **Error! Reference source not found.** as a potential candidate **Error! Reference source not found.**. This would allow a configuration where an AMQP enabled messaging broker (e.g. ActiveMQ) acts as a bridging federate (Figure

6), benefiting from its translation functionalities to further protocols.

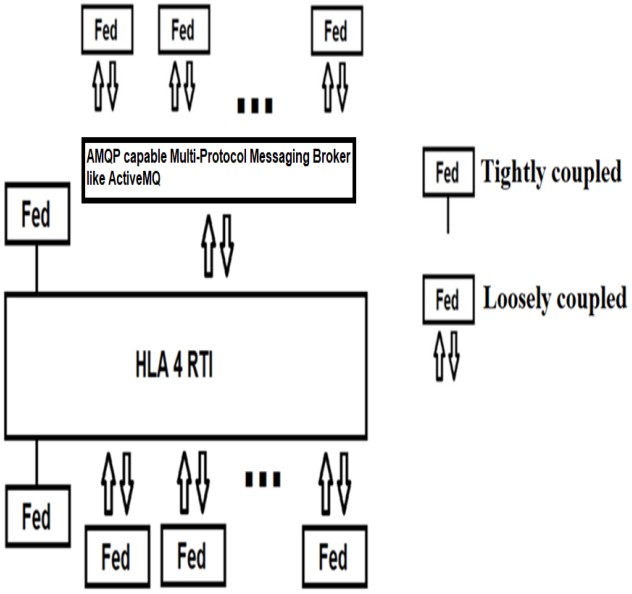

Figure 6. AMQP capable messaging broker as bridging federate

First results and discussion

Against the background that up until now, no HLA 4 RTI is released, a first test set-up refers to the protocol translation performance of an AMQP broker use case with ActiveMQ as the broker and the Simple (or Streaming) Text Orientated Messaging Protocol (STOMP) as the source protocol to be translated to AMQP (Figure 7).

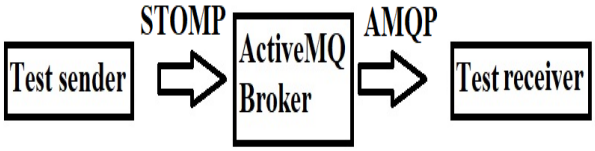

Figure 7. Test use case

One thousand System Wide Information Management (SWIM) compliant position messages (Figure 8) were transferred in a local test set-up (Table 1) with one node, one topic producer and one topic consumer. The broker's STOMP to AMQP translation performance was within 30 to 40 milliseconds (Figure 9), which is more than sufficient for position updates sent to Air Traffic Management (ATM) simulators, where in most cases 1 Hz, sufficient for full traffic scenario data updates, is used. The fact that the translation performance does not meet the 50 Hz real time flight dynamic model update request for human-in-the-loop simulations **Error! Reference source not found.** gets its relativization by the single producer thread set-up on a cheap computer (Table 1), representing low performance hardware. Fulfilling even this, can be anticipated with the usage of better performance hardware configurations in the future.

```
<?xml version="1.0" encoding="UTF-8"?>
<Flight xsi:schemaLocation="http://www.fixm.aero/fixm/4.1 Fixm.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.fixm.aero/flight/4.1">
 - <agreed>
    - <element>
       - <routePoint xsi:type="fb:DesignatedPointOrNavaidType" xmlns:fb="http://www.fixm.aero/base/4.1">
          - <fb:position xsi:type="TrajectoryPoint4DType" srsName="urn:ogc:def:crs:EPSG::4326">
              <fb:pos>39.8495 -75.099945</fb:pos>
             - <level xsi:type="fb:FlightLevelOrAltitudeType">
                 <fb:altitude uom="FT">230000</fb:altitude>
              </level>
           </fb:position>
        </routePoint>
     </element>
  </agreed>
 - <enRoute>
     <currentSsrCode>2234</currentSsrCode>
  </enRoute>
</Flight>
```

