Kernel Architecture for CAD/CAM in Shipbuilding Environments

Antonio Rodríguez, SENER Ingeniería y Sistemas, S.A., antonio.rodriguezg@sener.es
Carlos González, SENER Ingeniería y Sistemas, S.A., carlos.gonzalez@sener.es
Iñigo Gurrea, SENER Ingeniería y Sistemas, S.A., inigo.gurrea@sener.es
Lluis Solano, Universidad Politécnica de Cataluña, solano@lsi.upc.es

Abstract

The capabilities of complex software products such as CAD/CAM systems are strongly supported by basic information technologies related with data management, visualization, communication, geometry modeling and others related with the development process.

These basic information technologies are involved in a continuous evolution process, but over recent years this evolution has been dramatic. The main reason for this has been that new hardware capabilities (including graphic cards) are available at very low cost, but also a contributing factor has been the evolution of the prices of basic software.

To take advantage of these new features, the existing CAD/CAM systems must undergo a complete and drastic redesign. This process is complicated but strategic for the future evolution of a system. There are several examples in the market of how a bad decision has lead to a cul-de-sac (both technically and commercially).

This paper describes what the authors consider are the basic architectural components of a kernel for a CAD/CAM system oriented to shipbuilding. The proposed solution is a combination of in-house developed frameworks together with commercial products that are accepted as standard components. The proportion of in-house frameworks within this combination of products is a key factor, especially when considering CAD/CAM systems oriented to shipbuilding.

General-purpose CAD/CAM systems are mainly oriented to the mechanical CAD market. For this reason several basic products exist devoted to geometry modelling in this context. But these basic products are not well suited to deal with the very specific geometry modelling requirements of a CAD/CAM system oriented to shipbuilding.

The complexity of the ship model, the different model requirements through its short and changing life cycle and the many different disciplines involved in the process are reasons for this inadequacy. Apart from these basic frameworks, specific shipbuilding frameworks are also required. This second layer is built over the basic technology components mentioned above.

This paper describes in detail the technological frameworks which have been used to develop the latest FORAN version.

1. Introduction

CAD Systems are complex products, the main reason being that they involve several of the most sophisticated technologies used in the development of software applications. General purpose CAD Systems require high performance visualization, computational geometry complex algorithms, advanced ergonomic user interfacing techniques, etc.

These characteristics lead to the fact that every one or two years a new release of the most popular CAD Systems is launched on the market and that every two or three releases a new

version appears, which introduces major changes, mainly philosophical, that affect the design of the application.

In addition to this, there is a general trend in 3D graphic API standards, which every ten years suffer a drastic change motivated by several factors:

- Advancements in graphic and modeling knowledge.
- Increasing power on graphic acceleration chips.
- Evolution of operating systems.
- The choice of primary languages used for programming.
- The type of users for 3D graphic applications.

Examples of these important changes are the appearance of 3D GKS, PHIGS and PEX in the early 1980's and the appearance of OPEN GL and DIRECTX in the late 1980's. The first of these API's was accessed with programming languages such as FORTRAN and C and the second generation API's are accessed with C++.

Shipbuilding CAD Systems suffer somehow a very different evolution path. As these are CAD products targeting a very specialized segment of the market, with a very specific know-how, the capability to react to these important changes has been more limited compared with general purpose CAD applications.

As the periods between big jumps in the evolution of these CAD Systems oriented to shipbuilding increase, when they may accumulate much more technological changes. This paper is intended to describe the new generation of shipbuilding CAD systems based on technologies, which if not revolutionary, are at least innovative when used in this area.

This paper is basically an enumeration of these technologies and the way they are used in shipbuilding CAD Systems. Technologies described are:

- Database
- Visualization
- Topology
- Product model

The following paragraphs will explain how, by means of these technological frameworks, it is possible to build up the most sophisticated CAD tools for shipbuilding designers.

2. Database Framework

This framework must offer some of the most fundamental services when developing a CAD System. These are a persistence service, which is a basic infrastructure, and concurrent engineering facilities, which is a strategic added value to a shipbuilding CAD System. Each of these services is explained in the following paragraphs.

2.1. Persistence service

While the first shipbuilding CAD Systems used files to store information, the increasing complexity of the systems, covering more areas of the design, called for new tools to handle large amounts of information in a fast, flexible way.

At the same time the increase in the number of users working simultaneously on the same ship called for concurrent access to the ship's information. In response to these requirements databases started to be used to store the information, partly or totally.

In the first instances relational databases were used to store the attributes of ship components, while in some of the second instances proprietary databases were developed to store geometrical and topological information of the ship model, as well as attributes of its components.

During recent years there has been a trend to replace the proprietary databases of integrated CAD Systems with commercial databases with standard access languages and database server capabilities. The main reasons for this change have been the ease with which the users can develop new applications and the possibility to reduce the amount of information transferred between each user and the database. The latter is not very important in a LAN, but it is crucial in a WAN.

The use of object oriented databases, proposed in recent years, has not had wide acceptance, as the standardization, concurrency and access capabilities of object oriented databases are considerably behind those of current commercial relational databases.

