A PRODUCT MODEL FOR SHIP FUNDAMENTAL DESIGN
BASED ON WORKFLOW ANALYSIS

Hiroyuki Yamato, Takeo Koyama, Akira Fushimi, Hiroshi Masuda and Akiyoshi Iwashita.
University of Tokyo, Tokyo, Japan

Abstract

In general, the product model may be defined based on the physical structure of the product. However, the product model must comply with the design procedure. In this paper, the product modeling methodology is discussed to include the design process based on the workflow analysis. The design workflow is represented according to IDEF3 methodology to define the process and data transition. General Product Modeling Environment has Common Frame Library, which is a collection of the commonly usable classes of parts of ship. GPME has been used to add a specific class to represent the intermediate product definitions in the design process as a part of Extended Frame Library. And the effectiveness of the methodology is illustrated by the corrugated bulkhead structure design.

Introduction

The product modeling is the main concern for the informed shipbuilding industry. The single source data may be most effectively available for the Computer Integrated Shipbuilding from very basic conceptual design through the ship production and even for the maintenance. The product model must be easy to get and to give data in every stage of the design, production, operation until decommission. The product model can be mainly achieved by examining the physical structure of the product. However, it is rather difficult to take account of the work process, especially the design process, into the product model structure.

On the other hand, the workflow analysis is becoming one of hot issues recently in order to streamline the design and production process for higher efficiency in time and labor. Some effective methodologies such as IDEF3 have been proposed. The workflow analysis enables to clarify knowledge concerned with the work process. The result of workflow analysis is useful for the development of a product model to support the design process.

In this paper, authors would like to discuss the workflow analysis and product modeling methodology in the ship structural design to include the design perspective into the product model. To represent the workflow by IDEF3 to capture the data handled in each activity, and results obtained by the IDEF3 analysis is combined to the product model generation. General Product Model Environment was used to add the class corresponding with design procedure.

Workflow analysis

The workflow analysis is necessary to clarify the intermediate definitions in the design process. That is because the requisite intermediate definitions deeply depend on the design workflow and they are influenced by the way of design. In this chapter, the design workflow analysis methodology will be presented.
**Formal workflow representation**

The design process is not clearly defined especially in industries whose products are complex as shipbuilding industry. Ambiguity or options to take may exist in many aspects of the design process. Even though skilled designers cannot state their way of design in a concrete and unique manner. However, to implement the workflow on the computer, it is necessary to describe the design workflow in a formal way. And not only the process but also the transitive intermediate product data must be represented clearly.

**IDEF3 Process Description Capture Method**

In this analysis, IDEF3 Process Description Capture Method [1] was used. IDEF3 is one of IDEF, an acronym of Integrated Computer-Aided Manufacturing DEFinition, methods that are used to perform modeling work in support of enterprise integration [2]. IDEF3 provides a mechanism for collecting and documenting processes. It enables to express knowledge how a particular system, process or organization works accurately.

As shown in Figure 1, IDEF3 has two description modes to show the work and object transition in the process. Those are a Process Flow diagram and an Object State Transition Network, OSTN, diagram. The capability of describing in two different perspectives is a great advantage of IDEF3.

![Diagram](Image)

**Figure 1. Concept of IDEF3 Description**

456
Process Flow diagrams show the workflow as a chain of units of behavior, UOBs, to capture process-centered views of processes. Precedence links represent the order of UOBs and Junctions do the logic of process branching, ‘and’, ‘or’, ‘synchronous and’, ‘synchronous or’ and ‘exclusive or.’ The shadow of UOB indicates that one or more decomposition diagrams are associated with the UOB for more detailed description.

OSTN diagrams capture object-centered views of processes, summarizing the allowable transitions that objects can undergo throughout a particular process. The relation to Process Flow diagrams can be visible by using Referents, each of which corresponds with one of the UOBs in Process Flow diagrams. Referents are also used in Process Flow diagrams. OSTN diagrams are typically developed only for the important objects of the process description. OSTN diagrams are most often developed after Process Flow diagram; however, it is easier to begin with OSTN diagrams in some cases. There is no definite procedure to develop those two diagrams.

Each element in the two diagrams can have Elaboration to store detailed documentation. The elaboration document typically includes listings of the objects, facts and constraints, also a textual description of that element.

**IDEF3 application to the design process analysis**

As stated above, IDEF3 methodology was used in this analysis. But IDEF3 was designed for the analysis of generally various business processes not only for the design process analysis. It is necessary to determine how to apply IDEF3 methodology to the design process analysis for making the analysis more effective. So authors defined some policies of IDEF3 application to the design process analysis as following.

**UOB elaboration**

In this research, special UOB elaboration documents are attached to each UOBs. They enable readers to understand the meaning of UOB more clearly. Figure 2 shows the UOB elaboration document form used in this analysis. It includes a textual general definition of activity and listings of the work items, created data at this activity, reference data created in preceding activities and external information & knowledge. The work items become UOBs in the decomposition diagram of that UOB.

