Realtime Traffic Control using Publish/ Subscribe Distribution Technology

Prof. Dr. Erwin Mayer, Johannes Fröhlich, Mehran Shakeri

Verkehrskontrollinfrastrukturen sind von ihrer Natur her verteilte Systeme, häufig mit Leitstellen, die auf unterschiedlichster lokaler IT- und Systemarchitektur basieren. Trotz dieses föderativen Charakters ist es aus Sicherheitsgründen essentiell, dass alle beteiligten Kontrollzentren jederzeit einen gemeinsamen konsistenten Gesamtzustand der Verkehrssituation verwalten. Im vorliegenden Projekt wird eine neuartige Middleware entwickelt, die diese Aufgabe im Bereich der Luftverkehrskontrolle auf der Basis des OMG DDS (Data Distribution Service) Standards realisiert.

Todays' traffic support environments are distributed by nature. In many cases the monitoring, control and guidance of traffic is effected by a federation of coordinating centers, often managed by different organizations, using differing local IT technology and system architecture. Despite the federative character of such systems, maintenance of a consistent overall traffic state is indispensable for a safe operation. This project develops a new type of middleware supporting federative systems in the domain of Air Traffic Control (ATC), using OMG's DDS (Data Distribution Service) standard as contributor.

Introduction

The goal of the Single European Sky ATM Research Programme (SESAR) [5] is to increase safety and efficiency in European air traffic control. One driver to this is an improved cooperation of the independent Air Traffic Service Units (ATSUs), e.g. control centers and airports. Already today, adjacent units work together along a variety of standards like [7], in order to support the overflight of an airplane across multiple ATSUs' area of responsibility. However this is based to a high degree on bilateral exchange procedures between the neighboring centers, each center maintaining a separate flight state with proprietary attributes and update procedures.

The DDS-ATC (Data Distribution System for Air Traffic Control) project aims at developing a technical infrastructure such that a federative system of ATSUs (namely the European Air Traffic Control Infrastructure) is capable of accessing a joint representation of a virtual flight object with standardized static and dynamic attributes. Secured by access rights, members of the federation shall use this to oversee a specific flight from departure to landing. They shall dynamically and in realtime, read and update the flight's unique and common representation over a Wide Area communication network. For safety reasons a centralized storage of a flight database has to be avoided, instead the realization of the virtual flight object shall be based on a distributed replication scheme. The DDS-ATC project is funded by the BMWI (Bundesministerium für Wirtschaft und Energie) and is executed together with industry partner COMSOFT, German supplier for Air Traffic Control technology.

System Architecture

Figure 1 describes the underlying architecture of the proposed solution.

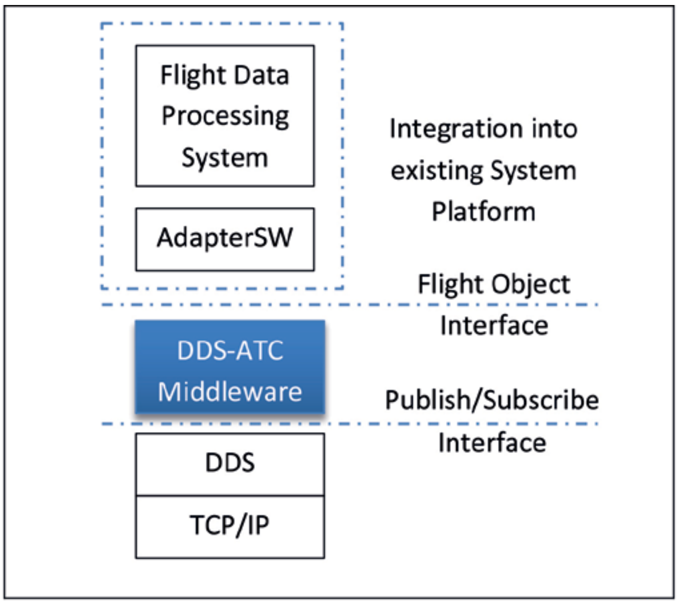

Fig. 1: Project Architecture

Based on a standard TCP/IP architecture (IPv4 and IPv6), OMG's DDS (Data Distribution Service) [1] is used as the lower layer communication platform. It offers a Publish/Subscribe Interface [4][8], allowing the asynchronous 1:N transfer of data between producers and consumers of information updates. On top of this, the DDS-ATC middleware, representing the core of the project, implements a replicated Flight Object database. Over a standardized interface [5][6], it is used by applications to read and update Flight Object attributes, much as if these flight objects were stored centrally. As part of its service the middleware performs data distribution and synchronization of the flight object replicas, ensuring data consistency across the sites. In a typical use case of the middleware the FDPS (Flight Data Processing System) of an existing ATSU would be upgraded, such that it can access these Flight Objects to communicate changes (e.g. a flight plan update or a flight delay) to other centers. The Adapter Software is implementing the missing link between FDPS and the middleware. Project partner COMSOFT implements the application-specific interoperation of centers, like the OLDI protocol [7], while the Hochschule provides the database management, replication control and recovery functionality of the middleware.

