AEROSTATION – A CORBA COMPONENT APPROACH TO THE AERODYNAMIC DESIGN FRAMEWORK

Christelle Casties *, Alain Soulard *, Eric Chaput *, Luis Barrera *, Jérôme Huchard ~

*** AEROSPATIALE MATRA AIRBUS**
316, route de Bayonne 31060 Toulouse Cedex03, France
~ Sema Group
56, rue Roger-Salengro 94126 Fontenay-sous-bois Cedex, France

Keywords: *CORBA,UML, wrapping, CFD, framework, DAMAS*

Abstract

An open architecture for a component framework was defined and implemented at the AEROSPATIALE MATRA AIRBUS aerodynamics department to provide traceability of the whole process by embedding organisation tools, commercial codes and in-house codes, and to free end-users from information technology details.

The objectives of this framework are to accelerate the engineering of research outcomes by allowing existing codes reusability, as well as to reduce the cost of development by shortening the aerodynamics design cycle by 30%.

This paper presents the methodology used for designing and deploying the aerodynamics design framework, the AeroStation.

1 Introduction

The worldwide competition between aircraft manufacturers puts strong pressure and high priority on the capability to adapt the design process and tools in order to reduce time cycle and cost for a new aircraft development.

At the same time, highly challenging projects, like the Airbus A3XX demand even more accurate and more reliable design tools. The design engineers need to be confident with their performance predictions at an early stage of aircraft development.

The aerodynamics department of AEROSPATIALE MATRA AIRBUS, anticipated, in the early 1990's, a dramatic evolution in the development of design tools and especially in the way numerical codes for computational fluid dynamics (CFD) could be more efficiently embedded in the engineering environment.

The increasing complexity of such an environment, developing interoperability between large software like CAD systems, mesh generator, CFD codes and complex computational flow post-processing (see Figure 1), forced us to consider a new approach for engineering applications.

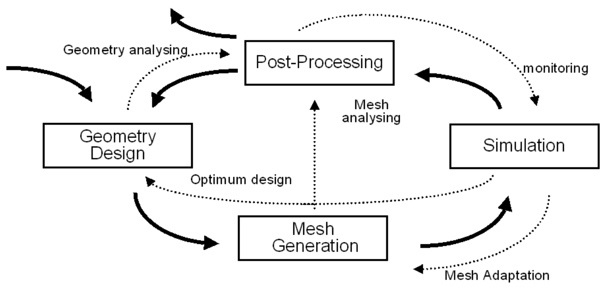


Figure 1: Interoperability between tools within aerodynamic design process

This was amplified by the outsourcing of codes development to research centers like ONERA and CERFACS. This decision was taken in order to concentrate the effort of industry on engineering activities, but keeping the means to remain the architect of our design tools.

The emphasis was put on the capability to mix and match best in class software components coming from different providers within an open architecture allowing the process reengineering on demand, and its traceability: management of links between design constraints, design products and design tools.

Process reengineering and traceability were considered of utmost importance within a multi-disciplinary and multi-site process, like it has to be managed for the aerodynamics design of the Airbus aircraft.

A wide range of tools is needed for design and these tools are continuously improved taking advantage of new research findings without additional work for end-users.

A system approach was thus adopted for building the aerodynamic design environment, the so-called *AeroStation*, within the Softair0 project (1995-1998).

The specifications of the project were to make an intensive exploitation of CFD codes by design engineers easier, providing a framework for "plugging and playing" any new tool and new process in order to inherit of existing services.

The provision for future implementation of processes like aerodynamics shape design and management, mesh generation and management, computation environment and exploitation of results with new generation of object-oriented CFD codes, made the temptation to be exhaustive in the specification of services strong, prescribing a lot of interfaces for the framework.

CORBA emerging technology was considered by AEROSPATIALE and Sema Group, by the end of 1994, to be a good candidate for the development of this aerodynamic design framework. The interface definition language (IDL) was seen to be well suited to a detailed specification of services and a full-framework was designed in order to allow a fine grain integration of processes and tools.

The deployment of *AeroStation* within the aerodynamics department started within the MEDEA project (1998-2001). This project deals with the systematic reengineering and integration of design processes or sub-processes in the *AeroStation*.

Section 2 of this paper gives a description of the architecture: design principles, component library, and component integration techniques adopted for the *AeroStation* framework.

The integration of in-house processes within the framework is then illustrated in Section 3 through the example of the simulation process reengineering.

Section 4 details the current deployment of the framework within the aerodynamics department, giving information about engineering environments already in use and the requirement and constraints on software and hardware architecture. Finally, the perspective for multi-site applications and new process integration are examined.

2 Architecture

The architecture of this engineering environment is based upon the Open Tool Bus (OTB) specification [3], defined by Sema Group with France Telecom's support, and complies with the international CORBA standard for distributed objects.

2.1 Design principles

The goal of the OTB specification is to define the reduced interfaces set between production tools and management tools, needed to manage and improve a production process by the mean of global traceability (see Figure 2).

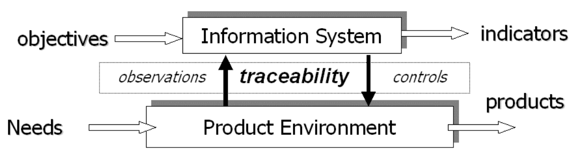


Figure 2: Production process and traceability

The design of the interfaces is based upon the following principles.

- Production tools are responsible for the production and storage of the production objects. Thus the OTB specification does not rely on a common database or a common file format but on common interfaces and services described using the IDL language and object oriented approach.