On the other hand, there is an important issue when using a relational database to store the product model created with an object oriented CAD System. The challenge is to connect both paradigms without losing the main advantages that each of them offers.

The object oriented paradigm is the natural path to develop a consistent and powerful product model and all the software components related to it. It is accepted that this would be the leading paradigm to which a relational model would have to adapt.

This strategy consists of using the relational database as a persistent layer for the object oriented world. From the object oriented perspective, it would be easy to replace this layer should a new storage technology appear. But this layer must be able to exploit the advantages that the relational databases technology offers to the application developer. This question will be analyzed in more detail in the following chapter devoted to explain the concurrent engineering facilities.

To cope with the problem posed in this way, the persistence service must supply the following features:

- Offer a transparent access to the persistent objects. The application programmer should not need to use SQL directly to receive and access an object stored in the database
- Support for inheritance
- Support for polymorphism
- Unique object identifiers
- Associations between objects (1 to 1, 1 to n and n to m)
- Automatic caching of objects in memory

- Automatic storing of modified objects (commit)
- Access via smart pointers to avoid multiple instances of the same object and keep the pointers always valid
- Access to multiple databases
- Centralized lookup and search mechanisms (search conditions)
- Support for proxy objects for accessing parts of an object, e.g. showing a list of equipments with description without reading all the information for the equipment objects. This may be solved by lazy on-demand materialization not only for objects but also for object attributes
- Transactions (commit, rollback, save-point)
- Multi user access
- Locking of objects
- Serialization of objects

All this complex functionality can be encapsulated for the object oriented developer, as the object model is leading the development process. The best way to do this is by means of generative programming. When creating the object model, the software developer could annotate with specific persistence attributes the code to specify the storage behavior of the objects. A pre compiler process would automatically generate the corresponding object factories to which would be delegated the responsibility of managing the objects life cycles.

The use of generative programming to cover this gap is the most productive way to accomplish this task. Additionally, generative programming is the best methodology to avoid error prone development process, to improve the quality of the software produced in this critical area.

2.2 Concurrent engineering facilities

Concurrent engineering has, for a long time, been an important issue for shipbuilding CAD Systems, specially in the area of detail design. This need led to the product model concept, as the single source of product information. The existence of a product model allowed the concurrent engineering practice, at least in the context of a local area network. CAD systems based on a single product model for all the disciplines are generically called integrated systems.

In this kind of system, the coherence and compatibility of the work performed by different users is obtained by using a single database to store one single ship model for all users. This method guarantees that the quality of the engineering work is far superior to that which can be reached when a ship model is not used.

This advantage can be obtained when all of the users are working in the same local area network (LAN) to access the common database containing the ship model. When the workstations are geographically distributed, it would be necessary to access the model through a wide area network (WAN) using public resources for data transmission and the necessary bandwidth at a reasonable cost.

Until recently, several factors have contributed in preventing the extension of the use of integrated systems to this environment in a proper way. One is the high communication cost associated with the required bandwidth, which is very high in the case of most integrated

systems. Another is the fact that the database management software used to manage product models has not been designed to reduce data transmission ratios.

There are two different strategies to cope with these communications problems that may result in a CAD System which is particularly well suited for concurrent use from remote sites:

1. Use a largely topological definition (rather than geometrical) of the product model. The volume of the database is lower by an order of magnitude and the amount of information to be transferred decreases accordingly.

2. Take full advantage of the infrastructures and capabilities that a relational database with SQL and database server may offer in this area. This reduces the required bandwidth by an additional order of magnitude.

The topological definition of the product model is explained in more detail in a specific chapter of this paper. With respect to the second point, two database architectures are available to deal with scalability issues.

The first one is preferred because it allows working in a WAN exactly in the same way as in a LAN. This architecture requires a single database to which remote users are connected via frame relay, ISDN or any other corporate communication infrastructure. In this way, a large number of users can be working concurrently from remote locations on the same ship model.

A good example of this is the case of the Kvaerner Masa-Yards company, which has implemented FORAN System for basic ship design in its Turku and Helsinki shipyards. The figure 1 represents the configuration at Kvaerner Masa shipyards. End-users at Helsinki or Turku run the FORAN modules from a local server. FORAN database is used from a local database server or from the other site, depending on the project. Some common files (such as hull forms and user-defined macros) are located in a shared disk to keep them exactly the same for both shipyards. There is a 100 Mbit/s network at each shipyard and the dedicated line between the shipyards is 30 Mbit/s.

