Nationaal Lucht- en Ruimtevaartlaboratorium
National Aerospace Laboratory NLR

NLR-TP-2001-338

ICT environment for integrated multidisciplinary aircraft design analysis

W.J. Vankan, B.C. Schultheiss and E.H. Baalbergen

Nationaal Lucht- en Ruimtevaartlaboratorium
National Aerospace Laboratory NLR

NLR-TP-2001-338

ICT environment for integrated multidisciplinary aircraft design analysis

W.J. Vankan, B.C. Schultheiss and E.H. Baalbergen

This investigation has been partially carried out in the MOB project under a contract awarded by the EC, contract number G4RD-CT1999-0172.

This report is based on a presentation held at the CEAS Conference, Cologne, Germany, June 25-27 2001.

Division:	Information and Communication Technology
Issued:	June 2001
Classification of title:	Unclassified

Summary

The large variety of software tools that is used in multidisciplinary aircraft design processes, and the associated data formats and heterogeneous computer infrastructure, impose undesirable ICT complications to end users. Efficient usage of present technologies such as CORBA, Java and HTTP, shields end users from these complications and facilitates close, multidisciplinary co-operation. SPINEware provides these technologies as a tool kit for the creation of user-oriented multidisciplinary design environments. Such design environment for multidisciplinary design and optimisation of blended wing body aircraft configurations is being developed in the MOB project.
This paper describes the functionality of SPINEware, and illustrates its application in the MOB project.

List of acronyms

BWB	Blended wing body
CAD	Computer aided design
CDE	Computational design engine
CFD	Computational fluid dynamics
CORBA	Common object request broker architecture
GUI	Graphical user interface
ICT	Information and communication technology
ILU	Inter-language unification system (free CORBA implementation)
LAN	Local area network
MDO	Multi-disciplinary design and optimisation
MOB	Project acronym for EU project: A Computational Design Engine Incorporating Multi-Disciplinary Design and Optimisation for Blended Wing Body Configuration
ORB	Object request broker
SPIRL	SPINEware resource locator
STEP	Standard for the exchange of product model data (ISO 10303)
Tcl/Tk	Tool command language/Toolkit
URL	Universal resource locator
WAN	Wide area network

Contents

(16 pages in total)

1 Introduction

Aircraft design requires a close co-operation of a number of different technical disciplines, such as aerodynamics, structural mechanics and flight mechanics. In each of these disciplines a large variety of software tools is available for design and analysis simulations, each of which requires a specific computer platform for proper operation. For example, the programs for CAD and results visualisation typically run on graphical workstations, while CFD flow solvers are usually executed on supercomputers. Moreover, most of the software tools use specific data formats for storage and exchange of information. The variety of software tools and the associated heterogeneous computer infrastructure and data formats impose complications to the close multidisciplinary co-operation that is required in aircraft design. The end users in the multidisciplinary aircraft design process should be shielded from these complications as much as possible.

Present technologies like CORBA (Common Object Request Broker Architecture), Java and WWW provide possibilities to build end-user oriented, integrated multidisciplinary design environments. For this purpose these technologies can be incorporated into middleware systems that facilitate data exchange and interoperability in multidisciplinary design environments, which are usually operated on distributed heterogeneous computer networks.

SPINEware is a middleware system that supports the construction and usage of working environments on top of heterogeneous computer networks [1]. A working environment realised using SPINEware presents a local- or wide-area computer network as a single and easy-to-use "metacomputer" on the user's desktop computer. The working environment provides uniform and network-transparent access to the information, applications, and other resources available from the computer network presented to the user through an intuitive GUI. To support flexibility, openness, and easy extendibility of SPINEware as well as the SPINEware-based working environments, state-of-the-art technologies and software are applied. In particular, the application of the CORBA standard facilitates the integration of commercial and other third-party software and to reuse the object-oriented support and communication services provided by software implementations of the CORBA standard (so-called ORBs).

A SPINEware-based working environment can be tailored to particular end usage and application areas. The metacomputing capability, the tailoring facilities, and the user-oriented desktop system capabilities enable the realisation of powerful and easy-to-use application environments that support non-computer experts to exploit the potentials of the underlying computer infrastructure. It provides a suitable basis for the realisation of a user-oriented multidisciplinary environment as required in the aircraft design process.