- Production objects are not only files, they may be fragments of files, objects in a database, or anything else produced by a tool.
- Reliable identification of production objects is a key to optimizing the way process management tools and traceability tools handle production objects.
- Production object evolution is a key to global traceability and an effective update mechanism.
- The traceability tools should have direct access to the production tools in order to display the production objects in their context. Thus what you see in the traceability tools is what you do in the production tools.
- The integration of existing production tool with management and traceability should be easy.

2.2 OTB/Skipper

OTB/Skipper is an implementation of the OTB specification.

To achieve the traceability objectives in a cost-effective way, OTB/Skipper provides an **architecture**, a **library of components** and efficient **wrapping techniques** to integrate tools in the development or design environment (see Figure 3).

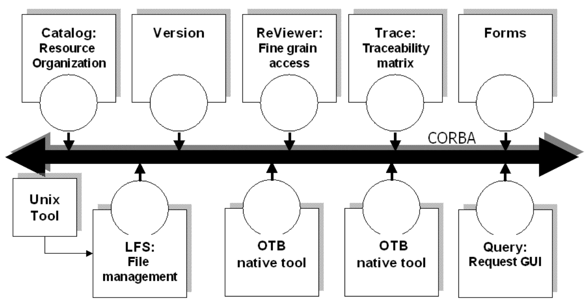


Figure 3: OTB/ Skipper Architecture

OTB/Skipper also provides a **query language** giving access to the whole set of trace links to enable computation of customized indicators and ratios such as requirement coverage, test plan coverage, convergence of maintenance operations (number of fixed and remaining bugs), etc.

With the component library, OTB/Skipper is able to build a full-generalized traceability service customized for a specific project. Customization covers adaptation of the OTB/Skipper components to the specific procedures, constraints and objectives of the project, development of the wrapping layers needed to integrate the tools used by the project, and writing of the needed customized queries.

The component library is made as follow.

- OTB/Session allows the different applications and the different objects of the framework to be accessed.
- OTB/Catalog is an application which presents and structures a group of resources.
- OTB/LFS (Logical File System) is a utility tool that makes all the files appear as full OTB resources.
- OTB/Tracelink allows traceability links between resources to be put.
- OTB/Tracematrix allows the computation of traceability matrices and the performing of coverage analysis.
- OTB/Versions is oriented to evolution link and version management information.
- OTB/Forms is a tool used to describe information with well-defined fields which correspond to precise semantics. It is used to define a task precisely or to summarize the results of a task
- OTB/Query is used to access resources via requests written in a proprietary language: CQL.

2.3 Wrapping integration

The wrapping technique [2] for the integration of an application consists in writing a software layer (wrapping code) to answer the queries addressed by OTB/Skipper to this tool (see Figure 4). These queries will essentially make it possible to:

- start the application and display a production object in its context,

- retrieve all the production objects included in a storage object or a version,
- retrieve the intrinsic links provided by the production application.

The purpose of the wrapping is to make all the information needed to trace the process available to the traceability tools through common interfaces.

Thus the traceability tools query all the production objects in a uniform way without knowing their implementation, storage and so on.

The *AeroStation* uses the libraries and tools provided by OTB/Skipper to wrap the simulation tools, mesh generator, CAD tools...

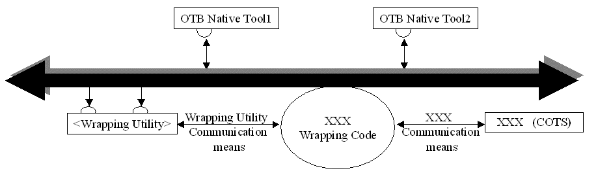


Figure 4: Wrapping method

2.4 Fine grain integration with CORBA objects

Another way to answer OTB/Skipper queries is to create CORBA objects providing OTB services.

In this approach, the application implements CORBA servers and CORBA Objects to provide resources, fine grain access and manipulation through CORBA/IDL API.

The integrated applications chooses the level of integration it provides.

The services offered by the application may include:

- visualization of data in the application context with its GUI,
- semantic manipulation of the data through CORBA/IDL interfaces,
- retrieval of the sub-objects needed for the traceability of the process,
- retrieval of the intrinsic links provided by the application.

Thus the traceability tools are directly connected to the application services by accessing the CORBA objects provided by the application.

3 In-house process integration

Now that the major principles of architecture are established, we can focus on the geometry designer core domain of competence: aerodynamic design process.

These processes represent the department know-how. Its integration in the *AeroStation* framework has three main objectives:

- allow the user to concentrate on aerodynamic issues and increase its efficiency,
- build up experience,
- reduce training time for new engineers.

To explain the methodology defined for any home process integration, we propose to follow the main phases: modeling, rationalization and optimization, traceability and management identification, process integration, through a highly simplified example: the simulation process which seems to be an easily understandable illustration.

3.1 Modeling

First of all, it is very important to analyze the process correctly with the aim of improving the course of the different operations. It consists in user interviews about its habits, its requirements, the tools used and the data handled. To allow several iterations and, in the end, a user validation, the process is represented using Unified Modeling Language (UML) formalism.

Starting from the existing state, the different objects handled, their relations with other objects, their attributes, and their operations are highlighted. This is shown in Figure 5 with the objects involved in the simulation process.

The main objects handled in this process are the structured meshes, made up of blocks and faces, and the calculation corresponding to one flow field solution that has been obtained through different steps. Every step needs both initial conditions and numerical parameters to run and produces a convergence history.

These two major objects constitute the model object. Some entities related to the calculation (numerical boundary conditions, initial conditions, and flow field) are linked to some mesh entities. Moreover, mesh entities have to be frozen before calculation creation in order to ensure the calculation consistency.