![UOB Elaboration Document](Image)

![Figure 2. UOB elaboration document form for the design workflow analysis](Image)
OSTN diagram

In this analysis, as the objects of interest are the product definitions, OSTN diagrams should be developed for capturing the transition of the product definitions through the process. It may be natural that existent drawings and technical documents in actual work are made objects of OSTN diagrams. But it is not convenient for the development of product model because the product definitions for designers are elements represented in drawings and documents, not drawings and documents themselves. The object of OSTN diagram in this analysis is regarded as a conceptual design object for the designer, such as the midship structure, the bulkhead structure, the transverse web structure, and so on.

ProSim

ProSim [3], a tool to support IDEF3 methodology, was used in this paper. ProSim is a software tool developed by the Knowledge Based Systems, Inc., and this provides the environment for developing IDEF3 descriptions. The IDEF3 analysis was achieved very easily by using this software.

Workflow analysis of a bulk carrier hull structural basic design

In this chapter, the workflow analysis of a bulk carrier structural design will be presented. Figure 3 is a picture of a typical bulk carrier considered in this analysis.

![Figure 3. Picture of a typical bulk carrier](image)

Hull structural basic design

In this research, the phase of hull structural basic design was analyzed. As shown in Figure 4, it locates between a preliminary design and a detailed design in whole hull structural design. The output of the hull structural basic design is called 'Key Plan', that is basic definitions of hull structure and contains diagrams of midship section, construction profile, deck plan, shell expansion, and so on. The intermediate definitions of hull structure in the basic design are not clear and accurate because the definitions of hull structure in this phase is mostly rough and abstract. The fact is a significant factor why systems in this domain are not integrated well and cannot provide sufficient support for design work.
Captured descriptions in this analysis

In this section, some of the captured diagrams in this analysis will be presented to show how to apply IDEF3 methodology for the design workflow analysis.

Figures 5a and 5b together show a top-level Process Flow diagram of hull structural basic design. It shows a workflow from the design policy decision till getting approval of the key plan. This is the most abstract but overall description of the hull structural basic design. Each UOB which has shadow contain a decomposition diagram for more detailed description.

Figure 6 shows the Process Flow diagram of midship section longitudinal member design, which is a decomposition diagram associated with the UOB 'Design midship section longitudinal member' in the top-level Process Flow diagram. Much more detailed decomposition diagrams exist under the shadowed UOBs. A lot of decomposition diagrams are captured like this.

Figure 7 shows an OSTM diagram for the midship structure definitions. Schematic descriptions of each oval were also represented in Figure 7. The latter part of the diagram is not included in this figure to avoid lengthy description. The development of OSTM diagrams is based on the descriptions of Process Flow diagrams. By making OSTM diagrams, the transition of product object definitions become clearly understandable like this.
Figure 5a. Process Flow diagram of hull structural basic design (1 of 2)

Figure 5b. Process Flow diagram of hull structural basic design (2 of 2)
Figure 6. Process Flow diagram of midship section longitudinal member design

Figure 7. Transitions of Midship structure definitions (fore part only)
EFL development based on the workflow analysis

GPME in this paper has the first version of Extended Frame Library for Shipbuilding, EFL/S. The first version of EFL/S can represent the basic hull structure. However, it is not sufficient for the ship design process to represent intermediate definitions of the product under design. Additional EFL development for more effective design support should be necessary, and this can be accomplished through the workflow analysis of the design process.

In this chapter, an overview of GPME and its frame libraries is described first and a suggestion and a prototype of extension for bulk carrier transverse corrugated bulkhead structure definitions is shown next as a concrete example of the additional EFL development.

**GPME and its frame libraries**

GPME is an environment to provide an object-oriented product model efficiently usable in the wide variety of the automotive, construction, shipbuilding and so forth [4] [5] [6]. GPME is a product of the consortium research project executed by Ship and Ocean Foundation and major Shipbuilders in Japan. The architecture of GPME is shown in Figure 8.

![Figure 8. GPME reference architecture](image)

GPME furnishes Common Frame Library to produce Extended Frame Library by OntoEditor. The product has the common structure in the form of the object-oriented database. The commonality may be used very conveniently in each field of industry and users need only to add their own specialty for their own product model to represent specific data in the product. GPME gives the environment to give user's own product model very efficiently. CFL is to represent common parts of the product in the same industry. EFL should be added on CFL. The end-user can define product models in a graphical way in OntoEditor and the system provide the data structure in an object-oriented database. And the
product instance can be browsed with ProdEditor. GPME has been evolved in the consecutive research project nick-named Advanced-CIM after the end of the original project in 1997. However, the original GPME without sophisticated CFl was used in this paper. The original GPME has the first version of EFL/S. Figure 9 shows distinctive classes introduced to the structural design FL in EFL/S. The intermediate definitions cannot be completely represented and the hull structural basic design are not sufficiently supported by this FL.

![Diagram](image)

**Figure 9.** Distinctive classes of the structural design FL in EFL/S

**An example: EFL to design corrugated bulkhead structures**

Figure 10 shows the corrugated bulkhead structure in a bulk carrier. It is composed of upper and lower stools, and a corrugated plate.