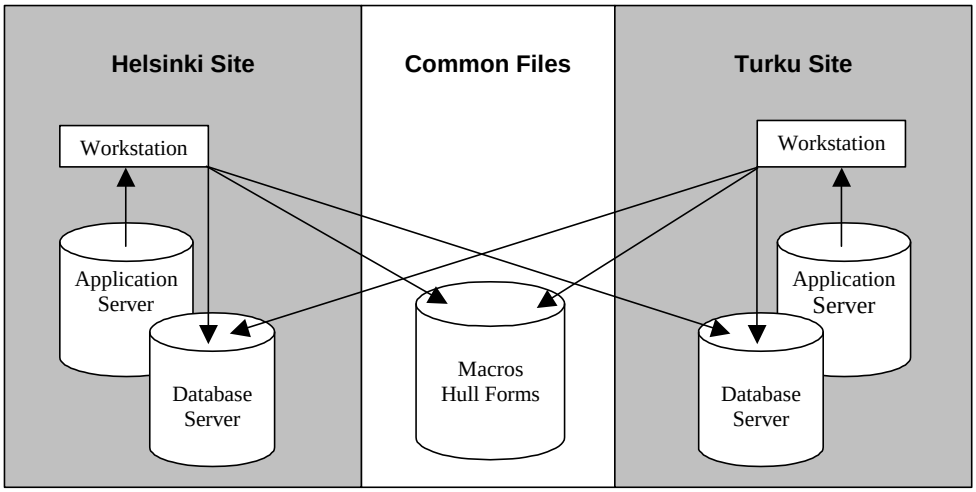

Fig.1: Kvaerner Masa-Yards collaborative engineering with FORAN

This solution allows concurrent engineering between the shipyards. The performance has proven to be acceptable with a scalable number of users.

When the number of remote users is very large or the available bandwidth is insufficient, the second architecture (distributed) can be used as an alternative or a complement. This architecture is based on the distribution or replication of databases between the master location (shipyard) and the remote sites (subcontractors). The coordination performed by the shipyard, owner of the ship model, can be made in a very easy and efficient way.

3 Visualization Framework

A visualization framework has been developed in order to display and manage the 3D graphic scene of a design. This visualization framework has been implemented as an application program interface (API) that works with a graphic library. Working with a set of visualization functions provided by the API instead of working directly with a graphic library, gives a great flexibility to the system visualization in, for example, the case of changes in the graphic library. At present, the visualization release of this API works with OpenGL, but can be easily changed to work with, for instance, DirectX.

The visualization framework provides the usual visualization tools that are needed in 3D design, such as zoom, pan, different point of views with high performance in order to deal with large visualization scenes. In this sense, a Level of Detail (LOD) management has been implemented to achieve the best performance in complex scenes.

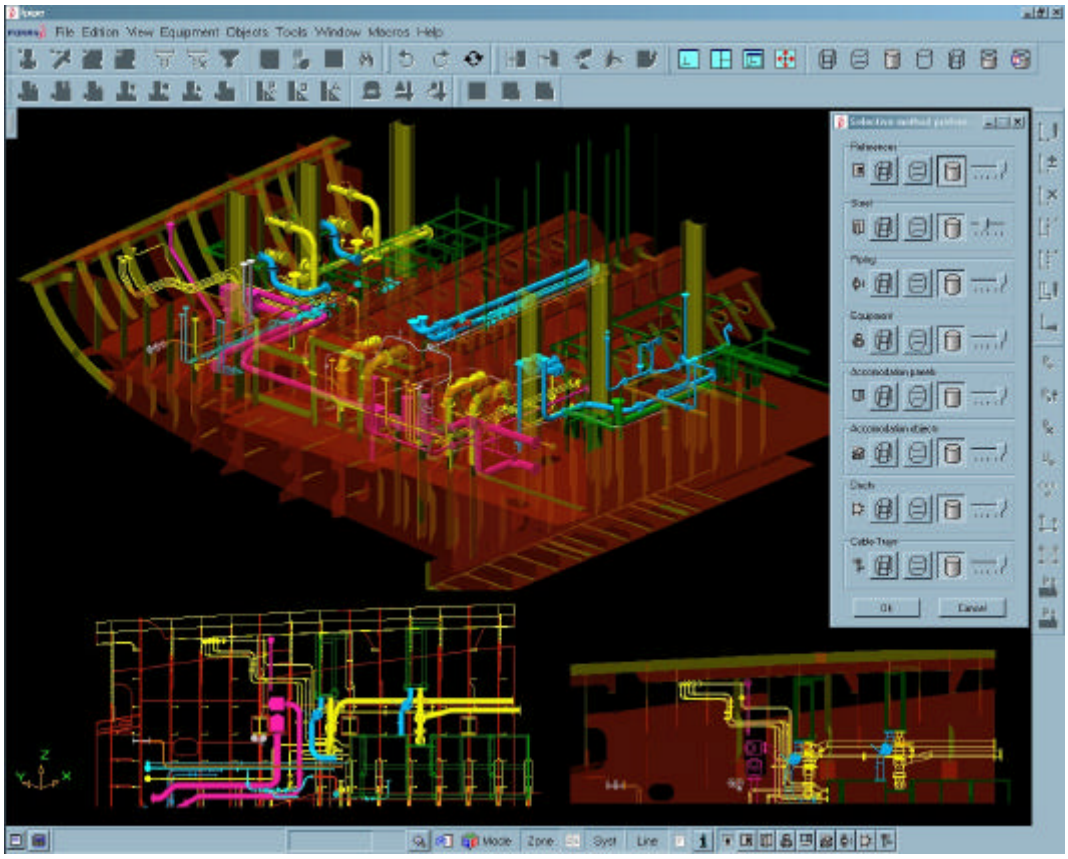

Fig.2: Outfitting model with different visualization methods (Shaded and Wireframe) and transparency

Functionalities for efficient selection and filtering tools are also available during the design process. There are also tools to hide scene components in order to speed up interactive

graphical manipulation of a 3D scene. Visualization attributes of the different scene components can be easily configured by user.