Recently SPINEware has been extended with web-based access. The SPINEware User Shell is now also available as a Java applet implementation. SPINEware object services can be accessed via a web browser using these SPINEware User Shell applets. The accessibility of and

interoperability within SPINEware working environments is thus much improved, especially in WAN environments where security regulations such as firewalls are to be dealt with.
This paper presents a description of SPINEware and its main components. In addition, an example of a WAN working environment for multidisciplinary aircraft design and optimisation that was set up using SPINEware is presented. In particular the integration of software tools as well-defined objects, the easy creation of tool chains, and the web-based access to this working environment are shown.

2 The SPINEware middleware system

2.1 Rationale

Effective multidisciplinary collaboration, as required in today's aircraft design processes, strongly depends on efficient usage of the ICT infrastructure. Although present hardware and software provide support for establishing the required ICT infrastructure, end users often get confronted with low-level details emerging from the underlying computing systems and networks. Moreover, to stay competitive, organisations invest in new computers, network components and software products are added, and old systems are replaced by faster, more advanced and fancier ones.
These innovations unfortunately lead to increased complexity of the ICT infrastructure. The complexity not only concerns the hardware and system software, but also the end usage by engineers, who are not computer experts in general. They constantly need to familiarise with the ever expanding and changing computing infrastructure, which usually leads to inefficient resource usage. Consequently, the organisation using the infrastructure for managing the processes and competence may suffer from the complexity and inefficiency arising from it.
Three significant characteristics that contribute to the complexity of today's computing infrastructures are distribution, heterogeneity, and system-level – instead of user-level – integration [2]. *Distribution* involves all aspects resulting from the infrastructure being a network of interconnected computers. In practice, the infrastructure forces the end user to be aware of the fact that several computers are – and usually even must be – involved in the engineering process. Inter-company networking also gives rise to security issues, such as firewalls, authentication, and encryption, which may burden the engineer with even more complexity arising from extra measures to be taken. *Heterogeneity* concerns the confrontation of the end user with a variety of hardware (e.g., computer systems varying from highly interactive desktop systems to batch-driven, number-crunching supercomputers, involving different byte ordering, sizes and ranges of numeric values, etc.), operating systems (e.g., Windows, Linux, and UNIX in many different flavours), applications, and data formats. *Integration at the system level* – as opposed to the user level – refers to the availability of a large

set of low-level utilities that indeed support the usage of interconnected computers (e.g., FTP for exchange of files, TELNET for remote login, and SAMBA for sharing files systems). However, these utilities still leave much detail to the end user or cover only part of the networking details.
SPINEware aims to provide a coherent and total solution for dealing with the low-level details emerging from distribution, heterogeneity, and system-level integration [1]. Its approach is to provide users involved in the multidisciplinary process, such as engineers, project managers and support staff, with an application and end user oriented, easily customisable, single computer (metacomputer) instead of a system-oriented network of stand-alone computers to which the user must adapt himself or herself. The metacomputer, being tailored for the user, contributes to easy and efficient operation, and hence to acceptation, of new systems by novice as well as expert users.

2.2 SPINEware system description

Two main components of SPINEware are the *SPINEware Object Server* and the *SPINEware User Shell*.
A *SPINEware Object Server* is a process that manages the availability, accessibility and interpretation of SPINEware objects. SPINEware is fully object-oriented: in a SPINEware working environment, all resources such as software tools, data and documents that are available from the connected hosts are modelled as objects. SPINEware provides a set of basic object classes such as File and Directory (for representing the native system's files and directories), ObjectFolder and WorkingEnvironment (for folder objects for organising SPINEware objects), AtomicTool (for advanced “wrapping” of executable programs), Workflow (for easy tools chaining), Job and Queue (for managing and scheduling tool execution). The user can modify existing classes and create new object classes - whether or not inheriting from existing classes. An object class has a set of methods associated to it, by which a user can manipulate the objects. In the *SPINEware User Shell*, the default method for an object class is a standard access operation on an object, where typical examples are *Edit* for File, *View* for Directory and ObjectFolder, and *Execute* for AtomicTool. Each object in SPINEware has a world-wide unique object identifier, usually represented as a *SPIRL* (SPINE Resource Locator). Beside object identification, SPIRLs can also specify object method invocation. SPIRL syntax is based on the well known URL naming scheme used in WWW. A complete SPIRL specifies the object’s class, a method name (if used for method invocation), its host’s Internet name, its location on that host (e.g. its file system path), and optionally any method arguments. For example, the *Edit* method for the file /home/user/john/file.text on host johnshost.domain.net is specified by the SPIRL:
spine://File:Edit@johnshost.domain.net//home/user/john/file.text?language=Dutch&font=small
. If this SPIRL is issued, SPINEware will invoke the object method *Edit* to the *File* object that