![Picture](image)

**Figure 10.** Picture of a corrugated bulkhead structure in a bulk carrier
Figure 11 shows the transition of all corrugated bulkhead structures captured in this analysis. According to the diagram, the following design process can be read and understood:

Before the hull structural basic design, only the positions of all bulkhead structures have already been determined. In the process of hull structural basic design, the midship bulkhead structures, i.e. bulkheads in the midship area, are designed first and the other bulkhead structures are designed later. This is shown in the mezzanine three ovals.

![Diagram](image)

Figure 11. Transition of all corrugated bulkhead structures definitions

Actually, more detailed object state transition is included in the UOB 'Design midship bulkhead structures' in the lowest four ovals. It indicates that four detailed object states exist between the object state 'BHD positions determined' and the object state 'Midship BHD structures designed'. This transition expresses the following:

In the design process of midship bulkhead structures, the basic dimensions of bulkhead structure shape is determined first. The border shape adjustment for individual bulkhead, which means the modifications of bulkhead structure shape based on the basic dimensions, is executed secondly. The configuration and attributes of primary members are designed thirdly. And the arrangement and attributes of secondary members are designed lastly.

It should be considered that definitions of the basic dimensions in the object state 'Basic dimensions determined' are actually intended for not only the midship bulkhead structures but also the other bulkhead structures. And the basic dimensions are designed by strength requirements. In strength requirements, all of bulkhead structures in a bulk carrier are regarded as one of the two types, i.e. watertight bulkhead and deep tank bulkhead. So two definitions of the basic dimensions normally exist. Each individual bulkhead structures are designed on the basis of the two definitions.
According to the above interpretation that is extracted from the OSTN diagram of corrugated bulkhead structures definitions, authors considered that the object structure of corrugated bulkhead structures definitions in the design process should be as Figure 12. It is composed of objects that express individual bulkhead structures and objects that contain the basic dimension definitions of bulkhead structure.

* No.5 hold is a ballast hold in this case.

Figure 12. Object structure of corrugated bulkhead structures definitions in the design process

The first version of EFL/S cannot support for representation of these instances. Authors tried to design additional classes to the first version of EFL/S for expressing and handling them. The key features of extension are as follows:

- Add a class that expresses a unit of individual corrugated bulkhead structure
- Add a class that contains the definitions of basic dimensions
- Add a function for rough shape calculation of individual bulkhead structures on the definitions of basic dimensions
Figure 13. Extension for corrugated bulkhead structures definitions

Figure 13 illustrates the extension for corrugated bulkhead structures definitions. Classes that are expressed by gray boxes are included in the first version of EFL/S and the other classes and relations are extension that is suggested in this paper.

This extension was developed on GPME and the sample program using the additional classes was also created. Figures 14, 15, 16 and 17 show the sample program result browsed with ProtEditor. In this sample program, trough shapes of individual bulkhead structures are automatically calculated on the definitions of basic dimensions, i.e., W.T.BHD and D.T.BHD. Figure 14 shows base shapes of all bulkhead structures. Figure 15 shows trimmed shapes of all bulkhead structures. Figure 16 shows a base shape of one bulkhead structure at frame number 70, i.e., FR70 BHD. And figure 17 does a trimmed shape of FR70 BHD. Those Figures show that the EFL obtained as shown in Figure 13 can support the design nicely.
Figure 14. Sample of BHD rough shape calculation 1 (all BHDs, base shape)

Figure 15. Sample of BHD rough shape calculation 2 (all BHDs, trimmed shape)
Figure 16. Sample of BHD rough shape calculation 3 (FR70BHD, base shape)

* Border shape of each BHD structure is adjusted in later process.

Figure 17. Sample of BHD rough shape calculation 4 (FR70BHD, trimmed shape)
Conclusion

In this research, firstly, Authors tried to clarify intermediate definitions of hull structure in actual design process by the workflow analysis of hull structural basic design in order to develop an effective product model. The workflow analysis based on IDEF3 methodology was made.

And secondly, Authors tried to discuss how to develop additional FL to the first version of EFL/S on GPME based on their analysis. In this paper, Authors exhibited a suggestion and a prototype of extension for bulk carrier transverse corrugated bulkhead structures definition as a concrete example of the additional FL development.

Finally, Authors have obtained following conclusions:

- Design workflow analysis with IDEF3 methodology is effective to clarify intermediate definitions of product in actual design process.

- The workflow analysis results can provide useful information for EFL development in GPME.

Acknowledgement

Authors would like to express their cordial thanks to the Ship and Ocean Foundation and Toyota Caelum Inc. for providing the development environment for prototyping. Authors would also express their gratitude to Mr. Ken Ito from MH1 for his help in this work. And the work presented is supported by the Ministry of Education, Science, Sports and Culture under a Grant in Aid for Scientific Research.

References

SHIPBUILDING INFORMATION INFRASTRUCTURE PROJECT
(SHIIP)

Tom Rando, Electric Boat Corp., Groton, CT
Lisa McCabe, Electric Boat Corp., Groton, CT

Introduction

The Shipbuilding Information Infrastructure Project (SHIIP) is a three year, $22 million project funded jointly by the Defense Advanced Research Project Agency (DARPA); MARITECH program and by industry. The MARITECH program, begun in 1993, is devoted to improving the competitiveness of the U.S. shipbuilding industry. The SHIIP project is led by Electric Boat Corporation (EBC). The SHIIP team consists of shipbuilders (Bath Iron Works, NASSCO, Atlantic Marine, Avondale, Todd Pacific) and technologists (NIIP/IBM, Computer Sciences Corp., Deneb Robotics, STEP Tools Inc., Data Access Technologies).

