Application of Data Distribution Service Middleware in Mixed-Criticality Airborne Systems

a briefing at the

AFRL Safe & Secure Systems & Software Symposium (S5)

14-16 June 2011
Dayton, Ohio
Outline

• Mixed-Criticality Application Context
  • Mixed-Criticality Motivation

• Data Distribution Service Middleware Introduction

• UAV Flight Software Motivating Example
  • Criticality Levels
  • On-Board Functions

• Data Distribution Service Middleware Applied to Flight Software Example

• Safety Aspects
Unmanned Platforms Summary

Engineering, Operations & Technology | Boeing Research & Technology

Copyright © 2009 Boeing. All rights reserved.
Mixed-Criticality Application Context

• Multiple Criticality levels in flight software
  • A common small Unmaned Aerial Vehicle (UAV) architecture – all functions to reside on common compute substrate
    – Common CPU(s) due to size, weight, and power (SWAP) constraints

• Increasing certification requirements would drive separating criticalities
  • Demand for demonstration of higher assurance for more critical functions
A Family of Boeing Small UAVs

ScanEagle

Over 467,000 Combat Flight Hours

USMC | US Navy

Coalition Forces

US Air Force

Integrator

Underwater Launch

ScanEagle Compressed Carriage (Sensors / Weapons)

Surfaces Deployed In Flight

Multiple SECCs in Air-Launch Palletized Containers

Underwater Launch

Multiple SECCs on Wing of a Strike Fighter
• TCP/IP based networking connects a few nodes, *doesn’t branch*

• Brokered designs branch through central servers, *but have chokepoints*

• Peer-to-peer decentralized publish-subscribe eliminates bottlenecks
Middleware in Avionics

- **Point-to-point**
  - Commonly used today
  - Hard to scale
  - Hard to be modular

- **Broker based**
  - Introduces single point of failure
  - Must run at highest level of criticality

- **Peer-to-peer publish/subscribe**
  - Well suited for airborne systems
  - Scalable and modular

Copyright © 2010 Boeing. All rights reserved.
Data Distribution Service Middleware

- Directly-programed data models
  - Like a database
- Independent modules
  - Like SOA
- Plug and play flexibility
  - Like a hardware bus
- Peer-to-peer performance
  - Like streaming protocols
- Interoperable
  - Like TCP/IP

Scalable, high performance, reliable infrastructure
### Criticality Levels of Interest

<table>
<thead>
<tr>
<th>Criticality Level</th>
<th>Required to prevent ~</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flight Critical</td>
<td>- Unintended loss of life.</td>
</tr>
<tr>
<td></td>
<td>- Unintended reduction of public safety.</td>
</tr>
<tr>
<td></td>
<td>- Substantial damage to the vehicle or its support structure.</td>
</tr>
<tr>
<td></td>
<td>- Loss of vehicle</td>
</tr>
<tr>
<td></td>
<td>- Unintended event (e.g., crash) into politically sensitive area</td>
</tr>
<tr>
<td>Mission Critical</td>
<td>- Unintended loss of the vehicle’s function (mission sensors or other effectors).</td>
</tr>
<tr>
<td></td>
<td>- Unintended loss of support structure function (e.g., ability of operator to use on-board UAV sensors).</td>
</tr>
<tr>
<td>Maintenance Critical</td>
<td>- Inefficient use of the vehicle’s resources.</td>
</tr>
<tr>
<td></td>
<td>- Inefficient use of ground support structure.</td>
</tr>
</tbody>
</table>
Sample UAV System Architecture

**Plan within vehicle capability**

- Min/Max V
- Nz response
- P response
- Data request
- Surface test allowed

**Act within vehicle capability**

- Vehicle sensor data
- Off-board data (including from control station)

**Assess vehicle subsystem health**

- Vehicle sensor and state data

---

**Guidance / Other**

- Flight Mgmt
- Subsystem Control

**Adaptive Control**

- Neural Net Correction
- Control Allocation

- Surface test requests
- Effector capability (gray-scale)

**Integrated Vehicle Health Management**

- Vehicle Capability
- Health Assessment

**Actuation (Control)**

- Motor Control
- Actuator Built In Test

- Motor commands
- Actuator sensor data
- Actuator health data

- Surface commands
- Actuator sensor data
- Actuator health data

- Sensor commands
- Weapons commands
- Data link Tx
Modular Separation

- **Flight Critical Component**
  - High Criticality
  - Guest OS

- **Mission Critical Component**
  - High Criticality
  - Guest OS

- **Mission Critical Component**
  - Medium Criticality
  - Guest OS

- **Maintenance Critical Component**
  - Low Criticality
  - Guest OS

- **Virtualization**
  - (MILS Separation Kernel or ARINC 653 Partition Scheduler)

- **Hardware**
DDS Example Data Flows

Flight Critical Component
High Criticality
DDS
Guest OS

Mission Critical Component
High Criticality
DDS
Guest OS

Mission Critical Component
Medium Criticality
DDS
Guest OS

Maintenance Critical Component
Low Criticality
DDS
Guest OS

Virtualization
Hardware

From high criticality component on another node
Data Centric Approach to Mixed Criticality

• Identify all of the data in the system
  • Based on the aircraft and mission

• Define criticality of data
  • Flight, mission, or maintenance critical
  • Level of criticality

• Define data delivery attributes
  • Rate of delivery
  • Reliable or best-effort

• This creates a “data contract”
  ➔ Publishing components must meet the contract
Data Distribution Service Middleware Applied to Flight Software Example

• UAV example
• Example data-centric model
  • Data criticality levels

<table>
<thead>
<tr>
<th>Flight Critical</th>
<th>Mission Critical</th>
<th>Maintenance Critical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motor commands</td>
<td>Imaging Sensor commands</td>
<td>On-board system health data</td>
</tr>
<tr>
<td>Surface commands</td>
<td>Weapons commands</td>
<td>Built-In Test results</td>
</tr>
<tr>
<td>Vehicle attitude</td>
<td>SIGINT Payload commands</td>
<td>Actuator Command /</td>
</tr>
<tr>
<td>Vehicle airspeed</td>
<td>Laser Designator Commands</td>
<td>Response data</td>
</tr>
</tbody>
</table>
• Design component to minimum assurance required by data it writes
Validation of a DDS Based Flight System

• Validate data contract is correct
  • Data type
  • Name
  • Quality of Service

• Sender/Receiver of the data is anonymous
  • Validate each component – does it conform to the data model
  • Validate the System – is there a producer at correct assurance level for each required data
Safety Aspects for DDS

- Subset of the full DDS API
  - Remove functionality not suitable for a safety-critical system, such as non-deterministic behavior
  - Reduce code size to enable certifiability of middleware
- Integrate with safe interpartition transport
- Enable delivery of high criticality data to lower criticality component
  - Lower criticality component cannot “talk back” to the higher criticality component
Summary

• Small UAV proliferation
  • Gaining in utility and functionality
  • Affordability is key

• Embedded software key to functionality and integration

• Mixed Criticality can support affordability
  • DDS middleware is an enabler