Figure 8. SWIM compliant position information

Computer	Lenovo MIIX 320-10ICR 4000 MB memory Intel® Atom(TM) x5-Z8350 CPU @ 1.44GHz Windows 10 Home 64bit (1909)
ActiveMQ version	5.15.12

Table 1. Test set-up parameters

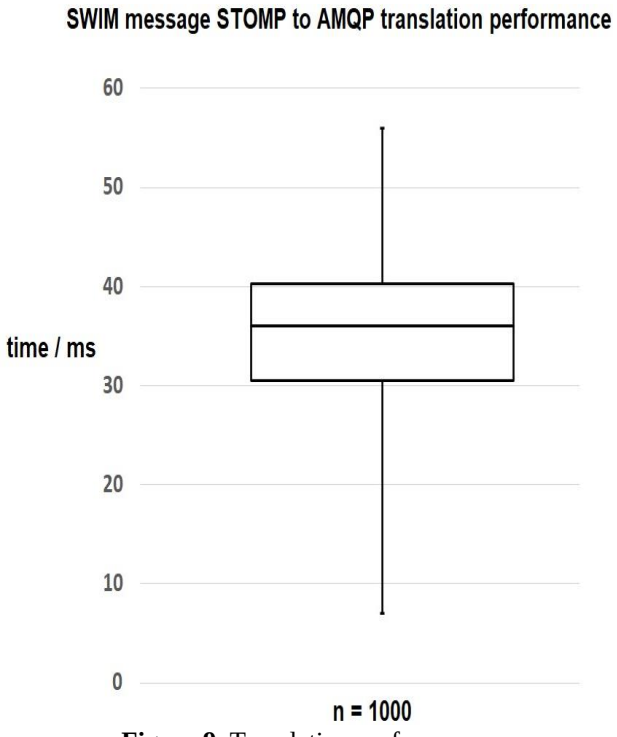

Figure 9. Translation performance

Outlook

Further tests in the future will cover the use of higher performance hardware as well as the integration of ActiveMQ Real Time, where the provision of an optional Java Message Service (JMS) API layer is planned. Stepwise distributed simulation setups with associated experiments will pave the way to a system of systems configuration.

References

[1] X-Plane 2020 "Developing Plugins", 13th of February 2019. [Online]. Available: https://developer.x-plane.com/article/developing-plugins. [Accessed 12 Aug. 2020].

[2] Nuclear Projects “X-Plane UDP data structure”, [Online]. Available: http://www.nuclearprojects.com/xplane/info.shtml. [Accessed 12 Aug. 2020].

[3] B. Möller, F. Antelius and M. Karlsson, Editors. Towards a Standardized Federate Protocol for HLA 4. Proceedings of 2018 Fall Simulation Innovation Workshop (SIW), (2018); Orlando, USA

[4] B. Möller, M. Karlsson, Editors. New Object Modeling Opportunities in HLA 4. Proceedings of 2019 Simulation Innovation Workshop (SIW), (2019); Orlando, USA

[5] AMQP Advanced Message Queuing Protocol. [Online]. Available: https://www.amqp.org/. [Accessed 11 Sep. 2020].

[6] B. Möller, F. Rodriguez, Editors. Towards HLA 4. Proceedings of SISO Seminar at ITEC 2017, (2017); Rotterdam, Netherlands

[7] G. Dussart, V. Portapas, A. Pontillo and M. Lone, in Flight Physics - Models, Techniques and Technologies, Edited K. Volkov, InTech, London (2018), pp.49-72

Author

Frank Morlang

1999 Diploma in Engineering in Materials Science of Technical University of Darmstadt

1999 - 2001 Project engineer at Aerodata company Braunschweig

2001 - today Researcher at Deutsches Zentrum für Luft- und Raumfahrt (German Aerospace Center), DLR, Braunschweig