The visualization framework includes tools to customize and manage different scene views. There are available different visualization methods such as wire frame, hidden lines removal or shaded view. In order to simplify and clarify the work in a complex scene, the visualization method works at component level, this means that in a same scene the designer can have different components displayed in different methods (see Figure 2). Also different levels of transparency can be applied to individual components or groups.

A basic navigation tool, for use during the design, is also provided by the visualization framework. This functionality allows a basic inspection mechanism through navigation in a scene. There is also a specific and high performance environment for inspection and navigation in large scenes.

4 Topology Framework

The management of the relationships and dependencies among the parts involved and the automatic updating of the geometry are necessary in order to improve the efficiency and the flexibility during the design tasks. The relationships defined among different components allow the establishment of dependencies for changes. If an element A is modified, the elements depending on A and related with A, must be updated in position, dimension or both. We define these dependency relations among the involved elements as *topological constraints*. The topology framework provides the structure and functionalities to manage the existing topological constraints in a ship design.

Topological constraints have to be kept in the product model to maintain changes in the design and to update the location, shape and dimensions of the components and parts.

Some of the structural parts are directly related with the hull surface. In fact, there are components such as frames, which are defined on the hull and follow a curve on the hull surface. This means that any change in the hull surface has to be transmitted to the related components. A modification or refinement in the hull of the ship implies the redesign of the dimension and position of the structural related elements such as deck surfaces, profiles and plates. Using *topological constraints* the elements can be updated in an automatic way.

Modeling parts are related with *topological constraints*, where the location and dimension of a part depend on the dimensions and locations of others parts. This can be understood as an adaptive modeling.

Due to the concurrent nature of the application model described, topological relations must be handled for all components of the database whether they have been read or not. This is performed by the use of ghost parts and database autoload procedures.

More specifically, it can be considered the example in Figure 3 that represents a simplified transversal section of one of the sides of the ship.

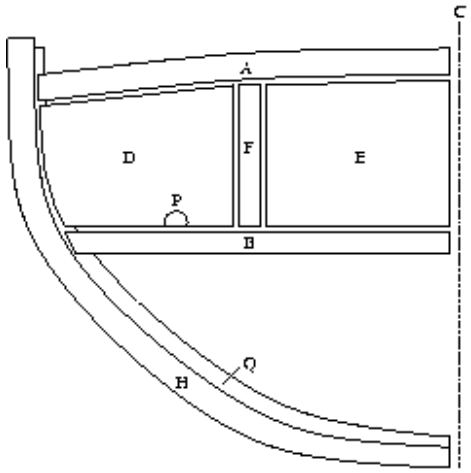

Fig. 3: Simplified transversal section.

Ship hull modifications must be propagated to the main deck, contiguous panels, lower decks and center plane reference, as they produce changes of their shape or location. The center plane reference must propagate its shape modifications to the different decks and plates. Changes in the shape or location of the lower decks can modify the position of the ship equipment; this forces specific modifications in the associated pipes which end up in changes in the panels that must be crossed by the pipes.

Figure 4 shows the corresponding topological constraints of Figure 3 components.

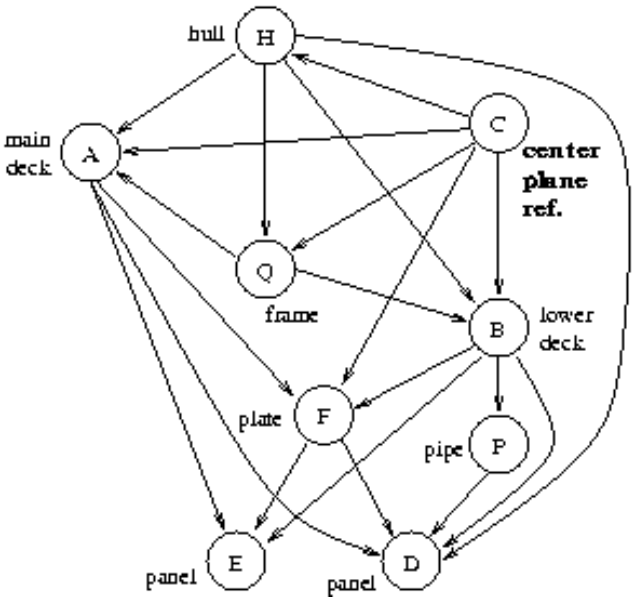

Fig 4: Example of topological constraints among ship components.