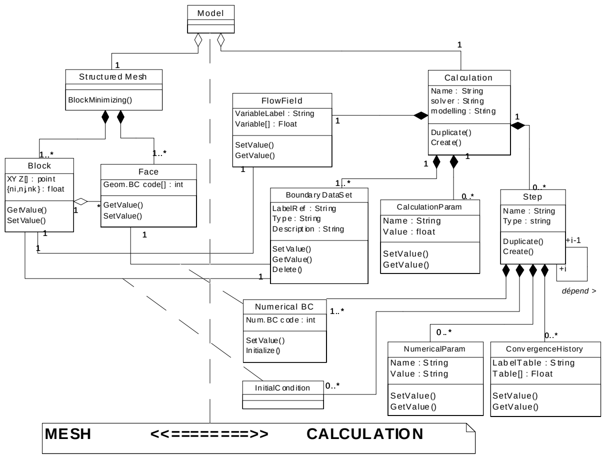


Figure 5: Object model for simulation process

After having described the object structuring aspects, and their relations, attention must be turned to the changes they undergo. To represent the object behavior, we use state diagrams and instance them with an event trace (see Figure 6)

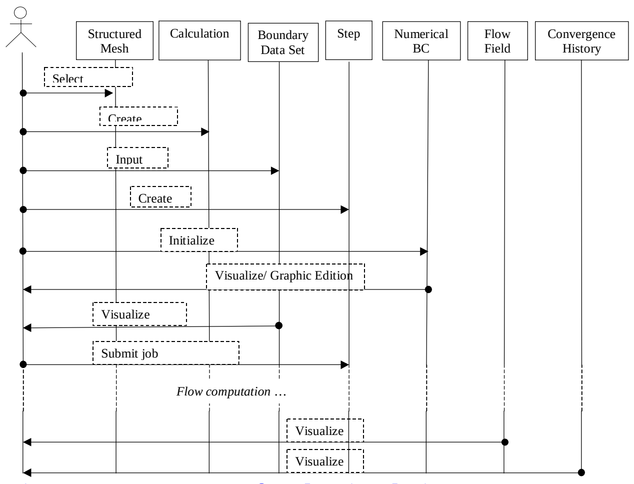


Figure 6: Event trace for the simulation process

This shows that the defined objects are independent of the method used for the calculation (flow solver).

3.2 Rationalization and Optimization

Once the model is written and validated, it is possible to rationalize the process. To do so, we proceed in four directions: data, graphical user interface, scenario and consistency with the other processes.

3.2.1 Data

A key tool of the integration is the sharing of standardized data. To achieve an efficient communication between several applications and to obtain automation, it is necessary to reduce the number of different data structures.

Furthermore, to ensure durability, archiving must be homogeneous.

To solve this problem, six years ago, we have analyzed several market tools. But, noting the lack of standard concerning CFD data, we developed the DAMAS storage device [7].

DAMAS, for *DAta for Meshes and Aerodynamic Solvers* consists of three components (see Figure 7):

- an access method allowing general data to be stored and read efficiently (SDA),
- a scheme representing in a structured way relations and links between entities corresponding to mesh data, simulation parameters and simulation results,
- an application programming interface to handle these data (LIB_DAMAS).

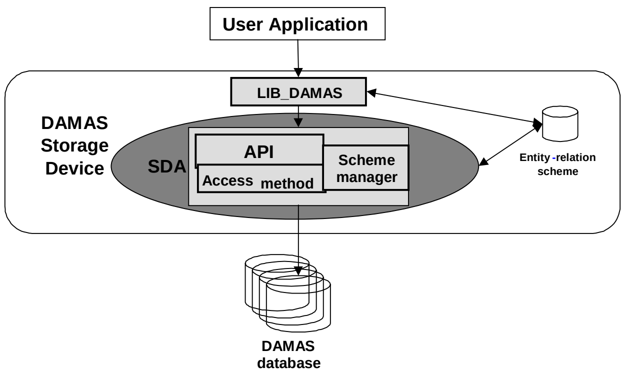


Figure 7: DAMAS principle

The main advantages of DAMAS are :

- consistency,
- extensibility,
- high performance access,
- minimum disk space.

Using DAMAS, we succeed in rationalizing data access for the different tools. The user doesn't need to translate data before using the various tools of aerodynamic processes (see Figure 8).

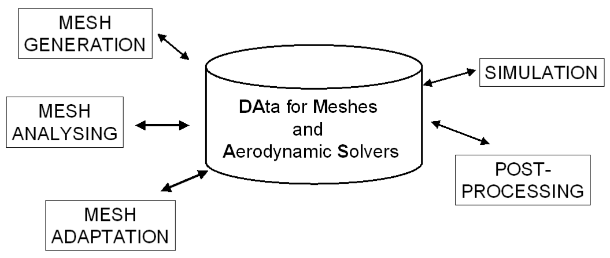


Figure 8: Use of DAMAS in the aerodynamic design cycle

Today, a standardization is becoming mandatory to allow people to work together within different departments and/or companies, so we are continuing to study emerging tools with the aim of replacing DAMAS by an equivalent standard tool.

CGNS, *CFD General Notation System* [4,5] seems to be a good candidate, once performance has been improved and it is accepted as an ISO standard.

3.2.2 Graphic User Interface

Having solved the data storage problem, the next task is to look into the way of chaining the different operations of the process.

The most user-friendly and easy to learn is the interactive way. A Graphical User Interface (GUI), designed as a user guide, can provide an intuitive sequence of actions, controls, and default values preventing bad inputs.

