Developed under NSRP ASE ISE-4

Technology Investment Agreement (TIA) 2005-381

Integrated Shipbuilding Environment Consortium (ISEC)

ISE-4 Final Report

Document Number
ISE-4-TEAM-0001

Version 1.0

Submitted by:

Dr. Burton Gischner

Electric Boat Corporation

Submitted on:

2006-07-25

Approved for public release; distribution is unlimited.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 ISE Project Goals

The Integrated Shipbuilding Environment Consortium (ISEC) was formed in 1999 as a consortium of several major shipyards and CAD vendors with the goal of developing an information interoperability capability for U.S. Naval shipbuilding. Detailed discussion of the information interoperability problem and the ISE solution can be found in Section 3 below.

ISEC has addressed these interoperability issues in a series of NSRP (National Shipbuilding Research Program) Projects known as the Integrated Shipbuilding Environment (ISE). The NSRP Advanced Shipbuilding Enterprise (ASE) was established in 1998 as collaboration of eleven shipyards with Navy and other federal agencies. It has attempted to solve interoperability and data exchange issues by awarding various projects over the past five years to this consortium of shipyards, CAD vendors, and universities. The projects awarded by NSRP in this area are known informally as: HARVEST, ISPE, ISE-1, ISE-2, ISE-3, and ISE-4, and they have successfully attacked various phases of the interoperability problem.

The HARVEST Project completed the STEP Shipbuilding Structural Application Protocols (AP215, AP216, and AP218) and secured their approval as International Standards (IS). ISPE determined the requirements for exchanging steel processing product model data in a shipyard independent format. ISE-1 outlined the requirements and architecture for an interoperability solution. ISE-2 implemented that solution in the structural and piping disciplines. ISE-3 continued these implementations in the HVAC and Common Parts Catalog (CPC) interface areas.

The ISE Consortium is continuing its efforts at developing tools for product data interoperability under the current NSRP ASE Project for “ISE Interoperability Modules” (ISE-4). This project will have been active from February 2005 through July 2006 and is extending the work of previous ISE Projects in four additional areas:

- Designing, developing, and demonstrating prototype exchanges of CAD and CAE data for information describing compartment geometry and properties while testing ISO product data model standards (Ship Arrangements Task)
- Enabling CAD-independent interchanges of steel fabrication work packages (Steel Processing Task)
- Enhancing and extending the results of the ISE-2 Project to exchange a richer and more complete product model using AP209 for Engineering Analysis data (Engineering Analysis Task)
- Developing the ability to use electrical standards to interoperate between ship design CAD models and SPAWAR datasets (Electrotechnical Task)

1.2 ISE-4 Tasks

The ISE-4 Project is divided into four tasks that operate fairly independently, although they did hold several joint meetings and put on a coordinated Final Project Demonstration.

The Ship Arrangements Task is led by Electric Boat Corporation with support from Atlantec Enterprise Solutions, Intergraph, NSWCCD, Northrop Grumman Ship Systems, Product Data Services, and SENER. It is described in detail in Section 4 of this report.

The Steel Processing Task is led by Northrop Grumman Information Technology with support from Atlantec Enterprise Solutions, Electric Boat, Intergraph, and Northrop Grumman Ship Systems. It is described in detail in Section 5 of this report.

The Engineering Analysis Task is led by Electric Boat Corporation with support from Intergraph and Northrop Grumman Ship Systems. It is described in detail in Section 6 of this report.

The Electrotechnical Task is led by Knowledge Systems Solutions, Inc. with support from Electric Boat, Intergraph, NSWCCD, and Northrop Grumman Ship Systems. It is described in detail in Section 7 of this report.

1.3 Participants

The participating organizations in the ISE-4 Project include:

- Atlantec Enterprise Solutions (AES)
- Electric Boat Corporation (EB)
- Gulf Coast Region Maritime Technology Center (GCRMTC)
- Intergraph Corporation
- Knowledge Systems Solutions, Inc. (KSS)
- Naval Surface Warfare Center - Carderock Division (NSWCCD)
- Northrop Grumman Ship Systems (NGSS)
- Northrop Grumman Information Technology (NGIT)
- Product Data Services Corporation (PDS)
- SENER

2 APPLICABLE DOCUMENTS, REFERENCE, AND GLOSSARY

2.1 Acronyms, Definitions, and Abbreviations

ADAPT: EB Software – Automated Design-to-Analysis Process and Transformation code

AES: Atlantec Enterprise Solutions

AIC: Application Interpreted Construct

AIM: Application Interpreted Model

AP: (Application Protocol); Documents that specify the format for representing product data within a set of related processes or activities.

API: Application Programming Interface

AP203: AP for Configuration Controlled Design

AP209: AP for Composite and Metallic Structural Analysis and Related Design

AP212: AP for Electrotechnical Design and Installation

AP215: AP for Ship Arrangement

AP216: AP for Ship Moulded Forms

AP218: AP for Ship Structure

AP227: AP for Plant Spatial Configuration

ARM: Application Reference Model

ASE: Advanced Shipbuilding Enterprise program of the NSRP

CAD: Computer Aided Design

CAE: Computer Aided Engineering

CAM: Computer Aided Manufacturing

CATIA: Dassault CAD system

CC: Conformance Class

CCS: Composite Constituent Shape

COTS: Commercial-Off-The-Shelf

CPC: Common Parts Catalog

DARPA: Defense Advanced Research Projects Agency

DIS: Draft International Standard

DoD: Department of Defense

DONXML: Department of Navy XML Repository created to provide guidance for use of Extensible Markup Language in a standardized way

EB: Electric Boat Corporation

ESTEP:	Evolution of STEP Task under ISE Project
EXPRESS:	A formal data specification language that specifies the product information to be represented
FEA:	Finite Element Analysis
FEM:	Finite Element Model
GCRMTC:	Gulf Coast Region Maritime Technology Center
GUI:	Graphical User Interface
GUID:	Globally Unambiguous Identifier
HM&E:	Hull, Mechanical, and Electrical
IDE:	Integrated Development Environment
IGES:	Initial Graphics Exchange Specification
IPDE:	Integrated Product Data Environment
IS:	International Standard
ISDP:	(Integrated Ship Design & Production); Intergraph's shipbuilding product suite
ISE:	Integrated Shipbuilding Environment Project
ISEC:	Integrated Shipbuilding Environment Consortium
ISO:	International Organization for Standardization
ISPE:	Integrated Steel Processing Environment Project
IT:	Information Technology
JSR94:	Java Specification Request (JSR) for rule engine APIs (JSR94) – http://jcp.org/aboutJava/communityprocess/review/jsr094
JVM:	Java Virtual Machine
KSS:	Knowledge Systems Solutions, Inc.
LEAPS:	Leading Edge Architecture for Prototyping Systems
MariSTEP:	DARPA MARITECH Program that implemented prototype STEP translators for ISO Shipbuilding APs.
NASSCO:	National Steel and Shipbuilding Company
NAVSEA:	Naval Sea Systems Command
NGSS:	Northrop Grumman Ship Systems
NPV:	Net Present Value
NSRP:	National Shipbuilding Research Program
NSWCCD:	Naval Surface Warfare Center Carderock Division
OSEB:	Object Serialization Early Binding
Part 21:	STEP Implementation Method - Clear Text Encoding of Exchange Structure
Part 28:	STEP Implementation Method - XML representation for EXPRESS-driven data

PATRAN: Analysis software program from MacNeil Schwindler Corp.

PDES: Product Data Exchange using STEP; A consortium of companies to accelerate the development and implementation of the STEP Standard

PDS: Product Data Services Corporation

PLM: Product Lifecycle Management

Postprocessor: A software unit that translates product information from an independent public domain product data format to the internal format of a particular computer system.

Preprocessor: A software unit that translates product information from the internal format of a particular computer system to an independent public domain product data format.

ROI: Return on Investment

SBIR: Small Business Innovation Research – Research and Developments Projects, funded by the Department of Defense (DoD) for small technology companies.

SDE: Shared Data Environment

SEDS STEP Enhancement and Discrepancy Reports

SOAP: Simple Object Access Protocol. A communication protocol used to establish communication between applications.

SPAWAR: Space and Naval Warfare Systems Center, San Diego

STEP: (STandard for the Exchange of Product Model Data); It is the familiar name given for the international standard ISO 10303 Industrial Automation Systems and Integration - Product Model Representation and Exchange. The objective is to provide a mechanism that is capable of describing product model data throughout the life cycle of a product. The standard is a collection of parts, each published separately.

STI: STEP Tools, Inc.

TC: Technical Corrigendum to an International Standard (IS)

TIA: Technology Investment Agreement

TIFF: Tagged Image File Format

TRIBON: AVEVA TRIBON Solutions CAD / CAM system

TWR: A 120 foot Torpedo Weapons Retriever, built in 1985, which is being used as the source of ISE test data

UoF: Unit of Functionality

WWW: World Wide Web

W3C: World Wide Web Consortium

XML: Extensible Markup Language

XSLT: Extensible Style sheet Language Transformation

2.2 Referenced Documents

1. ISO 10303-215 – “Application Protocol – Ship Arrangement”, International Standard, ISO TC184/SC4/WG3 N1226, 2004

2. ISO 10303-216 – “Application Protocol – Ship Moulded Forms”, International Standard, ISO TC184/SC4/WG3 N1133, 2003

3. ISO 10303-218 – “Application Protocol – Ship Structures”, International Standard, ISO TC184/SC4/WG3 N1350, 2004

4. ISO 10303-227 – “Application Protocol – Plant Spatial Configuration”, International Standard, ISO TC184/SC4/WG3 N1476, 2005

5. Gischner, Burton, "Ship Arrangements Preliminary Implementation Agreements", Document Number ISE-4-ARRANGE-0002, 2005-07-29

6. Gischner, Burton, "Ship Arrangements Test Plan and Test Cases", Document Number ISE-4-ARRANGE-0001, 2005-05-27

7. ISO 10303-46 - "Integrated Generic Resources: Visual Presentation", International Standard, 1994

8. B. Kassel, 120' TWR Product Model Data Repository, http://rabecs.dt.navy.mil/TWR/ProdModRep01.htm , NSWCCD, West Bethesda MD

9. NSRP Strategic Investment Plan, December 9, 2002 Revision 3, http://www.nsrp.org/strat_plan/sip_2002.pdf

10. ISPE Final Report, Northrop Grumman Ship Systems, January 27, 2003, http://www.nsrp.org/projects/ispe_deliverables.html

11. T. Briggs, T. Rando, K. Richard, ISE Steel Processing Requirements, May 31, 2005 Revision 1.0 http://www.isetools.org/eb-cgi-bin/inet/yabb_ISE/YaBB.pl?board=sp_deliverable_reports

12. Ernest Friedman-Hill, Jess in Action, Connecticut: Manning, 2003

13. ISO 10303-209 – “Application Protocol – Composite and Metallic Structural Analysis and Design”, International Standard, 2001

14. Gordon, S., Editor, "ISE-4 Engineering Analysis Test Plan and Test Cases", Document Number ISE-4-ANALYSIS-0001, 2005-07-31

15. ISO/DIS 10303-28 – “Part 28: Implementation methods: XML representations of EXPRESS schemas and data, using XML schemas”, Draft International Standard, ISO TC184/SC4/WG11 N258, 2006

16. “Software Architectures for Ship Product Data Integration and Exchange”, 12-07-1999, T2.5/D251, M. Grau, SEASPRITE consortium.

17. “AP212 Ship Electrical Usage Guide with SCM Addition”, 8-18-2000, Version 2.0, Electronic Commerce Center, Concurrent Technologies Corporation.

18. ISO 10303-212 Edition 1 – “Application Protocol – Electrotechnical Design and Installation “; Draft International Standard, ISO TC184/SC4/WG3 N1144; March 1, 2001

19. ISO 10303-21 - "Implementation Methods: Clear Text Encoding of Exchange Structure”; International Standard, 1994

20. Electrotechnical Design and Installation AP212 Schema Analysis; October 26, 2005

21. Wolsey, K., Editor, "ISE-4 ROI Task – Analysis and Recommendations", Document Number ISE-4-TEAM-0002, 2006-07-21

22. “Improving Interoperability through use of ISO 10303” by R. Wood, T. Rando, B Gischner, J. Mays, T. Briggs presented at12th International Conference on Computer Applications in Shipbuilding in Busan, Korea, August 2005

23. “Enhancing Interoperability Throughout the Design & Manufacturing Process” by B. Gischner, P. Lazo , K. Richard, and R. Wood presented at the Ship Production Symposium in Houston, Texas, October 2005

3 OVERVIEW – ISE STRATEGIC PLAN

The ISE-4 Project must be viewed in the context of an overall plan to develop an information interoperability capability for U.S. Naval shipbuilding. The roadmap for that plan is depicted in Figure 4 which includes boxes for each of the four tasks included in ISE-4. Following that roadmap will lead to successful completion of the ISE Strategic Plan as described below.

3.1 Computer-Based Ship Design and Engineering

In the 1990's the U.S. Navy and its shipbuilders began the migration from a design process that captured ship design information on 2D drawings to a process in which ship design information is formulated electronically (with geometry represented in 3D) and captured as a digital product model. The family of systems required to create and maintain a digital product model has come to be known as the Integrated Product Development Environment (IPDE). The process of developing and deploying an IPDE capable of dealing with the intricacies and scope of a modern U.S. Navy warship design entails substantial costs.

As new ship design and construction programs were authorized for complex ships, the use of an IPDE to create the design became imperative. The shipbuilder design agent then invested in an IPDE. The costs of this investment were borne by the U.S. Navy customer as part of the price of the ship design. Of course, the creation of hardware, software, training, and integration into legacy systems of advanced computer-based design capabilities is a business – with several vendors offering products to serve the IPDE market. Not surprisingly, this new technology (at this stage of its development) tends to be vendor unique and not interoperable. Furthermore, the technology advances rapidly so that subsequent products of the same vendor can be essentially incompatible with the vendor's own legacy offerings. In short, the situation is that the U.S. Navy has paid for several major versions of IPDE, installed by different shipbuilders design agents over the span of the last fifteen years. The Navy is understandably frustrated. It observes what seems to be redundant investment in IPDE, and it tries to determine how it will manage the maintenance and logistic support of the various ship classes through their decades long futures, serving in the fleet. Maintenance and logistic support require continuous ready access the ship class design and engineering information, information which is more easily used if it is available through an IPDE. Absent the needed standards, information created in an IPDE is not easily accessed except through the IPDE in which it was created.

- **The cost and schedule for the deployment of an IPDE adversely affects the cost and schedule for the design of Navy ships – with implications for the cost and schedule of the lead ship construction.**

Today, approximately one-tenth of the design costs for a U.S. Naval shipbuilding program are consumed by the deployment and support of its IPDE system. In some respects, this is not an unreasonable burden. However if it were much higher, the cost/benefit rationale would disappear. For any given design program, it is still a substantial cost encumbrance for the acquisition process. But, more importantly, because IPDEs have been developed as part of the design of each ship program, the Navy has been paying for versions of the same basic capability over and over in each new program. Moreover, IPDE development and deployment do not begin until the design contract is awarded. This approach, which is time-consuming and inopportune, inevitably impacts the design and construction schedule, resulting in wasteful costs that, while difficult to account for, are also substantial.

- **Two forces, competition (both among shipbuilder design agents and among IPDE providers) and the ceaseless advance of technology, have converged to create the shipbuilding IPDE dilemma.**

The Navy deals with shipbuilders that compete for design and construction work. The competition is motivated by a desire to improve quality and reduce cost and schedule. The shipbuilders seek advantage by selecting an IPDE from competing IPDE vendors. Imposing a mandated IPDE system on the shipbuilders would frustrate that competition, even supposing it were realistic to imagine that it would be possible to align the various software systems, versions and integrations at competing shipyards. Additionally, imposing a specific IPDE would put the Navy into a very difficult position with respect to the responsibility for performance on the design contract. Simultaneously, the enabling computer systems technologies have been evolving apace. No sooner is an IPDE system designed and validated than new and improved capabilities become available. So, each IPDE is a custom system, suitable for one program and for one shipbuilder design agent. Each program, unavoidably, builds its own IPDE from scratch.

- **A root cause obstructing a solution to this dilemma is that there is no consensus on the fundamental question of what constitutes a digital product model.**

The first generation IPDE systems were viewed by most as a means to facilitate the publication of 2D drawings. Eventually it became clear that the digital product model was itself the prime work product, enabling multiform efficiencies associated with the re-use of its data, and that it could, in fact, constitute the design deliverable. Unfortunately, there has not yet emerged an industry consensus on the fundamental question of what constitutes a digital design deliverable. This should be a key element of the requirements set forth in the ship's specifications. Without such a foundation, the selection and subsequent deployment of the IPDE system becomes an overbearing factor in the acquisition process. Moreover, deployment costs are exacerbated rather than decreased as each new program debates the question of what the digital design deliverable should be. Typically, this debate starts anew with each new program -- with participants who are at a disadvantage with respect to lessons learned from prior deployments.

This complex problem is further exacerbated by the collaborative nature of current naval shipbuilding programs. Because each IPDE system is built independently for each new program, the collaboration infrastructure needed for two or more shipbuilder design agents to work together in a program has to be built from scratch. The collaboration infrastructure itself has a price tag in the millions of dollars. It is even less repeatable than the IPDE it services because its links touch the IPDE and its surrounding proprietary support systems.

- **Complete the development of the ISE information interoperability infrastructure to serve as an enabler to a flexible, open, next-generation IPDE capability.**

The development of the next-generation IPDE capability, which has already begun within the Virginia program, is a vast undertaking, dependent upon many process and technology enablers. Information interoperability, the sharing of information across system, application and organization boundaries, is one such enabler. . The ability to archive the data in a neutral format to provide a migration path to the next-generation of systems is part of the issue. The following describes the costs, benefits and technical approach for completing this enabler.

3.2 Current and Future IPDE Situation

Each major U.S. Navy shipbuilding program in recent years has built its own IPDE from scratch. This is true even among different business units of the same corporation. Collaboration team members typically deploy a lesser, satellite instance of the lead yard's IPDE or rig up connectivity to the primary IPDE. As a consequence, the team member is compelled to support and maintain one or more ancillary IPDE environments. The landscape is illustrated in Figure 1.

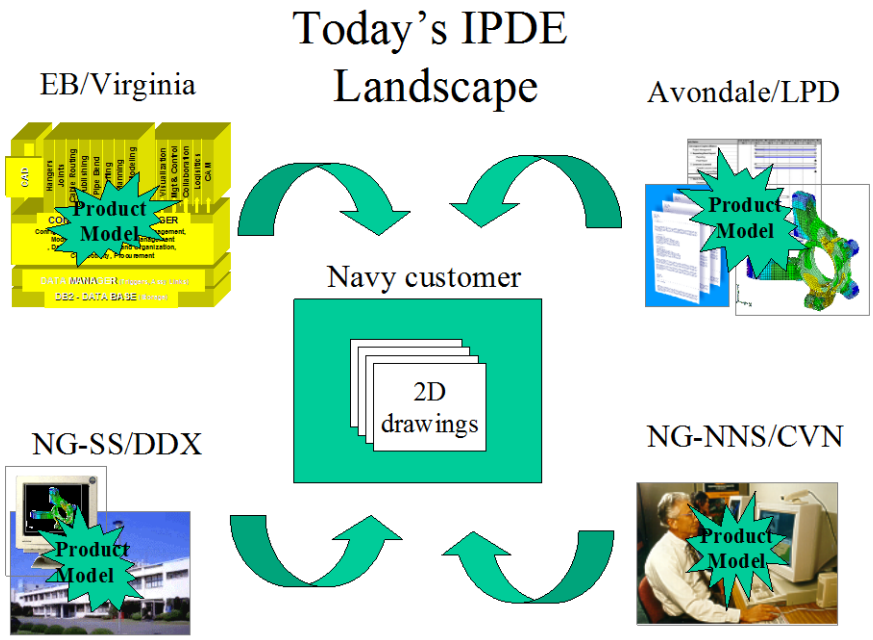


Figure 1: Today's IPDE Landscape

Each major program boasts a different IPDE implementation. The most widely publicized difference is the selected CAD platform, but, in fact, the more crucial difference is in the surrounding shipyard IT systems, the application systems, the data management systems and the plexus of integration threads that hold the system together. At the core of each IPDE is the digital product model which is authored, managed, used internally for analysis and manufacturing, and ultimately transformed into the design deliverable. However, the digital product models are different for each program. One reason for the discrepancy is that the constituent elements of the product model deliverable have not been unambiguously specified at a fundamental level. This deficiency impedes the prospects for collaboration of product data as well as for the collaboration of IPDE system components (i.e. plug and play).
Before discussing the development of the next generation IPDE some background is in order. An IPDE system entails the interconnections of software systems, business processes and human resources. Different shipyards use different names for the entire system or for major components of the system. Some examples include Integrated Development Environment (IDE) and Shared Data Environment (SDE). The acronym IPDE has been applied to the Virginia system and processes.