The above explained case is a simplification in order to state the problem. In fact, in real ship design, a large number of components in each section are involved. In average, there can be from fifteen to a hundred sections and from a dozen to several hundred structural components as frames, plates, decks, and panels.

A ship is designed hierarchically, by composition of simple parts into more complex ones. The natural way of representing the dependencies and relationships is by using a directed acyclic graph (Rappoport 1993). From a directed acyclic graph that represents the topological constraints among ship components, it is possible to compute the sequence to perform the automatic update of the components.

We define the *topological constraint graph* as a directed acyclic graph where nodes are components of a design and edges represent the topological constraints.

The update process of the components related with *topological constraints* has to be performed in a specific order. In Figure 4, the evaluation of node P has as a prerequisite the evaluation of nodes Q, B, C and H. To update the components after a change, it is necessary to process the nodes in such an order that no node is processed before any node which points to it.

Using a *topological sorting* (Aho 1983) it is possible to compute a linear ordering of the graph nodes, such that if there is an edge from node i to node j, then in the linear ordering, node i appears before node j.

Topological sorting is based on a graph depth-first search. The cost of a *topological sorting* is linear (O(n) where n= number of nodes). In the case of Figure 4, the *topological sorting* produces the sequence: C H Q A B F P E D.

A generic and integrated data structure, decoupled from the product model database, is proposed (figure 5). This data structure allows managing and updating of the geometric elements (parts) related with *topological constraints*. The data structure is based on a directed implicit graph where the nodes are ship structural components each one maintaining its own list of related components (*topological constraints*). The graph is explicitly traversed when the update process is needed. The ship structure update is performed by a set of class objects that have access to the ship structure components.

The present software development is based on object oriented techniques and pattern-based design (Gamma 1994) in order to have a high level of software reusability and decoupled data structure.

5 Product Model

One of the key factors that increase dramatically the complexity of a shipbuilding CAD System is the different modeling capabilities which are required through the different stages of the ship engineering process. This happens in the context of a very short design life cycle, except for navy shipyards, where the demands are for very flexible and powerful modeling tools.

In shipbuilding, the complexity of the product model increases through the different design cycles. Concept design contains a small amount of information, classification design contains much more information and detail design contains a very large amount of information.

In the initial (concept) and basic design cycles the most relevant elements of the product model are defined by means of surfaces. This model of moulded surfaces of the ship also constitutes a topological reference system for the definition of any other element of the model. This requires a close integration with any other modeling capability.