DDS Publish/Subscribe

As part of an effort to support distributed applications for realtime environments, the OMG (Open Management Group) defined the DDS (Data Distribution Service) Publish/Subscribe standard [1]. It uses a 1:N communication paradigm, decoupling senders and receivers of data by intermediate instances, so-called „topics". Senders of data write („publish") data into a topic without knowing the number and identity of receivers. If an application is interested in certain data it „subscribes" to the respective topic and from then on continues to receive the sent data, much like the other subscribers to the same topic (see Fig. 2).

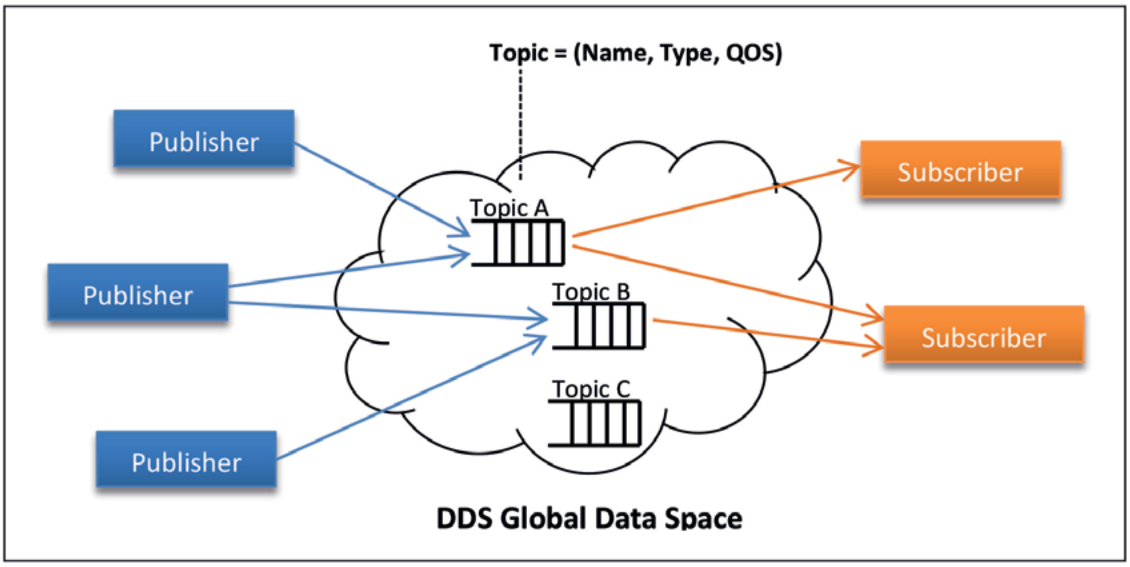

Fig. 2: DDS Publish/Subscribe Sublayer

A topic corresponds to a tuple (name, type, QOS). The name represents the handle to the data and is enforced to be system-wide unique. DDS supports „discovery" procedures to find topics based on this name. Unlike other Publish/Subscribe systems like JMS [8], DDS is aware of the data type of a topic and can use it e.g. for serialization, filtering and an embedding of data access functions into a higher layer programming language. This „data-centric" view of the distributed data is unique to DDS and largely simplifies development of distributed applications. Types are defined in an IDL (Interface Definition Language), in a similar way to CORBA (Common Object Request Broker Architecture) [3].

Also specific to DDS, there exists the possibility to define separate and dedicated QOS (Quality of Service) attributes for each individual topic. While for one topic a reliable transfer is mandated, another may be defined by the application as best-effort and the DDS provider system will adjust the communication stack accordingly. Further QOS parameters relate to deadline monitoring, provision of durability (persistence) or maintenance of delay budgets (ref [1]).

The DDS provider system can be configured to work in an entirely peer-to-peer manner, i.e. without central storage or broker, ideal to support the given project environment, with its independent ATSUs. Further benefit of using DDS as a lower layer is its performance characteristics: The DDS stack is optimized for realtime transfer of data with an efficient (zero-copy) buffer handling, exploitation of lower layer multicast technology and optional use of shared memory.

Next to the standardization of the API (application programming interface) [1], DDS also provides a "wire protocol", i.e. a unique specification of how data is exchanged, such that different nodes can use different DDS supplier stacks.[2]

Synchronization

While the use of DDS as a lower layer to the project already provides basic distribution functionality, a Replication Control scheme and a Recovery Management is required on top. Because copies of the data are stored at multiple locations transmission delays of the underlying network can lead to inconsistent views of the stored data and optionally to an inconsistent state of the data itself. Figure 3 shows an example. Two adjacent ATSUs trigger updates to the state of an individual flight object. The resulting updates to each of the replicated copies of the data arrive in a different order (1->2, 2->1) at two other ATSUs, which may lead to a loss of consistency between the replicas.