resides on the host *johnshost.domain.net* under */home/user/john/file.text* with editor options *language=Dutch* and *font=small*. As a result, SPINEware will launch an appropriate file editor that displays the contents of the specified file on the user's desktop. SPIRLs are used internally in SPINEware for object operations, but can also be issued by users directly to the command interpreter. SPIRL interpretation is performed by the *SPIRL broker* utility, which enables invocations of object methods from within tools, scripts, and command-line interpreters. When a user starts a SPINEware session, one SPINEware object server is started initially on the local host, which will handle all requests from the *SPINEware User Shell*. If an object on another host is requested, the initial SPINEware object server starts ("on demand") a SPINEware object server on the other host, and relays all subsequent method invocations (and the corresponding results) involving objects on that host. The communication among the SPINEware objects is based on CORBA, and is implemented using an off-the-shelf CORBA implementation, *ILU*. This product supports CORBA-based inter-object communication over LAN and WAN and is capable of being used in combination with firewalls.

The *SPINEware User Shell* provides an intuitive, tailorable Graphical User Interface (GUI). The user shell acts as the interface between the user and the local SPINEware object server, by presenting all the required information of SPINEware objects in the connected network on the user's local graphical desktop, and by translating the user's GUI input to SPIRL commands for the local object server. The SPINEware object information is presented through dedicated browsers for each of the SPINEware objects. Some examples of specific browsers for the ObjectFolder, WorkingEnvironment, AtomicTool and WorkFlow objects are given in figure 1.