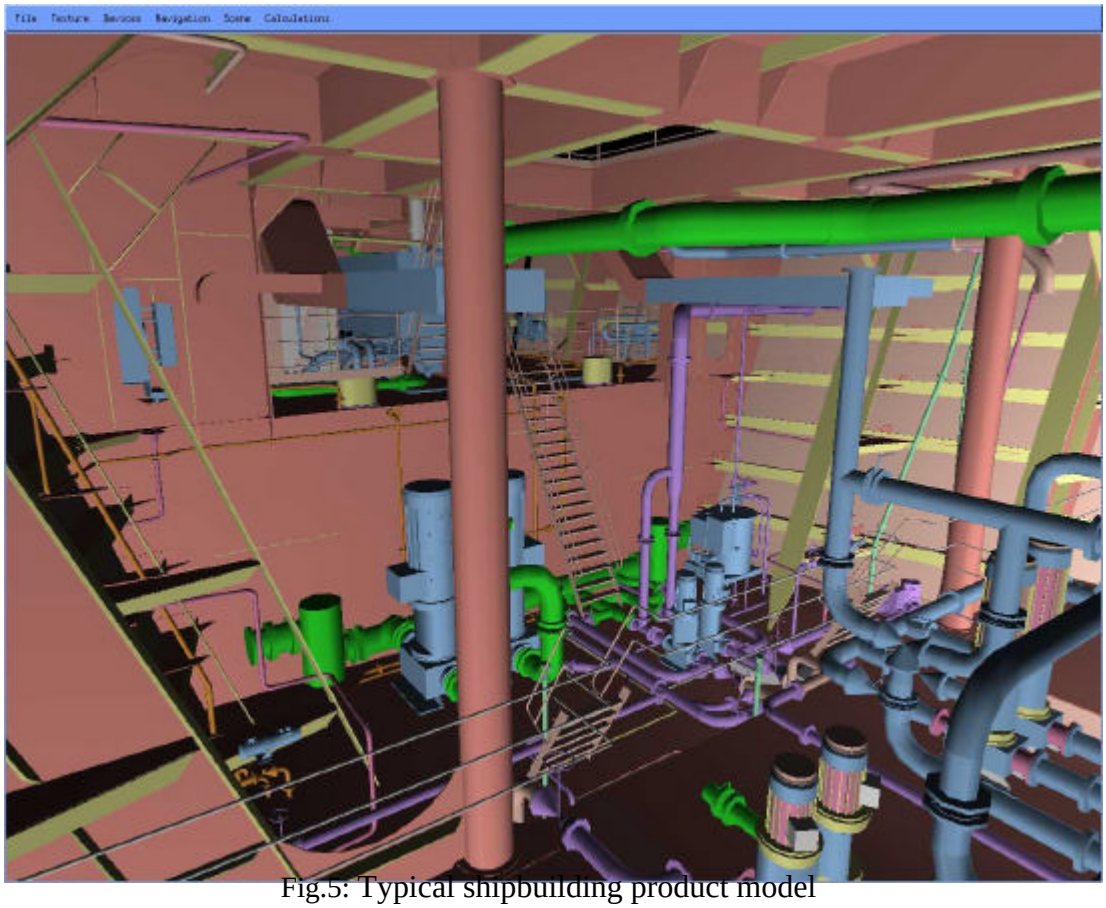

Fig.5: Typical shipbuilding product model

In detail design, the most critical requirement for the product model is the integration of all the disciplines, which in contrast is better implemented by means of a solid model with powerful interference checking capabilities. As mentioned previously, frequently the definition of the solid elements is related or based on the surfaces elements, which requires the extensive implementation of the topology with sophisticated algorithms for surface to solid conversions.

Each of these types of models is explained in more detail in the following paragraphs.

5.1 Surface Model

The surface model must contain all the valuable model information developed during the initial stages of the design. To be able to define all the moulded surfaces of the ship (hulls, decks, bulkheads, superstructures, appendages, etc) a powerful and easy to use formulation must be used.

The most difficult requirement that a surface model must cope with is the capability to

support the fitting and fairing process. Fitting and fairing tasks are usually the starting point of the ship design process. These tasks take a considerable amount of time to be completed, while the resulting product is essential information for the following engineering works.

To deal with this bottleneck, several strategies have arisen. The most effective is that presented by topological integrated systems. In this kind of system, as soon as a preliminary surface model is obtained, the following processes can start to work with this information. In parallel, the surface model is refined and faired, without preventing other tasks from going on. Once the surface model is finished, the work performed on the preliminary model is automatically updated, based on a topological definition. Topological definition is invariant for small changes (such as those changes introduced by the fairing process).

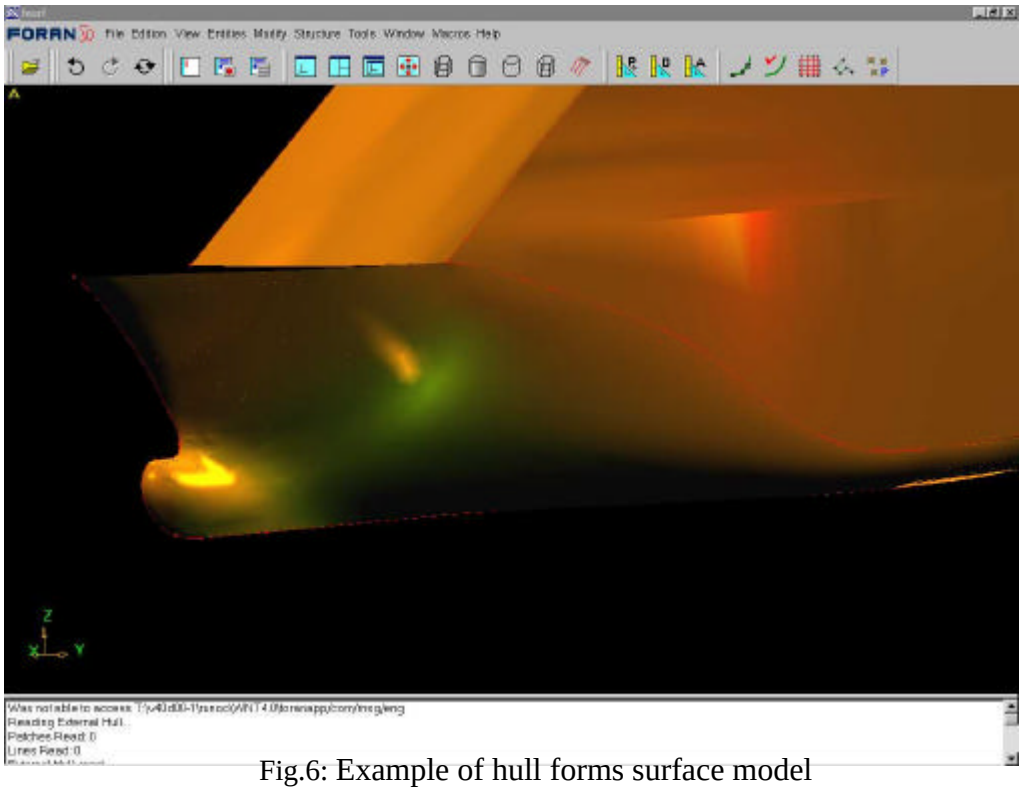

Fig.6: Example of hull forms surface model

Under this scheme, a complete and effective surface model must fit the following requirements:

- Fast fitting of the ship surface model within a stated accuracy (hull, double hull, superstructures, decks, bulkheads, etc)
- Easy and intuitive fairing process based on sophisticated algorithms, allowing the user to be freed from dealing with complex mathematical concepts.
- Oriented to a topological definition. The basic geometry is defined only once. Other information is defined with respect to this geometry. This includes management of ship surface model transformations and local and global changes.