Further inconsistencies may result in the context of node failures and recovery. Here it is essential that

1. a recovering node receives all missed updates for its local replica to maintain a consistent state, and
2. does not provide own replica updates to third nodes, before the state of the own copy of the flight object is up-to-date.

To implement these constraints, the middleware instances have to run a synchronization protocol, ensuring not only the reliable transfer of data, but also a consistently ordered delivery of 1:N (multicast) messages to overlapping groups of recipients.

Implementation Approach

Replication Management will be implemented by a variant of a Primary Copy replication control algorithm [9]. The approach assumes that one of the replicas assumes a distinguished role in that it synchronizes the updates to the other (secondary) copies. The DDS-ATC implementation refines this principle in two aspects: First, instead of having a single ATSU representing the primary copy for the whole database (which would be a safety risk), the granularity of the primary-copy-ship is chosen to be an individual flight, i.e. for two distinct flights two differing ATSUs typically host the respective primary copy. This caters for a strong independence and resilience against failures. Second, the primary-copy-ship is not static but transferred between ATSUs dynamically. In an ATC-application-specific manner, each time the operational responsibility and control for a flight changes between one ATSU and another (e.g. when an airplane crosses the border), implicitly on the level of the DDS-ATC middleware a transfer of primary-copy-ship is triggered.

Note that while the primary copy scheme ensures consistent updates for the most part of the scenario (namely whenever the airplane is in the scope of a single ATSU), the process of transferring primary-copy-ship is itself critical to the consistency conditions of the replica. Two updates, one immediately before the transfer and one immediately succeeding the transfer may otherwise in terms of order of execution again trigger an inconsistent replica status (see figure 3).

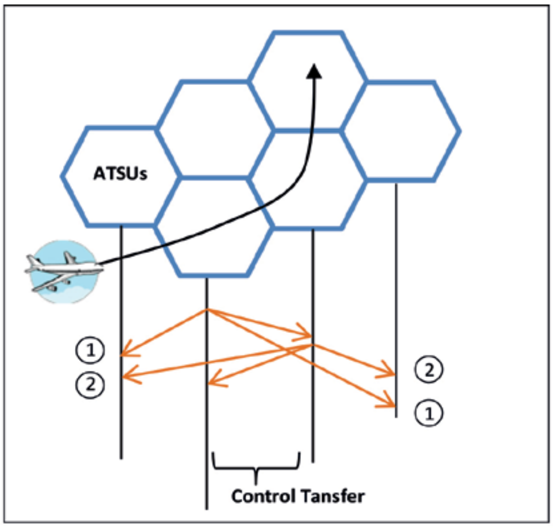

Fig. 3: Inconsistent Update Order as part of 1:N Distribution

Summary and Outlook

The DDS-ATC project will deliver a prototype for the realtime, peer-to-peer-based sharing of flight information within a federation of infrastructure centers controlling air traffic. It does so by means of a middleware implementing a fault-tolerant replicated datastore on top of the DDS Publish/Subscribe standard. Next to the functional validation of the proposed solution, a performance assessment will be on the agenda of the project. As part of this a distributed simulation environment will be developed, also allowing to inject errors and failure conditions. The project is currently in its initial phase and will continue until 2015.

Referenzen/References

[1] OMG: Data Distribution Service for Real-time Systems. Standards document V 1.2 (2007)

[2] OMG: The Real-time Publish-Subscribe Wire Protocol, DDS Interoperability Wire Protocol. DDS-RTPS standard specification V2.1, (2010)

[3] OMG: CORBA (Common Object Request Broker) Architecture. http://www.omg.org/spec/CORBA (2012)

[4] Mayer, V.: Evaluierung des DDS Publish/Subscribe Standards und Anwendung auf ein Air Traffic Control – Szenario. Master Thesis, Hochschule Offenburg (2013)

[5] Sesar (Single European Sky ATM Research) Programme, Joint Undertaking Site, http://www.sesarju.eu (2013).

[6] EUROCAE: Flight Object Interoperability Specification. ED-133, (2009)

[7] Eurocontrol: On-Line Data Interchange (OLDI). DPS-ET1.ST06-STD-01-01, Ed. 2.3, (2001)

[8] Eugster et al.: The Many Faces of Publish/Subscribe. ACM Computing Surveys, Vol. 35, (2003)

[9] Kemme, Jiménez-Peris, Patiño-Martíne: Database Replication. Morgan and Claypool Publishers. (2010)

AUTOREN

Prof. Dr. Erwin Mayer
Fakultät Elektrotechnik und Informationstechnik
Lehrgebiete: Computernetze, Verteilte Systeme, Systemprogrammierung
erwin.mayer@hs-offenburg.de

Johannes Fröhlich, B.Sc.
Angewandte Informatik
Wissenschaftl. Mitarbeiter
johannes.froehlich@hs-offenburg.de

Mehran Shakeri, B.Sc.
Computer Engineering
Wissenschaftl. Mitarbeiter
mehran.shakeri@hs-offenburg.de