Nationaal Lucht- en Ruimtevaartlaboratorium
National Aerospace Laboratory NLR

NLR TP 96411

Developing the EATMS architecture: think big - start small

P. Dieleman and E. Kesseler

DOCUMENT CONTROL SHEET

	ORIGINATOR'S REF. **TP 96411 U**		SECURITY CLASS. **Unclassified**

ORIGINATOR
National Aerospace Laboratory NLR, Amsterdam, The Netherlands

TITLE
Developing the EATMS architecture: think big - start small

PRESENTED AT
the "EATMS Architecture Workshop", June 11-13, 1996

AUTHORS **P. Dieleman and E. Kesseler**	DATE **960606**	pp **15**	ref **17**

DESCRIPTORS

Air traffic control	**Object-oriented programming**
Applications programs (computers)	**Protocol (computers)**
Architecture (computers)	**Real time operation**
Data management	**Systems engineering**
Distributed processing	**Systems integration**
Europe	**User requirements**
Flight plans	

ABSTRACT
The European Air Traffic Management System is characterised by a floating initial situation and continuous change because of the autonomous and evolutionary nature of its components. This is a key driver in the specification and design of the EATMS architecture.
The EATMS development strategy should be based on a clear operational philosophy and a clear realisation process. This will result in clear appropriate design guidelines. Appropriate modelling techniques should capture all system aspects that impact the EATMS architecture (e.g. by using the 5 views of the RM-ODP). In other words: "think big". At this moment the applicability of standards, primarily RM-ODP and CORBA, for the EATMS architecture development is uncertain. The feasibility of applying these standards is to be determined, for instance by developing prototypes in a style comparable to the Annette project. In other words: "start small". During the implementation of parts of the EATMS, every part should fit within the vision embodied in the operational philosophy, overall design and adopted process for realisation ("think big"), but should have a short realisation elapse time, providing short term return of investment ("start small").

Contents

2 Figures

(15 pages in total)

This page is intentionally left blank.

Developing the EATMS Architecture: Think Big - Start Small

Peter Dieleman, Ernst Kesseler
(E-mail: dieleman@nlr.nl, kesseler@nlr.nl)
National Aerospace Laboratory NLR,
Informatics Division,
P.O. Box 90502
1059 CM Amsterdam
the Netherlands

1. INTRODUCTION

The European air traffic has shown a steady and significant increase over the last years. This substantial increase is expected to continue for the foreseeable future. Consequently, the capacity of the European air traffic management system must increase accordingly, while maintaining or even improving on the good current level of safety. The changes in the airline industry, like the deregulation of the air traffic within the European Union due to the single market, have led to the additional requirement of maximising the cost effectiveness of the provided air traffic management services. This provides an important example of a change towards satisfying the customers needs. Due to the changes in the operation of the air transport industry, environment, like the impact of increasing environmental restriction, the speed of changes in the customers need has accelerated over the last years. Consequently the European Air Traffic Management (ATM) system should respond quicker to changes in its operating environment. This can also be worded as a move away from the traditional technology push towards market orientation. This trend also exemplifies the maturity of the air transport industry.

Eurocontrol has responded to this challenge (which can, or even should, be seen as a business opportunity) by the European Civil Aviation Council (ECAC) strategy for the 90's [1]. The prime solution is the proposed European Air Traffic Management System (EATMS), with the objective of reaching harmonisation and operational integration within the ECAC airspace. EATMS is a long term objective, with implementation in 2005 and beyond. EATMS can also be considered the fourth step of the European ATC Harmonisation and Integration Programme (EATCHIP) strategy, the third step of which is currently being implemented [2], after successful completion of the first steps. The EATMS is based on the following constraints:

- the ECAC states are autonomous. This implies different technical systems from different vendors as well as differences in the organisational, judicial and operating environments. These different systems form the constituent parts of the EATMS.

- from the airspace user point of view, EATMS has to provide a seamless continuous service for the entire ECAC geographical area during its continuous evolution.

- EATMS will be a distributed real time system. Due to its safety critical nature there are strong requirements on continuous availability and integrity.

- the EATMS will have to comply with the international standards like the International Civil Aviation Organisation (ICAO) Communication Navigation Surveillance(CNS)/ATM standard, and ICAO Standards And Recommended Practices (SARPS), as well as be able to interface with the disparate equipment and procedures of the adjacent states and the military. These interfaces will evolve continuously.