State of the art technology related to surface modeling indicates that the most comprehensive yet powerful formulation is the NURBS (Non Uniform Rational B-Splines). The cost to pay is complexity. This complexity is reflected with a joke about the acronym (Nobody

Understands Rational B-Splines).

To develop the powerful tools to facilitate the fitting and fairing process based on this complex formulation, sophisticated algorithms must be implemented. But even with these algorithms, the direct manipulation of a NURBS surface is a complex task for the designers. The shipbuilding designer is traditionally more used to working with curves representing a set of planar sections as a mean to deal with the complexity of the ship surfaces, and is able to perform the fitting and fairing process based on these sections.

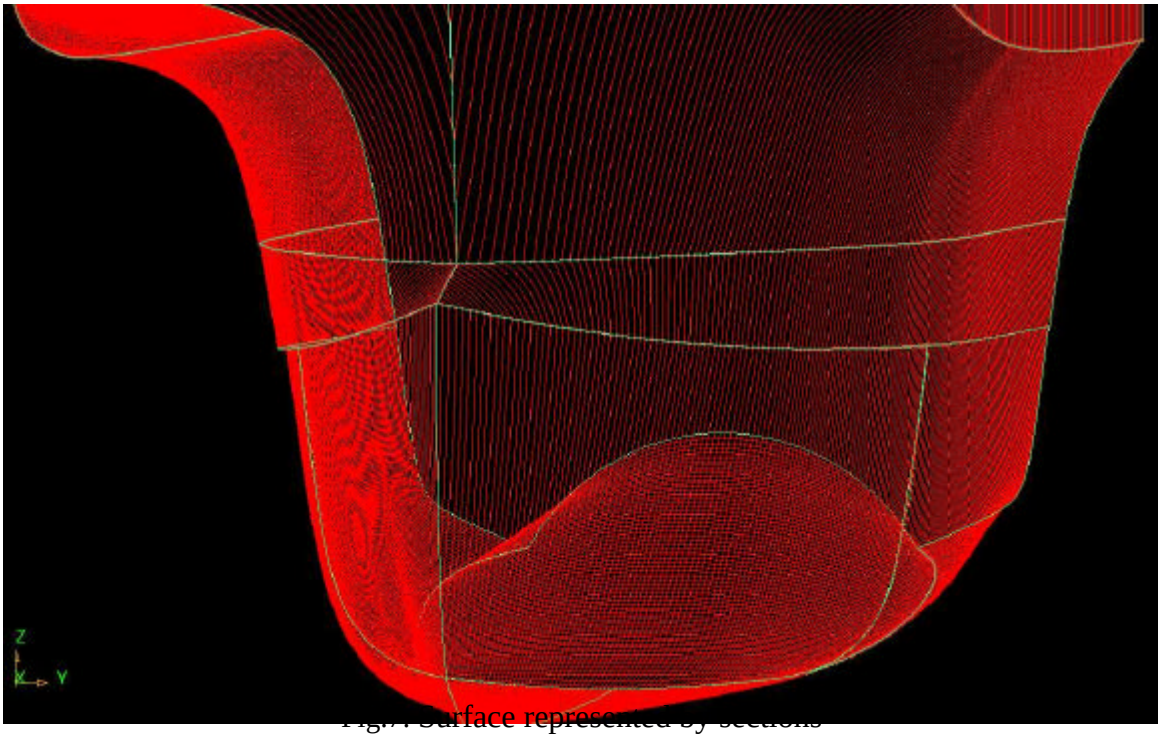

Fig.1 Surface represented by sections

Thus, the best approach to build a surface NURBS model must fit these requirements:

- The user should be able to work with curves, because it is much easier to manage curves than surfaces
- Surfaces should be constructed using curves to define their geometrical properties.
- Direct manipulation of surfaces (by moving the control points) should be reduced drastically
- The philosophy of the method should be that when the surface is produced it is close to the surface required, reducing drastically further manipulations
- Trimming of surfaces is necessary to avoid limitations of quad topology
- The user should be able to directly edit points on the surface
- Automatic fairing algorithms should be available
- The user should be able to interchange geometrical information using most common standards (IGES, STEP, DXF, etc)

These have been the premises which have been used in the development of the surface model framework of the FORAN System.

5.2 Solid Model

The main goal of the solid model is to have a 3D geometric representation of the different ship components. The solid model representation is a virtual geometric prototype of the ship that can be tested and validated. The solid model must be able to offer functionalities and modeling operations related with part design and assembly design. These are two main environments with different sets of requirements.

In order to achieve a high level of productivity, the part modeling is based on primitive creation functions. A set of 3D primitives is used to model the different parts. The part modeling enviroment allows to model more complex parts defining a group of basic primitives and the 3D location of each part. Using 3D primitive geometric elements it is possible to have an efficient geometric representation of the elements involved in ship design.