For example, in the simulation process, the GUI allows the user to input simulation parameters, to run a solver and to visualize the convergence, without any knowledge of the different tools.

The first window, see Figure 10, shows the different stages the user must perform:

- select/create/duplicate a Calculation Folder,
- prepare calculation (Boundary Data Sets),
- select/create/duplicate step,
- prepare step (Numerical Parameters),
- submit calculation job.

Figure 10 illustrates the step preparation screen. Most used numerical parameters can be set on the main screen, whereas a second screen is used for more advanced parameters (occasionally modified). The use of 3D graphic tools to edit numerical boundary conditions also helps to enhance the efficiency of user input.

3.2.3 Scenario

For a more experimented user, the scenario is a far more efficient tool.

In fact, several iterations are required, comparisons are needed, and the tasks are not always done sequentially, so the user must be free to compile different actions and to execute them in batch mode.

In a first stage, the scenario can be automatically generated during the interactive phase. Then, it can be edited to allow the user to regulate parameters, and insert control instructions and new commands (see Figure 9).

```
define model OTB:/Users/to02428/A3XX/m0.88a2.0
create  calculation  "CALCUL"  solver  NSMB  model
NST0E_BL

create    boundary_data_set    "reference"    type
"NST0E_BL" values
      MACH 0.85 ALPHA 2 BETA 0 PT/PT0 1.0 TT/TT0
      1.0
      RE1 12.3e6 SU/T0 0.38
create boundary_data_set "fan pressure"
      type "CONST. PRESSURE"  values  P/P0 0.861
create boundary_data_set "Fan massflow"
      type  "GLOBAL  MASSFLOW"  values   AREF  6.3
      EPS1  0.65

create step "step1"
create numerical_bc
create numerical_parameter CFL   5.0
create numerical_parameter NSTEPS 1000
…….
submit job  -time 28800 -memory 128 -clean y
```

Figure 9: Example of scenario

3.2.4 Consistency with other processes

At this stage, it is very important to check the consistency between the different processes of the aerodynamic design cycle (existing processes and foreseen ones)

In so far as these processes are linked together, we need to have homogeneous definitions of traceability and avoid redundancy.

From a user point of view, to improve efficiency, operating modes and organization information visibility must be coherent from one process to another. Similarity identification is an important difficulty for user requirement analysis.

Then, from a programmer point of view, a major benefit is expected from reusing existing codes. At this stage of analysis, it is necessary to pool developments and to identify the reuse possibility at several levels: data access, GUI, scenario and above all at technical services level. This frequently involves frequently an in-house aerodynamic code re-architecture.

3.3 Traceability and organization identification

From the object model and the user scenarios, we can highlight the traceability and management needs as regards data and tools.

3.3.1 Traceability

The know-how of the expert performing an activity is significantly captured by setting up links between fine grain elements used as inputs and fine grain elements of the resulting objects.

The major difficulty is to define the information level we need to trace, and to find a compromise solution to conserve performance and pertinence.

For instance, in the simulation process, at the highest level, the data traceability allows the meshes and the computations performed for a given CAD shape to be founded, and conversely the shape and the meshes configuration used for a computation (see Figure 11).

A more down level could be the link between the step and the solver version used.

Handling traceability includes several aspects:

- defining which information items are involved in traceability,
- capturing and storing meaningful links,
- maintaining links in spite of the evolution of objects,
- exploiting the trace information to help understanding and decision-making.

3.3.2 Organization

The organization needs can be divided into several categories:

- roles identification if there are several contributors,
- the different statuses of the data (private, public, confidential, official,…),
- the right of access (depending on roles and data status),
- the nomenclature rules and the data attributes,
- the work flows.

As in the case of simulation, the designer manages private data (simulation data and scenarios) is stored in its user data space. Nobody else can have access to this space except if he explicitly changes the permissions. Once the study is achieved, he can officialize the information. He provides mandatory attributes allowing everybody to find the information.

3.4 Process integration

3.4.1 Wrapping integration

In the wrapping approach, process data is encapsulated in a single Resource. Its simplified interface does not allow manipulation of data through specific resource operations, but only the start of a specific tool for resource data edition.

The wrapped Resource is then typed in order to allow:

- a visual recognition of data type thanks to icons,
- the start of the appropriate tool for data visualizing and edition.

Finally, the wrapping technique also allows us to access some sub-objects, and convert them into resources, provided the list of sub-objects with their unique identification key is supplied.

The wrapping technique has been used for Calculation process integration. The LFS resource has been used to store the calculation files (flow field data base, specific input files, returned logfiles …) in the UNIX directory attached to the LFS volume. For the time being, the management of sub-resources has not been considered essential. It could be implemented later, if new traceability needs appear.

Mesh and calculation are stored and managed separately in two resources (Mesh Folder, Calculation Folder) because meshes and calculations are obtained through separate processes with different management rules. When a Calculation Folder is created, a traceability link to the Mesh Folder is automatically installed, and the Mesh Folder is then frozen.

3.4.2 Fine grain integration with CORBA objects

Fine grain integration with CORBA objects consists in specifying sub-objects of a process (for example blocks in a mesh, steps in a calculation, graph view in post processing…) and managing them as workshop Resources.

A first prototype was built to serve as a guide for the fine grain integration. The Exploitation Folder was defined as a shared-folder where design engineers organize their post-processing of CFD computations.

The feedback considering this fine grain approach is the confirmation of the considerable advantage of IDL language for the standardization of processes. However the large number of sub-objects and calls to their IDL operations (CORBA communications) leads to a less efficient process. Moreover, due to dynamic activation of various servers, the integration tests proved to be complex and too time-consuming.

