Astronomical Data Analysis Software and Systems IX
ASP Conference Series, Vol. 216, 2000
N. Manset, C. Veillet, and D. Crabtree, eds.

Object Oriented Near Real Time Packet Distribution as Part of the FIRST Integrated Network and Archive System

E. Wieprecht, E. Wiezorrek, O. H. Bauer

Max Planck Instititut fuer extraterrestrische Physik, Garching, Germany

R. Huygen, B. Vandenbussche

Institute of Astronomy K. U. Leuven, Leuven, Belgium

P. Claes, J. J. Mathieu

ESA - Astrophysics Division, Noordwijk, Netherlands

Abstract. The Far InfraRed and Submillimetre Telescope (FIRST) is the fourth cornerstone mission in the original European Space Agency's HORIZON 2000 program. The Integrated Data Archive System (FINDAS) will be the base of the distributed ground segment.

This paper shows aspects of the development of a prototype application to test the feasibility of near real time distribution of Telemetry Packets to the different instrument control centers. FINDAS is based on a 3-Tier Architecture. We discuss the development of the middleware component (an object server) and front end (a real time client application).

1. The Far InfraRed and Submillimetre Telescope - FIRST

FIRST will be a 3.5 m telescope, covering the wavelength range of 80 to 670 μm and hosting three scientific instruments: the heterodyne instrument (HIFI), the photoconductor instrument (PACS) and the bolometer instrument (SPIRE).

Currently the launch is foreseen in early 2007. The satellite will orbit the second Lagrangian Point (L2) and will operate for about 4.5 years.

2. The FIRST Ground Segment

The FIRST ground segment consists of five major parts: The Mission Operations Center (MOC) for operating the satellite, the FIRST Science Center (FSC) for coordination of overall science activity and the three Instrument Control Centers (ICCs) for operation and characterization of the instruments.

For FIRST a novel geographically distributed ground segment is envisaged. Mission operations center, science center and instrument teams will not be at the same location. Goal is to reduce operational costs by avoiding traveling of experts for various aspects of the mission.

3. The FIRST Integrated Network Data Archive System

The FIRST ground segment will be based on the object oriented FIRST Integrated Network and Data Archive System - FINDAS. It will manage all information relevant to the FIRST project throughout all phases of the mission, from spacecraft/instrument development, through all tests, checkout, operations and post mission phase inclusive archiving. FINDAS is not only the repository for all FIRST related information, it also supports:

- Flexible, efficient and controlled access to up-to-date FIRST information, data and software for all ground segment elements,
- Homogeneous configuration management for all information, data and software,
- Homogeneous data analysis environment for development and operations for the different ground segment elements,
- Homogeneous data access and analysis environment for observers.

4. Real-Time Operations

The architecture of the operational environment will be kept similar throughout all phases of the development. Some elements in the FIRST ground segment will not be available in the early phases like ground tests on the integrated instrument level. Their places will be taken by mock-ups, pieces of software which perform a subset of their tasks. As test procedures require permanent online monitoring it is essential to have a near real time data access. This allows immediate recognition of abnormal behavior of the instrument and instantaneous reactions in the case of critical situations. Also "manual commanding" and execution of the command will be verified in real time. Furthermore the instantaneous data access allows efficient data analysis and test cycle optimization.

5. The FINDAS Real-Time Client Prototype

Goal of the prototype is to provide all players in the ground segment with the possibility to gain experience with a distributed object oriented database system, to get a feeling of the technological feasibility of the concept and to refine the user requirements of the whole system.

A real time application was chosen in order to demonstrate the technological feasibility of displaying real time data from FINDAS during ground based instrument tests and during certain phases of the mission rather than listening for telemetry packets "online".

5.1. Concept

A 3-tier architecture approach was chosen. The user interface is the TMViewer. The TMViewer communicates with an object server. The object server is extracting telemetry packets from the object database and is hiding the physical and logical structure of the database for the client application (TMViewer).