- due to the autonomy of the ECAC states the evolution from the current situation towards the EATMS will proceed at different speeds for the different geographical areas. Consequently there is a requirement for inter-operation of ATM systems at centre level with varying levels of sophistication. The requirement for continuous availability yields the need to support a diverse, ever changing configuration of system components (i.e. national ATM systems or their constituent subsystems).

These assumptions result in strong requirements on the EATMS architecture. Within the Open Distributed Processing (ODP) standard, ISO 10746 [7], these assumptions are worded as the characteristics heterogeneity, autonomy, evolution. Due to the nature of air traffic the remaining ODP characteristic mobility is inherently applicable. The EATMS system also exhibits all inherent characteristics of distributed systems mentioned in the ODP standard, such as remoteness, concurrency, lack of global state, independent failure of system components and asynchrony.

2. EXPERIENCE GAINED FROM ATM/CNS DEVELOPMENT, USING C/S TECHNOLOGY

When the multiformity of the European situation is not taken into account, the results of EATMS development will not get the required multi-national support. This is exemplified by the Air Traffic Land and Air Study (ATLAS) project, which, according to its charter, had to base itself on a single unified system in stead of a harmonised solution. ATLAS did not obtain its recognition even though various elements of the study approach as well as thoughts on new operational concepts like autonomous aircraft have influenced subsequent studies and developments. The lessons learned from this project are:

- system harmonisation is the way to go, above a single unified system
- a unified operational concept is needed as a basis for the harmonised implementations
- the approach to separate mission, objectives (or requirements) phases from the subsequent design (architecture) and implementation phases is advantageous
- a new ATM system should base itself on the evolution from the current situation. The clean sheet of paper approach (a client requested prerequisite of the ATLAS study) is not compatible with the European realities
- a single Yourdon SA/RT (Structured Analysis with Real Time extensions) based model is better then the two separate models used, one containing a pure functional decomposition while the other contains the data model. With the current state of the information technology object oriented methods might be more appropriate than structured analysis based methods. To capture all views of the single unified system, covering all relevant aspects of the ATM organisation, a single model will be either too generic to provide much detailed information or will become unmanageably large. Within the Open Distributed Processing Reference Model (RM-ODP, [7]) this is taken into account by identifying viewpoints (or specifications).

The current EATMS approach followed by EUROCONTROL takes these lessons into account.

The Common Modular Simulator (CMS, [14]), being part of the Programme for Harmonised Air Traffic Management Research in Eurocontrol (PHARE), aimed at providing a unified ATM research environment for the participating European countries (i.e. research establishments). The participating countries experience the highest current air traffic load. After consensus on the user requirements, the implementation was based on a single language. Consequently the continuous operation of a lot of already existing software could not be guaranteed. Hence the resulting implementation did not meet the expectations. The lessons learned are:

- harmonisation of requirements works well
- the diversity provided by local implementations (idiosyncrasies for outsiders) needs to be respected.

Within the Annette project [15] a distributed simulation has been set up between four European ATM research establishments. Standard (Internet) technologies were used. An interface was designed, limited to the functions needed for the single demonstration involving all 4 sites. Every partner was responsible for the local implementation of the common agreed interface specification. This approach resulted in a short turn around time for the demonstration. The lessons learned are:

- using (industry) standard technology helps
- concentrate on common agreed interfaces
- checking compliance to the agreed interface specification is important
- the CMS breakdown of an ATM simulator was used for Annette and worked well
- the restricted requirements allowed a simple implementation which could be realised quickly, provided immediate feed back.

The NLR ATC Research SIMulator (NARSIM, [11]) is currently being upgraded. The present NARSIM structure is based on old software technologies and shows the signs of the continuous upgrading of its capabilities during its operational life. The restructured NARSIM, called NARSIM C/S (client server) is compliant with the CMS breakdown. Taking the above mentioned lessons learned into account, NARSIM C/S uses a middleware approach to connect the existing software modules. This middleware should allow future application software modules from various sources to be added. The middleware takes care of all information exchange between the application software modules (called servers), thereby making each application software module more independent of the remaining software. It also improves the visibility of the interfaces. The lessons learned from this project are:

- the CMS partitioning works
- middleware technology used for NARSIM C/S enables distributed processing
- for interoperation of architectural components information technology standards are needed
- it is very important to agree upon the interfaces of the application modules
- the resource usage of the middleware is commensurate with real time ATM simulation on the current NARSIM hardware platform. However to achieve this performance for some servers intelligent agents (i.e. client stubs) are needed.