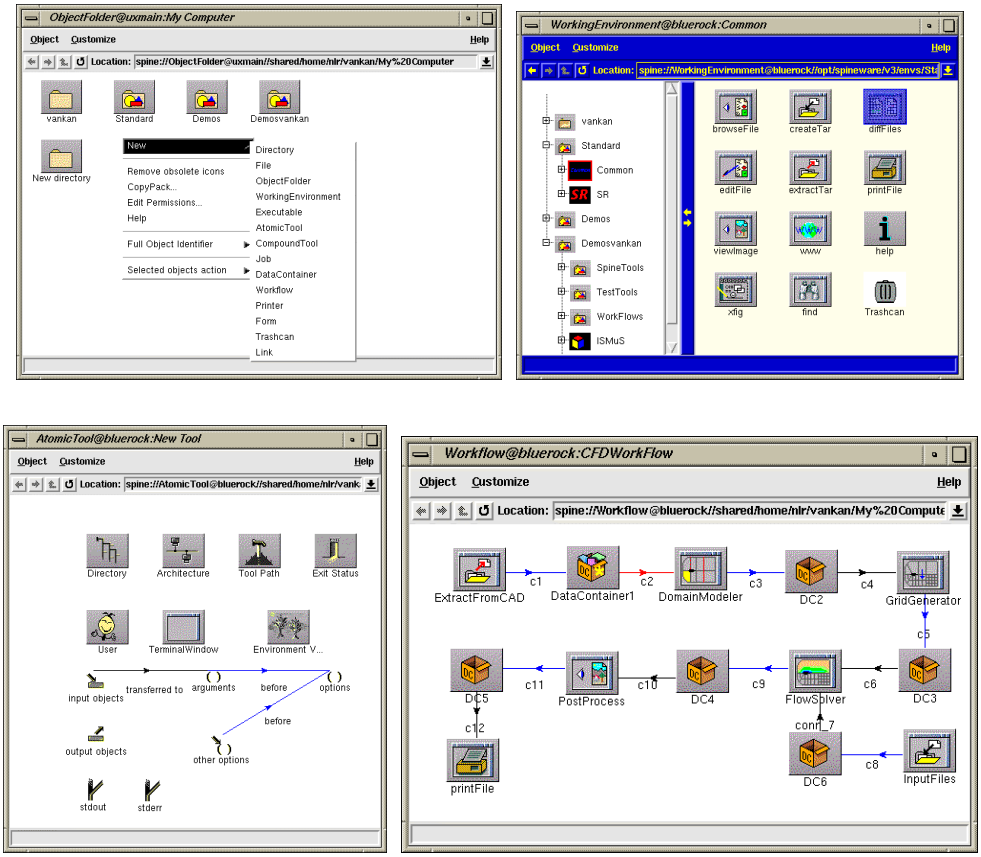

Figure 1 Some examples of specific browsers for the ObjectFolder, WorkingEnvironment, AtomicTool and WorkFlow objects (Tcl/Tk based GUI).

The object-specific browsers are designed such that the relevant information of the object (identification, methods, etc.) is clearly presented to the user and that the user is efficiently guided through the process of creating, editing or executing the objects. It should be noted that the objects that are presented to the user, may reside anywhere in the connected network and are accessed directly, either locally or remotely, by the user.

The user shell can be run in different ways. One way is that the user starts a local user shell process on his local system (e.g. desktop computer) which directly communicates to local or remote SPINEware object servers. The other way is to run the user shell by making use of Java applets and servlets to communicate to the SPINEware object servers [3]. The first way requires the user to have the SPINEware software installed on his local system, or a remote system he has graphical access to. The second way requires the user to have (as a minimum) only a Java-2 enabled web browser on his local system. The access to the system that runs the actual SPINEware software (which may be either remote or local) is arranged via an HTTP daemon with servlet support. For example, if a user logs on to the system via his web browser, the Login servlet is invoked. The Login servlet starts up the user's SPINEware session by publishing a

new instance of the SPINEware User Shell object to the CORBA naming service and launches a SPINEware object server on the user-specified host using a rexec command. Figure 2 explains the two different operational modes of the SPINEware user shell.

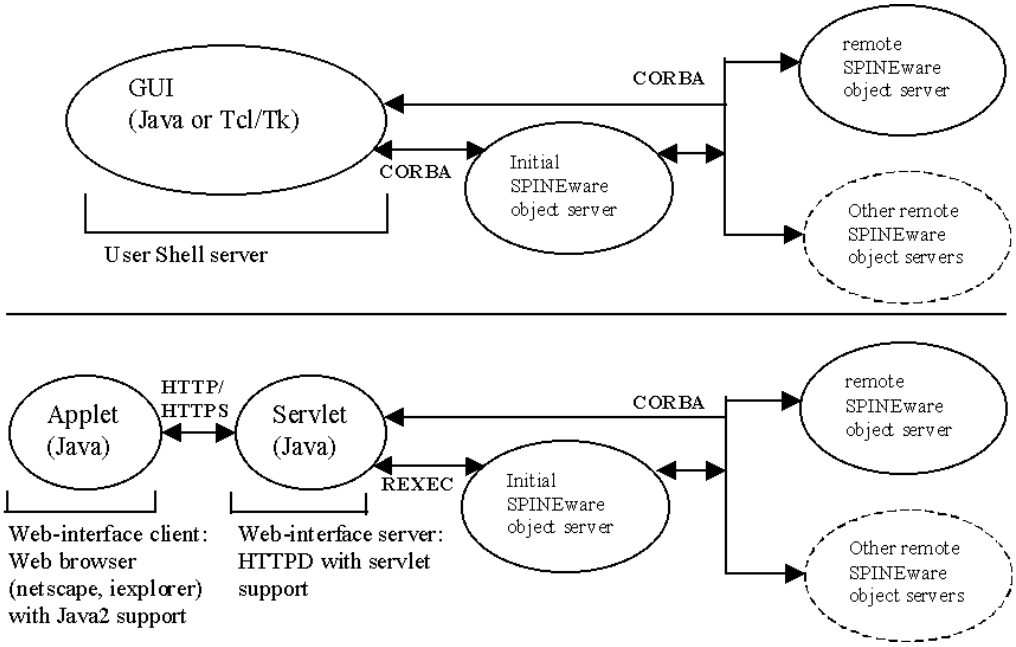

Figure 2 Two different ways of operation of the SPINEware user shell.