The relationship between a PLM software package and an IPDE system can be confusing. PLM is a software commodity. It is an implement that provides the backbone for an IPDE capability. The IPDE itself, of course, entails much more, including the functional applications, the business processes, and the interoperability of product information. The goal of the Navy's current PLM (Product Lifecycle

Management) development activity is to deploy a next generation IPDE capability (built upon a COTS PLM foundation) that is multi-program, multi-application, configuration managed, standards-based and open. In other words, the objective is to deploy a system that is integrated and deployed in such a way that its component elements can be re-used for different programs and within different computing environments. Of course, this does not mean the re-deployment is without cost; costs would include license costs as well as installation and maintenance costs, costs that are well-understood and not unexpected. The costs that are avoided are the costs to build the underlying IPDE capability.

A secondary benefit of the next generation IPDE system is that it facilitates collaboration across different IPDEs. As implied in Figure 1, the current IPDEs do not communicate with each other. Today collaboration is accomplished by standing up a local, scaled-down instance of the primary IPDE at the collaborating site. As a result of honoring the information interoperability ground-rules (being defined by ISE), the future IPDE has the opportunity to import product data as if it were created natively in its own systems. This capability is enabled by the work proposed for NSRP/ISE but is only brought to fruition after a program deploys the requisite supporting tools. The specifics of these prerequisites are described in the next section.

3.3 Role of Information Interoperability in Next Generation PLM-Based IPDE

There are a number of process-related and technological prerequisites to the next generation IPDE capability. One of those prerequisites is information interoperability which is the sharing of information across system, application and organization boundaries. Information interoperability is a key enabler of next-generation IPDE; it is a necessary but not a sufficient condition of its production deployment. Surprisingly, the primary focus of information interoperability is not on data exchange for collaboration. The vast majority of the information exchanges take place (thousands of times a day) among the application systems within a single IPDE.

The next step toward the NAVSEA vision of ONE SHIPYARD via the next generation PLM-based IPDE work, should be the establishment of a consensus regarding what constitutes the digital design deliverable. Today design deliverables are conveyed via drawings (which are human but not computer interpretable). These paper-based deliverables are specified to the minutiae (down to the shape and color of arrowheads and leader lines on schematics). No such consensus exists for the digital product model (even though it drives the entire acquisition process). We need an implementable and enforceable yet open specification of the digital product model for naval shipbuilding.

It should be emphasized again that information interoperability is the vehicle by which applications interact within the IPDE system itself and, consequently, to other applications within the IPDE enterprise. Only secondarily is it the vehicle for data interchange between companies (different IPDE's)

Finally, the ISE architecture (and prototypes) have defined how to create the formal documentation of the requirements for digital product model deliverable. This is an exacting and demanding process, but it should be the cornerstone of any Navy policy that seeks to sustain the ONE SHIPYARD vision. The goal of this activity is to specify the digital product model for shipbuilding, to a level of detail and aligned with current Web capabilities, that enables the production-worthy sharing of the product model between and within various IPDE deployments.

This is illustrated in Figure 2.

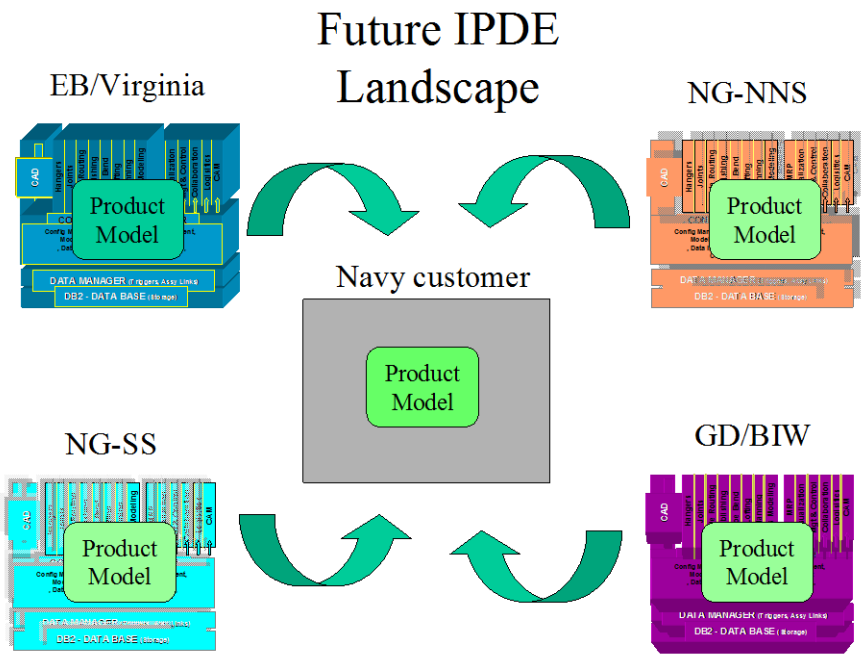


Figure 2: Future PDE Landscape

The centerpiece of the approach is that there is a common understanding of what constitutes the design deliverable/digital product model. Variations in representation are permitted, but the basic definition (the information elements) is unambiguously specified. Each shipyard, of course, deploys its own (PLM-based) IPDE instance; however, all of them respect the specified product model. There is no requirement or guarantee that every shipyard deploys the selfsame PLM system and supporting applications, though as a result of the next-generation IPDE capability, a large portion of that capability will be available in COTS product upon successful completion of the project.

3.4 Overview of the Information Interoperability Lifecycle for Each Application Domain

The shipbuilding problem universe is broad, but it is comprised of a finite number of well-understood application domains. The information interoperability lifecycle is virtually the same for each application domain. This fact can be used to help estimate costs per application domain (which can also be tailored if needed).

Figure 3 depicts the stages and responsibilities of the information interoperability life cycle. It should be noted that there is a role that is well suited to NSRP, but that role is not comprehensive and depends upon support from other stakeholders. The roles are illustrated as the swim lanes in Figure 3. The first role belongs to the international standards organizations (such as ISO and, at a more technological level, the World Wide Web Consortium -- W3C). Work to define the digital ship product model has been underway since the 1980's and is now substantially complete. This work consists of the definition of the universe of information elements needed to represent a ship across the stages of the product development life cycle.

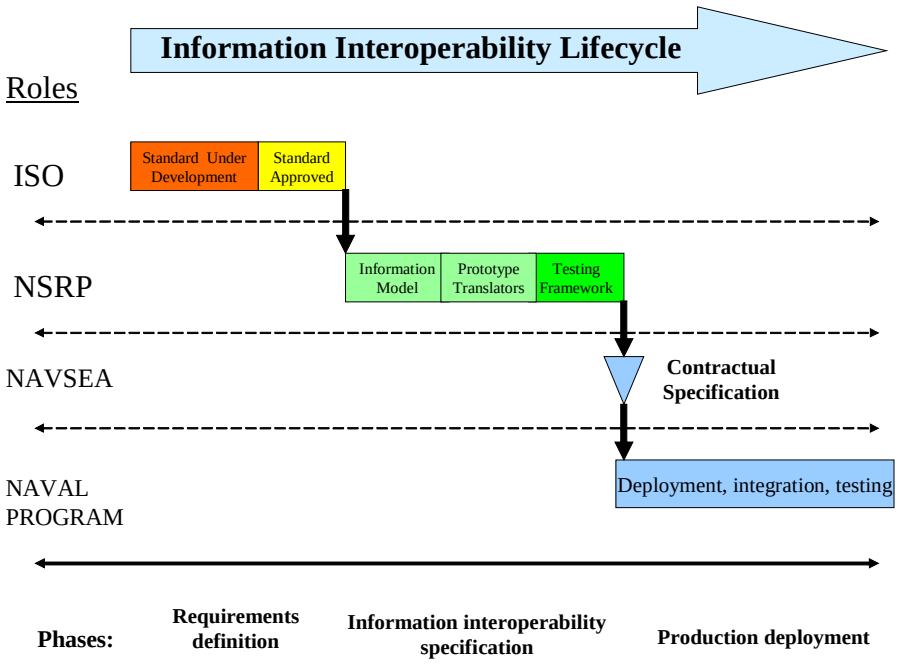


Figure 3: Information Interoperability Lifecycle

The second role belongs to NSRP; it entails the production level refinement of the standards-based information requirements into a form suitable to provide information interoperability. This work has an element of R&D and is independent of any single Navy program. The ISE Project has developed a very detailed technical approach. It is important to understand that broad directives to use STEP, XML or any IT standard, while well-intentioned, are NOT SUFFICIENT to provide information interoperability.

A detailed description of the ISE process is available on the ISE website. The following summarizes the technical approach. The international standards published by ISO/STEP initiate the move toward implementation level consensus, but they are not sufficient; the ISE process elaborates the STEP standards, making them suitable for production use by simplifying, bounding and testing the standard. The process steps (illustrated in green in Figure 3) have been proven, prototyped and demonstrated by the ISE Project in selected application domains.

1) Information model: this step consists of the definition of a use case specific information model, that is, an unambiguous delineation of the information elements needed for each significant ship design/build/support usage scenario. Please note an important technical detail. The information must employ end-user friendly information elements, not the impenetrable STEP-specific jargon. The bulk of the effort at this stage consists of simplifying and scoping the information requirements. This entails correcting deficiencies and eliminating ambiguities that still exist in the broad standards. Once the content is resolved a consensus must be developed on format issues for various representations. As one example, the Navy and industry are moving toward standard name spaces that can be utilized by XML implementations to facilitate data interoperability and the resulting schemas are being placed in the DONXML Repository.

2) Development of prototype translators: This pertains to CAD as well as other applications. The primary purpose for developing prototype translators is the need to ensure completeness of each interaction in a way that cannot be ensured by a mere paper review. This stage generally results in the

realization of some commercially available translators but does not guarantee a production capability or deployment. Since this stage is fundamentally part of the testing process, it need not cover the whole population of available CAD platforms.

3) Test framework: The ISE work has revealed that two stages of testing are needed. The first stage consists of automated testing using ISE tools; the second consists of end user testing with actual data files. The second stage is now done catch as catch can; in the future it should become part of the NSRP ISE SOW.

Upon completion of the NSRP/ISE work there are still important steps that remain before production deployment is accomplished. First, the specifications (the information requirements) must be mandated by the Navy customer. This is the only way the technology will be deployed. This does not differ materially from prior practice whereby the Navy specifies the content and meaning on a drawing. It is imperative that specifications be detailed enough to be implementable. Statements such as "STEP shall be used to represent product data" are so vague that they are meaningless and unenforceable. Today there is still no consensus on basic issues such as what comprises the digital model of a piping system. ISE has found the right level of specificity and flexibility. An unambiguous specification eliminates many of the problems associated with IPDE startup. For example, without a clear definition of information requirements it is easy for IT technologists to obfuscate the completeness of their tools. The published specification also obviates, or should obviate, the situation in which a vendor can sell an application (for an IPDE) which does not even create a complete design model (just the graphics that look like a model). Moreover, without a consensus specification, it is virtually impossible to aggregate a complete product model for parts that originate in different applications. In the end, all this does not guarantee that all vendors will support, but it will make it possible and fairly easy to determine which ones do.

This stage should not be thought of as a data exchange exercise, the definition of the information models represents the definition of the fundamental information requirements (and, consequently, the required modeling capabilities) of each application system in the IPDE.

The final stage is deployment: The formal specification of the digital product model deliverable as part of the acquisition cycle is the penultimate step. Deployment remains. At this point there is still no deployed system. Each program must be motivated to execute the deployment guided by the specifications, which define the information requirements for each application in the (PLM-based) IPDE. It is important to note that the real life deployment of future IPDE's will still involve multiple, competitive vendors' products. Some assembly will be required at each shipyard, especially with respect to integration. The next generation Virginia IPDE will provide a tested, COTS-based, open reference implementation. It will be readily deployable at other shipyards (obviously, that shipyard would have to purchase the system). Its deployment would, of course, entail installation, integration, and configuration costs, but not new IPDE software development costs. On the other hand, other vendors (PLM vendors, infrastructure and/or application vendors) will undoubtedly seek to participate. Their path is clear. If they can support the creation and maintenance of the specific digital deliverables, they can have a role. The conformance testing tools from the ISE Project can be used to grade the level of each vendor's conformance. Finally, the employment of a commonly defined digital deliverable simplifies the process of inter-organization information sharing; the last piece of the puzzle is the ability to import/export (portions of) the product model from the IPDE. This capability is implied by virtue of the fact that the customer expects its final deliverable in the prescribed format; mediating between the representations of different IPDEs is a capability that is, then, supported by ISE tools.

3.5 Roadmap

Figure 4 is the Information Interoperability Roadmap; it shows the current status, what's done and what's left. Each box correlates to one application domain. The current progress of information interoperability development is indicated by the color code (relative to the legend on Figure 4). Upon completion of this roadmap (when all the boxes are blue), a production capability will be in place by which IPDE information requirements are specified; at least one PLM-based IPDE reference implementation will be in place; the capability will exist to share information between components of that IPDE system; it will be possible to roll out and configure the IPDE implementation (or alternative implementations) at other shipyards; it will be possible to test the completeness of the modeling capability of a proposed IPDE deployment.

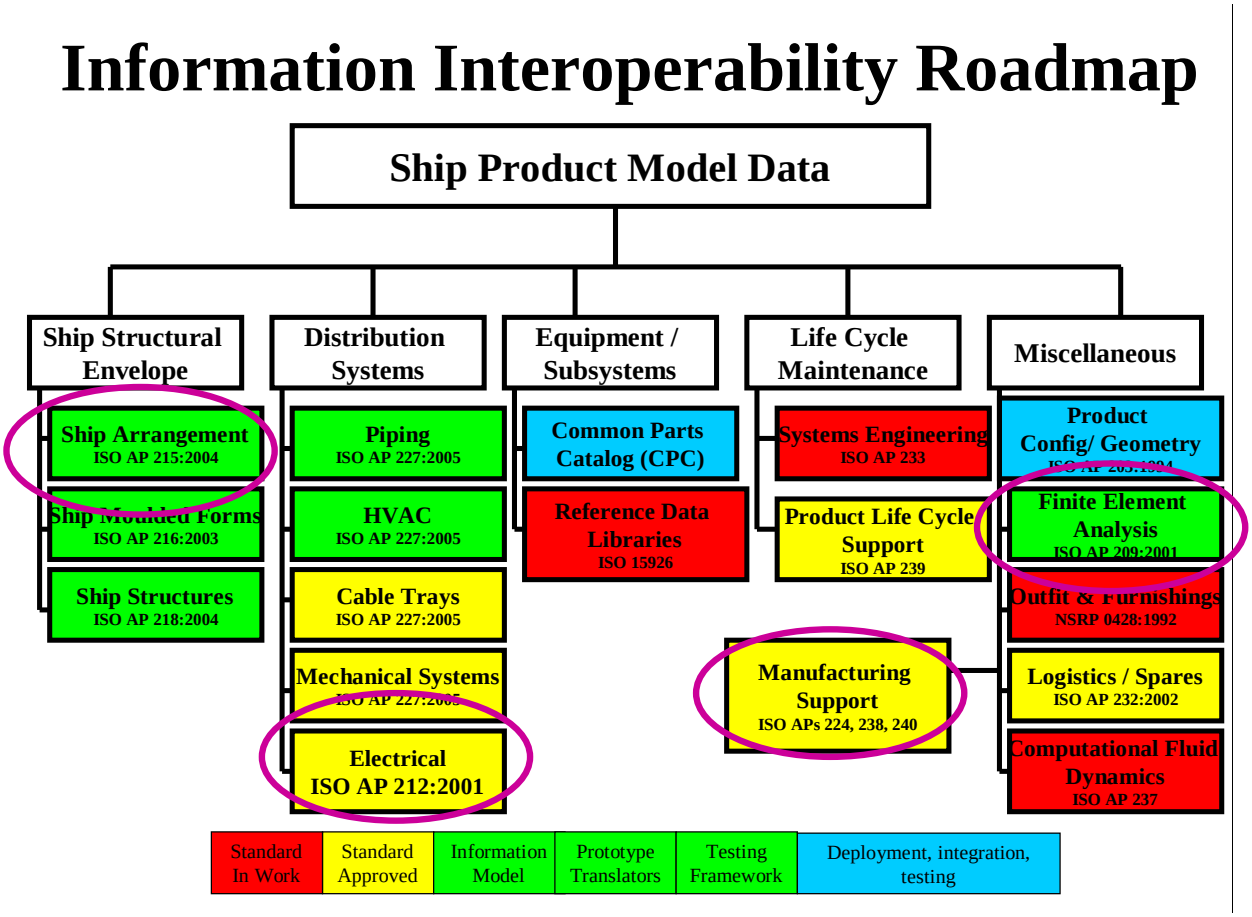


Figure 4: Information Interoperability Roadmap

The boxes within the purple ovals represent the disciplines addressed by the ISE-4 Project. Ship Arrangements and Finite Element Analysis were prototyped and tested under the ISE-4 Project so they are colored green for: "Prototype Translators Complete". As with other APs, final production testing remains and needs to be carried out under the framework established by the NSRP Systems Technology Panel. Prototyping efforts on Electrical and Steel Processing are continuing throughout 2006, so those boxes remain yellow to indicate that the translators are not yet completed.

4 Ship Arrangements Task

4.1 Task Description

The ISE Ship Arrangements Task developed translation software between several major CAD systems in use within U.S. shipyards and the Navy to support the product data model defined by the ISO 10303-215: Ship Arrangements Application Protocol (Reference 1). This portion of the STEP standard, known as AP215, specifies the international product model data standard for ship compartmentation, including geometric, functional, and property data, loading conditions for design analysis and operations, tonnage measurement, and damaged stability analysis. The data sharing capabilities implemented by this project were demonstrated in April 2006 with the exchange of compartmentation data between several shipyard and Navy CAD systems.

During preceding phases of the ISE program, the shipyard, Navy and CAD technology team has developed STEP-based data sharing capabilities based upon ISO 10303-216: Ship Moulded Forms (Reference 2), ISO 10303-218: Ship Structure (Reference 3), and the HVAC portions of ISO 10303-227: Plant Spatial Configuration (Reference 4). The translator software developed utilize a combination of the two primary methods of exchange specified by STEP, namely ASCII text files (known as STEP Part 21), and XML (known as STEP Part 28). The Ship Arrangements standard and the associated translators developed under ISE-4 built upon this previous work in both the use of the software architecture developed previously by ISE, and in the use of the previously developed translation capabilities for sharing the ship's hull and structural bulkhead moulded form surface definition data.

Within the STEP series of standards, ship structural envelope data is partitioned into three application protocols. STEP AP216 defines the zero-thickness moulded surface definition for the ship's hull and major structural bulkheads. STEP AP218 defines the ship's plate and profile parts and systems based upon these AP216 surfaces. STEP AP215 also refers to these AP216 surfaces for the definition of internal compartmentation and adds additional product data definitions for the functions of various types of compartments and tanks and for many non-geometric properties of the compartments and groups of compartments (zones). This data is created at various phases of the ship construction process and is shared throughout the ship's lifecycle to support initial concept through detail design, manufacturing, and post-delivery operations. Figure 5 illustrates the types of CAD and analysis data defined within AP215.

