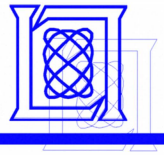


Design and Implementation of the TFDM Information Management Architecture

William Moser

MIT Lincoln Laboratory

This work was sponsored by the Federal Aviation Administration under Air Force Contract No. FA8721-05-C-0002.
Opinions, interpretations, conclusions and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

Outline

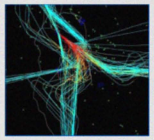
- Brief Introduction to TFDM
- Information Architecture Objectives
- Survey of Architectural Components
- The TFDM Simulation Environment
- Summary

Tower Flight Data Manager (TFDM) System Overview

External Sources

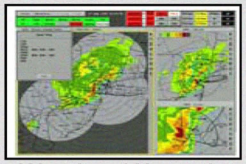
Terminal and Surface Surveillance

Flight Plan Data



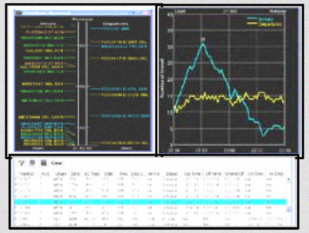
Traffic Flow Constraints

Flight Operations Data



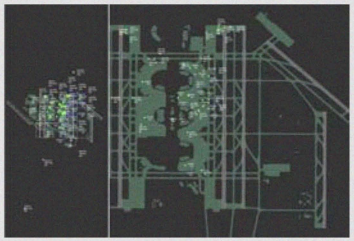
Weather / Hazards

Tower Flight Data Manager

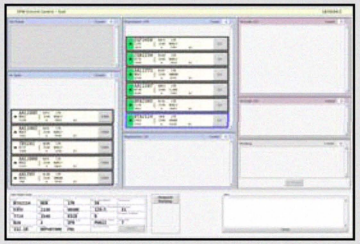


Arrival / Departure Management Tool (ADMT)

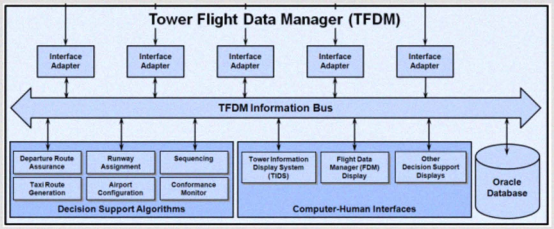
Remote / enhanced visual awareness



Surveillance display



Flight data manager



Net-centric infrastructure

Enablers

- Consolidated tower systems
- Enhanced cross-domain information exchange
- Decision support tools

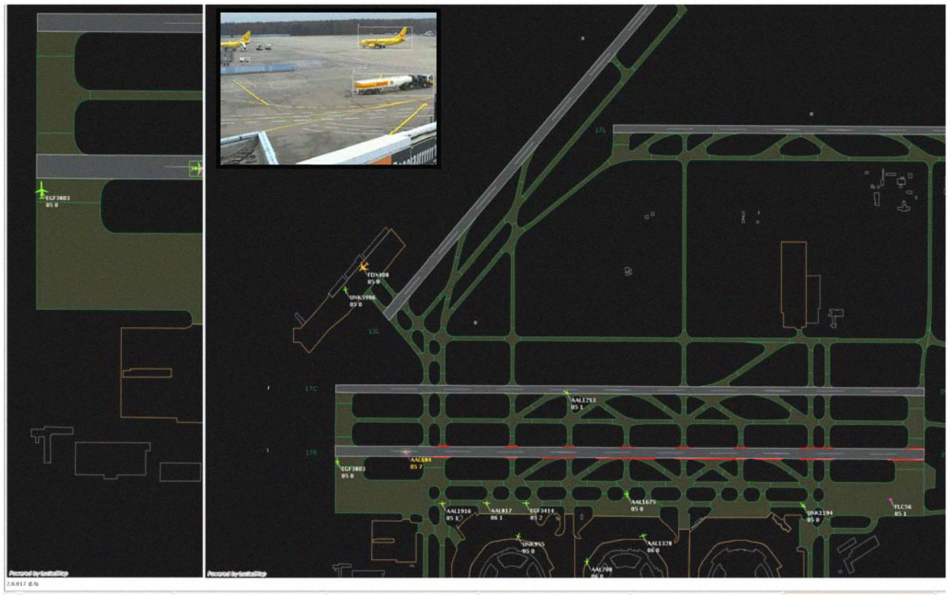
Benefits

- Robust operations
- Reduced delay, fuel, environmental impact
- Enhanced safety
- Ability to support remote operations: Staffed NextGen Tower (SNT)

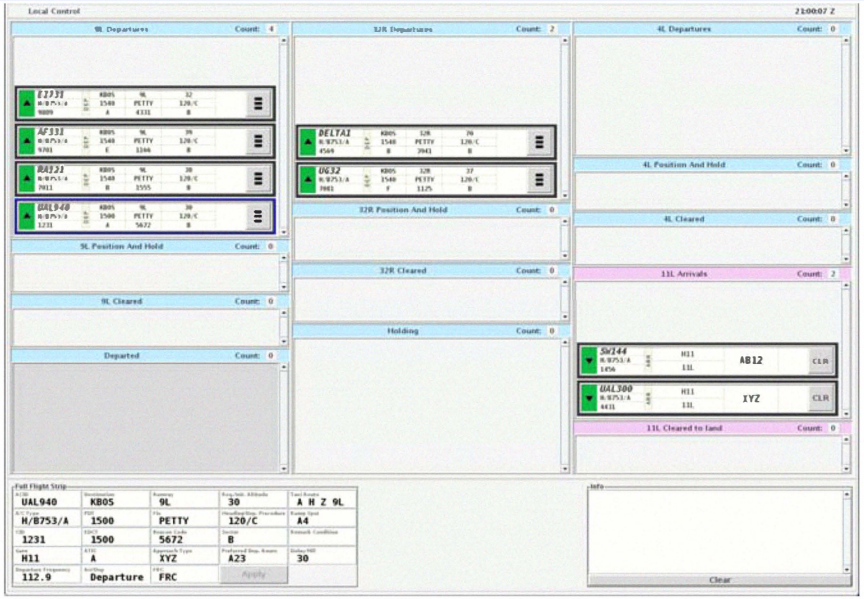
Operational Users

- Tower controllers
  - Flight data, Clearance, Ground, Local, Supervisor
- Terminal Control
- Flight Operations Centers
- Ramp Tower
- Airport Authority