The EURATN (European Aeronautical Telecommunication Network) project [17] has provided the first implementation of an Aeronautical Telecommunication Network (ATN) compliant with the ISO/Open Systems Interconnection (OSI) standards. The EURATN project produced a working system. However due to the new technologies involved it took a long time to obtain results. As a result some of the standards used are (from a information technology point of view) obsolete, before the implementation had been completed. The EURATN lessons learned are:

- use standards, preferably industry or de-facto standards with sufficient (commercial) support (i.e. use Common-Of-The-Shelf (COTS) software)
- system evolution has to proceed with manageable steps, each with an implementation time which takes the general progress in industry into account. Smaller steps resulting in intermediate products which provide immediate benefits to the customers also allow quicker return of investment, that is “think big” (provide a vision for the future), “start small” (short implementation elapse time, combined with immediate benefits).

3. EATMS ARCHITECTURE SCOPE AND REQUIREMENTS

The motivation and objectives for developing EATMS are outlined in the EATMS Mission, Objectives and Strategy Document [4]. Key objectives are safety, capacity, efficiency, cost effectiveness, uniformity and environment. To achieve these objectives for a complex system like EATMS the system architecture plays a crucial role. The present paper primarily focuses on aspects of the software architecture.

The term (software) architecture has many definitions. Here we take as a starting point “The set of rules to define the structure of a system and the interrelationships between its parts” as defined in [7, part 2].
These rules should translate the EATMS keywords listed above into workable concepts and methods to model and structure the EATMS system. This includes the engineering of internal and external interfaces and development of mechanisms and techniques for interworking of its structural parts.

Drivers for the EATMS architecture and the architecture development process are its overall context, the operational approach foreseen, the envisaged building blocks and the policy towards existing systems.

The EATMS context focuses on the services as provided by EATMS as a whole, including humans, machines and their respective procedures, and their relation with EATMS-external elements (i.e users and providers), which are (see [5]):

- airspace users (or : “flight”)
- aerodromes
- operator
- other parties, such as:
  - meteorology
  - air defence systems
  - ATC units of adjacent airspace

This identifies the role of the system in support of air transport industry, and sets the functional and geographical boundaries for the EATMS architecture.

From both the operational ATM concepts for EATCHIP III [2] and EATMS [3, initial version] it can be concluded that the CNS/ATM architecture should enable interoperability in different ways:

- Between ATM planning layers (phases within the planning process, with varying time horizons, between long term strategic planning, i.e. half a year, up to tactical control, i.e. seconds to minutes), and the associated control loops
- Between Air Traffic Controller (ATCo) operations and automated system operations
- Between multiple ATCos operations, mediated through the automated system
- Between systems operations (including Air/Ground)

The interoperability should prevent EATMS from being a set of stand-alone automated tools requiring humans to draw conclusions and take actions based on diverse information provided. The EATMS concept of Flight Phases as described in [5] is of relevance for the architecture due to requirements on dynamics and timing aspects for EATMS services. An overview of the required services, grouped per user category is given in [5, figure 7].

EATMS can be decomposed into a functional hierarchical tree yielding a set of building blocks (components/units) that constitute the system [5, figure 8].

From a technology point of view the transition from, and partial incorporation of, existing (legacy) systems as part of an evolving EATMS is an important factor in the transition process from EATCHIP III. The system architecture is to support this process. This sets a strong requirement on the system architecture, especially in the context of the assumed autonomy and continuous evolution of system elements.

An initial set of functional and non-functional specifications can be derived for the identified EATMS building blocks. Functional specifications state what services have to be provided. Non-functional specifications list the constraints for the services, that may have an impact on the required Quality of Service (QoS), such as:

- Availability
- Accuracy
- Timeliness
- Reliability
- Resilience

The QoS provided by the EATMS to its users could be expressed in an overall QoS metric. The Quality of Service can be considered as a metric for architectural quality. This overall QoS is a combination of the QoS of each major service within EATMS. During the design of EATMS, information on the resulting QoS of the system should be available to optimise the EATMS architecture and its constituent servers.