In addition to the compartment’s geometric shape, which may be defined either through explicit exchange of the compartment's geometric boundaries or by references to surface data from an AP216 exchange file, AP215 supports sharing of design analysis data for damage stability, loading conditions, tonnage, and weights. Zones may be identified for such purposes as fire control, positive pressure areas for chemical and biological defense, or for division of the ship for construction planning and configuration management.

Product data definitions are included in the AP to identify the operational functions for which a compartment has been designed, and extensive required, as-designed, and as-built properties are included to define such information as compartment volumes and centers of gravity, arrangeable areas, naming and identification, manning levels, temperature ranges for HVAC design, illumination levels for lighting design, and security classifications. Tank definitions include such data as volumes, areas, and moments of inertia, those associated piping system design parameters that are dependent upon the tank geometry, coating specifications, and the definition of operational fluids and fuels, or bulk, liquid, or gaseous

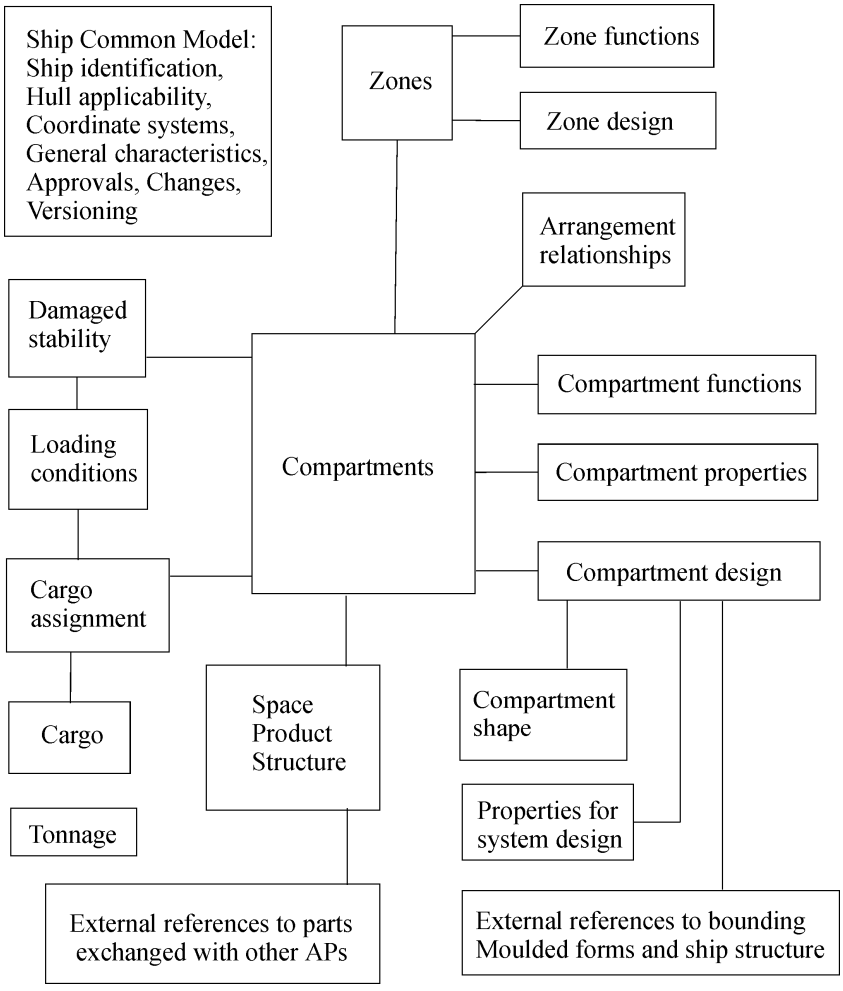


Figure 5: ISO 10303-215 Ship Arrangements Data Planning Model

cargos that will be contained in the tanks or cargo spaces. Two types of product structuring capability are included in the AP. One communicates the arrangement of multiple compartments upon a particular deck; the other identifies the engineering parts such as machinery, piping, HVAC, and structural items that are contained with a single space.

Each of the STEP APs contains product data definitions that may be shared for various purposes through the entire design and operational lifecycle of a ship program. This data may be shared between various types of organizations such as the design agent, shipyards, classification societies and regulatory bodies, and the ship's owner/operator. These varied organizations utilize assorted CAD and text-based software systems for particular tasks. The ISE team has found it particularly useful to further analyze the numerous expected uses of the data supported in the STEP APs to identify specific finite exchange scenarios at various points in the lifecycle to more clearly identify which of the data is required to support successful sharing at various points in a design teaming environment. Under previous ISE phases this has been accomplished through development of detailed data process flows, exchange scenarios, and use cases that build upon the application activity models published within the standards. The ISE Ship Arrangements team has produced similar process flow analysis, exchange scenario, and use

case information for the arrangements product data model, culminating in the documentation and XML definition of the required STEP data objects needed for each particular exchange use case.

4.2 Scope and Participants

ISE-4 – Ship Arrangements is a task of the NSRP ASE sponsored, industry led project titled "ISE Interoperability Modules." This program is a collaborative effort that includes several U.S. shipyards, software vendors, and research institutions. The term "shipyard" is used to mean not only the commercial shipyards, but also the "virtual" shipyard represented by the U.S. Navy and the NAVSEA CAD-2 program.

The ISE-4 –Ship Arrangements team is led by Electric Boat Corporation (EB) with members from the following organizations: Atlantec Enterprise Solutions (AES), Intergraph Corporation, Naval Surface Warfare Center - Carderock Division (NSWCCD), Northrop Grumman Ship Systems (NGSS), Product Data Services (PDS), and SENER.

4.2.1 System Overview

Under the ISE-4 – Ship Arrangements Task, much of the ESTEP team (which was active during ISE-2 and ISE-3) was able under NSRP ASE to continue its efforts to create product model standards for the shipbuilding industry and to implement translators supporting the exchange of this data. The team members listed above include several shipyards, and CAD vendors supporting these and other U.S. shipyards. Table 1 below indicates the relationships between the CAD Systems, vendors developing translators for those systems, and the U.S. shipyards using those systems.

Table 1 Translator Development Participants

Shipyard	CAD Vendor	CAD Environment
Kvaerner Philadelphia	SENER	FORAN
NASSCO	Atlantec Enterprise Solutions	TRIBON
NSWCCD	Product Data Services	LEAPS
NAVSEA & NGSS (Avondale Operations)	Intergraph	ISDP & Intelliship

4.3 Translator Implementations

4.3.1 Strategies Used

A major effort of the ISE-4 – Ship Arrangements Task members was to develop a common understanding of and a common way to apply the Ship Arrangments Application Protocol (AP215) to enable the transfer of arrangements and compartmentation product model data among various Shipyard / CAD system environments, and to document this understanding as a series of implementation agreements (see Reference 5). Product data relating to ship structure or ship moulded forms is also required in a complete arrangements exchange, and is transferred using APs 216 (Ship Moulded Forms) or 218 (Ship Structures). Thus, implementation agreements relating to the use of those Application Protocols have also been included in Reference 5.

The ISE-4 – Ship Arrangements Project implemented Conformance Class 3 of AP215. As problems were discovered with the AP structures, they were recorded and documented to be resolved in future

Editions, Amendments, or Technical Corrigenda (TCs). Several ISE-4 team members participate actively in ISO TC184/SC4 (the organization responsible for maintaining and updating the STEP Standards), and are thus well positioned to see that these required modifications are incorporated into the AP.

4.3.1.1 Part 21 vs. Part 28

Traditionally when STEP is used for product data exchange, the translators produce a neutral data representation known as a STEP physical file, or a Part 21 file.

Some of the ISE-4 –Ship Arrangements preprocessor translators are producing files in this format. However, with the growing use of the internet and the World Wide Web, a new STEP format has been developed, called Part 28, which utilizes XML structures to encapsulate the STEP data.

Processors have been developed during the ISE (Integrated Shipbuilding Environment) Project to convert the Part 21 files into XML-based Part 28 files to enable interactive exchanges of Ship Arrangements data using the Web. Mediator programs have also been developed to go in the other direction, converting XML-based Part 28 files into the more traditional Part 21 format. Thus some of the ISE-4 – Ship Arrangements translators produce files in Part 21 format while others create Part 28 files, but the publicly available tools developed under the ISE Project enable conversion of these files between Part 28 and Part 21.

The Implementation Agreements apply to both formats to enable seamless exchanges among ISE-4 participants, no matter which neutral format they have selected.

4.3.1.2 Preprocessors vs. Postprocessors

A preprocessor reads the model in the native CAD system and generates a STEP neutral file. The preprocessors developed for ISE-4 – Ship Arrangements were validated by careful analysis of the Part 21 or Part 28 file created to see that the required entities and attributes are properly captured in this file. A detailed checklist was created and checked to verify the accuracy of the STEP files created for ISE-4 – Ship Arrangements.

A postprocessor reads the neutral STEP file and converts it to a model on the receiving CAD system. The postprocessor is validated by checking the attributes and functionality of the product model on the receiving system.

4.3.1.3 Test Case Rationale

4.3.1.3.1 Simple to Complex

For Ship Arrangements, eleven test cases of multiple levels of complexity were selected. The test cases ranged from simple to complex and were designed to simulate the exchange of many types of properties relating to various zones and compartments on a ship.

The test cases are described in more detail in a report issued by the ISE-4 Project on “Ship Arrangements Test Plan and Test Cases” (Reference 6).

They are all developed from real compartments on an actual U.S. Navy ship – the TWR 841.

4.3.1.3.2 Conformance Classes

There are five Conformance Classes in AP215 describing information required in different phases of the life cycle. The ISE-4 – Ship Arrangements translators have implemented Conformance Class 3 (CC3) to support exchanges for Detailed Design data.

Postprocessors are required to successfully handle a valid AP215 CC3 file, even if it contains some attributes not supported by the receiving CAD system.

Preprocessors must produce a valid AP215 CC3 file, even if the sending CAD system does not support all the attributes required.

4.3.1.3.3 Life Cycle Stages

The ISE-4 – Ship Arrangements AP215 exchanges focused on data needed to support the Detailed Design phase of the ship's life cycle.

4.3.1.3.4 Unsupported Attributes

Attributes that are required for Ship Arrangements exchange, but not generally supported in a given CAD system, must be handled by the AP215 postprocessors developed for ISE-4. The generally approved solution is to add this attribute to the product model on the receiving CAD system, to ensure its availability for future exchanges.

4.3.1.3.5 Optional Attributes

In AP215, attributes may be defined as "Optional" or "Required", giving the sending CAD system the ability to decide which ones to support.

However, ISE-4 – Ship Arrangements Task determined a minimal set of attributes that are required for any successful exchange. The presence and handling of these attributes was verified during the ISE-4 testing.

4.3.2 Implementation Agreements

4.3.2.1 Reasons for Implementation Agreements

Agreements were required among the ISE-4 – Ship Arrangements team members with respect to their implementations of translators for the Ship Arrangements Application Protocol (AP215).

The ISE-4 – Ship Arrangements Task represents one of the first implementations of a new Application Protocol (AP) for the transfer of arrangements and compartmentation product model data. These agreements enabled the vendors to implement the relevant portions of the standard in an unambiguous manner that can be interpreted by translators for other CAD systems. These agreements and lessons learned should eventually be incorporated into a Usage Guide for AP215 so they can be applied to all future exchange efforts.

Implementation agreements are required due to the following reasons:

- **Sub-setting of large models to ease implementation**

  Reducing the number of different types valid for the exchange means that less code needs to be developed. An example would be a reduction of the number of curve subtypes that are used in the ISE-4 implementations.

- **Removal of not required elements to reduce complexity**

  Parts of the standard that are not used or that are out of scope will not be translated. For example, much of ISO 10303-46 (Visual Presentation) (Reference 7) can be used in CAD systems, therefore, partners may agree to exclude this from their translators though it may be a part of the standard that they use.

- **Clarification of ambiguities to achieve compatible translators**

  Where definitions are ambiguous or incorrect, statements are made to provide a common interpretation of them. For example, assembly structures may be generated for many purposes (construction, maintenance etc.) and in many ways; therefore, these are clarified and re-worded.

4.3.2.2 Categorization of Types of Implementation Agreements

These agreements are considered mandatory for translators developed and tested in the ISE-4 – Ship Arrangements program. However, it is useful to distinguish the agreements that are specific to the ISE-4 program from those which are recommended practice. To this end, the agreements are classified as follows:

- **Incorporate into ISO standard**

  These agreements will be recommended by the ISE-4 Project for incorporation into the ISO STEP standard. They are not currently part of the ISO STEP standard.

- **Current STEP Practice**

  These agreements reflect current practices for implementing STEP translators, but are not part of the ISO STEP standard.

- **Recommended**

  These agreements reflect the recommended usage, but are not part of the ISO STEP standard.

- **ISE-4 Project**

  These are agreements made specifically for ISE-4 Project, typically to reduce complexity or ease implementation and testing.

- **ESTEP Task**

  These agreements were made specifically for ESTEP Task, typically to reduce complexity or ease implementation and testing.

4.4 Test Plan and Test Cases

4.4.1 Testing Methodology

This section defines the methodology used to test the ISE-4 – Ship Arrangements translators.

4.4.1.1 Overview

The test data for the ISE-4 team exchange tests is derived from the data for a U.S. Navy ship - a 120 foot Torpedo Weapons Retriever (TWR), built in 1985. The drawings for this ship have been cleared for public release and were provided to the ISE-4 Team by NSWCCD. Each shipyard environment has modeled selected portions of the TWR in their native environment using standard modeling techniques for their environment. Each environment then produced test files based upon this model and posted them to the ISE-4 web host. The intent was to produce a set of standardized test files that are representative of tools, techniques, and standards specific to each shipyard environment. These test files are also expected to provide the basis for STEP AP usage guides and abstract test suites.

The ISE-4 team test environment, in actuality, consists of many environments:

1) The individual shipyard environments in which the tests are performed

2) The ISE-4 web site where test files are stored and distributed to the shipbuilder / CAD vendor teams for testing

The ISE-4 exchange tests employ a pair-wise exchange between shipyard environments. Each shipyard environment then tested their translator with the set of test files created by other environments. This is illustrated below in Figure 6.

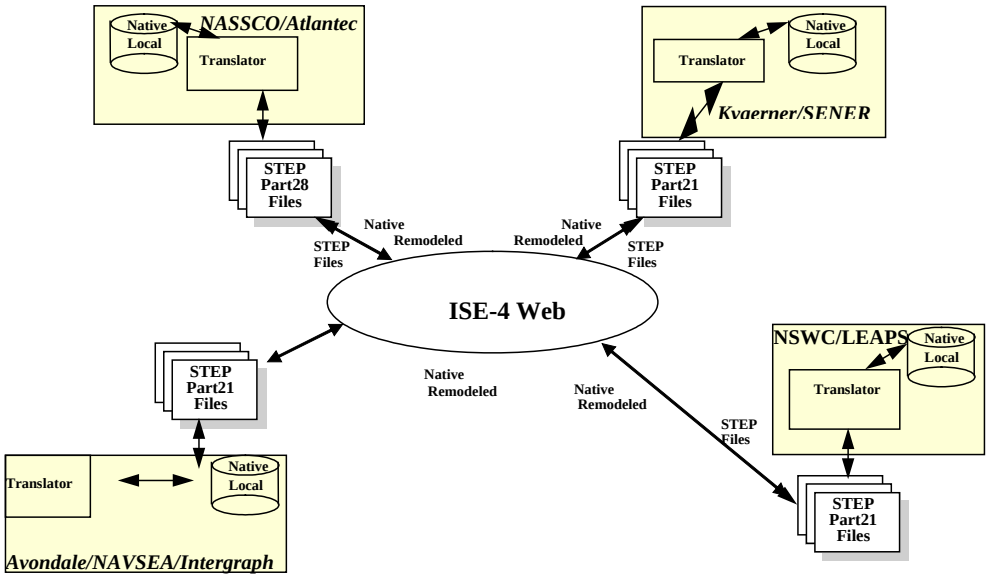


Figure 6: ISE-4 Test Environment

Each team test is performed within each shipyard environment. For a given preprocessor test, the appropriate native remodeled test data set is output as a STEP file by each shipbuilding environment and posted to the ISE-4 web. For a given postprocessor team test, each shipbuilding environment downloads the STEP files applicable for that test, and postprocesses the files. Thus, each environment produces one test data set for a given preprocessor test, but processes three test data sets for a given postprocessor test.

This testing methodology was previously used in the MariSTEP, ISE-2 ESTEP, and ISE-3 HVAC programs with good results. The test files produced by individual shipyards did in fact contain significant variations, which challenged every postprocessor. Other exchange scenarios can be developed, but it is felt that this methodology provides very good test coverage for the vast majority of scenarios that are useful to shipyards. Typically these other exchange scenarios build upon the pairwise exchange, e.g. shipyard A sends a file to shipyard B who modifies the data and then sends the file to shipyard C. The result of these other exchange scenarios is merely a refinement of the test coverage provided by a pairwise exchange. Such scenarios are useful for demonstration purposes, but do not add significantly to general translator testing.

STEP test files can potentially be written in either Part 21 (standard STEP neutral file format) or Part 28 (XML format). Most STEP translators traditionally have required Part 21 files. However, several vendors in the ISE-4 team are producing Part 28 files. The data content of either format is equivalent and mediator tools were developed under the ISE-4 Project to convert between Part 21 and Part 28 formats.

4.4.1.2 Exchange Tests

The ISE-4 team exchange tests focus on testing a particular entity or group of related entities for AP215. The tests progress in complexity with each successive test. The idea is to start with simple tests and "build" complexity by adding different, but related entities. Test data sets are to be reused wherever possible to take maximum advantage of "proven" data sets in testing multiple exchange files and/or external references. Each test typically contains one or more related sub-tests that further refine the entities tested. The test cases for this specific AP are defined in Section 4.4.4.

4.4.1.3 Test Method Details

Each shipbuilding environment performed each team exchange test using the ISE-4 test worksheets and entity / attribute checklists specific to that test. For each test, there is a single ISE-4 template which contains a test overview, a preprocessor worksheet, a postprocessor worksheet, and an entity / attribute checklist for that test.

4.4.1.3.1 Preprocessor

To test the ISE-4 preprocessors, each shipbuilding environment modeled / remodeled each test data set within the environment's native system. Each environment executed a preprocessor test using the native data set to produce a STEP Part 21 or Part 28 file. This resulting file was analyzed to ensure that the values within the native model are correctly represented and that a valid model was produced. At the conclusion of the test, a validated STEP Part 21 or Part 28 file was posted to the ISE-4 web.

During a preprocessor test execution, results are recorded directly on the preprocessor worksheet and later summarized on the Preprocessor Test Report for that test. Checklists for simple tests will typically contain most entities in the test case, however checklists for more complex tests will check only for selected entities.

4.4.1.3.2 Postprocessor

To test the ISE-4 postprocessors, each shipbuilding environment obtained the STEP Part 21 or Part 28 files applicable to that test from the ISE-4 web. Each shipbuilding data set was postprocessed separately and the entities / attributes in the Part 21 or Part 28 file were verified to ensure correct implementation.

During a postprocessor test execution, results are recorded directly on the preprocessor worksheet and later summarized on the Postprocessor Test Report for that test. Checklists for simple tests typically contained most entities in the test case, however checklists for more complex tests checked only for selected entities.

4.4.1.3.3 Problem Resolution

Since shipyard environments produce test files as well as test translators, it is often unclear whether a problem lies with the translator or the test file. It is important to identify problems as they are encountered during testing. These should be captured in the test template. If problems are found in a test file, they should be corrected and a new version posted to the ISE-4 web, time permitting.

4.4.1.4 Naming Conventions

Standardized naming conventions are required to effectively manage the ISE-4 test team environment.

4.4.1.4.1 Shipyard Environment and CAD Vendor Names

Abbreviations that were used to identify each shipyard environment and CAD vendor are included in the list of Acronyms, Definitions, and Abbreviations (Section 2.1).

4.4.1.4.2 Test Names

The test naming convention is based on the AP and the test. The convention is as follows:

<AP>-<test>-<subtest>

For example, *AP215-2-3,* denotes AP215, test 2, sub-test 3.