The goal of the SHIIP project has been to support the integration of systems technologies within the U.S. shipbuilding industry. The SHIIP project has sought to identify, develop and deploy standards-based protocols that can be adopted by the U.S. shipbuilding industry as a whole. A standards-based protocol may be endorsed by an international standards body, it may be an emerging standard, or it may be a de facto standard. The salient pre-requisite has been that the protocol be implemented by more than one vendor so that the use of the protocol does not unduly tie a shipbuilder to any one technology provider. To a certain extent, U.S. shipbuilders have been at the mercy of systems technology providers. In its current state the industry does not have the wherewithal to make the technology providers respond to its needs. Rates of production are low. The number of U.S. shipyards is small and compartmentalized; first tier yards have systems technology requirements that are quite different from those of second tier yards. Moreover, the industry is highly competitive and has rarely spoken with one voice.

Recent advances in systems technologies, especially technologies developed for the Internet and based on the Java programming language, have for the first time made it possible for an enterprise to deploy an information infrastructure on which its industry-specific application components can be assembled. The shipbuilding information infrastructure provides the systems integration foundation which formerly was available only within high-priced, proprietary, monolithic software applications. Many of these technology advances were developed and proved out by the National Industrial Information Infrastructure Protocols (NIIP) project.

SHIIP Deployment

Although it is an infrastructure deployment project, the SHIIP project has chosen a particular shipbuilding business domain in which to focus its efforts. To date most of the systems technology investment in the shipbuilding industry has been in support of design and engineering processes. At Electric Boat there is substantial amount of design information available in digital form. However, there has been very little activity devoted
to making this information available to the shipyard work force in ways that would streamline the production processes. While deploying broad-based information infrastructure technology, the SHIIIP project has also used the infrastructure to support applications which make more information available to shipyard team leaders, foremen and mechanics.

The SHIIIP project is a shipyard initiative. The SHIIIP project works closely with EBC's production information systems team. In cycle 2 the SHIIIP project selected the Foreman Work Assignment application, which was under consideration by EBC's production systems developers (see Figure 1). The SHIIIP team participated in the gathering of functional requirements, which was led by EBC's operations. After the first round of functional requirements was collected, the SHIIIP project led the development and demonstration of a prototype deployment of the Foreman Work Assignment application. This work was performed in parallel with the ongoing architecture definition work of the supporting information infrastructure. The demonstration was then used to elicit feedback from the users in order to refine the functional requirements. At this point the application development was turned over to EBC developers for production deployment. Finally, the architecture and selected components have become available for re-use in subsequent production applications.

**Figure 1 - SHIIIP Deployment Process**

In some ways, the SHIIIP infrastructure is a comprehensive Intranet for shipbuilding as shown in Figure 2. Central to the infrastructure are the communication protocols that connect the SHIIIP services and applications. Although several protocols were chosen, they are all practical, Internet protocols, and the objective of the project has been to enable only required services and to find the most appropriate protocol for each service. Table 1 summarizes the protocols selected:
Table 1 - SHIIP Communication Protocols

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypertext Transfer Protocol (HTTP)</td>
<td>Web server support for the other protocols</td>
</tr>
<tr>
<td>Remote Method Invocation (RMI)</td>
<td>Distributed object computing</td>
</tr>
<tr>
<td>Hypertext Markup Language (HTML)</td>
<td>Document content without structured data</td>
</tr>
<tr>
<td>eXtensible Markup Language (XML)</td>
<td>Document data intended for human reading as well as computer interpretation</td>
</tr>
<tr>
<td>Lightweight Directory Access Protocol (LDAP)</td>
<td>Directories for the location of enterprise data objects</td>
</tr>
<tr>
<td>Virtual Reality Markup Language (VRML)</td>
<td>Distributed 3D visualization for moderate sized models</td>
</tr>
</tbody>
</table>

![Diagram](image)

Figure 2 - SHIIP Intranet

Since the SHIIP infrastructure is comprised of distributed services and information objects, a directory service is essential so that users can readily locate objects as needed. These directories can be as straightforward as the directories of users and their roles or as complex as directories that represent the ship's product and parts objects and their relationships.

One of the most significant services of the infrastructure is the enterprise data management service. The purpose of this service is to provide uniform access to enterprise information, which may be stored in legacy datastores as diverse as file systems, hierarchical databases, relational databases or legacy applications. This information also spans a number of business process domains — such as up-front
requirements and concept formulations, physical ship design information, and ship assembly and outfitting information. The selection of technology for the enterprise data management service was the most difficult challenge of the project.

A service that provides users with timely notification of significant business events has been identified in the Integrated Development Environment (IDE) concepts of operations described in a number of recent naval shipbuilding programs. The SHIIP project has designed and demonstrated such a service using software agents.