The Java implementation of the SPINEware user shell ensures that it is available for a large variety of platforms. For more information on the SPINEware system is referred to [3].

2.3 SPINEware application

The practical application of the SPINEware middleware system is that it provides an integration platform for heterogeneous computer networks in which users can operate in pre-defined working environments, build up and operate in their own working environment, and exchange or share their working environment or its objects with other users. The user can for example browse through the working environment's contents, create and execute tools, and submit jobs using point-and-click and drag-and-drop operations. In addition to browsing through information and launching tools on the user's local host, the user may open browsers for accessing resources on other hosts as well. Data can simply be moved and copied by dragging

and dropping icons from one browser window to another. Working environments may be tailored for particular end users and application areas.

3 The ICT environment for aircraft design

3.1 Introduction

In the MDO project [4] it was recognised that the most relevant aspects of concurrent multidisciplinary design, as opposed to the sequential mono-disciplinary approach, were to accurately capture the MDO process, to decrease dependency on individual team members, and to provide control mechanisms on different levels of the calculation process [5]. SPINEware offered the functionality and the flexibility to account for each of these aspects in the MDO engineering environment. An important conclusion from the MDO project was that, due to the required iterative improvements of the MDO process, ICT is not just supporting the other disciplines but is one of the MDO core activities. Other relevant conclusions were that the use of integrated product models and automatic multidiscipline model generators was essential for the different disciplines to collaborate effectively, and that the open and flexible engineering environment was essential for the integration and exchange of data and processes [6]. These aspects are supported by SPINEware.

The experiences from the MDO project are further exploited in other projects, in particular the MOB project [7]. This project focuses on the development of a distributed engineering environment for multidisciplinary design and optimisation. This engineering environment, which is referred to as Computational Design Engine (CDE), is applied to the design of a blended-wing-body (BWB) aircraft configuration. Like in the MDO project, the engineering environment is developed on the basis of SPINEware. Other than in MDO however, the aim in MOB is to develop this engineering environment as a single environment accessible from different sites of different partners, and supporting more integrated analysis and automated data exchange on the basis of STEP.

3.2 The MOB CDE

In the MOB project a distributed engineering environment for multidisciplinary design and optimisation of BWB body aircraft configurations is developed. The CDE that is being developed in the MOB project shall be a flexible system, allowing the MOB partners to integrate their design and analysis tools. Moreover, the partners must be able to access, generate and exchange the relevant analysis results for their part of the design task over the WAN, and to automate certain standard procedures in the design process. The CDE facilitates the design process for the BWB, which is intrinsically multidisciplinary, multi-level, and multi-model. The design process is executed in the multi-site environment of the MOB partners. Consequently the

CDE is distributed over the different sites of the partners in the WAN. SPINEware is used for the construction of the CDE, in order to effectively deal with accessibility, interoperability and exchange of tools, data and users. The CDE as such is considered as a SPINEware WorkingEnvironment object, which is distributed over, and accessible from, the sites of the different MOB partners. Specifically for the CDE, the main functionality provided by SPINEware is the following:

1. Support WAN operation of the CDE, which is covered by the WAN definition of the SPIRL.
2. Shield networking details, such as logins to remote systems and file transfer, from the end user of the CDE. This is covered by applying CORBA for data communication.
3. Provide a flexible and intuitive GUI. This is covered by the SPINEware User shell, which enables the user, for example, to view or edit files, copy and move directory contents and create, edit and execute tools, either with or without input files, by simple point-and-click and drag-and-drop operations on the objects' icons.
4. Provide a flexible framework for integration of the CDE components This is covered by the WorkingEnvironment object, which supports environment management, enabling customisation of personal environments that can be subsets of the main WorkingEnvironment or CDE.
5. Provide a common object oriented interface to definition, integration, execution and interconnection of tools in the CDE, supporting easy usage and exchange of data, tools and processes. This is mainly covered by the functionality of the AtomicTool object.
6. Support composition, editing, exchange and execution of standard or routine chains of activities or tools, which is covered by the SPINEware WorkFlow object
7. Provide additional functionality, like information management, software version management and repositories for data and documents. This functionality is standard available and can be incorporated into the CDE if necessary

The main objective in creating the CDE is the development of tools and methods to design an aircraft by distributed teams using several discipline-based programs and approaches integrated into a distributed CDE. The design incorporates concept, preliminary aircraft and main phase design, which implies that the CDE has to ensure continuity of information flow through the design cycles. It also implies that multiple levels of fidelity are involved within the CDE. A preliminary global architecture of the CDE has been defined. This architecture is based on the global design and optimisation procedure as it is envisaged for the MOB project.