4.4.1.4.3 Test File Names

The test file naming convention is based on the AP, test, and data source. The convention is as follows:

<AP>-<test>-<subtest>-<source>

For example, *AP215-2-3-EB,* denotes test data for AP215, test 2, sub-test 3 which originated at EB

4.4.1.4.4 Test Template Names

The test templates are named as follows:

ISE4-<AP><test><subtest>

For example, *ISE4-215-3-1* contains the preprocessor worksheet, postprocessor worksheet, and entity / attribute checklist for AP215 Test 3, Sub-test 1.

4.4.1.4.5 Naming Conventions in Test Data

Standardized naming conventions are also required for certain types of test data. Test data must contain a standardized set of material types and catalog part numbers. To ensure uniqueness and provide traceability of data, GUIDs must contain the shipyard environment as their company id. Other naming conventions are specified in the Implementers Agreements, e.g. naming of surfaces and other types of geometry.

4.4.2 Translator Environments

4.4.2.1 ISDP/Intelliship (Intergraph)

4.4.2.1.1 Overview

The AP215 Arrangement STEP translators are designed to export and import Part 21 or Part 28 STEP files. The preprocessor was implemented in Intergraph's ISDP (Integrated Ship Design and Production) suite of shipbuilding CAD products. The postprocessor was implemented in Intergraph's IntelliShip shipbuilding CAD product. Both preprocessor and postprocessor extend the AP215 Arrangement translators developed during ISE-2.

4.4.2.1.2 Interfaces

ISDP Preprocessor (Export)

A command is provided that allows the user to enter the name and location of the STEP file to be created or to accept a default name and location.

IntelliShip Postprocessor (Import)

A command is provided that allows the user select a file from the list of files in the default location or to enter the name and location of the STEP file to be translated.

4.4.2.1.3 Hardware and Software Environment

Target Platform

ISDP Preprocessor: Solaris x86

IntelliShip Postprocessor: Microsoft Windows

Development Tools:

Intergraph is using ST-Developer by STEP Tools, Inc for the postprocessor..

Operations

1. ISDP Preprocessor (Export) The user will do the following:
   - Select the *Export to STEP* command
   - Select AP215 from the list of AP's that can be exported
   - Enter the name and location of the output STEP file or accept the defaults
   - All AP215 Compartmentation and Zone data within the is translated into the selected STEP file based on arrangements property mapping file
2. IntelliShip Postprocessor (Import) The user will do the following:
   - Select the DataExchange Task
   - Select the *Import* command from the DataExchange Vertical Toolbar
   - Select AP215 from the list of AP's that can be exported
   - Select the input file from a list or Enter the name and location of the STEP file
   - All AP215 Arrangements data within the STEP file is translated into the IntelliShip workspace and stored in the database

4.4.2.2 TRIBON (Atlantec-es)

NASSCO is using the AVEVA Tribon System for structural design, piping and machinery. Atlantec-es developed AP215 (Arrangement) translators, utilizing the Tribon Compartment System in support of the ISE-4 Ship Arrangements Task.

This section describes how Atlantec-es' ISE-4-developed translator(s) interact with company systems and external systems. Besides a technical hardware/software perspective, this section describes interfaces along the paths that data traverses.

4.4.2.2.1 Overview Diagrams

Figures 7 and 8 below show the data flow on import and export of AP215 information.

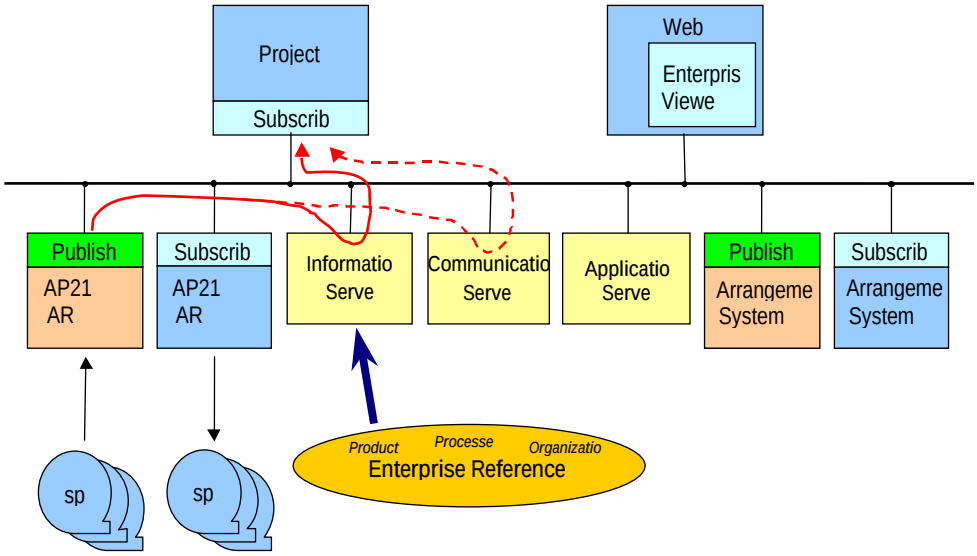


Figure 7: Data Flow on Import of AP215 Information

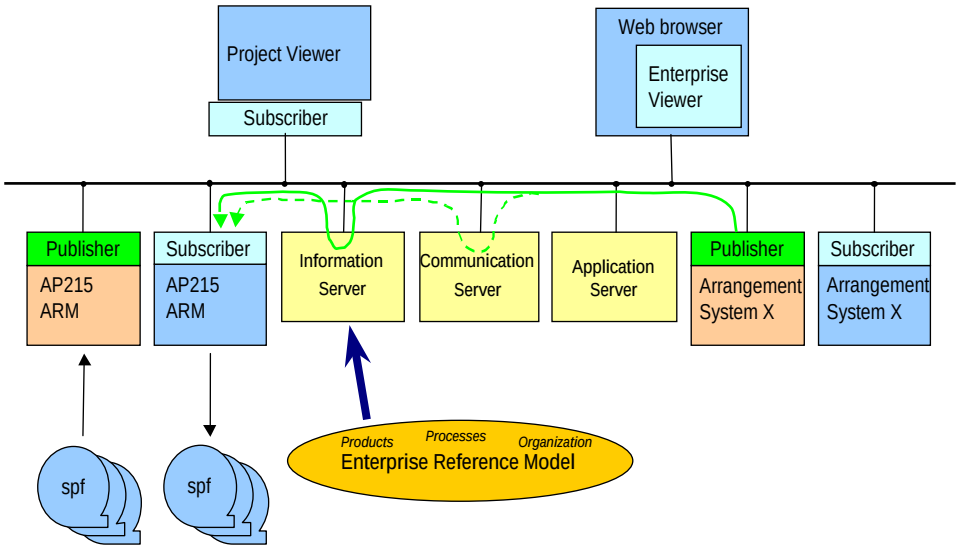


Figure 8: Data Flow on Export of AP215 Information

4.4.2.2.2 Interfaces

The CAD system that is used as the data source for the export of AP215 arrangement data is the Compartment Module of the Tribon Initial Design.

a) Postprocessor

AP215 data is imported into Atlantec-es's *Topgallant*® Information Infrastructure. Access to the information is available through tools such as Project Viewer.

4.4.2.2.3 Hardware and Software Environment

a) Target Platform: any JVM enabled platform

b) Development Tools:

Implementation is in Java 1.3. Atlantec uses its own tools to handle XML, EXPRESS and STEP Physical Files and to generate APIs from EXPRESS models.

Differences between a Test environment and anticipated Production environment:

Virtually none

4.4.2.2.4 Operations

a) Preprocessor

An AP215 Subscriber Adapter accesses available arrangement data from Information Server. A user interface -part of the subscriber – enables users to request data that is exported to STEP Physical Files. Subsequently the subscriber adapter requests data through Communications Server from the originating data source, for example, Tribon Initial Design.

b) Postprocessor

An AP215 Publisher Adapter monitors data locations in for changes. The Adapter recognizes the arrival of information... Once “aware” a publishing adapter scans the file and notifies Information Server that new information is available. However, the Adapter does not copy new data unless it is requested by Communication Server. If requested, an AP215 Publisher Adapter provides the information. by requesting it from a Subscriber Adapters\. These Subscriber Adapters can be part of external systems such as Tribon, or they can be part of *Topgallant*® application such as Project Viewer.

4.4.2.3 FORAN (SENER)

4.4.2.3.1 Overview

FORAN’s STEP translator for AP215’s ship arrangements is designed for both exporting and importing Part 21 STEP files from, and to, FORAN V50 forms database.

4.4.2.3.2 Interfaces

Preprocessor (Export)

The information to be exported by the STEP translator is stored in FORAN forms database.

The preprocessor is completely integrated with FORAN’s basic design and naval architecture modules. FORAN’s AP215 STEP translator has the same visualization and selection facilities as FORAN’s basic design and naval architecture modules, allowing in this way either a partial or complete export to a Part 21 STEP of the ship arrangements information existing on the database for a particular ship.

Postprocessor (Import)

AP215’s ship arrangements data is imported from a Part 21 STEP file.

A project associated to the ship must exist on database before starting the import process. Such project should have hull forms and decks already defined.

A validation process of the contents of the STEP file being imported will be carried out before loading the ship arrangements data in FORAN forms database. If the validation process ends

successfully, the information from the STEP file will be integrated with the rest of ship information, being so accessible for any other process or operation.

4.4.2.3.3 Hardware and Software Environment

Target Platform:

Microsoft Windows NT. Also a Linux version could be available with little additional effort due to the fact that FORAN system also runs under this operating system.

External Tools:

An EPM STEP toolkit existing in market is used in both the preprocessor and the postprocessor.

Differences between a Test environment and anticipated Production environment:

There are no differences anticipated.

4.4.2.3.4 Operations

Preprocessor

Starting from a project defined in FORAN forms database that will contain hull forms and decks information, visualization of the ship's arrangement elements and a user-driven selective process is carried out.

Ship arrangement's elements selected by the user are exported to a Part 21 STEP file.

Postprocessor

After having defined a project in FORAN forms database that will include hull forms and decks, the input Part 21 STEP file will be read and validated. If this process runs correctly the ship arrangements information on file will be transferred to FORAN's forms database and such information will be accessible for further visualization, selection and interaction with the rest of information existing in the database.

4.4.2.4 Leading Edge Architecture for Prototyping Systems (LEAPS)

4.4.2.4.1 Overview

LEAPS (Leading Edge Architecture for Prototyping Systems) is a product design and analysis system developed and supported by Naval Surface Warfare Center Carderock Division (NSWCCD) for use by NAVSEA and its contractors. LEAPS may be used in the design and analysis of multiple product types through the specification of product-specific business objects that specialize the underlying generic LEAPS data constructs and property types. In the case of ship design, the FOCUS object model specifies the appropriate ship design and analysis business objects and properties. LEAPS serves both as a central repository for ship design data and operational analysis properties and as the link to numerous software programs used by NAVSEA to perform specific performance analysis tasks for a ship design project. NSWCCD has developed direct translators between the LEAPS repository and various performance analysis systems used in a ship design, such as ASSET (see Figure 9). NSWCCD has also developed IGES translation capabilities to import and export geometric design data from various CAD system utilized by NSWCCD and NAVSEA, as well as data from the numerous CAD systems used by Navy contractors, most of which have IGES data exchange capabilities.

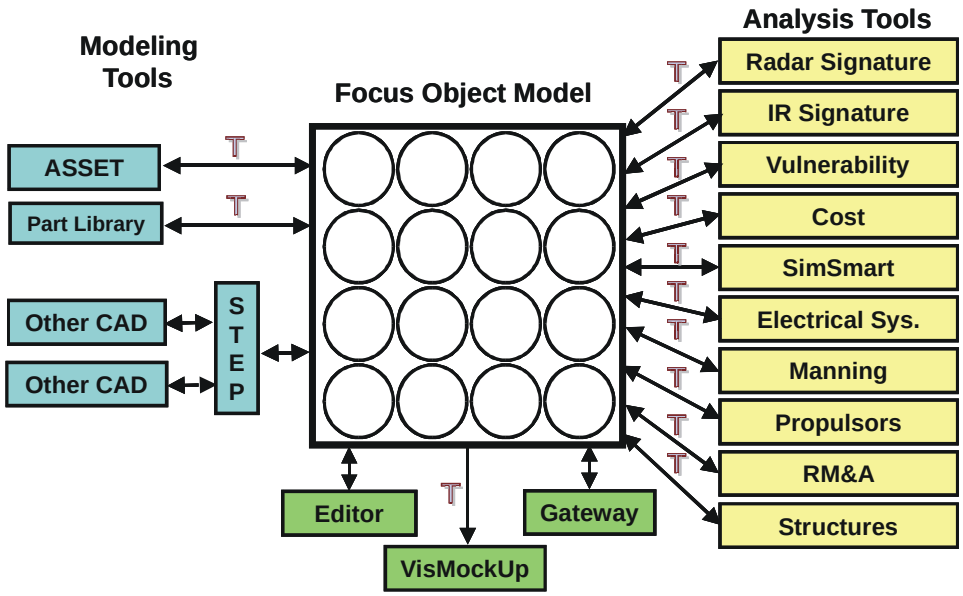


Figure 9: The Navy Leading Edge Architecture for Prototyping Systems (LEAPS)

During 2003 and 2004, the Navy eBusiness office funded development of the initial STEP import and export translators for FOCUS/LEAPS to STEP AP216, the STEP Application Protocol for the exchange of Ship Moulded Forms.

The AP215 import and export translator software consists of DLL libraries with C++ classes for each data object in each of the STEP AP's ARM and AIM schemas and corresponding C++ methods to accomplish the data conversion required by the AP's Mapping Specification. This data is then imported or exported from the LEAPS database by additional C++ classes and translator methods that act between the STEP ARM data population within the temporary ST-Developer repository, and the methods available within the LEAPS system for reading and writing to the LEAPS database and associating Focus object types to the data. In the case of a STEP data import, the AP215 AIM data is mapped by these methods into a population of the AP215 ARM. The translator C++ methods then read this data, format it into the LEAPS data structures with the correct naming as specified in the Focus 2003 data model, and populate the LEAPS database with the data from the input STEP physical file. The export translator performs similar operations in reverse to export LEAPS data selected by the user, translating it through the AP215 ARM model through the implementation of the AP's Mapping Specification for export as STEP AP215 AIM file. See Figure 10.

4.4.2.4.2 Interfaces

Preprocessor (Export)

The information to be exported by the STEP translator is stored in any LEAPS database. The preprocessor is accessed using a Graphical User Interface (GUI) which works directly against the selected LEAPS database. The GUI allows selection of the STEP AP for which LEAPS data is to be exported, i.e., either STEP AP215 or AP216. Upon selection of an AP215 export, user-selectable variables are provided to the user to select the particular LEAPS database, ship, and version (study). Options are also provided to handle various data options within AP215. For example, during the AP215 export processing, the user may choose to export an AP216 File that contains that bounding AP216 Moulded Form objects for the compartments or zones selected for export. Similarly user input is accepted for instructions to either export the compartment data for

those compartments within a zone, or alternatively the user may instruct the processor to export only external instance references to the constituent compartments' globally unique identifiers for a compartment arrangement or for any other space product structure type.

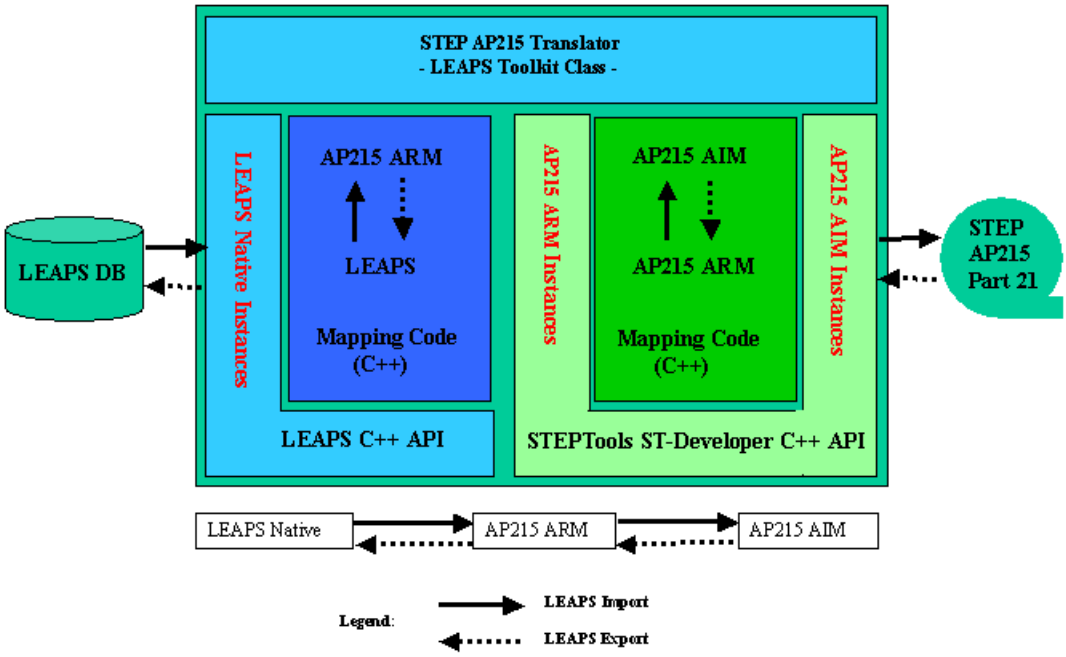


Figure 10: LEAPS STEP AP215 Import/Export Translator Architecture

Postprocessor (Import)

The information to be imported by the STEP translator is stored in any existing or new LEAPS database. This database may be empty for an initial import to create a single ship study, or the imported data may be used to populate an alterative ship study in a LEAPS database previously populated with one or more other studies.

A validation process of the contents of the STEP file being imported is carried out during the import process to check the incoming data against the requirements of AP215 AIM schema. Error logging is provided to identify any warnings or errors in the STEP file selected to be imported in the import process. If the validation process ends successfully, the information from the STEP file will be populated in the LEAPS database.

A branch of the same GUI described previously is used for the import process, allowing the user to select an AP215 import, the STEP file to be imported, and the LEAPS database where the data is to be populated.

4.4.2.4.3 Hardware and Software Environment

Target Platform:

Microsoft Windows XP.

External Tools:

A STEP Tools, Inc. STEP toolkit is used in both the preprocessor and the postprocessor.

Differences between a Test environment and anticipated Production environment:

None.

4.4.2.4.4 Operations

Preprocessor

A Graphical User Interface is used to specify the output name and location of the STEP file, to select the LEAPS database, ship, and study from which compartmentation data is to be exported, and allow selection of the particular compartments or zones which are to be exported. User selectable options instruct the processor to perform various export options as described previously.

The selected data is exported to a Part 21 STEP file conforming to the AP215 Application Interpreted Model (AIM).

Postprocessor

The input AP215 AIM Part 21 STEP file selected by the user through the GUI is read and validated to be in conformance with the AP215 AIM schema. If this process is successfully completed, the AP215 data is imported to the LEAPS database, ship, and study specified by the User.