5.2. Test Goals

The TMViewer client should prove that a system build on this architecture can provide us with packets within the time frame we envisage, one 65 kB TM-Packet every two seconds up to eight 4 kB packets every single second. In order to verify this, the following issues are tested while retrieving telemetry from the FINDAS prototype: the integrity of each packet, the order of retrieval and missing packets, the performance under different circumstances (turn around time between instrument status change and update of the view in the TMViewer), to connect to database with multiple clients, different data rates (200 Kbits/sec to 4MBits/sec), packet sizes (4 kB to 65 kB) and packet modes, has the size of the DB or complexity of the data model an influence on the performance?

5.3. Remote Object Access

For remote object access, necessary for the communication between the three players in this architecture, several possibilities were explored:

- Simple socket programming which is in general very effective - but not well suited for complex data types,
- Simple Remote Procedure Calls (RPCs) which is based on sockets, however mostly easier,
- Distributed Component Object Model (DCOM) from Microsoft - which is operating system dependent,
- Common Object Request Broker Architecture (CORBA) which is platform independent and support JAVA and C++. But the implementation needs some effort,
- Remote Method Invocation (RMI) - which is platform independent and easy to implement. But client/server applications have to be written in JAVA.

For the prototype of the real time TMViewer the combination C++ / CORBA and JAVA/RMI have been tested. We used the ORBIX CORBA from IONA (version 2.7/3.0) which is supporting CORBA (version2.0/2.1) and JAVA version 1.2.

6. Real Time Data Access from the Database

A difficult trade-off to make is the eagerness of the different components. The object server could be constantly "polling", i.e. query for the next object, if not available wait and try again. Some simple intelligence can be built in to optimize the frequency of issuing a new query based on the frequency of success. Another possibility is to use notification/signal mechanisms of modern OO databases triggering a signal upon the creation of a new instance of class.

The latter solution might be better to avoid unnecessary load on the database, but the first solution might result in a better throughput and response time if the data rate is close to the capacity of the system.

7. Multiple Development on a Single Data Model

The development of applications to test the FINDAS prototype is also used to learn how to co-operate with several groups writing applications that work on objects in a single data model.

For the design of the data-model, UML diagrams are taken as "communication language" between the different parties.

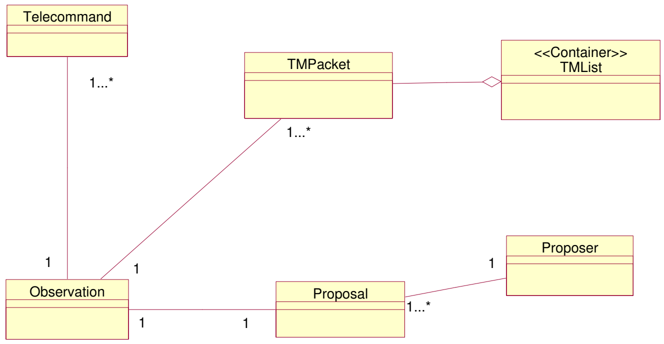

Figure 1. UML: The prototype data model.

The **proposal handling system** builds the front end to the test observer. It creates and maintains **proposals** and **proposal objects**. The **MOC/FSC simulator** converts the **proposal** to an **observation**. The **TMGenerator** is analyzing the **observation** and creates the **TMPackets**. The **Real Time Object Server** polls the DB for new **TMPackets**. If a new **TMPacket** is available it is extracted and the information is displayed by the **TMViewer**, the interface to the test person.

All the components of this scheme are build by different parties. The data model is the base for the developers. See Figure 1.

8. Conclusions

We were able to write a simple application in a 3-Tier architecture in four man weeks (inclusive learning simple basics about remote object access and OODB).The exercise was rather helpful for defining the User Requirements on the whole system. UML communication was essential for the distributed s/w development process. A software tool supporting UML (e.g. Rational Rose) proved to be useful.The C++/CORBA requires more learning/understanding then JAVA/RMI. We were troubled by ORBIX2.7/CORBA2.0 - ORBIX3.0/CORBA2.1 inconsistencies. Until now, we only tested the polling mechanism. Further tests, based on this configuration, will show us whether the technology used is suitable to serve as backbone for a scientific satellite ground segment.

Acknowledgments. Ekkehard Wieprecht acknowledges financial support from the Deutsche Forschungsgemeinschaft (DFG).