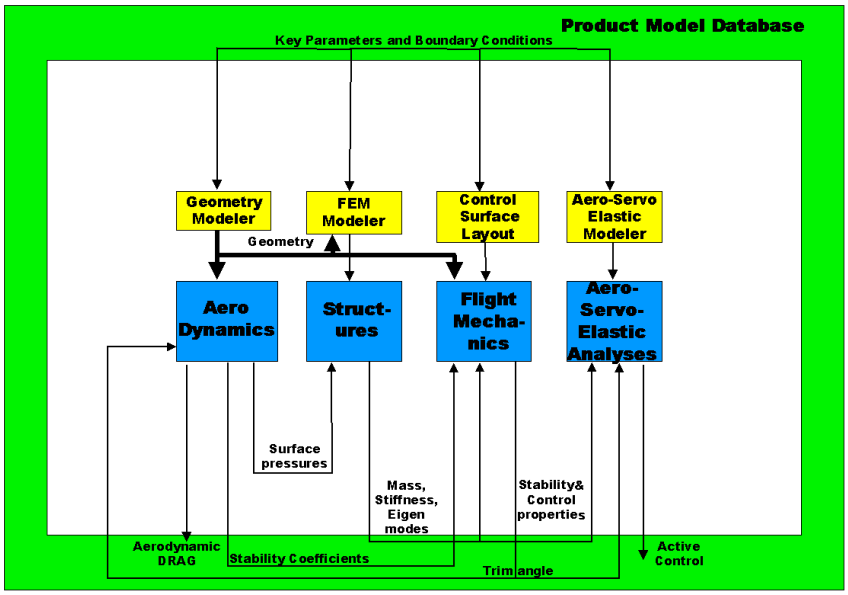

Figure 3 Preliminary CDE architecture for the global objective evaluation.

The CDE is currently being developed in the MOB project. The MOB partners perform the developments of their CDE components locally at their own sites. Once completed, these components can be integrated into the central CDE. The main requirement for integration is the accessibility to the consortium of the component. For this purpose, integration can be achieved either by installing a component into the central environment, which is currently installed at NLR, or the partner's component is directly accessed via the central environment. The central environment at NLR may be accessed via a SPINEware User Shell running locally at NLR, or via an HTTP daemon using a Java 2 enabled web browser at any of the partners' sites. Figure 4 presents the current top level of the central CDE accessed via an HTTP daemon using MS Internet Explorer on a PC.

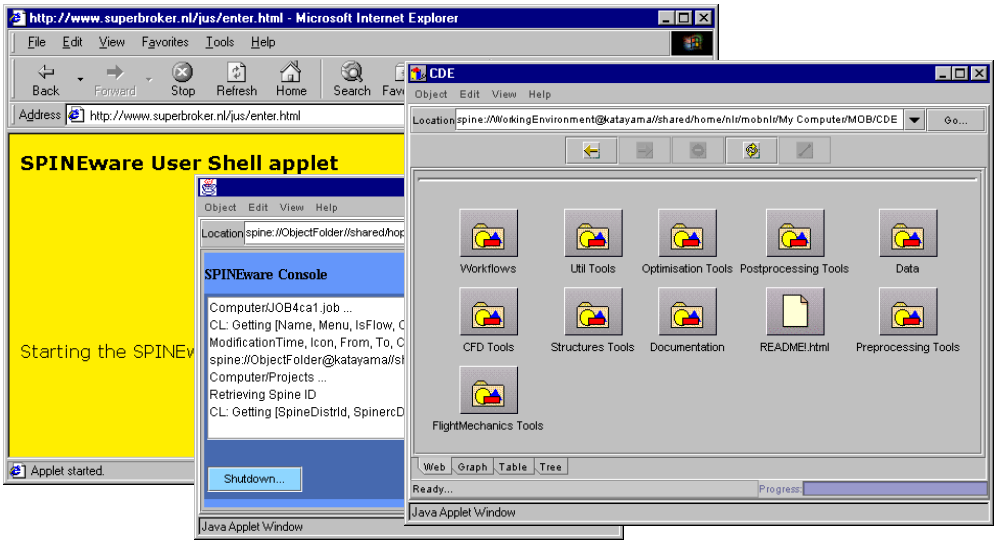

Figure 4 Top level CDE WorkingEnvironment browser (in foreground) accessed via the SPINEware Java user shell via MS Internet Explorer (shown in background).