Apart from above characteristics the EATMS architecture should exhibit characteristics which make it an open distributed system (for detailed specification see [7]):

- Openness
- Integration
- Flexibility
- Modularity
- Manageability
- Security
- Transparency

4. EATMS ARCHITECTURE DEVELOPMENT PROCESS

As user requirements, operational concept and functional specifications, of the system are expected to change continuously over the EATMS life cycle, the architecture development process should not focus too much on present specifications.

The approach to the software architecture development process should take as a starting point:

- EATMS Context and Scope [5]
- EATMS Operational Concept and Requirements
- Realistic assumptions on the general nature of the CNS/ATM system (e.g. distribution, networking, legacy, heterogeneity)
- Basic information technology design principles, based on the object-oriented paradigm using the object model (including concepts such as abstraction, encapsulation, see also [13])
- Basic software architecture approach and mechanisms, based on accepted and proven modelling concepts and vendor-independent technology.

The modelling of EATMS in accordance with the Reference Model for Open Distributed Processing (RM-ODP, [7]), is considered to be a sound basis for describing the system using the different viewpoints it advocates. The elaboration of the RM-ODP and the viewpoints used in it is beyond the scope of this paper. At present NLR has no practical experience yet in applying RM-ODP in ATM projects. Based on a study of the documentation it was preliminary concluded that:

- Information and Computational viewpoints are crucial to specify the functional architecture of the system, As the two viewpoints seem to be non-orthogonal they may be combined into a single model.
- The Engineering viewpoint is essential to specify the logical architecture.
- The Technology viewpoint is closely linked to the implementation process and somewhat separate from the design, although technology is a key factor in achieving the required Quality of Service (QoS).
- The detail of modelling should be driven by the design efficiency, i.e. focus on models that directly contribute to realising the system objectives.
- No specific methods (e.g. OMT or Shlear&Mellor) are prescribed. These are considered (including associated tools) important success factors when working in international teams.

Although the RM-ODP provides a complete set of views to model a complex system like EATMS, it does not provide the model for the complete life cycle of an information system. The Enterprise viewpoint is concerned with the business activities of the specified system. This implies a definition of what will be part of the system and what will become its environment, the system objectives, its stakeholders, etc. The Information, Computational, Engineering and Technology viewpoints detail the solution to be provided by the system from their respective viewpoints. It is not clear where this leaves the ATM operational concept [2, 3], which satisfies the system objectives but allows several realisations (or system implementations). The construction of the Information, Computational, Engineering and Technology views will be a sequential activity. However, for large systems such as EATMS, several iterations impacting multiple views are to be expected during the realisation process.

From these models covering the complete system (“think big”) implementation can start. It is

expected that several prototyping activities will take place to verify and validate the design ("start small"). It is recommended to perform this prototyping at a limited scale during the modelling (e.g. concerning system performance). Actual implementation should be carried out based on a transition plan that introduces the new architecture into the existing CNS/ATM systems and replaces systems with newly developed systems.
Traceability of requirements (including mapping of RM-ODP views) and appropriate standardised documentation should be handled and managed throughout the life cycle to assure design consistency, and ensures that the system complies with its requirements.

5. EATMS ARCHITECTURAL SOLUTION: AN OBJECT-ORIENTED APPROACH

Whereas RM-ODP does not prescribe an object-oriented approach, this section takes this approach as a starting point. This is motivated by the well-accepted theoretical background of object orientation (e.g. [13]), the EATMS assumptions of evolution and autonomy (see section 2) which are well-supported by the object oriented paradigm and by the notion that object technology is, and becoming more and more, widely accepted by industry by enabling software re-use, simplified maintenance and fast-prototyping support.

A system having the characteristics of EATMS (e.g. distributed, heterogeneous, starting from legacy systems) should benefit from the conceptual object model and architectural reference model, called Object Management Architecture (OMA), as developed by the Object Management Group (OMG).