The user interface into the shipbuilders' Intranet is a key element. The SHIIP project has designed and demonstrated a user interface component called the SHIIP Desktop. The SHIIP project has designed a Desktop with two modes, and both have been demonstrated in the second year of the project. In conjunction with providing the user interface, the SHIIP Desktop supports the strong authentication of users to the enterprise system.

The service most in demand by the shipyard work force is the distributed visualization of the ship product model. The SHIIP project has identified a number of use cases for distributed visualization. These use cases have been formulated in detail, and appropriate visualization technologies have been mapped to each use case. The conclusion of the SHIIP project is that it would be a mistake to attempt to apply any one visualization technology to all use cases.

The remainder of this paper describes how each module of the infrastructure was deployed in the second year of the SHIIP project.

Enterprise Data Management: CORBA or Java?

An enterprise data object (sometimes referred to as a Business Object) is a specialized type of data object. An enterprise data object inherits the features that are needed in order to be used in an enterprise software application: persistence, transactions, security, and load-balancing. What signifies that an object is an enterprise data object is that a large number of users need to have access to the object simultaneously. Such an object is significantly different from an object that has been designed to run in a standalone personal computer application.

If a user today processes an enterprise data object, such as an assembly in the ship’s product structure, the expectation is that that assembly will still be available tomorrow (persistence). If a user moves a part from one assembly to another, the expectation is that if the system fails after the part is deleted from the original assembly, the part will not disappear from the product structure (transactions). If a user is not authorized to change parts in the product structure, the expectation is that the system will prohibit that activity (security). And finally, if a hundred users need to access parts at the same time, the expectation is that the system will not make ninety-nine users wait until the first user is finished (load-balancing).

So called “middleware” technology represents the state of the art in enterprise data management today. This technology supports a multi-tiered architecture, in which applications are separated from the underlying database in order to make it possible to keep database management functions independent of application or business logic. One goal is to begin to free application developers from the overwhelming complexities of today’s information technologies. There are three prominent middleware technologies:
Microsoft's DCOM, the Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA), and Sun Microsystems' Enterprise JavaBeans (EJB). Because of its commitment to standards-based, non-proprietary technologies, the SHIIP project has focused on CORBA and EJB as the two alternatives for evaluation. The objective was to identify the systems requirements of the shipbuilding industry and to determine if either technology is better suited to satisfy these requirements. As part of the evaluation process, the SHIIP project implemented two prototypes of the Foreman Work Assignment objects – one using CORBA business objects and one based on the EJB specification as shown in Figure 3. As a result of the evaluation and of the prototype work, the SHIIP project concluded that EJB is the technology that is best-suited to satisfy the requirements enterprise data management for the shipbuilding industry.

![Diagram of SHIIP Middle Tier](image)

**Figure 3 - SHIIP Middle Tier**

The goal of the CORBA standard is to support the technical interoperability of distributed objects. At the time the OMG was formed, there was a perception that industry's highest priority was to be able to construct complex systems from software modules written in different programming languages. Moreover, the OMG felt that the requirement for "technology independence" was incumbent on them, that is, in order to be universally applicable, OMG standards should not be tied to any implementation technology. When the SHIIP project began, CORBA was the only available distributed object standard. One objective of the project has been to determine whether CORBA could be used to benefit the shipbuilding industry. Consequently, a substantial amount of the prototyping in the first two years of the SHIIP project was based on CORBA. The results of that prototyping and investigation have produced a number of lessons learned:
• Technology independence, while useful in theory, should not be pursued at the expense of developing working implementations.

• Technical interoperability is not a requirement for the development of an industrial information infrastructure. Supporting applications and software components for the information infrastructure are homogeneous enough to be implemented in a single implementation technology.

• OMG, while advocating separation of services, has produced, in the arena of domain standards, wrappers for monolithic applications.

• OMG has been hampered by the inability to reach timely consensus.

The Enterprise JavaBeans specification was published in April, 1998 – near the end of the second year of the SHIIP project. Java technology has a different prime requirement than CORBA technology. Java is dedicated to portability, the notion of software development whose goal is “write once, run anywhere.” Our experience has been that this capability is much more valuable to the shipbuilding industry than technical interoperability. Across alliances of shipyards, throughout the maritime supply chain, and within a single shipyard, it is very important to have access to software that can be run on any hardware/operating system platform available. Java provides a universal computing environment.

In addition, Java and Enterprise JavaBeans brought along some unexpected benefits that quickly persuaded us to turn to this technology to provide the foundation of the shipbuilding information infrastructure. Java provides an implementation technology. Java is not a conceptual object definition language. In fact, Java incorporated the strengths and purged the weaknesses of the C++ programming language with such success that Java developers are consistently more productive than C++ developers. These productivity gains are not only widely reported but were also noticed within the SHIIP project. Sun Microsystems has also been very successful at delivering timely specifications (and implementations). Finally, Java comes at such a low cost of entry that it promises to be adopted by second tier as well as first tier shipyards.