4.4.3 TWR as Source of Test Data

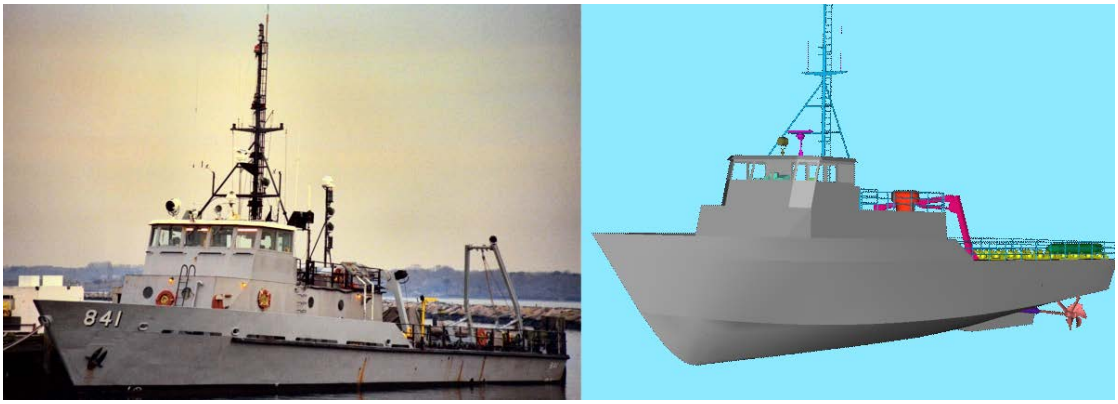


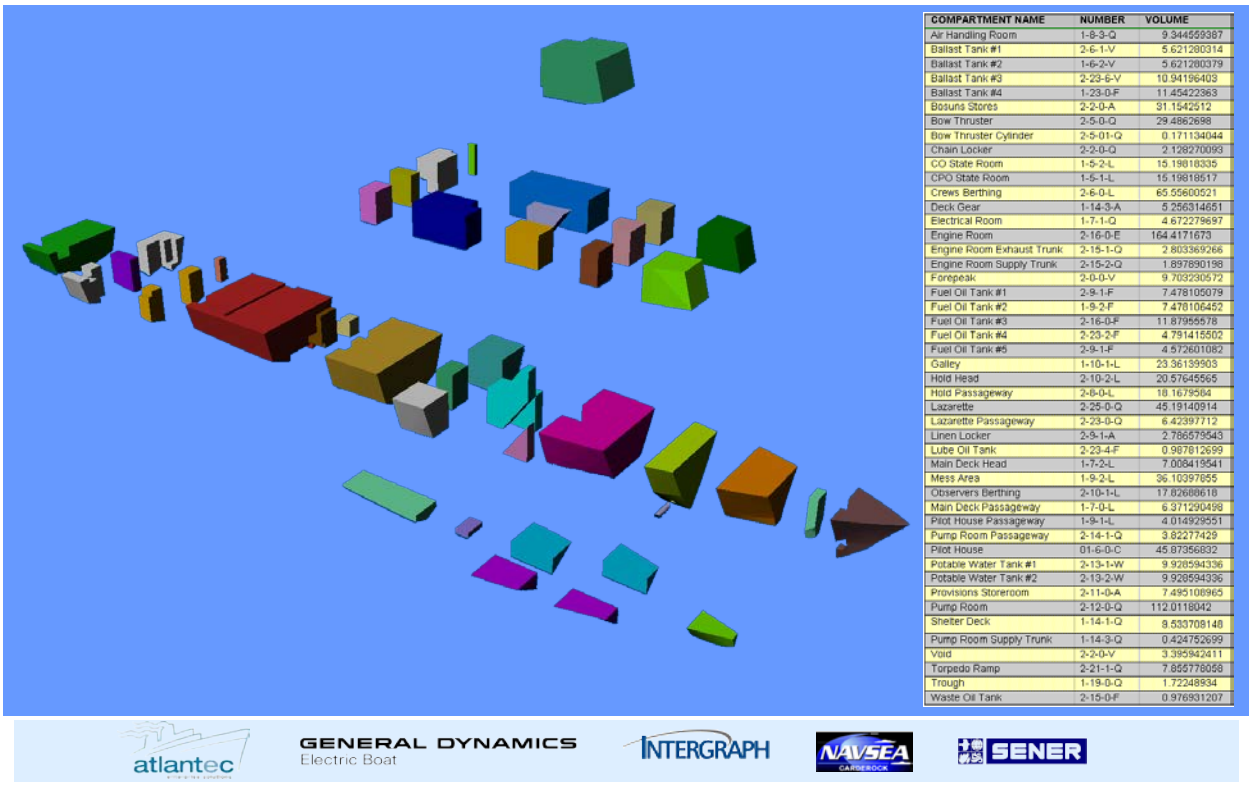
Figure 11: TWR Picture and CAD Model

For as long as we have been working on the selection of CAD systems, the development of data translators to support the marine industry, and the development of marine product model technology, the availability of test data has always been a problem. Data from the commercial shipbuilders has always had the problem of being proprietary. Data about U.S. naval combatant vessels has always had its own unique security problems. This problem became very apparent during the MariSTEP program in which each of the participants were unwilling to provide a dataset which would be provided to their competitors. During MariSTEP, the Navy was in the midst of a project to develop a new concept for the arrangement of an engine room. There were however some problems with this dataset from the perspective of completeness. Coinciding with the ESTEP program another source of data was located, the Torpedo Weapons Retriever. Finally, we have found a ship that meets the needs to the information technology side of the marine industry.

The Torpedo Weapons Retriever has been selected as the test platform by several Information Technology initiatives because:

- Its data is unclassified
- Its data is non-proprietary
- The ship is still operational
- The ship design uses traditional HM&E technology
- There is a relatively small volume of data
- The ship design encompasses a full range of systems and components

There are currently 172 drawings collecting 782 drawing sheets into a traditional paper-based description of the 120' Torpedo Weapon Retriever. These drawings are currently available on-line as scanned Tagged Image File Format (TIFF) images. Additional information required to develop a product model of this ship is also available. Currently, the amount of product model data is limited, but as programs such as ISE create product model data in either native of neutral format, it is expected that a copy and a description of the product model data will be entered into the TWR product model repository (Reference 8). The TWR has already been used to define moulded forms in support of ISO 10303 Part 216 (AP216), structure in support of ISO 10303 Part 218 (AP218), compartmentation information in support of ISO 10303 Part 215 (AP215), and to support the development of distributed systems models in support of ISO 10303 Part 227 (AP227). The TWR product model repository can be accessed without restriction from rabecs.dt.navy.mil



COMPARTMENT NAME	NUMBER	VOLUME
Air Handling Room	1-8-3-Q	9.344559387
Ballast Tank #1	2-6-1-V	5.621280314
Ballast Tank #2	1-6-2-V	5.621280379
Ballast Tank #3	2-23-6-V	10.94196403
Ballast Tank #4	1-23-0-F	11.45422363
Bosuns Stores	2-2-0-A	31.1542512
Bow Thruster	2-5-0-Q	29.4862698
Bow Thruster Cylinder	2-5-01-Q	0.171134044
Chain Locker	2-2-0-Q	2.128270093
CO State Room	1-5-2-L	15.19818335
CPO State Room	1-5-1-L	15.19818517
Crews Berthing	2-6-0-L	65.55600521
Deck Gear	1-14-3-A	5.256314651
Electrical Room	1-7-1-Q	4.672279697
Engine Room	2-16-0-E	164.4171673
Engine Room Exhaust Trunk	2-15-1-Q	2.803369266
Engine Room Supply Trunk	2-15-2-Q	1.897890198
Forepeak	2-0-0-V	9.703230572
Fuel Oil Tank #1	2-9-1-F	7.478105079
Fuel Oil Tank #2	1-9-2-F	7.478106452
Fuel Oil Tank #3	2-16-0-F	11.87955578
Fuel Oil Tank #4	2-23-2-F	4.791415502
Fuel Oil Tank #5	2-9-1-F	4.572601082
Galley	1-10-1-L	23.36139903
Hold Head	2-10-2-L	20.57645565
Hold Passageway	2-8-0-L	18.1679584
Lazarette	2-25-0-Q	45.19140914
Lazarette Passageway	2-23-0-Q	6.42397712
Linen Locker	2-9-1-A	2.786579543
Lube Oil Tank	2-23-4-F	0.987812699
Main Deck Head	1-7-2-L	7.008419541
Mess Area	1-9-2-L	36.10997655
Observers Berthing	2-10-1-L	17.82688618
Main Deck Passageway	1-7-0-L	6.371290498
Pilot House Passageway	1-9-1-L	4.014929551
Pump Room Passageway	2-14-1-Q	3.82277429
Pilot House	01-6-0-C	45.87356832
Potable Water Tank #1	2-13-1-W	9.928594336
Potable Water Tank #2	2-13-2-W	9.928594336
Provisions Storeroom	2-11-0-A	7.495106965
Pump Room	2-12-0-Q	112.0118042
Shelter Deck	1-14-1-Q	8.533709148
Pump Room Supply Trunk	1-14-3-Q	0.424752699
Void	2-2-0-V	3.395942411
Torpedo Ramp	2-21-1-Q	7.855776056
Trough	1-19-0-Q	1.72248934
Waste Oil Tank	2-15-0-F	0.976931207

Figure 12: Compartmentation Breakdown of the TWR

Figure 12 shows the compartmentation breakdown of the TWR that was used as test data for the Ship Arrangements implementations.

4.4.4 Definition of Test Cases

These test cases are compartments, zones, and associated requirements and properties selected from the TWR product model. Various properties exchanged are extracted or calculated from the Navy's TWR database and the ship's drawings. In some cases, if corresponding actual ship data is not available, properties and other product model data may be created in the various CAD systems solely to test the exchange.

These test cases are based on drawings of the TWR 841. The test case specifications include a Product Model system independent description of the test cases suitable to support the generation of native models required to exercise a STEP preprocessor. This includes a formatted text file to represent the non-graphical attributes, and STEP AP203 files to represent a nominal geometric configuration of the data.

In the Test Cases described below, Test Cases 2, 6, 9, and 11 formed the basis of the ISE-4 Demo at the end of the project.

The following are the high-level test case descriptions for exchange of compartmentation information.

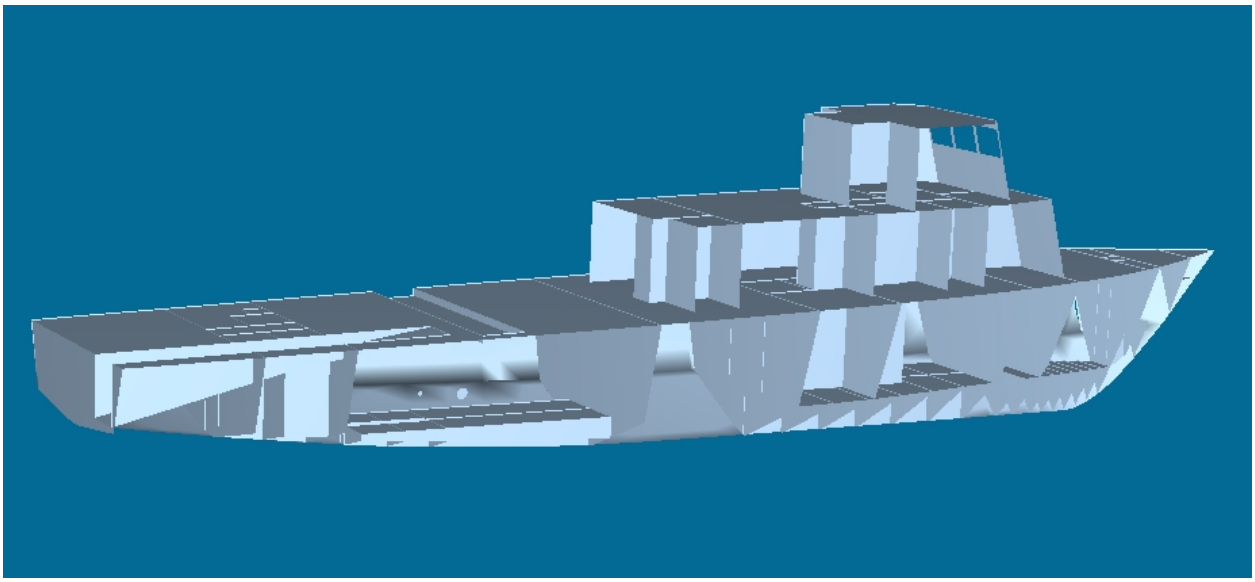


Figure 13: TWR Compartments

4.4.4.1 Compartment mission requirements w/o geometry

4.4.4.1.1 Purpose

This test case consists of a single compartment item with compartment functional definition and one or more compartment design requirements, without compartment design definition or properties. No geometry is exchanged in this testcase.

4.4.4.1.2 Exchange Context

The context for this test case is the exchange of initial text description of design requirements between ship owner and design agent.

4.4.4.2 Compartment required properties in initial deck arrangement w/o geometry

4.4.4.2.1 Purpose

This test case consists of multiple compartment items in a deck zone's compartment arrangement space product structure, with compartment functional definitions, design definitions without geometry, compartment identification properties and required minimum and/or maximum values for a selected set of compartment properties. No geometry is exchanged in this testcase.

4.4.4.2.2 Exchange Context

The context for this test case is the exchange of an initial non-geometric compartment arrangement and required design property values between ship owner and design agent.

4.4.4.3 Single compartment design with explicit trimmed boundaries

4.4.4.3.1 Purpose

This test case consists of a single compartment detail design with compartment functional definition and design definition, including explicit trimmed geometry for the compartment's shape representation.

4.4.4.3.2 Exchange Context

The context for this test case is the exchange of a compartment's explicit geometric shape between owner, design agent, classification society or shipbuilder.

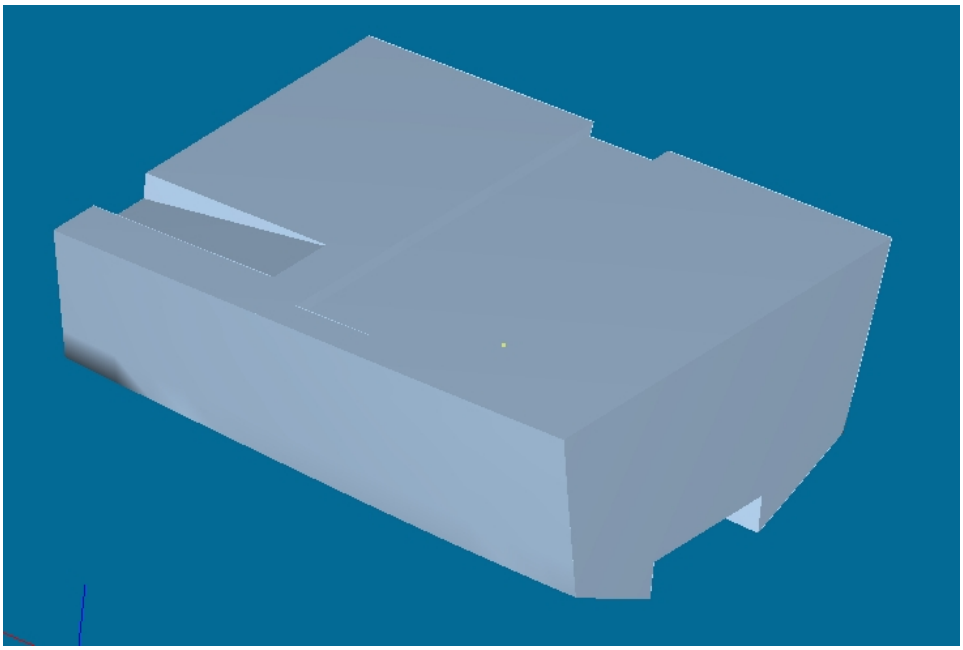


Figure 14: Test Case 3 – Single compartment design with explicit trimmed boundaries

4.4.4.4 Single compartment design with AP216 implicit boundaries

4.4.4.4.1 Purpose

This test case consists of a single compartment detail design with compartment functional definition and design definition with untrimmed geometry for the compartment's boundaries. Associated AP216 file

shall be exchanged with this testcase in order to resolve external instance references to Moulded Form GUIDs.

4.4.4.4.2 Exchange Context

The context for this test case is the exchange of a compartment's implicit geometric shape between owner, design agent, classification society or shipbuilder.

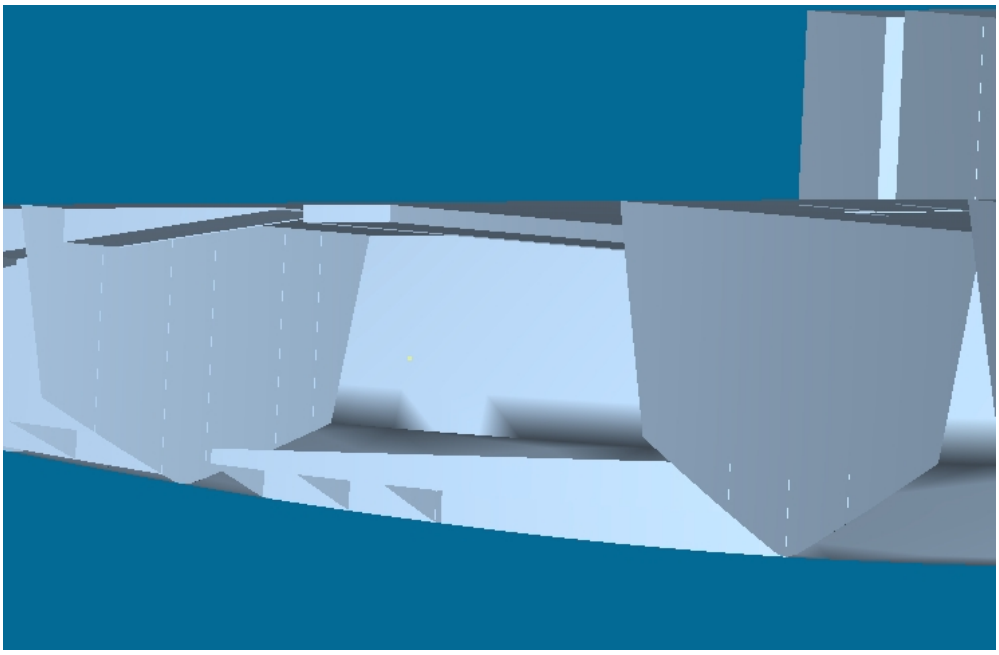


Figure 15: Test Case 4 – Single compartment design with AP216 implicit boundaries

4.4.4.4 Single compartment detail design with as-designed properties

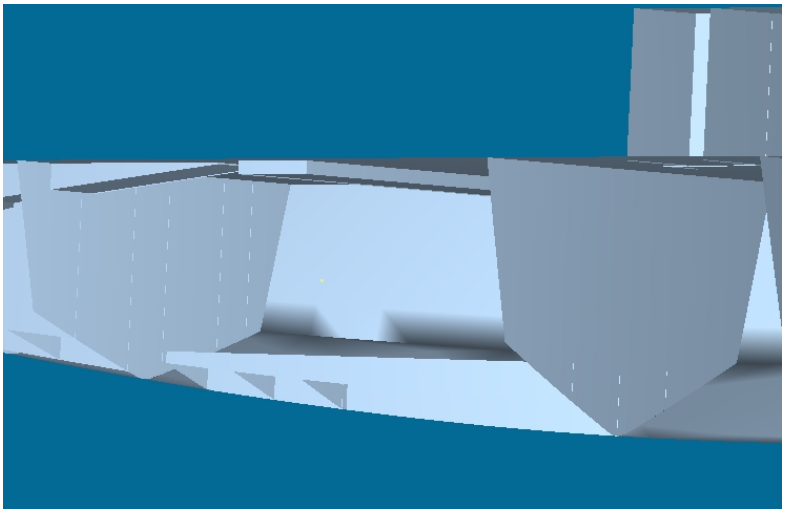


Figure 16: Test Case 5 – Single compartment detail design with as-designed properties

4.4.4.5.1 Purpose

This test case consists of a single compartment detail design with compartment functional definition, design definition with both explicit and implicit shape representations, and as-designed values for a selected set of compartment identification, volume, and area properties.

4.4.4.5.2 Exchange Context

The context for this test case is the exchange of a compartment's detail design function, geometric shape, and properties between design agent, owner, classification society or shipbuilder.

4.4.4.6 Multiple compartment detail designs with as-designed tank properties

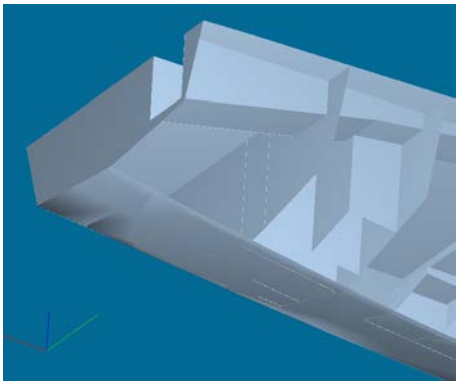


Figure 17: Test Case 6 – Multiple compartment detail designs with as-designed tank properties

4.4.4.6.1 Purpose

This test case consists of multiple tank compartment detail designs with compartment functional definitions, design definitions with both explicit and implicit shape representations, and as-designed values for a selected set of compartment identification, volume, area, and tank properties.