To sum up, a compromise solution has to be found between dynamic and modular architecture, and reliable and efficient tools, so the wrapping technique seems to be most advantageous.

4 Deployment

4.1 Use in the aerodynamics department

The *AeroStation* framework is used in an industrial way in the AEROSPATIALE MATRA AIRBUS aerodynamics department for CAD Shapes, mesh management and simulation process.

40 aerodynamic designers within the AeroStation framework handle about 2500 CAD shapes and 500 meshes per year.

The prevision for the simulation process is about 2500 converged computations per year.

Figure 11 shows the CAD shape-mesh-computation traceability through the Trace Matrix tool.

4.1.1 Shape management

The shape management process in *AeroStation* consists of an application providing different options depending on the user connected. Two user categories are defined: the shape designer and the administrator.

The shape designer can:

- Save his shapes in a safe data space,
- Ask the administrator to officialize shapes,
- Ask the administrator to export or import shapes.

To do these operations, he must give some shape characteristics (scale, positioning, format …).

For an officialization, or an export/import operation, he has to fill in a form containing the instructions (shape list reference, destination or origin, …)

Then the administrator can perform officialization and export/import operation.

He can have access to CAD tools (ICEMSURF, CADDS5,…) to control the shape quality and management tools to move and freeze the shapes.

Both shape designer and administrator can interrogate the databases via pre-defined requests to get a shape history.

4.1.2 Mesh management

The mesh management process in *AeroStation* looks like the previous one. The different users are the shape designer, the mesh designer and the administrator.

The shape designer can:

- Ask the mesh designer to make the mesh,
- Execute a pre-defined scenario to produce the mesh automatically (for specific configuration and topology),
- Ask the administrator to officialize meshes.

In the first two cases, he must fill in a form with the mesh specifications (CAD Shapes reference, structured/unstructured …). Meshing tools can read them and fill in the mesh characteristics automatically during the mesh operation.

For an officialization, he just sets the form status to "to be officialized".

The mesh designer freely disposes of meshing tools. He can:

- use them interactively, or in batch mode,
- create scenario,
- ask the administrator to officialize meshes and scenarios.

Then the administrator performs officialization using management tools to move and freeze the meshes and the scenario.

Requests are available to search for meshes and scenarios from any attributes, and to consult traceability (mesh-CAD Shapes for example).

4.1.3 Simulation process

The simulation process has been described in Section 3 through several examples. For the time being, the GUI is already available (UNIX environment). It then allows the designers to use and validate the simulation process, carrying out Euler, Navier-Stokes and coupling boundary layer calculations.

The simulation process will be deployed in the framework during the year 2000.

4.2 Deployment architecture

Before describing the deployment of the *AeroStation* framework, we now consider the constraints of the existing architecture and those of the framework.

4.2.1 General constraints

The aerodynamics department has a total of thirty workstations (26 SGI, 4 Sun). The user's disk space is distributed via NFS. A user can have access to any application whatever the workstation via NFS. There are a great number of different tools:

- Computer Aided Design (Icemsurf, Cadds5),
- mesh generation (Icemcfd),
- computation (Euler, Navier-Stokes, boundary Layer solvers),
- post processing (QuickView, Gsharp).

To submit jobs to Cray, they use NQE (Network Queuing Environment).

The framework consists of three different kinds of components:

- OTB components,
- fine grain integrated component,
- wrapped components.

The communication between the different components is made with an ORB Object Request Broker. The implementation chosen is Orbix from IONA.

4.2.2 Framework constraints

The *AeroStation* framework is characterized by a strongly client-server architecture. We can distinguish two kinds of client-server protocols: Orbix and Objectstore, and two kinds of persistence:

- file persistence: the user's data (CAD parts, meshes and aerodynamic results) are stored in LFS. It needs a large disk space,

- object data base persistence: fine grain integrated tools have a data base persistence based on Objectstore.

The OTB servers: OTB/Catalog, OTB/Session, OTB/Graph, OTB/Forms have their own persistence based on file storage.

Orbix is the implementation of the standard CORBA that is used to perform operations on resources, in particular to obtain the identifier or transmit the identifier whatever the software, and whatever the host. An Orbix daemon must be running on all the workstations with the same Internet port.

The opening of a client session or the opening of the tools activates servers per client which dialogue with the OTB servers. Each framework must have a specific Orbix configuration to have a correct communication between the components of the framework.

A framework is a configuration characterized by a unique port, a unique identifier, a set of OTB tools, a set of integrated components, and a set of wrapped components.

4.2.3 Architecture deployment

The main characteristics of the deployment are as follow.

- User sessions are launched from the SGI workstations or the PCs connected to the workstations via PCxware. The SGI workstations can access the LFS databases via NFS on a DEC server which was used before the deployment of the framework.
- The Objectstore databases, the Orbix-shared OTB servers and their persistence are centralized on a SUN server.
- The computing requests are submitted from the SGI workstations to a DecAlpha or a Cray via the NQE tool (Network Queuing Environment).

This distributed architecture optimizes the CPU and disk resources by dedicating a server to the framework operations and databases.

On the other hand, the SGI workstations are used to launch CPU-consuming software such as CAO, Meshing, Pre or Post Processing, locally.

We realized a successful deployment with respect to the constraints of the existing tools and the constraints of the framework.

4.3 Expected benefits

The traceability of the aerodynamics design process is seen as the most valuable outcome of the present reengineering work The lesson learned designing an aircraft can be capitalized through an automatic recording of relevant links between intermediate products like CAD-surfaces, and their corresponding data and tools at each step of the optimization. The design engineer know-how can then be attached to a link between two successive aerodynamic shapes in order to keep track of the decision process leading to the optimization of the product.