In this development cycle, the project prototyped two middleware technologies, CORBA business objects and EJB objects. The business object infrastructure was used to implement the Foreman Work Assignment application. With this application, a foreman can access EBC’s MRP system through the SHIIP Desktop in order to assign work items to members of his crew. Ultimately, mechanics could use the system to designate their own work plans. The first challenge was to retrieve the Work Order data from the legacy IMS database on the mainframe. An Oracle procedural gateway was used for the prototype. Two teams of developers produced Work Order business objects for the middle tier application. One team modeled the objects using OMG’s Interface Definition Language (IDL), and the other modeled the objects using Java and the EJB specification. Since there was no commercial EJB server available at the time, SHIIP developers simulated the EJB environment by adhering strictly to the EJB specification. On the client side, the EJB specification was also used to define the remote interfaces to the Work Order objects. Even the CORBA client stubs were wrapped with these interfaces. In this way it was possible to develop one set of user interface components that supported both prototypes.
Notification Service

Figure 4 illustrates an overview of the SHIIP Notification Service. This includes work that has been completed as well as work that is scheduled to be done in the final year of the project (which is indicated by the shaded box). The foundation of the Notification Service is the delivery system. Notification of interesting business events is performed by an asynchronous messaging system. The message conveys information regarding the occurrence of a business event to a user. The messaging system must be portable enough to deliver messages to a wide range of clients. The first prototype was based on the CORBA Event Service. This approach depends on the availability of CORBA clients throughout the shipyard. The next prototype will be implemented using the Java Messaging Service (JMS). Both the CORBA Event Service and JMS implement a publish-and-subscribe design pattern. After a business event is generated, it is turned over the messaging system. The messaging system permits users to subscribe for types of events. When events of this type are received at the messaging system, they are delivered to interested users. If there are no subscribers for a message, it is not delivered. The final delivery to the user will be selected by the user and can range from an email message, to a pager message, to a notification on the Desktop.

![Diagram of SHIIP Notification Service]

Figure 4 - SHIIP Notification Service

One innovation of the SHIIP Notification Service is the use of software agents. A software agent is a small software application that can be highly personalized to meet the needs of an individual user. Software agents are often rule-based and are intended to play the role of “agents” for their users. The enterprise notification service is an especially apt application for software agents. An automatic notification system may threaten to deluge users with unwanted messages. The ability to subscribe only for designated types of
events may not be sufficient to protect a user from such a deluge. The notification agent
can be configured by each user to provide a much more intelligent filter for potential
messages. Through a simple interface, the user can specify not only what type of events
are of interest but also under what conditions the user wishes to be notified or not be
notified of the event. The agent makes these decisions based on rules that have been
added to the agent and based on the attributes attached to the event. For example, a
foreman may request to be notified if a stop work order is attached to one of his jobs —
but only if that job is currently on the schedule horizon.

The next version of the prototype will focus on the event definition and event
generation portions of the service. Until now, events have been generated by polling
the database for significant state changes. This approach is non-obtrusive but may impose
burdensome overhead for a database with a high transaction volume. In the next cycle,
the Notification Service will become more integrated. The design and prototyping of the
Notification Service has demonstrated a need for an explicit model of the events that are
of interest throughout the shipbuilding process. Following traditional practice, the SHIIP
project has developed object models (data dictionaries) for the entities and relationships
that are contained in the enterprise data management system. Object definition languages,
however, do not typically provide facilities for the modeling of Business Events. The
SHIIP project has begun to define a methodology for a formal definition of Business
Events.

The inclusion of Business Events as part of the shipbuilding object model
suggests that the enterprise data management system should be more closely integrated
with the Notification Service. The use of JMS in the next version of the Notification
Service prototype is designed to accomplish that goal. With this approach it will be
possible to implement transactions that encompass both database updates and resulting
events. The transaction will guarantee that if an update is requested but fails, the message
will not be delivered.

**SHIIP Desktop**

One responsibility of the SHIIP Desktop is to authenticate the user, that is, to
guarantee to the enterprise systems that the user is really who he claims to be. One typical
method of authentication is to provide a user id and password. This approach tends to
become inconvenient to users when they are suddenly given access to scores of services
through the information infrastructure. Inevitably, different services end up using
different user id’s; and as services demand new passwords, it becomes challenging to
keep track of which passwords are currently in effect for which services. Another method
relies on public key/private key technology. With this method, each user is provided with
a private key (often on a floppy disk), the corresponding public key is managed in the
enterprise person directory. This method becomes inconvenient when the user needs to
move about the company. In this scenario, the user must bring his private key file with
him and install it on each client machine that he uses.

The SHIIP Desktop has prototyped a third method, based on iButton technology.
The iButton is button-sized Java computer. It can connect through an inexpensive adapter
to a serial or parallel port on a PC. It is powered by its host computer. The iButton is
capable of storing a user’s private key, distinguished name, and the unique identifier of
the iButton. In addition, it can generate time stamps. With these capabilities, the iButton can generate a string of characters that represent the user's "credentials". Figure 5 illustrates the authentication interactions that have been prototyped by the SHIIIP project.