The CDE components that are integrated by the partners, such as design tools and analysis tools, are to be combined into complex systems of calculations. Therefore generic descriptions of these tools in terms of tool interfaces, execution properties like hostname, options and arguments, etc., must be defined for each tool. The SPINEware AtomicTool object is able to store all this information. Hence the partners integrate their tool components as AtomicTool objects by making use of the graphical AtomicTool editor, as shown below in figure 5.
The assembly of AtomicTool components into chains of tools for automatic execution of design (sub)processes, is achieved by creation of SPINEware WorkFlow objects via easy drag-and-drop operations. An example of such a WorkFlow, representing the distributed BWB geometry generation and CFD and structural mechanics analysis processes, is given in figure 6.below.

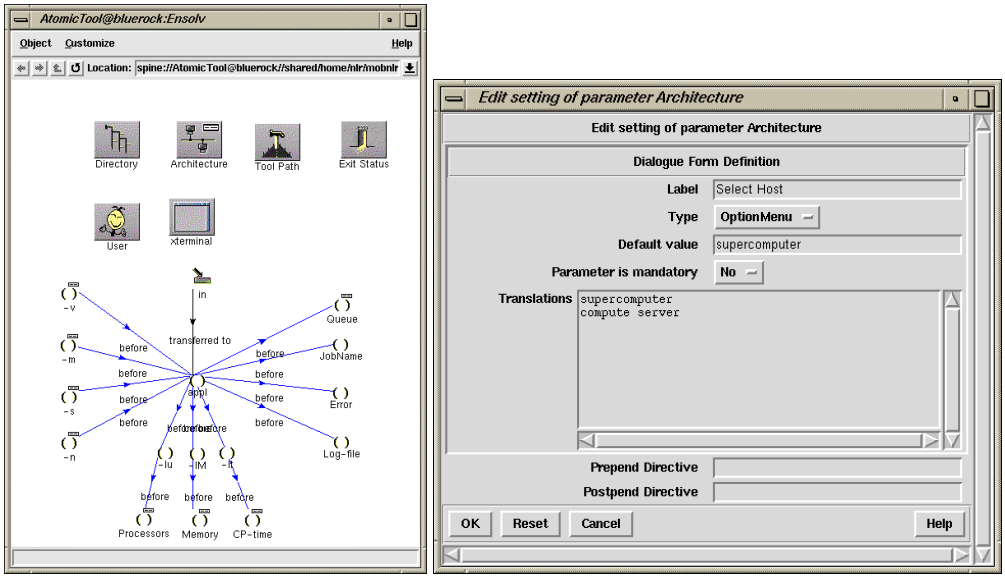

Figure 5 Examples of SPINEware AtomicTool editor windows. On the left the main editor window for the Ensolv CFD solver is given, where the icons represent AtomicTool sub-objects. On the right the sub-object Architecture-window for specification of the execution host is given.