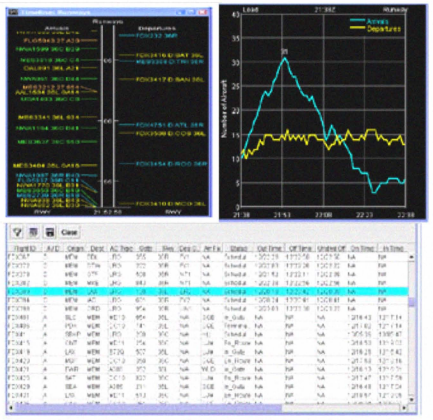
Primary Computer Human Interfaces (CHI)



Tower Information Display System (TIDS)

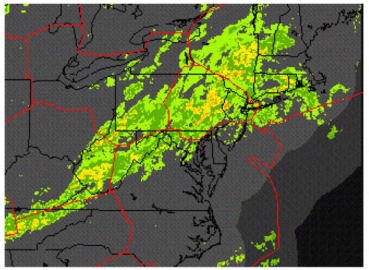


Flight Data Manager (FDM) Display

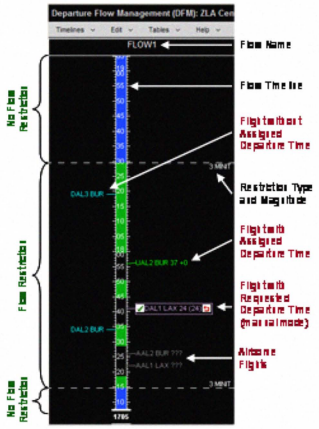


Supervisor / Traffic Management Display

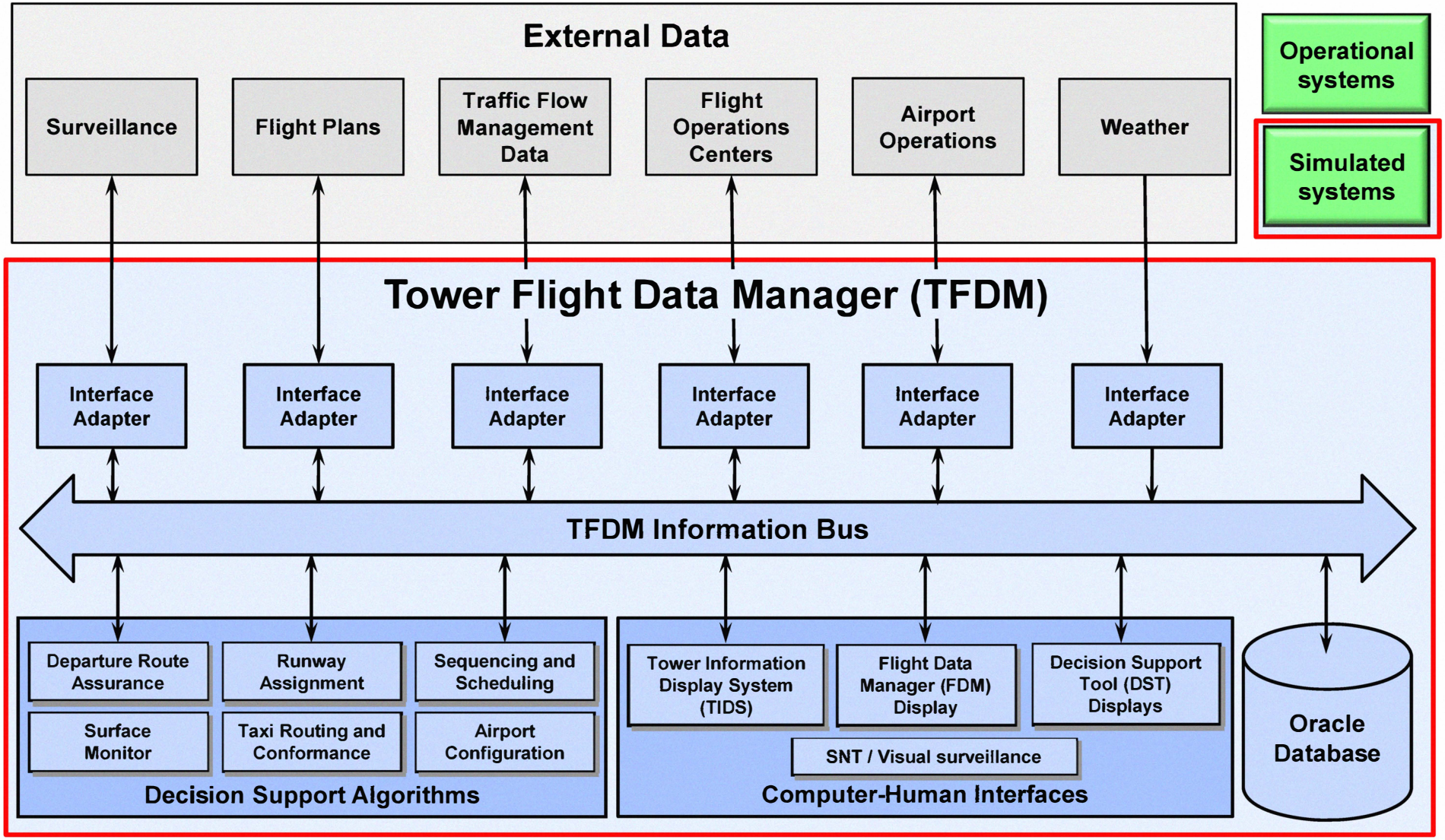
* Integrated display concept being developed *