Even if such a service does not provide any immediate time cycle reduction, it is expected to save a lot of time when this information has to be retrieved and applied to a new aircraft, all the more so by an inexperienced engineer.

Traceability also provides the capability to re-use existing surfaces, meshes and/or expensive computational results for a purpose other than the one they have been done for first, thus saving the time necessary to produce them again.

The second most noticeable source of benefits comes from the rationalization of process, required before its integration within the *AeroStation*. The potential time saving are then highlighted as soon as the process modeling is concerned.

As the framework gets well stocked in elementary services and complex tools it is possible to consider the reengineering of the tools themselves in order to take advantage of their possible complementarities and interactions between tools and process. This can be achieved by a finer grain integration of tools within a coupling scenario which allows a more efficient implementation of any multi-disciplinary simulations.

The engineering of coupled methods like mesh adaptation, optimum design, and monitoring of unsteady computations, is made easier and faster once each code has adopted an open architecture.

The global benefit expected for the whole aerodynamics design process of Airbus engine-airframe integration is a 30% reduction in design time cycle.

4.4 Perspectives

In order to achieve the complete integration of aerodynamic design processes, and then the expected gains, the developments planned for the years 2000 and 2001 are the following:

- Shape Design,
- Mesh generation and edition,
- Mesh Adaptation,
- Post-processing ,
- Automatic Generation of synthesis documents.

The extension of the *AeroStation* principles described above, to the whole Airbus aerodynamics process, requires to consider a multi-site framework or at least the communication between frameworks at distant sites of Airbus partners.

In order to anticipate the issues due to the management of distributed objets through a network, we made early this year a numerical simulation demonstrator distributed between two sites: Toulouse and Paris.

A cooperative simulation between two solvers embedded within two different CORBA framework using different mesh strategies: an inviscid fluid solver and a boundary layer solver driven by a common management process, have been set-up.

The demonstrator works and we observe a good convergence of the simulation. In this case, the use of CORBA allows the cooperation between two CFD worlds (structured meshes and unstructured meshes) on geographically distant sites and on heterogeneous hosts.

This approach could be extended to multidisciplinary methods such as aero-acoustics or fluid-structure for example.

5 Conclusion

A CORBA component framework has been defined at AEROSPATIALE MATRA AIRBUS for the integration of the aerodynamics design process. The process integration principles have been presented. Strong emphasis is put on process modeling, traceability and management identification, and the necessary rationalization of the process before its integration by wrapping. This last technique is preferred to a finer grain integration which proved to be more cost consuming due to the numerous interfaces to be defined.

With the *AeroStation* framework, the complete traceability of the design process is ensured automatically and the users can work together, sharing data, working methods and scenarios.

Mixing and matching existing software, commercial codes, and capturing designers' know-how within a distributed environment has been achieved using OTB/Skipper as a means of integration.

It provides easier design process reengineering capability. Using the wrapping technique when appropriate is reducing the cost of new process integration even further.

This framework will make the work done with partners easier, allowing communication between the different tools and the data exchange management.

The integration of the remaining processes is in progress, and multi-site implementation has been experimented.

Acknowledgments

Many of the ideas in this paper come from work done with Y.Dubois, V.Gachelin, E.Bedessem and S.Walter from CR2A-DI, and E.André from Sema Group. T.Hautesserres also provided information.

Everyone involved in SOFTAIR, MEDEA and OTB/Skipper projects has contributed to the drawing up of this paper.

References

[1] V.Treguer-Katossky, J.Huchard,. *Building a CORBA-compliant Aerodynamics Engineering Environment: The SoftAir Project*. DASIA'97 Conference, Seville, 1997.

[2] DGA/DCE/CELAR/Sema Group *OTB/Skipper Wrapping Guide* 3.1 1999.