4.4.4.6.2 Exchange Context

The context for this test case is the exchange of a compartment's detail design function, geometric shape, and properties between design agent, owner, classification society or shipbuilder.

4.4.4.7 Compartments in zone

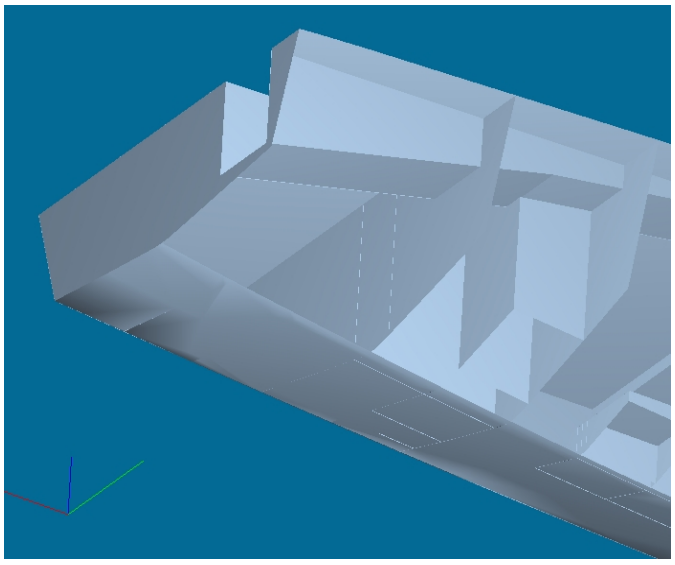


Figure 18: Test Case 7 – Compartments in zone

4.4.4.7.1 Purpose

This test case consists of the exchange of a design zone, including the same tank compartments as in testcase 6 as the constituent compartments, with a selected set of as-designed properties applicable to the entire zone.

4.4.4.7.2 Exchange Context

The context for this test case is the exchange of a zone, with identification of the constituent compartments within the zone, between design agent, owner, classification society or shipbuilder.

4.4.4.8 Compartment arrangement product structure on Main Deck 1

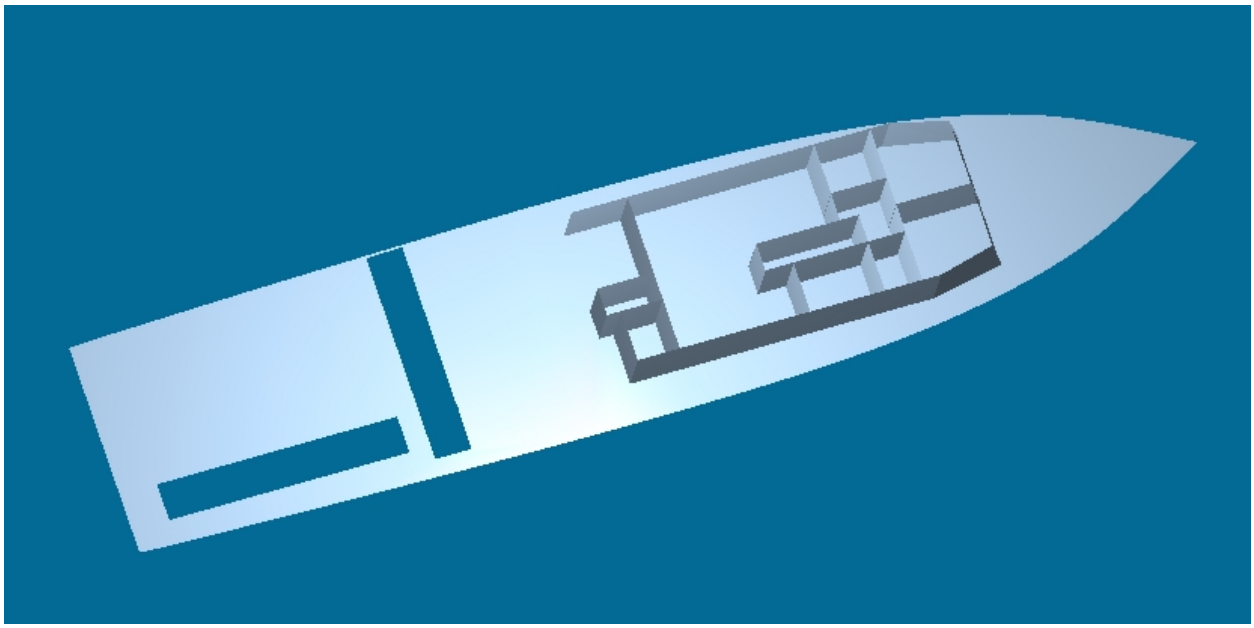


Figure 19: Test Case 8 – Compartment arrangement product structure on Main Deck (Deck 01 plating removed for illustration)

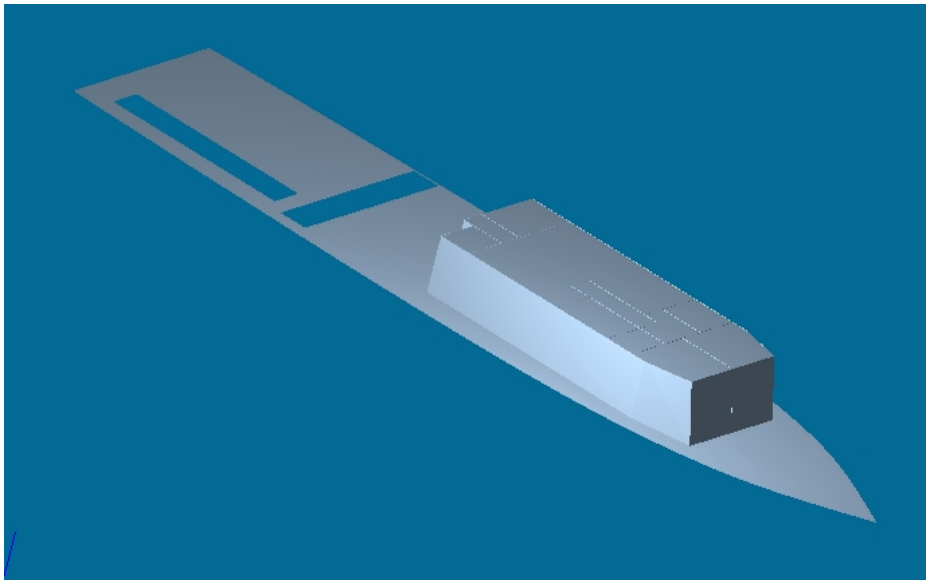


Figure 20: Test Case 8 – Compartment arrangement product structure on Main Deck

4.4.4.8.1 Purpose

This test case consists of the exchange of a space product structure for the Main Deck deck zone, including exchange of the product structure items, the items being all of the compartments on Main Deck.

Issue: Include space adjacency relationships for compartment access analysis?

4.4.4.8.2 Exchange Context

The context for this test case is the exchange of an arrangement zone, with exchange of the constituent compartment's detail designs, between design agent, owner, classification society or shipbuilder.

4.4.4.9 Compartment arrangement product structure on Main Deck 2

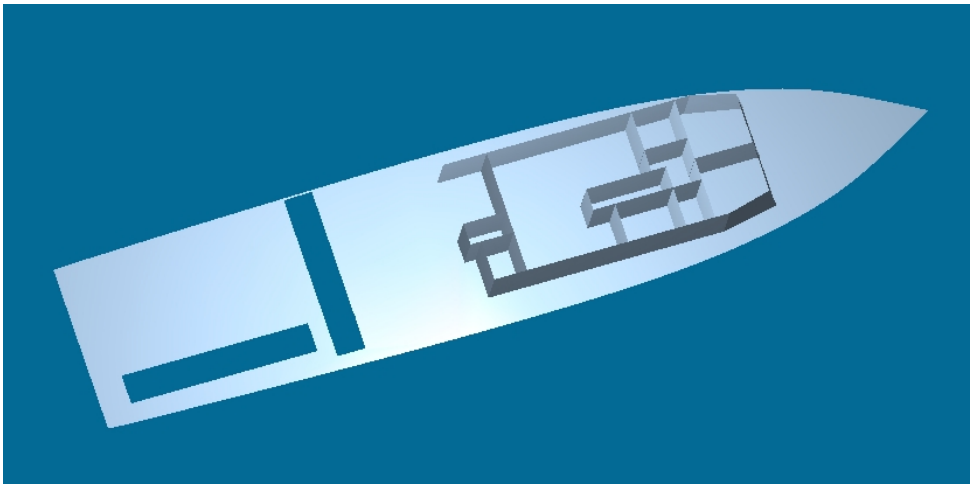


Figure 21: Test Case 9 – Compartment arrangement product structure on Main Deck (Deck 01 plating removed for illustration)

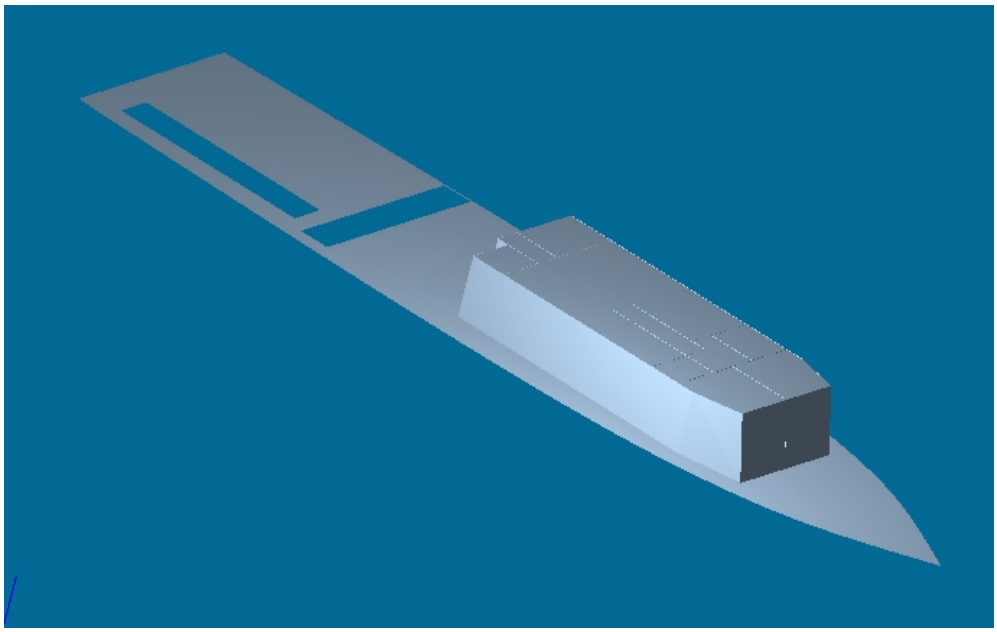


Figure 22: Test Case 9 – Compartment arrangement product structure on Main Deck

4.4.4.9.1 Purpose

This test case consists of the exchange of a space product structure for the Main Deck deck zone, including exchange of the product structure external item GUID identification, the external items being all of the compartments on Main Deck.

4.4.4.9.2 Exchange Context

The context for this test case is the exchange of an arrangement zone, with exchange of the constituent compartment's detail designs, between design agent, owner, classification society or shipbuilder.

4.4.4.10 Fire zone

4.4.4.10.1 Purpose

This test case consists of the exchange of a Fire zone for the forward berthing and stowage areas, with a selected set of properties for the zone.

4.4.4.10.2 Exchange Context

The context for this test case is the exchange of a fire zone in habitable compartments, with exchange of the constituent compartment's detail designs, between design agent, owner, classification society or shipbuilder.

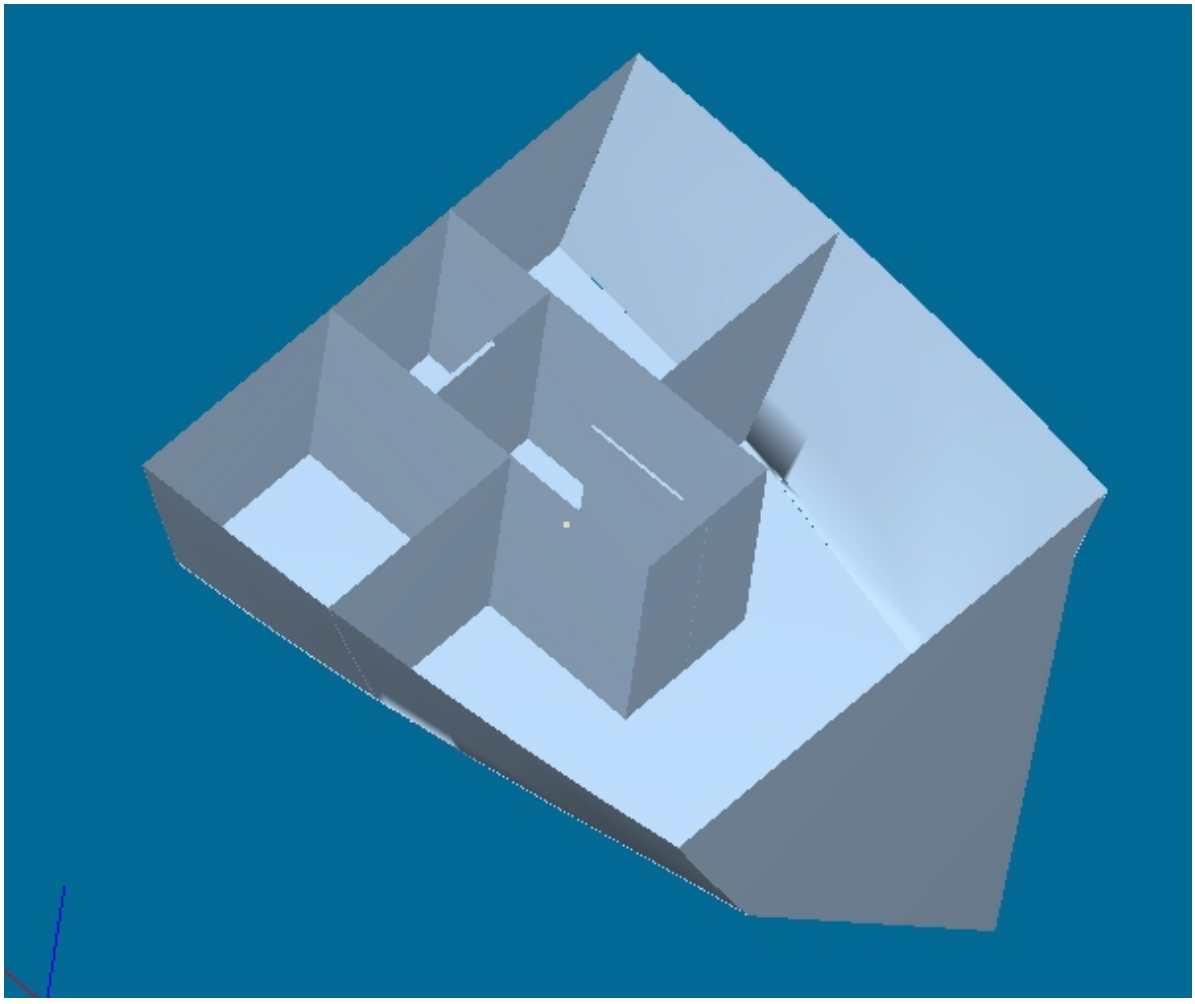


Figure 23: Test Case 10 – Fire zone

4.4.4.11 Items in a compartment space product structure

4.4.4.11.1 Purpose

This test case consists of the exchange of a space product structure with identifying GUIDs for all of the mechanical, structural and distribution system items within a compartment item.

4.4.4.11.2 Exchange Context

The context for this test case is the exchange of an "items in compartment" space product structure for a compartment, between design agent, owner, classification society or shipbuilder.

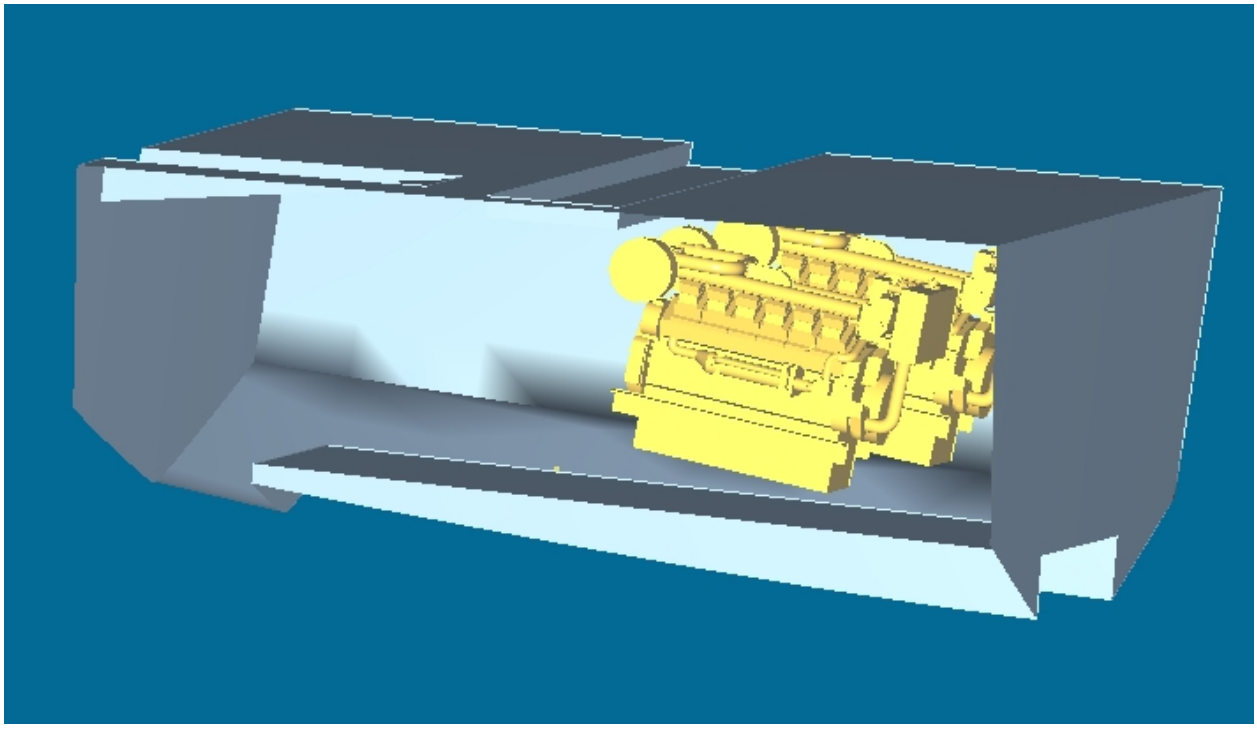


Figure 24: Test Case 11 – Items in a compartment space product structure

4.4.5 AP215 TEST CASE ENTITY USAGE TABLES

4.4.5.1 Entities Included in the Test Cases

The following table indicates which AP215 entities are used by which test cases. For convenience, entities are grouped by usage area.