![Diagram showing authentication process]

**Figure 5 – Authentication**

When the user starts the SHIIIP desktop, an authentication window prompts the user to attach his iButton to the client computer. The authentication window drives an authentication applet, which asks the iButton to provide the user's credentials. The iButton generates a string with the user's distinguished name, the ring id, and a time stamp. The iButton encrypts part of this string using the user's private key; this represents an authentication message "signed" by the user. The authentication applet sends the credentials to the authentication server. The authentication server is responsible for determining whether the user is, in fact, who he claims to be. The authentication server relies on the enterprise Person Directory to make this determination. The Person Directory is implemented using the LDAP protocol and an LDAP server. LDAP is the Internet protocol for directory services. The directory entry for the user is located in the directory based on user's distinguished name. The user's directory entry manages the user's public key, which is sent back to the authentication server. If the authentication server can successfully decrypt the credentials with the user's public key, then the user is strongly authenticated. The authentication server allows the user to access the Desktop.

The LDAP server is also the service that manages the roles that the user is authorized to play in the enterprise. This service is the foundation for managing access to enterprise services based on role. Access control within the information infrastructure will be designed and prototyped in the next year of the project. Also scheduled for the next prototype is the integration of the strong authentication service with a commercial off-the-shelf EJB server.
The SHIIP Desktop is illustrated in Figure 6. As the figure shows, the SHIIP Desktop is designed to work in conjunction with a Web browser. The SHIIP Desktop is implemented using JavaBeans technology. The Desktop is divided into two regions. The Type Manager Bar houses icons which control operations applicable to “types” of objects, such as creating, deleting, and locating. For example, the Work Order (WO) type manager would be used to create a new Work Order. The palette region houses business object instances, icons, and lists of icons. The Work Order user interface component is used to present the contents of Work Order to the user. Every business object has two possible representations on the Desktop – as a complete user interface and as an icon. Each icon represents a business object. Double clicking the icon causes the Work Order user interface to open; dragging and dropping the icon adds a Work Order to some other Desktop object. The Desktop also manages lists of business objects, which represent results sets from some query against the system.

![Figure 6 - SHIIP Desktop](image-url)

The Desktop presents a familiar graphical user interface in which objects appear to be resident in the user’s personal computer but are, in fact, dispersed across the enterprise. When a user drags a file icon to trash can on a stand alone PC, the file itself is on the PC, and so is trash service which deletes the file. When a user drags a Work Order icon to the trash bin on the Desktop, the Work Order may actually be an object that resides on an enterprise server and is stored in a mainframe database, and the same is true of the service which deletes the Work Order. The Desktop extends the range of the personal computer to provide the authorized user with access and control over every component in the enterprise's information infrastructure.

In order for the Desktop to be an effective user interface into the enterprise information infrastructure, there are a number of technical challenges or requirements that must be satisfied. The Desktop must, first, provide an interface to distributed business objects. It must be practicable to deploy across thousands of personal computers. It must enable the co-operability of business objects from a variety of application domains. It must be able to represent objects implemented across the entire range of
enterprise component and Web technologies. Finally, the Desktop must be customizable to suit the requirements and preferences of the entire population of end users.

As an element of the enterprise information infrastructure, the Desktop will potentially be deployed on hundreds, possibly thousands, of computers. In a production environment the result is a substantial configuration management burden. Whenever there is a new version of client-side code, every client must be re-configured. If the reconfiguration process should ever falter, the result is that there will be client computers in the field with incompatible combinations of application code. Some combinations may seem to work for some scenarios but mysteriously fail for other scenarios. Some combinations may not work at all. In any case, the potential disruption to production users is a nightmare. In the virtual enterprise, where resources are managed by entirely different support staffs, the problem is even more severe.

The SHIIP project has deployed two styles of Desktop and has attempted to document the architectural considerations that identify the scenarios in which one style is preferable to the other. The second style of Desktop has been called the lightweight Desktop. It is illustrated in Figure 7. One use case that was identified as part of the SHIIP project was delivery of work packages to the mechanics through a browser. The idea was that this approach would streamline the current process in which voluminous printed documents represented the only format for trade work packages. There were many complaints about the paper-based process:

- it was not possible to print only the page or two at a time that were needed;
- it was difficult to trace the cross-references throughout the many pages;
- the volume of paper led to inefficiencies;
- it was time-consuming to track down reference documents.

This use case presented requirements that were quite different from the Foreman Work Assignment application. The application was largely read-only; at most the user would need to fill in a form or two. Use of the system would be more intermittent and users would expect a quicker response time for these short-lived sessions. The process could be further streamlined if the client was a handheld or portable computer. For the Electronic Work Package task, the Lightweight Desktop was used. The Lightweight Desktop is based on the Web browser. Only HTML pages are delivered to the client. There is no requirement for the client to download or process sophisticated JavaBeans applets.

Nevertheless, it was possible to reuse significant portions of the Foreman Work Assignment objects for the Electronic Work Package application. As illustrated in Figure 7, the Electronic Work Package uses the EJB objects related to Work Orders. Work Orders are stored in an IMS database on a mainframe. Work Order data is accessed through an Oracle procedural gateway in order to populate the EJB business objects. These objects could be accessed directly by the Desktop client. However, in the Electronic Work Package application these business objects are accessed by a server-side Java program known as a servlet. The servlet is a client to the business objects, and it translates Work Order data into HTML forms, which are sent to the Lightweight Desktop. In the next development cycle, XML will be used in place of HTML. XML is
an Internet protocol that has the added feature that it can represent data that can be interpreted by computer applications.