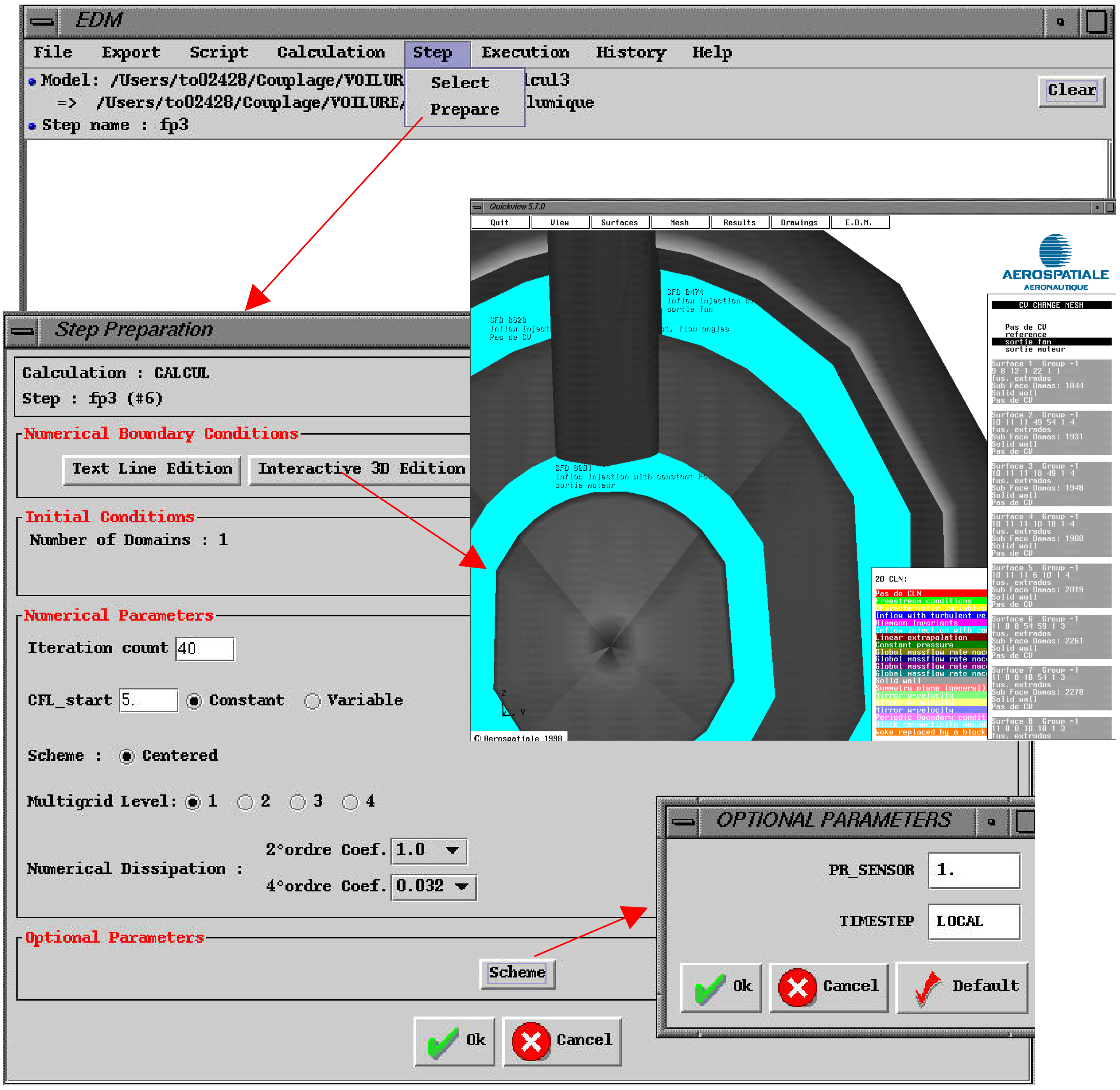
[3] France TELECOM/CNET DGA/DCE/CELAR /Sema Group *OTB Specifications 2.3 1999*.

[4] D.Poirier, S.R.Allmaras, D.R.McCarthy, M.F.Smith, F.Y.Enomoto AIAA-98-3007. *The CGNS System*, 1998.

[5] D.M.A.Poirier, R.H.Bush, R.R.Cosner, C.L.Rumsey, D.R.McCarthy AIAA-2000-0681. *Advances in the CGNS database standard for aerodynamics and CFD*, 2000.

[6] P.Farail, J.Huchard DASIA-2000. *Software requirements traceability , why and how? An avionics case study*, 2000.

[7] C.Casties AEROSPATIALE MATRA AIRBUS NT536.1547/99. *Study of the CGNS storage system for aerodynamic meshes and calculation data*, 1999.



Model: /Users/to02428/Couplage/VOILURE... lcul3
=> /Users/to02428/Couplage/VOILURE/... lumique
Step name : fp3

Step Preparation

Calculation : CALCUL
Step : fp3 (#6)

Numerical Boundary Conditions

Initial Conditions
Number of Domains : 1

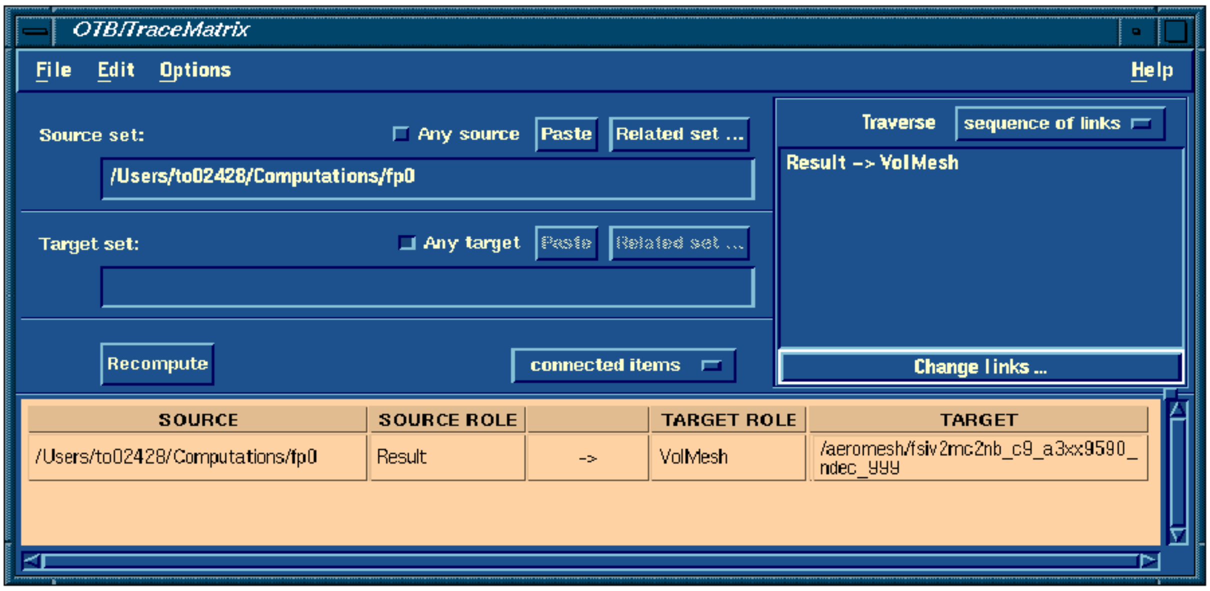
Numerical Parameters
Iteration count 40
CFL_start 5. [✓ Constant] Variable
Scheme : [✓ Centered]
Multigrid Level: [✓ 1] 2 3 4
Numerical Dissipation : 2°ordre Coef. 1.0
4°ordre Coef. 0.032