The Object Management Group (OMG) is dedicated to producing a framework and specifications for commercially available object-oriented environments. The Object Management Architecture Guide [6] provides an architecture with terms and definitions upon which all supporting interface specifications are to be based. Part of this architecture is a Reference Model which classifies the components, interfaces and protocols which compose an object system architecture into four areas (see figure 1):

■ The **Object Request Broker (ORB)** enables objects to make and receive requests transparently in a distributed environment. It is the foundation for building applications from distributed objects and for interoperability in homo- and heterogeneous environments. The technology adopted for ORBs is known as the Common Object Request Broker Architecture (CORBA, see [8]).
The ORB establishes the client-server relationships between objects. Using the ORB, a client can invoke a method on a server object transparently, which can be on the same machine, or across a network. The ORB is responsible for all the mechanisms required to find the object implementation for the request, to prepare the object implementation to receive the request, and to communicate the data making up the request.
The interface the client sees is completely independent of where the object is located, what programming language it is implemented in, or any other aspect which is not reflected in the object interface. A server interface defines the server object based on the Interface Definition Language (IDL), which is programming language and network neutral. A detailed description is considered to be beyond the scope of this paper.

■ **Object Services** is a collection of services (interfaces and objects) that support basic functions for using and implementing objects. These services are necessary to construct any distributed application and are always independent of application domains. The Common Object Services Specification (COSS) include services for (see [10] for details): Object naming, object event notification, object lifecycle, object persistence, concurrency control, externalisation, object relationship, object transaction, object security, object time. For example, the life cycle service defines conventions for creating, deleting, copying and moving objects; it does not limit objects implementation for a specific application.

- **Common Facilities** is a collection of services that many applications may share, but which are not fundamental as the Object Services (see [9]). For instance, a system management facility could be classified as a horizontal (i.e. market/application independent) common facility. CNS/ATM services, such as Flight Plan services, that are used by many clients in the CNS/ATM application domain, are candidates to be implemented as vertical (i.e. market/application specific) common facility.

- **Application objects** are products of a single vendor or in-house development group which control their interfaces. Application Objects correspond to the traditional notion of applications, so they can not be standardised by the OMG. Instead, Application Objects constitute the uppermost layer of the OMA Reference Model. The granularity of the application objects can be quite diverse (e.g. a single radar could be considered as a relatively small application object, while also some complete ATC centre could be treated as a relatively large application object). The granularity applied is a design decision, which is partly driven by the prescribed use of legacy systems forming part of EATMS.

Fig. 1 The Object Management Architecture Reference Model

Interoperability requires standardisation of formats and interfacing conventions between software systems. Within CORBA ORB-interoperability specifies a comprehensive, flexible approach to support networks of objects that are distributed across and managed by multiple, heterogeneous CORBA compliant ORBs [8]. The ORB Interoperability Architecture provides a conceptual framework for defining the elements of interoperability and for identifying the compliant points. The architecture clearly identifies the roles of different domains of ORB-specific information.

The key element of ORB interoperability is that objects are enabled to make and receive requests transparently in a distributed environment, even if the objects are implemented on different ORBs.
When an interaction takes place across a (ORB-)domain boundary, a mapping, or bridge is required to transform relevant elements of interaction as they traverse the boundary (see figure 2). There are essentially two approaches achieving this: mediated bridging and immediate bridging:

- With mediated bridging, elements of the interaction relevant to the domain are transformed at the boundary of each domain, between the internal form of that domain and an agreed common form.
- With immediate bridging, elements of the interaction relevant to the domain are transformed at the boundary of each domain, directly between the internal form of one domain and the internal form of the other.

By the use of bridging techniques, objects can interoperate via multiple ORBs, without the ORBs involved knowing any details of each others implementation, such as what particular IPC or protocols are used to implement the CORBA specification.

As OMG has all major computer and software vendors as members the standards set are major candidates for becoming de-facto standards. Also a wide range of implementations of the standard by different vendors will result in semi vendor-independent software which is mutually compatible. This makes CORBA a candidate for usage as COTS in support of EATMS architecture development.

Fig. 2 Generic ORB-to-ORB Bridging

An important requirement for EATMS is to support legacy system migration from EATCHIP III, such that the migrated target system remains fully operational in new environments. CORBA provides a distributed object-oriented approach to integrate legacy applications and enable developers to write an object wrapper to encapsulate the legacy system. With CORBA, the developer writes a description of the service provided by the legacy system using IDL, then writes code which invokes the appropriate set of actions in the legacy application when one of the object wrapper's methods is invoked. This approach has been proven efficient and effective [12].