Route	Trend	PIG	2155	2200	2205	2210	2215	2220
N90 HAPIE	—							
N90 MERIT	—	25						
N90 GREKI CAM	▼		33 ENR	34 ENR	30 NEAR	30 NEAR	29 NEAR	30 NEAR
N90 GAYEL J95	▲		34 N90	36 N90	35 N90	34 N90	34 N90	34 N90
N90 COATE J36	—		34 N90	36 N90	35 N90	35 N90	34 N90	37 N90



Tower Flight Data Manager (TFDM) System



= Lincoln responsibility for prototype system

Outline

- Brief Introduction to TFDM
- Information Architecture Objectives
- Survey of Architectural Components
- The TFDM Simulation Environment
- Summary

Information Architecture Objectives

- Integrate multiple heterogeneous information sources
  - Flight plans, weather, en route traffic information, surveillance, etc.
- Accommodate system evolution as information and decision support capabilities are deployed, decommissioned, or updated
- Export terminal-area information to stakeholders
  - Other NextGen facilities and systems, airlines, data archives, etc.
- Comply with relevant requirements, standards and guidelines
  - FAA NextGen/SWIM
  - LL Mission Assurance Office
  - LL Net-Centric Toolkit

Information Management Architecture, External View

A/DMT, ERAM, ATC personnel, etc

Publishers insert data incrementally as XML documents. Insertions are validated against schema definitions.

Consumers receive filtered XML-formatted data via *persistent query* subscription mechanism.

Pub/Sub-enabled WFS Server

TFDM SOA implemented as Open Geospatial Consortium's (OGC) Web Feature Service (WFS)

Data automatically aged out of real-time database, into data warehouse

Oracle Database

Long-term data store

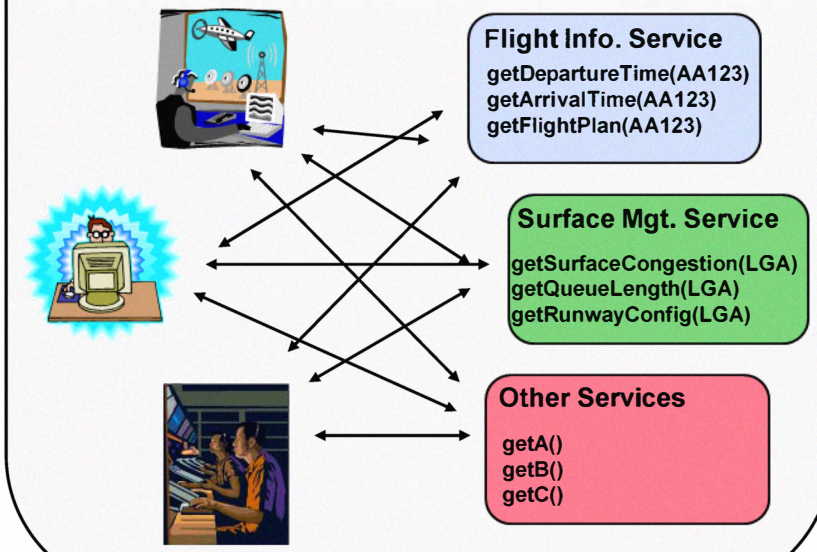
TFDM provides a single, flexible service (OGC's WFS*) to other entities in the System-Wide Information Management (SWIM) SOA

*The Open Geospatial Consortium's Web Feature Service

Scalable Information Architecture

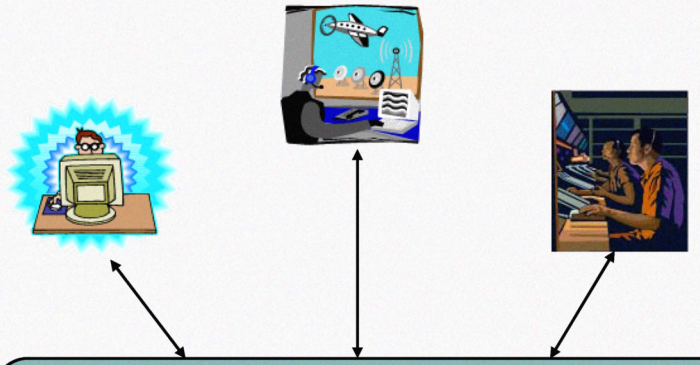
Conventional SOA design is not scalable!

- N-squared problem as more users and service providers connect
- Semantic divergence as service providers provide variations on the same theme (e.g. pub/sub)



TFDM Information Management Architecture (TIMA) scales well as users and services are added

- Filtered data access interface makes it easy for users to ask for the data they need
- Producers logically decoupled from consumers



Many distinct services vs. single flexible service

System-Wide Information Management (SWIM) Container

Progress Software's FUSE product selected as the SWIM container in August 2008

Function	Technology	TFDM Role
Enterprise Service Bus (ESB)	ServiceMix	Hosts ADMT modules, provides standard interfaces to external components
Message Broker	ActiveMQ	Pub/sub infrastructure for inter-process communication
Mediation Router	Camel	With ActiveMQ, forms backbone of TFDM information routing architecture
Services Framework	CXF	Implements information sharing via Web Feature Service (WFS)