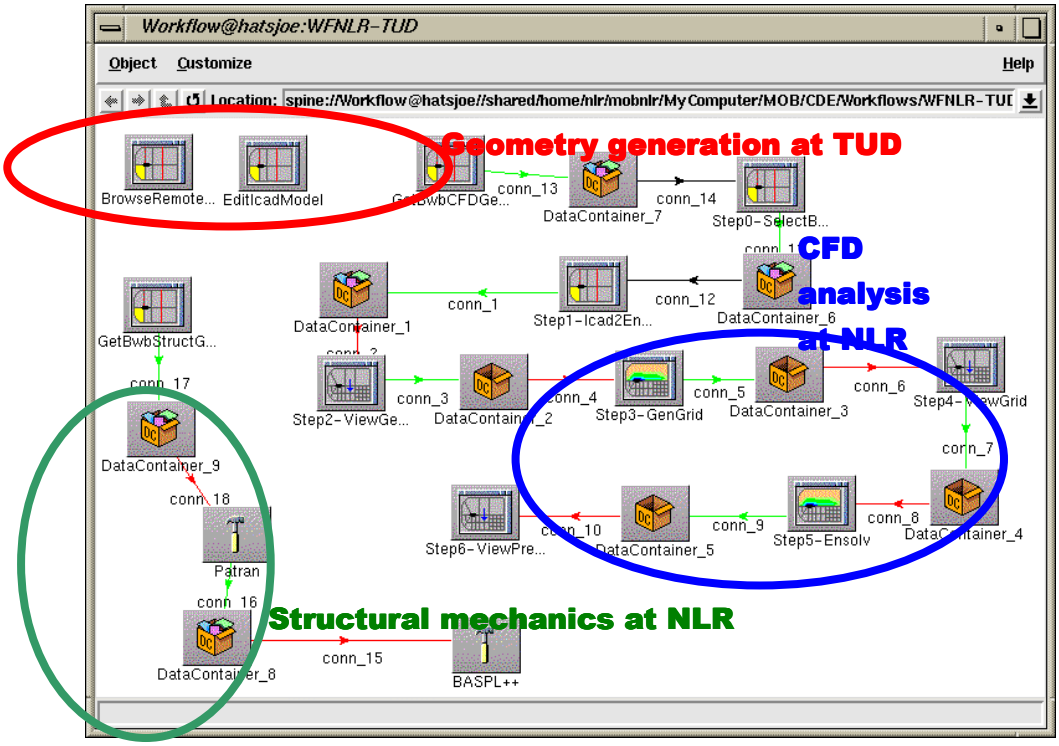

Figure 6 SPINEware WorkFlow object of the distributed BWB geometry generation and CFD and structural mechanics analysis processes, executed at the MOB partners TUD and NLR.

4 Conclusions

Close multidisciplinary co-operation, as found in aircraft design, requires end-user oriented, integrated, multidisciplinary design environments. Such design environments can be built using present ICT technologies like CORBA, Java and WWW. SPINEware is a middleware system that provides these technologies as a tool kit for the creation of such design environments. SPINEware has been applied in several multidisciplinary multi-partner projects to set up such environments for integration, execution and exchange of design and analysis tools. The environment that is currently being developed in the MOB project, the CDE, is built up from the CDE components that are developed by the MOB partners. These components can be integrated into the central CDE, which can be made available to the MOB partners' web browsers via HTTP servers. A preliminary version of this web-based implementation of the CDE is presently available.

5 References

[1] Baalbergen, E.H., SPINEware: A practical and holistic approach to metacomputing, in: proc. International Conference and Exhibition on High-Performance Computing and Networking HPCN Europe 1998, Lecture Notes in Computer Science 1401, Springer, pp. 1008-1011, 1998. NLR TP98478.

[2] Baalbergen, E.H. and H. van der Ven: SPINEware – a framework for user-oriented and tailorable metacomputers, Future Generation Computer Systems, 15, pp. 549-558, 1999.

[3] Schultheiss, B.C. and E.H. Baalbergen: Utilizing supercomputer power from your desktop, HPCN 2001 conference, Amsterdam, 2001.

[4] Allwright S., Multi-discipline Optimisation in Preliminary Design of Commercial transport Aircraft, ECCOMAS'96, Paris, France, September 1996

[5] Vogels M.E.S., P. Arendsen, R.J. Krol, M. Laban and G.W. Pruis, From a mono-disciplinary to a multidisciplinary approach in aerospace: as seen from information and communication technology perspective, in: ICAS 98, Melbourne, 1998a. NLR TP98226.

[6] Vogels M.E.S., S.E. Allwright, M. Stettner, Ph. Sims and P. Bartholomew, The information and communication technology's contribution to the MDO project, in: MDO 98, Royal Aeronautical Society, London, 1998b. NLR TP99029.

[7] MOB: A Computational Design Engine Incorporating Multidisciplinary Design and Optimisation for Blended Wing Body Configuration, EC 5th Framework Programme, Contract Number: GRD1-1999-11162, 2000.