At NLR a project has started early this year in which a CORBA 2.0 implementation (named ORBIX) is used to verify the CORBA concepts and aspects as addresses above.

6. CONCLUSIONS

The EATMS system is characterised by a floating initial situation and continuos change because of the autonomous and evolutionary nature of its components. This is a key driver in the specification and design of the EATMS architecture.

The EATMS development strategy should be based on a clear operational philosophy and a clear realisation process. This will result in clear appropriate design guidelines. Appropriate modelling techniques should capture all system aspects that impact the EATMS architecture (e.g. by using the 5 views of the RM-ODP). In other words: “think big”. At this moment the applicability of standards, primarily RM-ODP and CORBA, for the EATMS architecture development is uncertain. The feasibility of applying these standards is to be determined, for instance by developing prototypes in a style comparable to the Annette project. In other words: “start small”. During the implementation of parts of the EATMS, every part should fit within the vision embodied in the operational philosophy, overall design and adopted process for realisation (“think big”), but should have a short realisation elapse time, providing short term return of investment (“start small”).

The applicability of standards, primarily RM-ODP and CORBA, for EATMS development should be verified. Note that these standards should also be considered for building a new generation of CNS/ATM simulators.

The Quality of Service (QoS) provided by the EATMS to its users could be expressed in an overall QoS metric. This overall QoS is a combination of the QoS for each major service within EATMS. During the design of EATMS, information on the resulting QoS of the system should be available to optimise the EATMS architecture and its constituent servers.

The EATMS architectural framework should be implemented using - where possible - common-of-the-shelf systems, which are vendor independent (i.e. use a - possibly de-facto - standard), are proven technology and meet the baseline EATMS architectural requirements.

CNS/ATM Common Facilities should be developed in case CORBA-based technology is adopted. These CNS/ATM specific facilities could than act as generic ‘vertical' services within EATMS, and could be based on existing specification or even implementations of existing services.

7. REFERENCES

[1] Air Traffic Control in Europe, ECAC Strategy for the 1990’s, Meeting of ECAC Transport Ministers, Paris, 24 April 1990
[2] EATCHIP Operation Concept Document for the EATCHIP Phase III System Generation, EATCHIP Doc: OPR.ET1.ST02.1000-OCD-01-00, Edition 1.1, 28 June 1995
[3] Concept for the European Air Traffic Control Management System (EATMS), Initial

Operational Concept Document, Eurocontrol Doc.No. 937044, Issue 1.2, 15 October 1993
[4] EATMS : Mission, Objectives and Strategy Document, Draft 0.D, 9 June 1994
[5] EATMS : Context and Scope Document, EATCHIP Doc: FCO.ET1.ST02.DEL01, Edition 1.0, 15 September 1995
[6] Soley, R.M. (ed.) and Stone, Ch.M., The Object Management Architecture Guide, John Wiley & Sons, third edition, 13 June 1995
[7] Information technology - Basic Reference Model of Open Distributed Processing, Draft International Standard ITU-T Rec. X.901-i | ISO/IEC DIS 10746-i (four parts),1994/1995
[8] Object Management Group, The Common Object Request Broker: Architecture and Specification, Revision 2.0, July 1995.
[9] Object Management Group, Common Facilities Architecture, Revision 4.0, OMG Document 95-1-2, 3 January 1995
[10] Object Management Group, Common Object Services, Volume I, Revision 1.0, First Edition, 1 March 1994
[11] Michiels, R., A survey of the NARSIM C/S middleware, NLR TP 96220, April 1996
[12] Mowbray, T. J., Zahavi, R., The Essential CORBA; Systems Integration Using Distributed Objects, John Wiley & Sons, 1995
[13] Booch, G., Object Oriented Analysis and Design with Application, Second Edition, The Benjamin/Cummings Publishing Company, 1994
[14] Welman, C.G.M. (ed.), EUROCONTROL Common Modular Simulator, Architecture and Application Programming Interfaces of PARADISE, V1R3, 16 July 1993
[15] Reijers H.A. et.al., Annette: A Feasibility Study for an Air Traffic Management Research Network - Work Accomplished, NLR CR95072 L, January 1996
[17] EURATN, System Implementation Specification, version 3, 22 October 1993