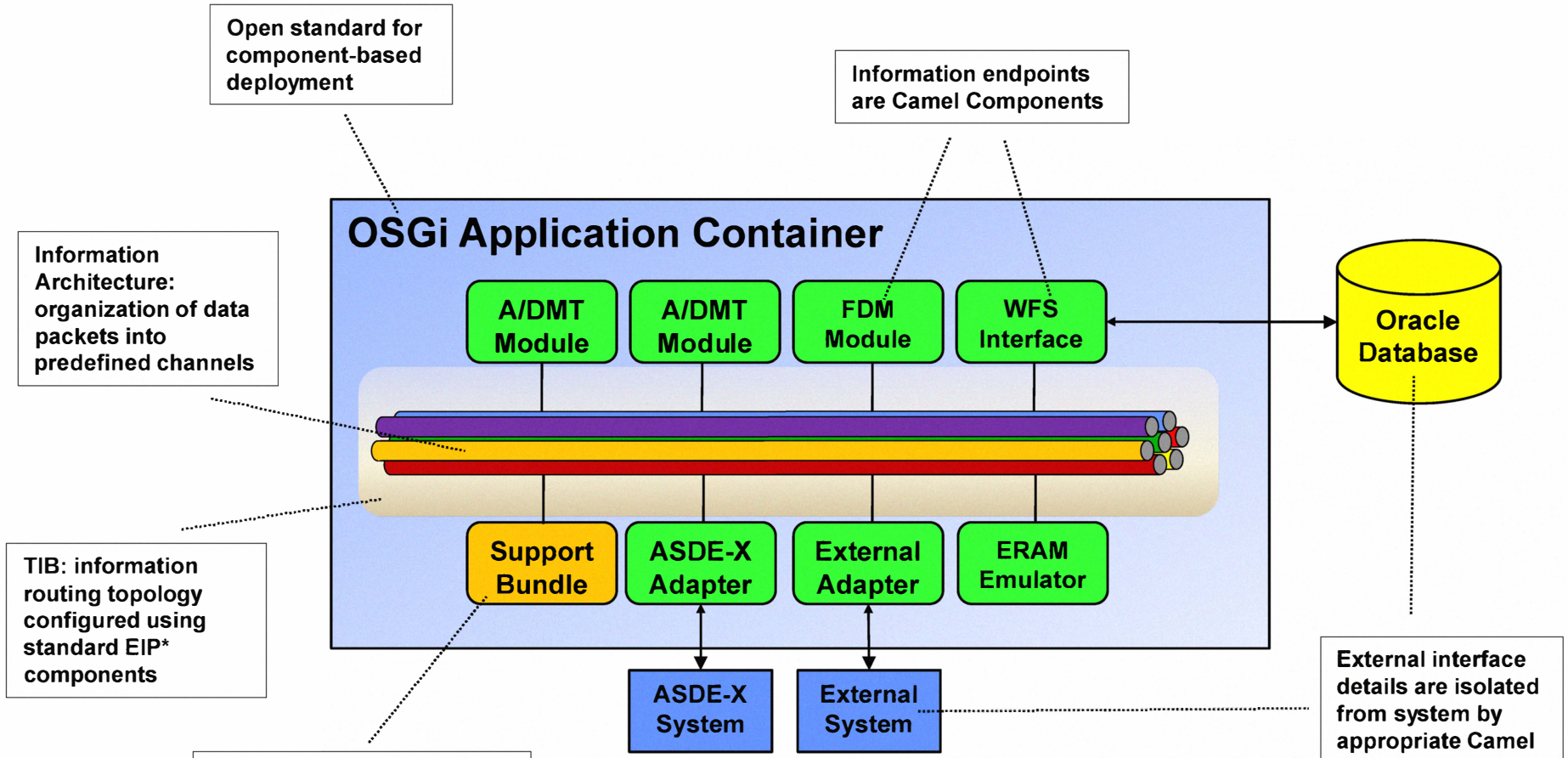
Outline

- Brief Introduction to TFDM
- Information Architecture Objectives
- Survey of Architectural Components
- The TFDM Simulation Environment
- Summary

The TFDM Information Management Architecture (TIMA)

- TIMA is comprised of components which address
  - Data archiving and retrieval
  - Information exchange among processing components
  - Organization of information
- Web Feature Service (WFS)
  - Web Service standard developed by the OpenGeospatial Consortium
  - TFDM's "public" access point as a SWIM service
- TFDM Information Bus (TIB)
  - Apache Camel, JMS-based approach to data transport
- TFDM Information Architecture
  - "Information Channel" overlay on the TIB

TIMA Design Overview



TFDM Information Management Architecture developed in accordance with SWIM guidelines