Table 2: AP215 entities Used in Each Test Case

TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10	TC11	Usage	Entity
X	X	X	X	X	X	X	X		X	X	Space - Compartment	Compartment
X	X	X	X	X	X	X	X		X	X	Space - Compartment	Compartment_functional_definition
	X	X	X	X	X	X	X		X	X	Space - Compartment Design	Compartment_design_definition

TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10	TC11	Usage	Entity
	X						X	X			Space - Deck Zone	Deck_zone
	X						X	X			Space - Deck Zone	Deck_zone_functional_definition
							X	X			Space - Deck Zone Design	Deck_zone_design_definition
						X			X		Space - Zone	Zone
						X			X		Space - Zone	Zone_functional_definition
						X			X		Space - Zone Design	Zone_design_definition
											Adjacency	Adjacent_space_surface_area
	X										Adjacency	Space_adjacency_relationship
	X										Adjacency	Space_arrangement_relationship
											Adjacency	Space_connection_relationship
											Adjacency	Space_enclosing_relationship
											Adjacency	Space_positional_relationship
											Adjacency	Cargo_bay_definition
X	X										Document	Document
X	X										Document	Document_portion
X	X										Document	Document_reference
X	X										Document	Document_reference_with_address
X	X										Document	External_reference
X	X										Document	External_storage
X	X										Document	Universal_resource_locator
		X		X	X	X	X	X	X	X	Explicit Geometry	Non_manifold_surface_shape
			X	X	X	X	X	X	X	X	External Instance Reference	External_instance_reference
							X	X		X	Product Structure	Space_product_structure
					X	X					Properties	Capacity_properties
											Properties	Cargo_compartment_property

TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10	TC11	Usage	Entity
											Properties	Coating_level
	X			X	X	X					Properties	Compartment_abbreviated_name
											Properties	Compartment_acceleration
											Properties	Compartment_access_authorization
									X		Properties	Compartment_air_circulation_rate
	X			X	X	X			X		Properties	Compartment_area_property
				X					X		Properties	Compartment_coating
	X			X	X	X					Properties	Compartment_horizontal_cross_sectional_area_property
											Properties	Compartment_illumination
									X		Properties	Compartment_insulation
									X		Properties	Compartment_naval_administrative_property
											Properties	Compartment_noise_category
											Properties	Compartment_nuclear_classification
				X					X		Properties	Compartment_occupancy
	X			X	X	X					Properties	Compartment_property
									X		Properties	Compartment_safety_class
											Properties	Compartment_security_classification
											Properties	Compartment_stiffened_surface_area_property
				X	X	X			X		Properties	Compartment_tightness
											Properties	Compartment_unstiffened_surface_area_property
	X			X	X	X					Properties	Compartment_vertical_longitudinal_cross_sectional_area_property
	X			X	X	X					Properties	Compartment_vertical_transverse_cross_sectional_area_property
											Properties	Compartment_volume_permeability_property
	X			X	X	X					Properties	Compartment_volume_property

TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10	TC11	Usage	Entity
											Properties	Compartment_ziplist_number
											Properties	Corrosion_protection
	X			X	X	X			X		Properties	General_compartment_property
					X	X					Properties	Moments_of_inertia
					X	X					Properties	Tank_compartment_property
					X	X					Properties	Tank_geometric_parameters
					X	X					Properties	Tank_piping_design_properties
X	X	X	X	X	X	X	X	X	X	X	Required	Definition
	X	X	X	X	X	X	X	X	X	X	Required	Design_definition
X	X	X	X	X	X	X	X	X	X	X	Required	Functional_definition
X	X	X	X	X	X	X	X	X	X	X	Required	General_characteristics_definitio n
X	X	X	X	X	X	X	X	X	X	X	Required	Definable_object
X	X	X	X	X	X	X	X	X	X	X	Required	Global_id
X	X	X	X	X	X	X	X	X	X	X	Required	Item
	X	X	X	X	X	X	X	X	X	X	Required	Item_structure
X	X	X	X	X	X	X	X	X	X	X	Required	Ship
X	X	X	X	X	X	X	X	X	X	X	Required	Versionable_object
X	X	X	X	X	X	X	X	X	X	X	Required	Principal_characteristics
X	X	X	X	X	X	X	X	X	X	X	Required	Centre_location
X	X	X	X	X	X	X	X	X	X	X	Required	Derived_unit
X	X	X	X	X	X	X	X	X	X	X	Required	Named_unit
X	X	X	X	X	X	X	X	X	X	X	Required	Precision
X	X	X	X	X	X	X	X	X	X	X	Required	Space
X											Requirement	Class_bulk_load_requirement_d efinition
X	X										Requirement	Class_compartment_requiremen t_definition
X											Requirement	Class_deck_load_requirement_ definition
X	X										Requirement	Class_tank_requirement_definiti on
X	X										Requirement	Compartment_design_requirem ent
X											Requirement	Vehicle_load_description
X	X										Requirement	Design_requirement

TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10	TC11	Usage	Entity
X											Ship Char	Carrier
											Ship Char	Class_and_statutory_designation
											Ship Char	Class_notation
											Ship Char	Class_parameters
											Ship Char	Freeboard_characteristics
											Ship Char	Lightship_definition
											Ship Char	Lightship_weight_item
											Ship Char	Loadline
X	X	X	X	X	X	X	X	X	X	X	Ship Char	Navy_ship
											Ship Char	Owner_designation
											Ship Char	Regulation
											Ship Char	Research_ship
											Ship Char	Ship_designation
X	X	X	X	X	X	X	X	X	X	X	Ship Char	Shiptype
											Ship Char	Shipyard_designation
											Ship Char	Working_ship

4.4.5.2 Entities Excluded by the Test Cases

The test cases do not include entities from the following Units of Functionality (UoF) in AP215:

- Cargoes,
- Coatings,
- Configuration Management,
- Damaged_stability,
- Hull_class_applicability,
- Loading_conditions,
- Location_concepts (except global_axis_placement),
- Tonnage,
- Weights.

4.5 Conclusions

The Ship Arrangements Task demonstrated the successful implementation of STEP AP215 for the transfer of compartment and ship arrangements data among various CAD systems in use in the U.S. shipbuilding industry. It validated the application protocol and developed a set of implementation agreements and proposed enhancements to the ISO Standard to facilitate the success of these exchanges (see Appendix A).

Among the important aspects demonstrated in the ship arrangements transfer was the exchange of data applying to various phases of the ship life cycle from preliminary design through detailed design and on to manufacture and life cycle support. The interrelationship of the shipbuilding APs was also demonstrated as the AP215 exchange referenced moulded forms transferred previously using AP216.

5 STEEL PROCESSING TASK

5.1 Problems Addressed

Shipbuilding today demands flexibility and agility in design and production. Teaming agreements and out sourcing have become increasingly common. However, sharing work between shipyards has remained problematic. Exchanging work packages among yards involves more than addressing data translation issues. Disparate manufacturing processes and tools along with unique yard capabilities and capacities create challenges for work sharing among yards when information originating from a lead yard is passed on to a follower yard. In addition, manufacturing rules and processes may be undocumented, embedded in the logic of software applications, or confined to the minds of a relatively small set of 'subject matter experts'. When manufacturing data is exported to a yard's production system it still requires additional information and modification to conform to the processes employed within the yard. Manufacturing processes specified in work packages, originating from a lead yard, often need to be backed out of the work package and re-engineered for the processes of the follower yard. This significant non-valued added effort makes multi-yard work share inefficient.

The research conducted for the Steel Processing Task addressed these issues by creating a system that captures manufacturing processes as a set of rules and applies rules technology to execute these rules against a given manufacturing product model. Rules sets contain all of the manufacturing logic required to ensure that a given manufacturing data set conforms to the requirements of a specific environment. Since the rules are maintained separate from the software system, they can be easily modified/updated independent of the system to support process changes and new equipment specifications.

5.2 Background

The Steel Processing Task continues the work accomplished by the NSRP Integrated Steel Processing Environment (ISPE) Project. That project's final deliverable included a consolidated set of steel manufacturing processing and data requirements for 4 major US shipyards and included a concept of operations for a system that addressed the same goals pursued by this effort; namely to:

- Follow the guidance of the NSRP Strategic Investment Plan (Reference 9) and develop an industry-wide technology that uncouples Computer Aided Manufacturing (CAM) from Computer Aided Design (CAD) platforms while maintaining interoperability
- Build on the research and standards development investments already made by the NSRP
- Stabilize the critical interface between the CAD systems and the shipyard manufacturing environment
- Provide a cost and resource-saving technology for the shipbuilding industry that enables work-sharing among US shipyards

The Steel Processing pilot software discussed within this document is a prototype for the system conceptualized as part of the ISPE Project. It is thus a burgeoning program, in its early exploratory phases. Resource restrictions for this task limited the development of functionality for the pilot system

although concepts related to the development and processing of manufacturing rules were explored. Future development on this program will explore additional functional areas addressed by ISPE.

5.3 System Description

5.3.1 Context

The Steel Processing System will be used as part of Manufacturing Engineering providing support for various functions within the Hull Engineering and Lofting disciplines; the primary function being the creation of Manufacturing Product Model data. The initial implementation of the Steel Processing System will be a pilot/prototype, meant to prove the concept of applying externally defined business logic to standards-based design data to yield a product model that conforms to the manufacturing processes the data is intended to support. The resulting manufacturing product model could thus be tailored to support different requirements simply be means of modifying the applied business logic. The approach described in the sections that follow use a Rules-Based system to provide the framework for managing and executing manufacturing rules.

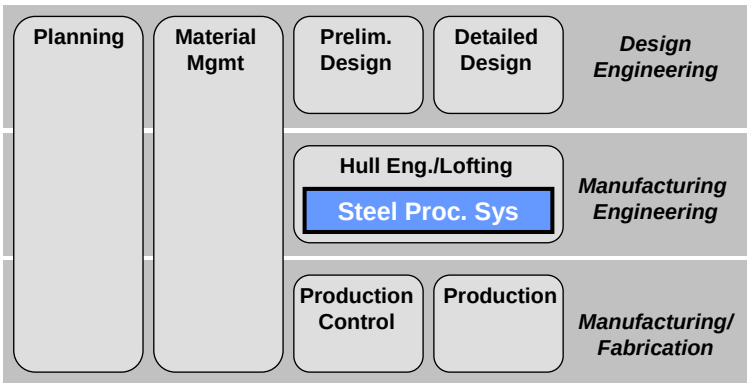


Figure 25: Ship Design/Manufacturing Processes

There are two primary types of user functions for the Steel Processing system: Rules Developer and Systems Analyst.

The *Rules Developer* is responsible for creating the rule sets that the Steel Processing system will process. They must possess an in-depth knowledge of the format of the data files used and of the manufacturing processes that are to be expressed in the rules language. Knowledge of the data file, and associated knowledge base maintained the Steel Processing system (see Section 5.3.4) is necessary to identify patterns on which the rules will execute. The business logic associated with each pattern defined in the rule determines what the system should do when the pattern is encountered. This functionality can be expressed in the rules language or in custom software modules written in other software libraries.

The *Systems Analyst* is responsible for using the Steel Processing application. Note that the current version of this software is in its prototype stage with limited functionality. An overview of the proposed complete suite of functionality is defined in Section 4 of the ISPE Final report (refer to Reference 10). The System Analyst would make the decision on which rule sets to run. There will be one or more rule sets for each function of the system (see Section 5.3.2).

5.3.2 Data Flow View

For the purposes of the Steel Processing Task, the ship manufacturing process is defined as a series of high-level operational scenarios. Operational scenarios decompose the lifecycle of ship design and manufacturing engineering processes into discrete functions and characterize the various stages of the information model as it matures through the application of these functions. Section 4 of the Steel Processing Requirements document (Reference 11) provides more information on the operational scenarios as they were originally envisioned for the Steel Processing system. Figure 26 shows the development of a ship product/information model as it progresses from preliminary design through manufacturing. The scope of the Steel Processing system, as it would exist in its fully-developed state, encompasses the Production Engineering, Piece Part Development, and Manufacturing Definition functions.

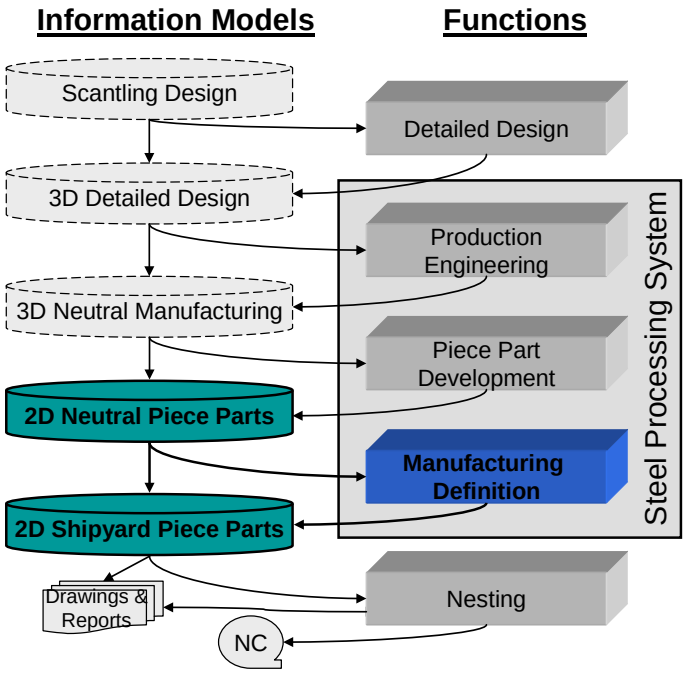


Figure 26: Steel Processing Lifecycle

The Steel Processing System pilot includes functionality to support the Manufacturing Definition function. The Manufacturing Definition function entails generating all of the yard-specific information necessary to direct/instruct steel production processes in the construction of a ship. The nominal 2D Neutral Piece Part model will contain all flattened (unwrapped) parts with bevel and weld information, part relationships, edge treatments, and assembly information. The nominal 2D Shipyard Piece Part model will define all added/removed material needed for fabrication including assembly information, tabs, labels, marking lines, etc. The data contained in this model is defined so that it conforms to the processes of a specific yard/shop. The test data set defined for the Steel Processing pilot application contains a subset of the data included in the 2D Shipyard Piece Parts information model.

5.3.2.1 Context Schemas

Aligning with ISE information interoperability architecture, each specific interchange specification (as depicted in Figure 26) is formally documented by means of a context schema. The context schema

provides the basis for the multitude of generated information work products, including the XML schemas for validating STEP AP218 exchanges and the mediators for translation among the various representations and encodings of the data.

The information use cases are described in Section 4 of the Steel Processing Requirements Document (Reference 11). The following describes the location for the context schemas to support the information models described above:

3D Detailed Design - http://www.isetools.org/xml/metadata/contexts/edo218-feat/edo218-feat.xml

3D Neutral Mfg Model - http://www.isetools.org/xml/metadata/contexts/edo218-cam/edo218-cam.xml

2D Neutral Piece Parts - http://www.isetools.org/xml/metadata/contexts/edo218-cam2d/edo218-cam2d.xml

2D Shipyard Piece Parts - http://www.isetools.org/xml/metadata/contexts/edo218-loft/edo218-loft.xml

5.3.3 User interface

This section provides an overview of the content and use of the Steel Processing system pilot application. Comprehensive user documentation for the system is beyond the scope of this document. Nonetheless, the description provided here will give the reader a good understanding of the system's functionality.

The design of the Steel Processing system interface serves two purposes: to display data set contents (both geometry and text attributes) and interface with the Rules Engine. The graphical user interface (GUI) is thus organized into two main sections: data view interface and rules interface (refer to Figure 27).

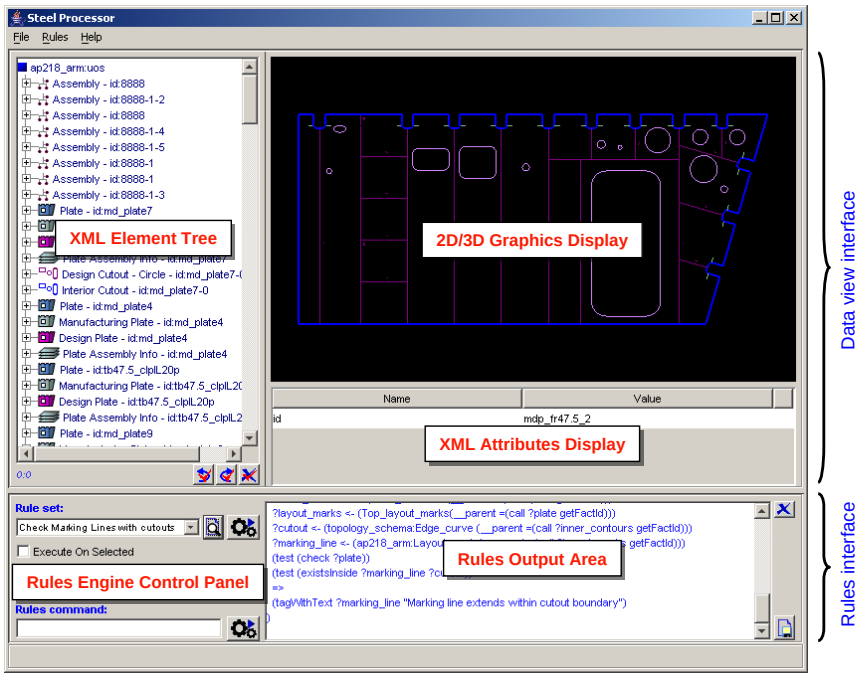


Figure 27: Steel Processing System User Interface

The data view interface enables the user to view the contents of the data set in text format as well as any associated graphical content. The data view interface includes the XML Element, XML Attribute, and 2D/3D graphics displays. The XML Element tree provides a means of navigating the hierarchical/tree structure of the data set. Elements selected in the tree have their attributes displayed in the XML Attribute display area. If there is graphical content at the selected element level or any child in its hierarchy, then that will be displayed in the 2D/3D Graphics display. Figure 27 shows the `ap218_arm:Plate_manufacturing_definition` element selected. The geometry associated with the `Outer_contour`, `Inner_contours` and `Top_layout_marks` child elements is displayed in the 2D/3D Graphics view panel. The element attribute – `id` – associated with the `ap218_arm:Plate_manufacturing_definition` element is displayed in the XML Attribute display.

The rules interface enables a user to identify a rule set, execute it, and review any output produced by the processing run. The selection of rule sets is determined by the system's configuration file. User's can restrict the application of a rule set to the selected elements in the data view interface or apply them to the entire data set. Any messages produced from executing a rule set are displayed in the Rules Output view panel.

Each of these user interface components, along with the system's configuration settings, is explained in detail in the following sections.

5.3.3.1 Interface Components

5.3.3.1.1 Configuration Settings

The configuration settings for the Steel Processing system allow several areas of the interface and application functionality to be customized. These include the icons associated with each element, the color of the geometry associated with an element as rendered in the 2D/3D, the rule sets available to the user, and other, general system variables. Configuration parameters are stored in a separate file, the location of which is specified as part of the command-line parameters of the application. The configuration parameters for the displays of element data and the rule sets are displayed below.

```
<ELEMENT name="Top_layout_marks"
         icon="marking_line_icon.gif"
         geomColor="128,192,128"
         overrideChildrenColor="true" />


<RULESET name="Thickness Throw"
         file="c:\myrules\thickness_throw.clp"
         description="Adds thickness throw indicators" />
```

The `ELEMENT` tag specifies the configuration parameters for the display of the XML element in the XML element tree display and the attributes for the corresponding graphical components in the 2D/3D graphics display. The 'name' attribute maps the parameters for this tag with a specific XML element. The 'icon' attribute identifies the name of an icon file to use next to the element in the tree display. All

icon files should be stored with the class files in the `com\ng\steel\images` directory. The 'geomColor' attribute is optional and specifies the red, green, and blue components of the color for all geometry associated with the corresponding XML element. Valid values are from 0 – 255 for each component. The 'overrideChildrenColor' attribute is also optional and specifies whether or not the color specified in this tag should be applied to all child elements. If this attribute is included, the 'geomColor' attribute should be included as well.

The RULESET tag specifies the configuration parameters for a rule set. Each rule set tag included in the configuration file will have a corresponding element in the rule set selection list in the rules interface control panel. The 'name' attribute is used to identify the rule set. The value supplied here will be used in the interface list. The 'file' attribute identifies the rule set file. This should be a full path and file name to the rule set file. The 'description attribute' provides a brief description of what the rule set does. It is displayed in the interface once the rule set list element is selected. The length of this value should be about 40 characters or less so as to fit in the space allocated in the interface.

One final note about the configuration parameters – there is no schema validation applied to this file. Users are responsible for applying valid information for each parameter.