![Diagram of Lightweight Desktop](image)

**Figure 7 - Lightweight Desktop**

**Distributed product visualization**

One of the most promising opportunities for increasing shipyard efficiency with improved information sharing is distributed product visualization. The SHIP project conducted numerous workshops with shipyard personnel in order to identify the shipyard’s visualization requirements. From these meetings it became clear that the shipyard’s visualization requirements were not exactly the same as the visualization requirements of the design and engineering processes. Yet most of the visualization research and development among shipbuilders has been driven by design and engineering organizations. The shipyard visualization requirements can be summarized as follows.

**Efficient display of large models**

In order to aid in the installation and outfitting processes, it is necessary to provide visualizations up to the size of a work area. A ship may be the most difficult of all industrial visualization challenges, being comprised of as many as ten million parts. The efficient display of large models entails two aspects: time to load the model and response time when viewing.

**Ability to extract a portion of the ship for viewing**

In today's shipbuilding design applications, the ship is represented as CAD models which are organized in accordance to the needs of the design work force. This organization cannot possibly suit the needs of all production and test processes. The
shipbuilding visualization system must be capable of representing the ship in a number of views for a number of different users. Some of these views include:

- planner’s view by schedule activity,
- operations’ view by work order by hull,
- operations’ view by work area,
- testing’s view by system by test section.

In order to support the diversity of required views, the visualization must be capable of providing an arbitrary visualization based on query that can designate items at the part level.

**Guarantee that the visualization accurately reflects the design configuration baseline**

Because of the duration of the ship’s design life cycle and the inevitability of changes to follow-on ships, there are often several versions of the design of the ship available at any time. Different versions are applicable to different hulls. In practice, the visualization service will likely be loosely coupled with the design databases. In any case, the system must guarantee that when the shipyard worker requests a visualization, it corresponds to the latest version in the design configuration baseline.

**Enable the user to navigate from the visualization to the appropriate design data**

The visual representation of the ship is very valuable information; however, there is other valuable information available in the design and engineering databases. The system must provide links so that the user can retrieve design and engineering data for each part in the visualization.

**Visualization Use Cases**

Three visualization use cases were identified by the SHIIP project, and solutions were prototyped in this development cycle. The scenarios are illustrated in Figure 8. The first, and most demanding use case, involves the production startup team meeting. The idea is that the team leader or foremen uses the visualization service to conduct a briefing with the shipyard workforce. In this use case, it is necessary to visualize an entire work area of the ship and to highlight work items that are scheduled for installation. It should be possible to use this meeting to schedule the sequence of imminent installation steps and to illustrate possible interferences. The design for this service depends on a dedicated visualization server that is capable of displaying thousands of pieces at a time. In addition, the SHIIP prototype employed a cache (library) of visualization data that had been extracted from the company’s CAD models. (In the next development cycle, the SHIIP project will focus on the synchronization of the visualization and CAD data.)
A second use case involves the situation in which a user needs to review the contents of a team meeting visualization session from a remote client PC. In the SHIP prototype this was accomplished by enhancing the visualization server with the capability of freezing each view of the session as a two-dimensional image. This image is sent to a client PC. Since the image format is GIF, only a browser is needed to display it. The image contains controls that enable the user to manipulate the view in three dimensions. In this way, the browser provides all the view navigation capabilities of a 3D viewer. This approach is especially promising for very large models because the size of the image that moves over the network is independent of the size or complexity of the model. All geometric processing takes place on the visualization server. An entire work area can be displayed on a laptop.

The third use case entails the visualization of a single work package by a mechanic responsible for performing the work. This use case differs from the other two in that the size of the typical visualization is substantially smaller than the work area visualization. In fact, for a work package visualization, today's technology is adequate for client-side viewing of the 3D model. In this scenario, the complete 3D visualization of the work package is downloaded to the client. The SHIP has prototyped the use of the Virtual Reality Modeling Language (VRML2.0) for work package visualization. There are some benefits that accrue from the use of VRML (as compared to the use of the 2D images). VRML viewers are based on Internet standards and are widely available. VRML 2.0 supports the interaction of the visualization session with other Java processes on the client. Finally, the client-side processing of model visualization contributes to the load-balancing of the enterprise's computing resources. If all geometric processing is performed on the visualization servers, these machines could be overburdened. If the
client can be used to perform the visualization processing for an entire category of visualizations, more server resources will be freed up. There is no additional imposition on client resources because the same clients would be used for the 2D GIF-style visualization. The SHIIP visualization systems have been architected so that the same underlying visualization data can be used in either scenario – making it much easier to switch from image to model visualization for the purpose of the balancing visualization load among resources.

**Conclusion**

At the end of its second year the SHIIP project has successfully demonstrated the use of several advanced technologies for the shipbuilding industry. The work of the SHIIP project has formed the basis of the Advanced Shipbuilding Enterprise’s (ASE) Strategic Investment Plan for Systems Technology. The ASE will be the primary vehicle for collaborative research and development for U.S. shipbuilding. As a result, the SHIIP project has provided a foundation on which systems technologies can be applied to improve the competitiveness and the quality of the U.S. shipbuilding industry into the twenty-first century.