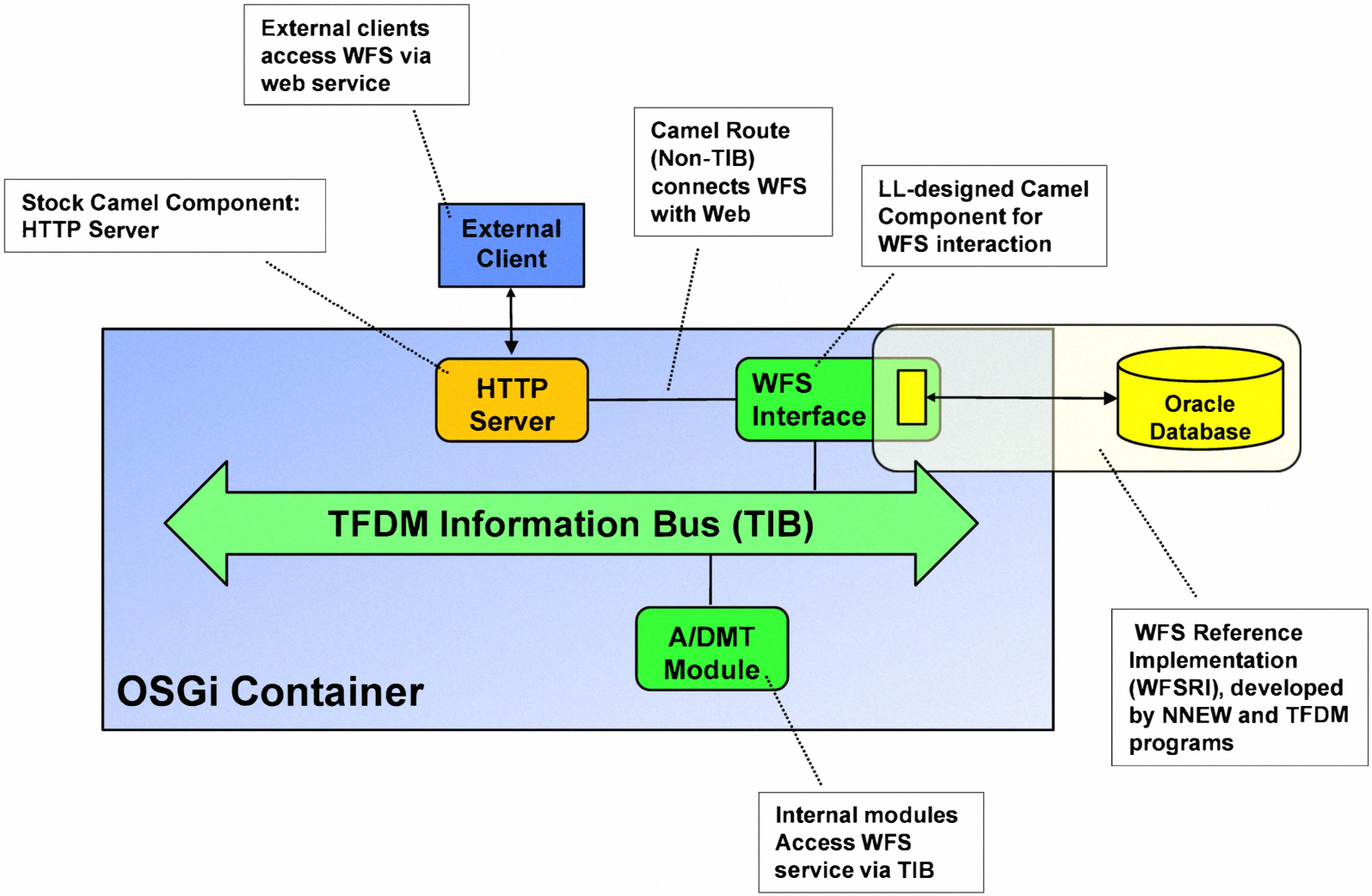
*Hohpe and Woolfe, Enterprise Integration Patterns, Addison-Wesley 2004

The Web Feature Service (WFS)

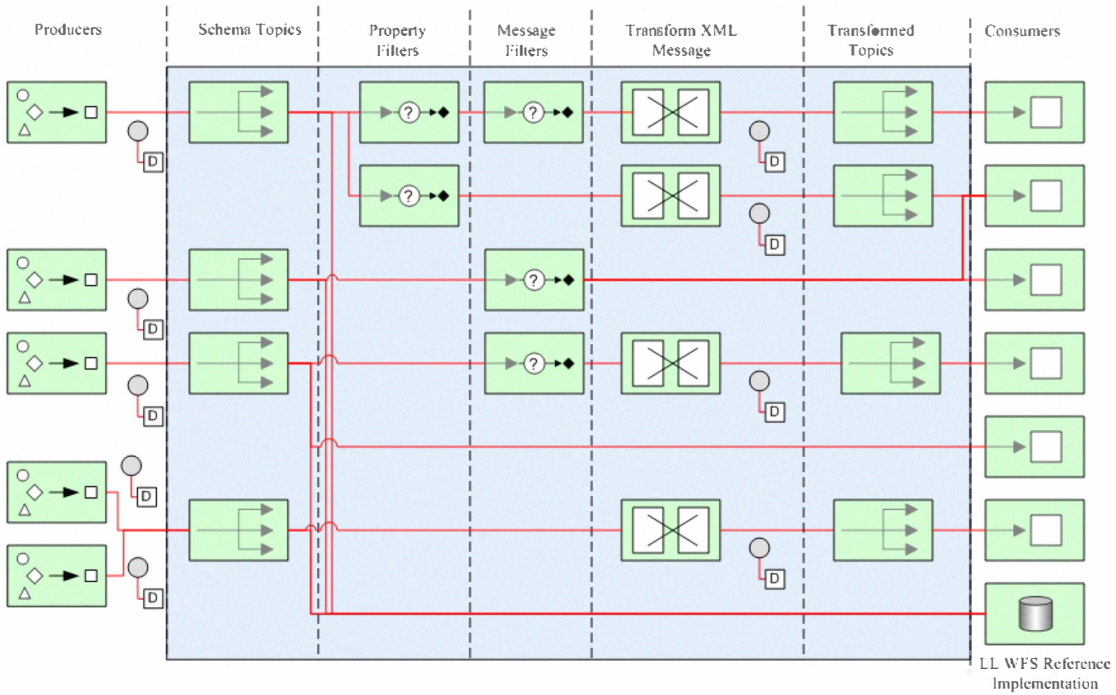
- Open Geospatial Consortium (OGC) standard for storage and retrieval of XML data
- Data must conform to the OGC's Geographic Markup Language (GML) standard
- WFS operations include INSERT, UPDATE, DELETE, QUERY, and DISCOVERY
- WFS specification includes WSDL for service interoperability
  - http://schemas.opengis.net/wfs/1.1.0/wsdl
- G43 NNEW team extending to include a publish/subscribe operation

WFS serves as the internal database for TFDM, as well as the external service interface in the SWIM SOA

WFS/TFDM Integration



The TFDM Information Bus (TIB)