The assembly design environment provides a logical structure for grouping and organizing parts into assemblies and subassemblies. The user is able to identify individual parts, keep track of associated part data, and maintain the relationships among parts and subassemblies. Tools for assembly modeling have been developed in order to achieve a high level of productivity. These tools include navigation, inspection and manipulation of large, complex assemblies. Components can be selected by all types of attributes or spatial location.

Assemblies can be created using the top-down or bottom-up design methodology or a combination of both methods. Bottom-up design is the traditional method. In bottom-up design, parts are created, inserted into an assembly, and relationships are established as required by the design. Bottom-up design is the methodology used when the design is performed with previously constructed parts. In bottom-up design component parts are designed independently and thus allows the designer to focus on the individual parts. Top-down design allows to design a component in the assembly environment. The geometry of an assembly to define a new component and its relationships with the other parts is used to fix the location.

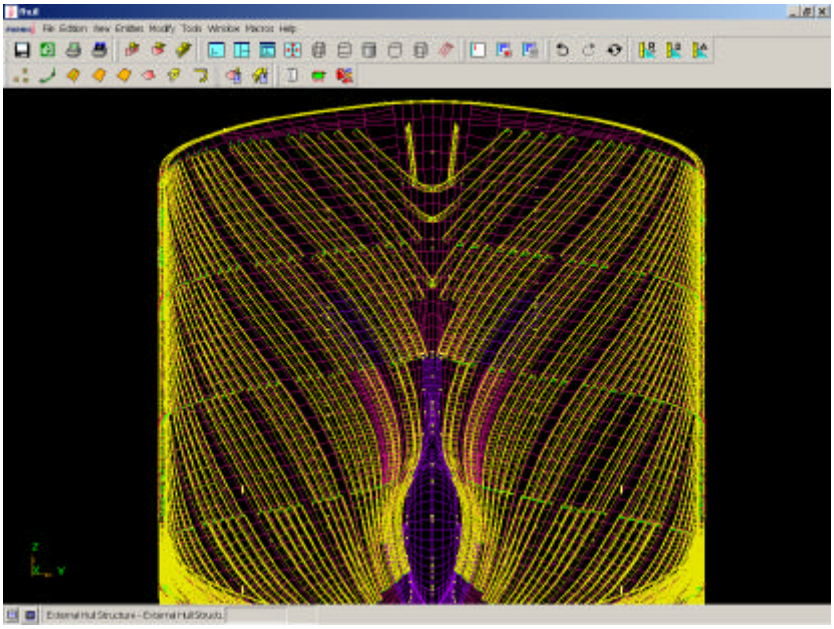

Fig.8: Example of shipbuilding specific parametric objects for hull

The solid modeling enviroment has been designed to optimize shipbuilding specific modeling environments such as piping and structural parts modeling. Automatic tools for piping are required to achieve a high level of productivity. In the same way, modeling of structural parts also has a set of modeling operations for high performance in the design of profiles, plates and panels (Fig. 8).

Finally the solid model environment provides a set of different file formats in order to export and import components parts or assemblies designed in other CAD applications. Specific methods have been implemented to optimize the geometry representation of the imported components.

References

RODRIGUEZ, A.; VIVO, M.; VINACUA , A.(2000), *New Tools for Hull Surface Modelling*, 1st Int. Conf. Computer and IT Appl. Maritime Ind., Potsdam

SOLANO, L.; GURREA, I.; BRUNET, P.(2000), *Topological Constraints in Ship Design.* Proceedings 4th IFIP WG 5.2 Workshop on Knowledge Intensive CAD KIC-4, U.Cugini and M.Wozny eds, Parma.

ALONSO, F.; CEBOLLERO, A.; GOMEZ, A.; RODRÍGUEZ, A.(2002), *Collaborative Engineering in Shipbuilding*, ICCAS'2002

SOLANO, L.; GURREA, I.; BRUNET, P.(2002), *Topological Constraints in Ship Design*, in 'From Knowledge Intensive CAD to Knowledge Intensive Engineering'. U.Cuggini and M.J. Wozny edtors. Kluwer Academic Publishers.

Aarnio, M.,(2000). Early 3-D Ship Models Enables New Design Principles and Simulations. In *1st International Euroconference on Computer Applications and Information Technology in the Maritime Industries (COMPIT'2000)*, pp. 5-17.

Alonso, F., Andujar, C., Brunet, P., Garcia, L., Navazo, I., and Vinacua, A. (1996). Virtual Reality Tools In Shipbuilding Design. In *TeamCAD'97*, pp.39-43.

Duran, J., Puzas, M.J., Alonso, F., and Garcia,L. (1995). The use of a 3D product model orientated CAD/CAM System in a small/medium size shipyard. In *CADAP'95*.

Garcia, L., Fernandez, V., and Torroja, J. (1994). The Role of CAD/CAE/CAM in engineering for production. In *ICCAS'94*.

Aho, J., Hopcroft, J., and Ullman, J. (1983). *Data Structures and Algorithms*, Addison-Wesley.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.