Optional Parameters

OPTIONAL PARAMETERS

PR_SENSOR 1.
TIMESTEP LOCAL

Figure 10: Simulation process GUI

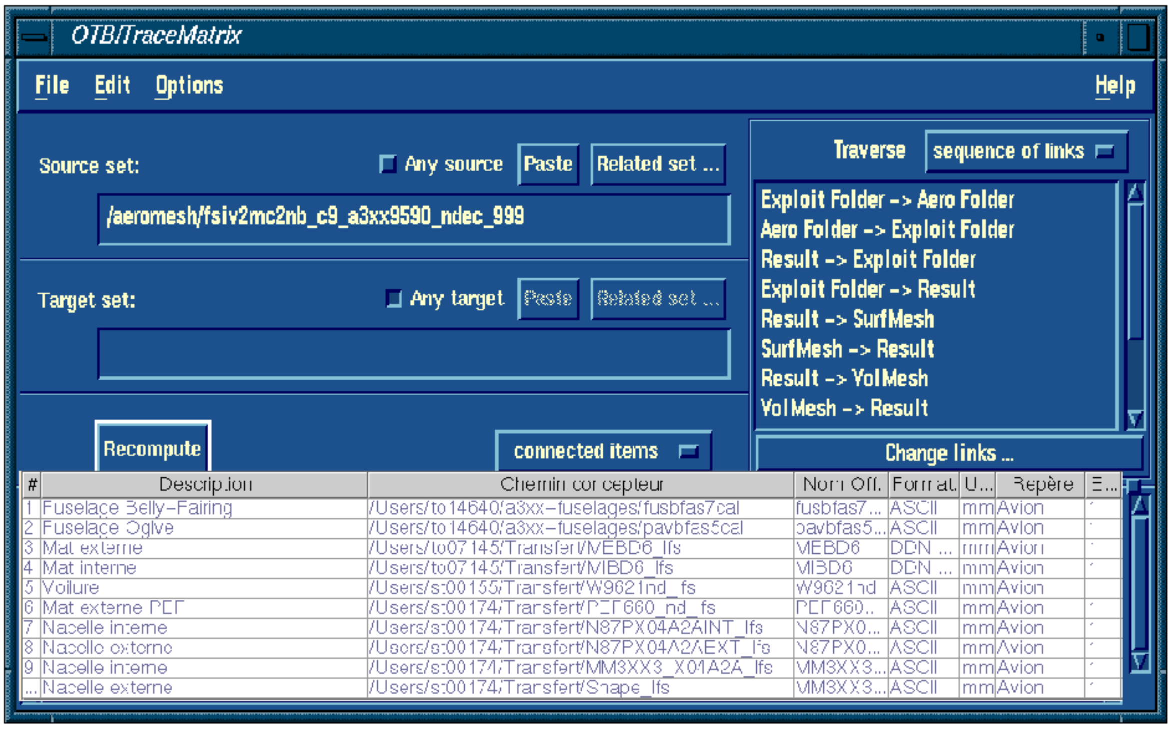


Source set: /Users/to02428/Computations/fp0

Target set:

Result -> VolMesh

SOURCE	SOURCE ROLE		TARGET ROLE	TARGET
/Users/to02428/Computations/fp0	Result	->	VolMesh	/aeromesh/fsiv2mc2nb_c9_a3xx9590_ndec_999



Source set: /aeromesh/fsiv2mc2nb_c9_a3xx9590_ndec_999

Target set:

Exploit Folder -> Aero Folder
Aero Folder -> Exploit Folder
Result -> Exploit Folder
Exploit Folder -> Result
Result -> SurfMesh
SurfMesh -> Result
Result -> VolMesh
VolMesh -> Result

#	Description	Chemin concepteur	Nom Off.	Format	U...	Repère	E...
1	Fuselage Belly-Fairing	/Users/to14640/a3xx-fuselages/fusbfas7cal	fusbfas7...	ASCII	mm	Avion	'
2	Fuselage Ogive	/Users/to14640/a3xx-fuselages/pavbfas5cal	pavbfas5...	ASCII	mm	Avion	'
3	Mat externe	/Users/to07145/Transfert/MEBD6_lfs	MEBD6	DDN ...	mm	Avion	'
4	Mat interne	/Users/to07145/Transfert/MIBD6_lfs	MIBD6	DDN ...	mm	Avion	'
5	Voilure	/Users/s:00155/Transfert/W9621nd_fs	W9621nd	ASCII	mm	Avion	
6	Mat externe PEF	/Users/s:00174/Transfert/PEF660_nd_fs	PEF660..	ASCII	mm	Avion	'
7	Nacelle interne	/Users/s:00174/Transfert/N87PX04A2AINT_lfs	N87PX0...	ASCII	mm	Avion	'
8	Nacelle externe	/Users/s:00174/Transfert/N87PX04A2AEXT_lfs	N87PX0...	ASCII	mm	Avion	'
9	Nacelle interne	/Users/s:00174/Transfert/MM3XX3_X01A2A_lfs	MM3XX3...	ASCII	mm	Avion	'
...	Nacelle externe	/Users/s:00174/Transfert/Shape_lfs	MM3XX3...	ASCII	mm	Avion	'

Figure 11: Computation-Mesh-CADShape Traceability