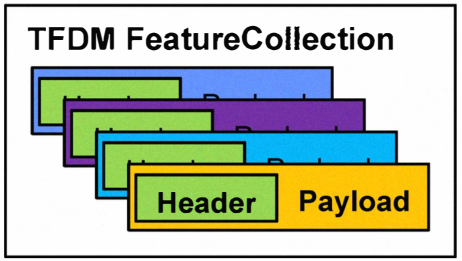


- **"No-code" data transport**: information bus configured from existing SWIM container components -- *No additional software development required*
- **Transport-independent applications**: applications configured to route data appropriately – *applications require no knowledge of information bus, or transport protocols*
- **Information-oriented**: Applications need only know what information they require, not who provides it

The TFDM Information Architecture

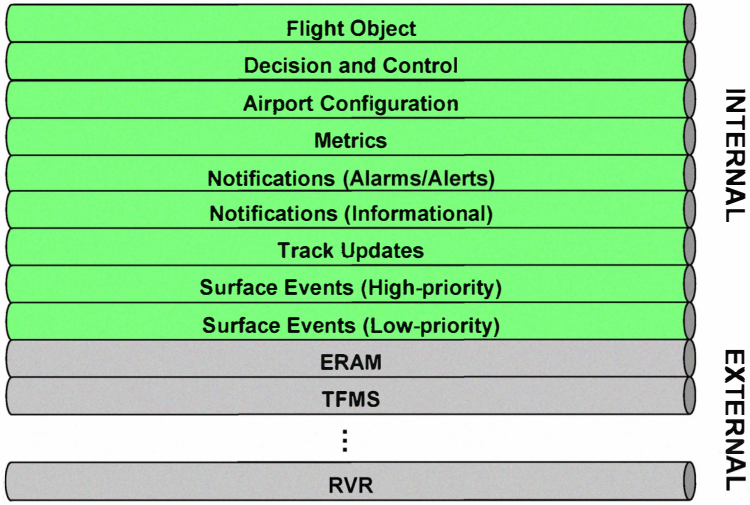
Representation: Information structure

- Standardized approach to XML schema development
- Basis for information transmission, archiving, and retrieval
- Data types ("message payloads") are realized as TFDM Features
  - GML Features with a common "header"
- Related Features may be grouped into FeatureCollections for transmission



Transmission: Information channelization

- FeatureCollections defined for major information categories
  - Flight Object, Decision & Control, etc
- Data for each category transmitted on dedicated "channel"
  - Channels implemented as JMS topics



Uniform standards for both representation and transmission are the foundation for robust information exchange in an evolving system

Outline

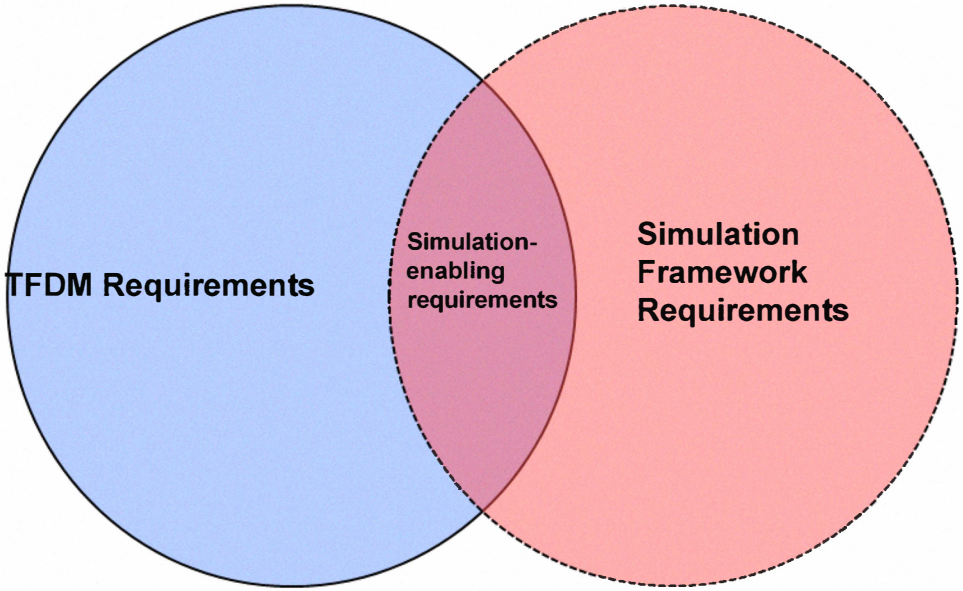
- Brief Introduction to TFDM
- Information Architecture Objectives
- Survey of Architectural Components
- The TFDM Simulation Environment
- Summary

Role of Simulation in TFDM

- Provide high-fidelity software test environment for developers
- Provide realistic environment for human-in-the-loop testing
- Qualify and integrate new features prior to field deployment
- Enable exploration of ATM concepts
- Provide a platform for benefits analysis

Ultimate purpose is risk mitigation for TFDM procurement

Simulation Requirements

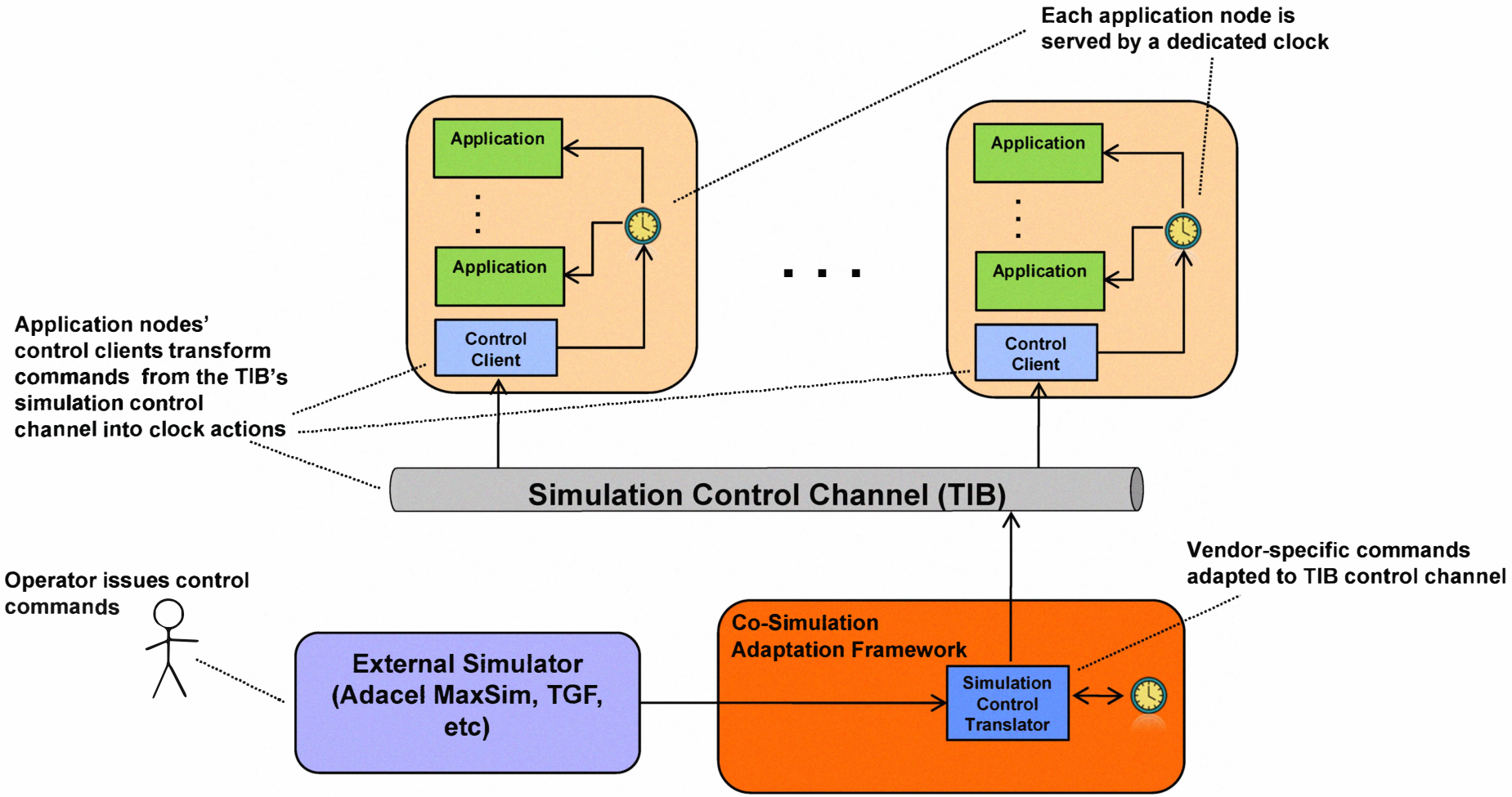


- *Simulation requirements are minimal, enough to enable TFDM modules to interact in simulations*
- *We need to develop a simulation framework to run simulations*
- *The simulation framework is not itself part of TFDM – we have not developed a separate requirements document for this*

Simulation Architecture Goals

- Scalable to multiple nodes
  - Primarily LAN-based, with limited WAN possibilities
- Robust, transparent time synchronization among system components
- TFDM applications unaware of simulation environment
- Lightweight, adaptable control mechanism
- Easily integrated with external simulation systems
  - Systems must have compatible time synchronization and accessible I/O mechanisms

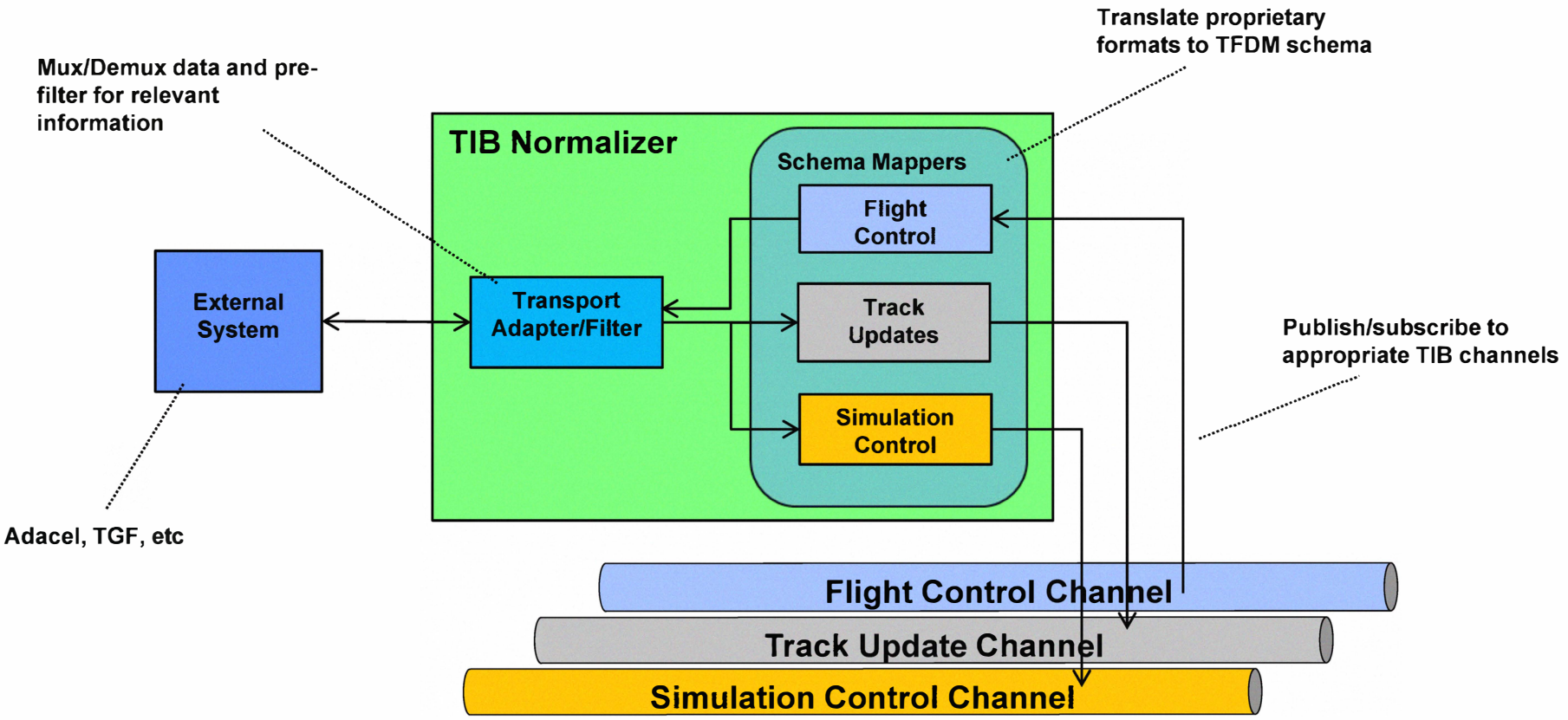
Basic Control Framework



Synchronization scalability maintained by using systems' *hardware clocks* and NTP*

*Network Time Protocol: protocol for synchronizing computer system clocks across networks

Co-simulation Integration Architecture



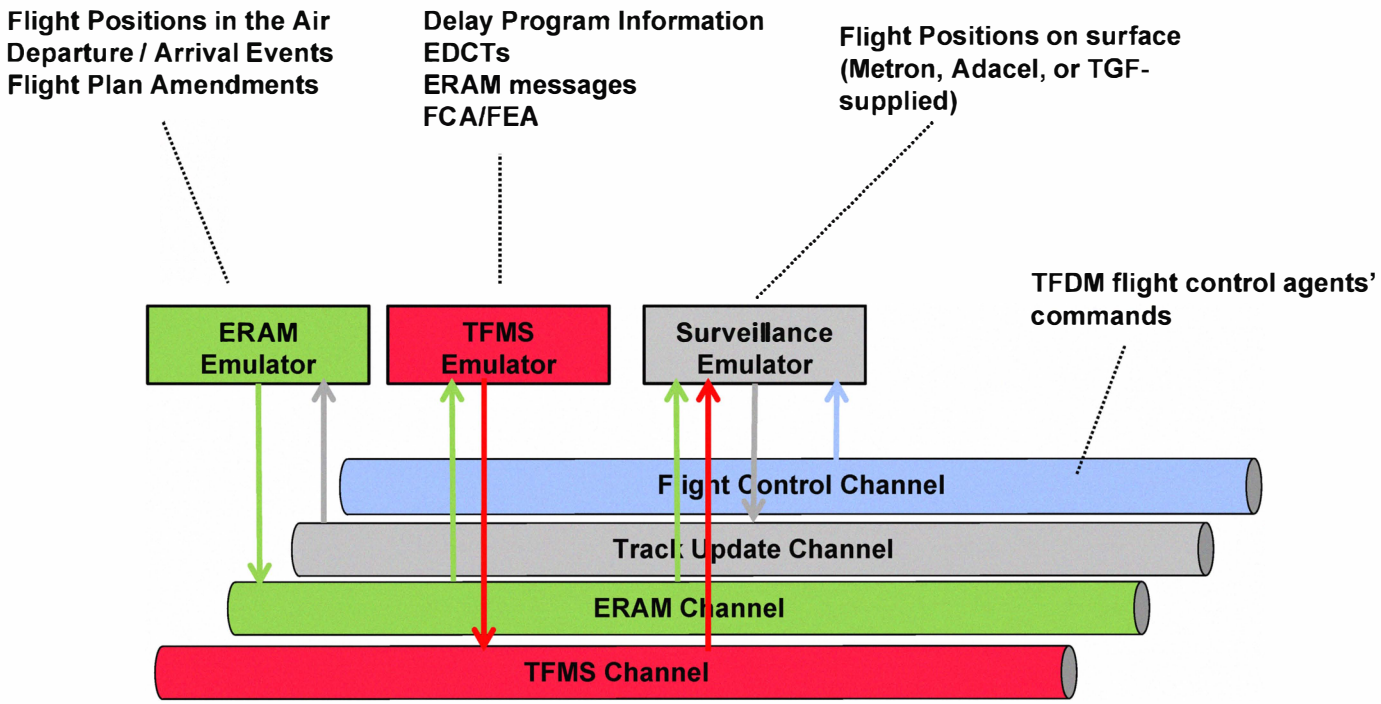
Data and protocol normalization is tailored to the specific co-simulation system

Simulation Engines and Modules

- **NAS Subsystem Emulation components**
  - **Emulators for TFMS, ERAM, ASDE-X (Lo-Fi)**
  - **Integrated into TFDM as "native" components**

- **Controller Simulation Environment**
  - **Adacel MaxSim system with software interface enhancements**
  - **High-fidelity terminal-area traffic simulator with detailed out-the-window views**
  - **FAA deploying to multiple locations around the country for controller training**

End-State Emulator Design



End state design modularizes functionality and enables integration with other simulation engines

Outline

- Brief Introduction to TFDM
- Information Architecture Objectives
- Survey of Architectural Components
- The TFDM Simulation Environment
- Summary

TFDM Information Management Architecture Summary

- Event-driven
  - Receipt of information, timer expiration, external system change
- Information oriented
  - Flexible communication mechanism and service interface
- Distributed
  - Deployment topology is transparent to applications
- Built on open-source tooling and open standards
  - SWIM-compliant
- Designed to integrate with simulation systems

The TFDM Information Management Architecture is intended to scale:
- *internally*, using an implicit invocation paradigm to accommodate evolving configurations of decision support tools
- *externally*, accommodating changes in information content provided without the need for interface retooling on either the client or server side