5.3.3.1.2 Graphics display

The graphics display panel is used to render any geometry associated with the elements selected in the XML Element Tree list. It is used for viewing only. All child nodes, containing geometry data, associated with a selected element are displayed in the scene. If the user selects the top-most root node, then all geometry will be display in the graphics display. The rendering characteristics for geometry within a node or node hierarchy are controlled by settings in the system's configuration file (see Section 5.3.3.1.1). Geometry, associated with elements that have been tagged, will be highlighted in yellow.

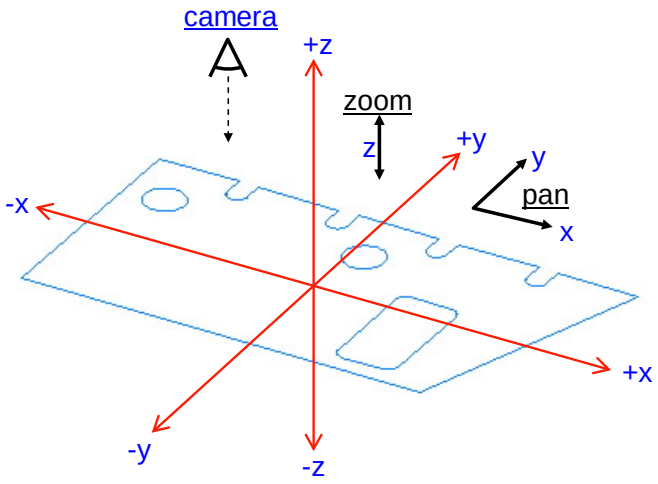


Figure 28: Graphics Environment

Graphical elements within the view panel can be selected to reveal the XML element associated with the selected graphic. Users can place the mouse cursor on a graphic element, such as a line, and select it

using the left mouse button. Once selected, a pop-up menu will be briefly displayed next to the selected area displaying the name of the associated XML element. If this popup menu is then selected with the left mouse button, the XML Element Tree will automatically scroll to the corresponding XML element, which will be briefly highlighted with the icon ( ).

Figure 28 shows the orientation of the virtual camera and the graphical components rendered in the 3D scene with respect to the primary axes. Although the interface restricts the view to only one direction, the graphical components rendered in the scene exist in a three dimensional space. Users are allowed to pan and zoom but rotation along any axis is not allowed. Zoom refers to the position of the virtual camera along the z axis. The resulting visual effect is the enlarging and collapsing of the scene. All views are fixed in the negative z direction. Panning refers to the movement of the camera along the x and y axis. The resulting visual effect with panning is the movement of the rendered geometry with respect to the window. Prior to any zooming or panning, the initial scene will be established such that the complete set of geometry will be visible and centered. That is, the virtual camera will be positioned at a point in the middle of all geometry being rendered and set at a distance (zoom) that allows the display of all the geometry in the scene. The data set used for the initial implementation of the system exists in 2D only; the z component of all geometry is 0.0.

There are two types of graphics components that are rendered: polylines and text. Polylines will show the outline of all plates, interior cutouts, and marking lines. Text and symbols will be used to show labels. Arc geometry is generated based on a given set of three coordinates depicting the start, middle, and end points on the curve. All arcs are circular and the number of points rendered along the arc is a function of the arc's radius.

5.3.3.1.3 Element Tree Display

The XML Element view panel displays the contents of the input data set. The tree-like interface is set up to display the top level elements once the data set is loaded. Child branches can be expanded by selecting the + next to the element in the tree. The tree element is labeled from the corresponding XML element's type.

The icon used in the tree for each element is set based on whether the element has children and whether a separate icon is set in the system's configuration file.[1] By default, elements that have children have a ● icon. Elements that have children and that currently have their children elements exposed have a ■ icon. Finally, elements that have no children - leaf elements - have a ○ icon. These icons are overridden if there is an icon specified in the system's configuration (see Section 5.3.3.1.1). Also, tagged icons are rendered differently as well.

Figure 29: Tagged Selection Interface

The XML Element display has an interface that allows the user to traverse elements that have been tagged. All tagged elements are referenced internally by a separate list that gets generated each time a rule set is executed. The interface has two parts. The numbers on the left indicate the index of the currently selected tagged item and the total number of elements that are tagged. The buttons on the right allow the user to traverse through the tagged items in either direction. Each time the next or previous

[1] All icon images are stored, and referenced from, the images directory within the class directory path for the application – `com\ng\steel\images`.

tagged element is selected, the tree view will dynamically expand and scroll to the selected element. The Graphics and Attribute displays will update accordingly. The last button on the right removes the tagged state from all elements.

5.3.3.1.4 Attribute Display

The XML Attribute view panel contains a single table interface. The contents of the table are updated each time an element is selected in the XML element tree. Attributes are simply name/value pairs and may include the content (data) if the corresponding element contains it. They are displayed on separate rows of the table. Element content is distinguished from attribute data by the name '`Element Content`', which gets displayed in the name column of the table. If there are a sufficient number of attributes, a vertical scroll bar will appear. Columns within the table can be resized by moving the bar to the right of the name column header cell.

5.3.3.1.5 Rules interface

The Rules Engine interface panel allows the user to identify a rule set, as specified in the system configuration (see Section 5.3.3.1.1), display it, and execute it. All rule sets that have been configured to run with the system will have a separate item in the drop-down list. The name displayed in the list is specified in the configuration for the rule. There is also a brief description of the rule displayed, in blue, within the interface. The item displayed in the list is the currently selected rule set. Selecting the display rule icon ( ) will display the contents of the rules file, associated with the selected rule set, in the rules output window. Selecting the execute icon ( ) will execute the current rule set. For the most part, the execution of a rule set is completely dependent on the specific rules within the rule set. However, the 'Execute on Selected' check box will set a state flag in the rules environment that can limit the subset of facts that the rules will process. When selected, only the facts associated with the selected tree of elements will be processed.[2] Finally, the text input box and execute icon allow the user to type a single rules engine command and execute it. The output of this execution is displayed in the Rules output window.

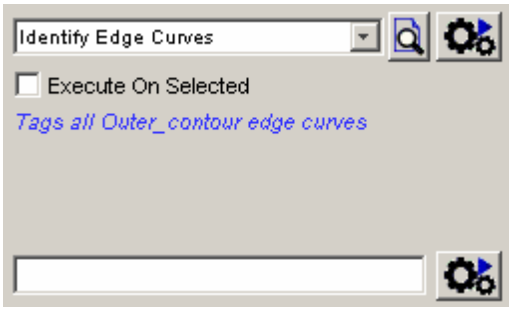


Figure 30: Rules Control Panel

5.3.3.1.6 Rules output

The rules output text area is used to display output from the rules engine from the execution of rule sets or individual commands. If the content being displayed is greater than the window provided, a scroll bar will be added to allow the user to scroll through the rest of the data. The size of the output window can

[2] This check box only sets/unsets a flag that is accessible from within the rule environment; rules must specifically check the flag and the 'selected' state for each fact.

be adjusted by moving the content split pane divider. The clear output button ( ) clears the contents of the output window when selected. Once erased, the contents can not be restored.

5.3.4 System Architecture

At the highest level the Steel Processing system will:

1. Ingest steel manufacturing data (defined primarily using STEP AP218),
2. Apply a set of rules and/or functions to this data, and
3. Produce a set of data that encompasses the logic of the applied rules.

The identification and description of the steps or sub-functions, along with the architectural components, required to transition from step 1 to step 3 is the subject of this section. The Execution View section describes the processes involved in implementing one or more Use Cases. The Data Management section describes the elements that are maintained within the system's knowledge base. Finally, the Rule Processing section describes the actions of the rules processor and how it acts on the knowledge base.

5.3.4.1 Execution View

The Steel Processing system will process manufacturing data at various stages in the lifecycle (refer to Section 5.3.2). At each stage, in-coming data is enriched by the application of externally-defined rules that define, refine, and/or validate this data and subsequently produce an exported data set that includes these enhancements. A different rule set is applied for each stage. Figure 31 shows the Steel Processing system and the high-level functions required to process data at each stage. An iteration of these functions is referred to herein as a 'processing run'. A processing run consists of preprocessing the data, applying the rules and exporting the resulting data.

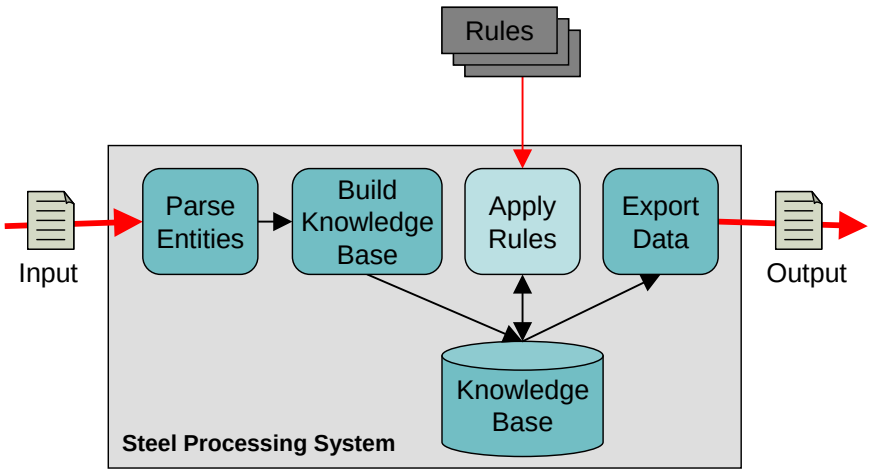


Figure 31: Execution View

5.3.4.1.1 Preprocessing

Preprocessing refers to the process of reading in and storing the incoming data set. The incoming data set is formatted using STEP Part 28, which is an XML implementation. Having the data formatted as

XML makes the parsing and interpretation of the data easier since the system can rely on publicly available XML parsers and schema validators to read in, and ensure the integrity of, the data. The process of parsing and filtering the entities involves reading the input data set, validating its content, and ignoring all entities that don't apply to the applied rules. The filtering process may be necessary in anticipation of the memory requirements traditionally required by rule-based systems. Separately defined Filter Criteria will identify the appropriate XML entities to be extracted and stored in the Knowledge Base.

All data, on which the rules will apply, must be formatted and stored such that the data can be accessed by the Rules Engine. The *Build Knowledge Base* process formats the entities parsed from the input data set and stores it into the Knowledge Base. The configured rule sets will act on and update the Knowledge base as the rules are exercised. The Knowledge Base therefore contains the working memory for the system. Currently, the Knowledge Base exists entirely in memory; there is no current plan on integrating a database management system.

5.3.4.1.2 Applying the Rules

The application of manufacturing rules is the most complex and intensive processing in the Steel Processing system. The Steel Processing System will incorporate a commercially-available Rules Engine into the architecture[3]. Rules Engines are domain independent. That is, a rules engine does not encode functionality that is specific to any given domain. The decision to use a Rules Engine is reflected in the basic premise behind the inception of the Steel Processing System; separating the processing requirements (rules) from the underlying data. Using rule-based systems enables the definition of an underlying business logic that is separate from the system's mechanization. This separation contrasts with most software applications where logic is embedded in the software code. Making modifications to such systems involve modifying the system's source code and building a separate system release. Using a rule-based system, only the rules database is modified to make such a change.

The rules for the Steel Processing System will be written in an ASCII-based language and stored in separate files for each rule set. Given that these rules instruct the system on how to implement the manufacturing data, the system can scale and/or be modified easily by editing the existing rule sets or by defining new rules. Separate rule sets will be defined for each target yard and stage of processing. Thus any instantiation of the process specified in Figure 31 will act on a specific rule set.

At the most basic level, rules are a set of 'If/Then' statements where the 'If' part of the statement associates with a pattern of data. The Rule Engine will access data stored in the Knowledge Base. The 'Then' part of the statement instructs the system on what to do for each pattern found in the input data. Some of the logic for instructing the rules engine on how to process this data can be defined as part of the rule itself. Geometry-related parsing and/or interpreting functionality is/are beyond the scope of the rule language. The Rule Engine invokes custom software modules to process this type of data. All data processed by the Rules Engine is stored back into the Knowledge Base.

5.3.4.1.3 Exporting the Data

After the rules have been processed, the data within the Knowledge Base is extracted and reconstituted back into the STEP format. All additional and/or modified data, as defined by the rule application process, will be included in this data.

[3] The Steel Processing System will use JESS, a rule engine developed for the Java platform. Refer to http://herzberg.ca.sandia.gov/jess/index.shtml.

5.3.4.2 Data Management

The Data Management module includes functionality to parse and reconstitute STEP data, validate this data, and manage the Knowledge Base.

All STEP data is assumed to be formatted in XML. Given this assumption, the STEP Data Manager incorporates an XML parser into the architecture. The XML parser will be used to read in and store the STEP AP218 entities (elements) and attributes. All STEP entities are stored as objects (Java Beans). Java Beans are simple Java objects that adhere to an interface specification for access to the object's content. All Java Beans stored in the Knowledge Base have a name attribute and a unique identifier, which will identify the type of STEP entity the bean pertains to. There is also additional information to allow the STEP Data Manager to rebuild the STEP data for export once the rules have been applied. The entities included in the input data set will contain more information than will be processed by the system. The Steel Processing rules will operate on a subset of this data. Refer to the Requirements Document, Section 4.2 (Reference 11) for a description of the STEP AP218 entities that pertain this system.

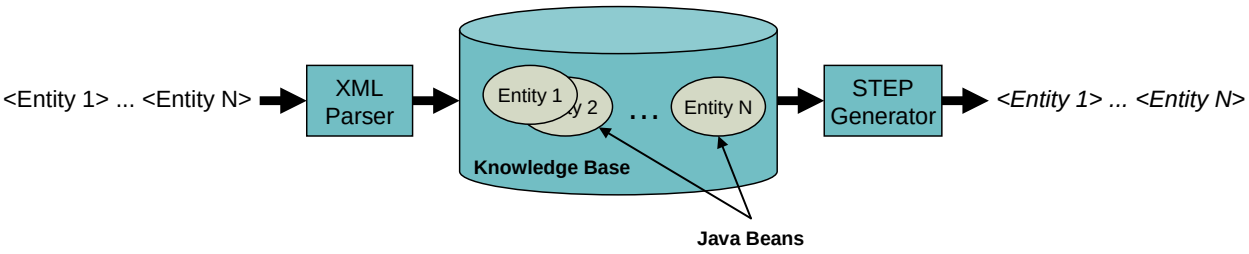


Figure 32: Knowledge Base

Once the rules have been executed the STEP Generator will extract the information in the Knowledge Base and rebuild the STEP file with all original, modified, and added entities included. The reconstructed STEP file will conform to the appropriate information model schema (refer to Section 5.3.2).

5.3.4.3 Rules Processing Module

The Rules Processor module sets up and executes a set of rules. All domain-specific functionality associated with ship steel manufacturing will be embedded in this module. It includes three main components: Rules Engine Interface, Rules Processor, and Custom Rule Modules.

5.3.4.3.1 Rules Engine Interface

The Rules Engine Interface consists of a set of generic interface classes used for the User Interface modules to interact with the Rules Engine. All interaction with the Rules Processor will be handled using this interface. The primary purpose for having this layer is that the Rules Engine can be exchanged for a different version or implementation and the User Interface does not have to change.

5.3.4.3.2 Rules Processor

The Rules Processor component is the heart of the Steel Processing System. It interfaces directly with

the Rules Engine using the Rules Engine API.[4] It's mode of functionality is thus based directly on the specific Rules Engine integrated with the system. At a high level, the Rules engine will require some form of initialization, an impetus to begin the execution of the rules, a mechanism to handle anomalies, and a method of retrieving the results.

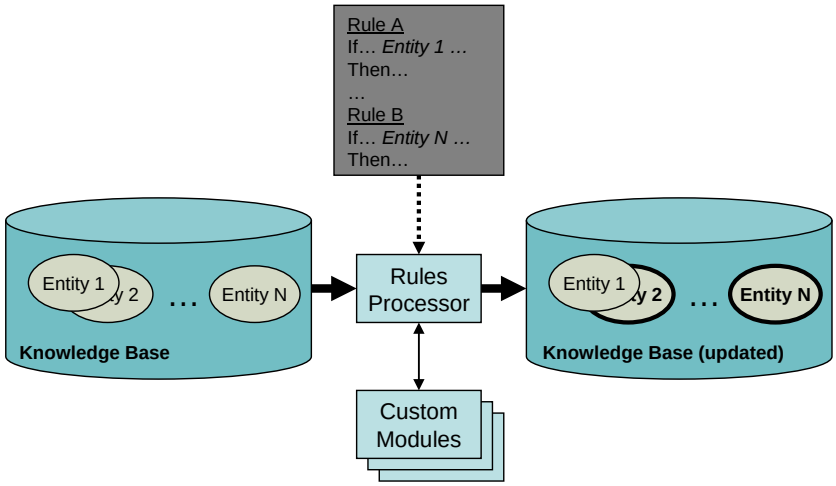


Figure 33: Rules Processing

Initialization consists of identifying and loading the appropriate rule set(s), initializing runtime variables to default values or those chosen from the system's configuration or user interface, and loading the working memory. The rule sets will be defined separately in ASCII text files. The language used to construct the rules may be specific to the integrated Rules Engine. Some rules engines define the rules using XML. Regardless, the schema/language for the rules will depend on the implementation. The initialization of the rules engine is commanded from the system's user interface (see Section 5.3.3.1.5). The user interface includes provisions for specifying values to configure the rules engine.[5] The knowledge base is the space that the Rule Engine uses to query the information that the rules depend on and store the results of the applied rules. The entities stored in the Knowledge Base represent the objects (elements) on which the rules will execute.

The User Interface controls when the rules engine should begin its processing. Once the process is started, the rules engine will create an *agenda* that identifies the specific rules to execute. As the rules execute the knowledge base is updated. The structure of the rules and the result of their execution determine the set of additional rules to execute. It is likely that as the rules are executing there may be isolated errors or anomalies detected that require user input.

5.3.4.4.3.3 Custom Rules Modules

The functionality required for implementing the rules cannot exist exclusively in the rule set. Although the syntax for defining the rules contains all the constructs for defining the necessary logic, there are no mechanisms for handling manufacturing-specific data. For example, the bulk of the data for any

[4] See the Java Specification Request (JSR) for rule engine APIs (JSR94) - http://jcp.org/aboutJava/communityprocess/review/jsr094

[5] The Detailed Design phase will identify the configuration options for the system. System configuration may be implemented using separate configuration files and/or user interface-driven.

information model processed by the system consists of geometry, which defines the shape of the steel parts. Depending on the nature of the rules, custom rule modules will be developed to process the geometry and provide information for the rules to process. A custom rule module may be developed, for example, to calculate the surface area of a steel plate entity or the center of gravity for a collection of parts. The Rule Engine integrated into the pilot system includes ability for interfacing with external software or defining custom software as a means of tailoring the rules engine. Custom Rules Modules may also be used for preprocessing, adding content to the Knowledge Base based on the logic defined by the rule set, and post processing the data once the execution of the rules has completed.

5.3.4.4 Rules Engine

The rules engine module contains the integrated COTS rules engine. The Steel Processing pilot application will use the Jess rules engine. Jess is a Java-based rules engine developed by Sandia National Laboratories. It is available, for a fee, through that organization. More information on this application can be found at http://herzberg.ca.sandia.gov/jess/index.shtml.

The decision to use Jess was based on the following:

1. Jess is an object-oriented system with a Java API

   One of the design criteria for the Steel Processing System is to use Java as the development platform. The functionality of Jess can be extended through the system's API.

2. Jess supports XML as a medium for specifying rules

3. Jess is a mature product

   The current version of Jess is 7. It has been in development since the late 1990s. Working with a system that has been around for several years and one that has a fairly broad and extensive user base helps mitigate risks associated with integrating external software packages.

4. There is sufficient documentation available for the product

   The product comes with a full set of documentation. A description and use of the system is also published in a book specifically written for this package (Reference 12).

5.4 Test Data Set and Final Schemas

5.4.1 Overview

This section provides a description of the data that was used to develop and demonstrate the Steel Processing system pilot application. A graphical schema is provided to show the XML elements used to build the data set and how they are associated with the entities that are a part of the STEP AP218 specification. An explanation the rule sets developed is also provided.

5.4.2 Test Data Set

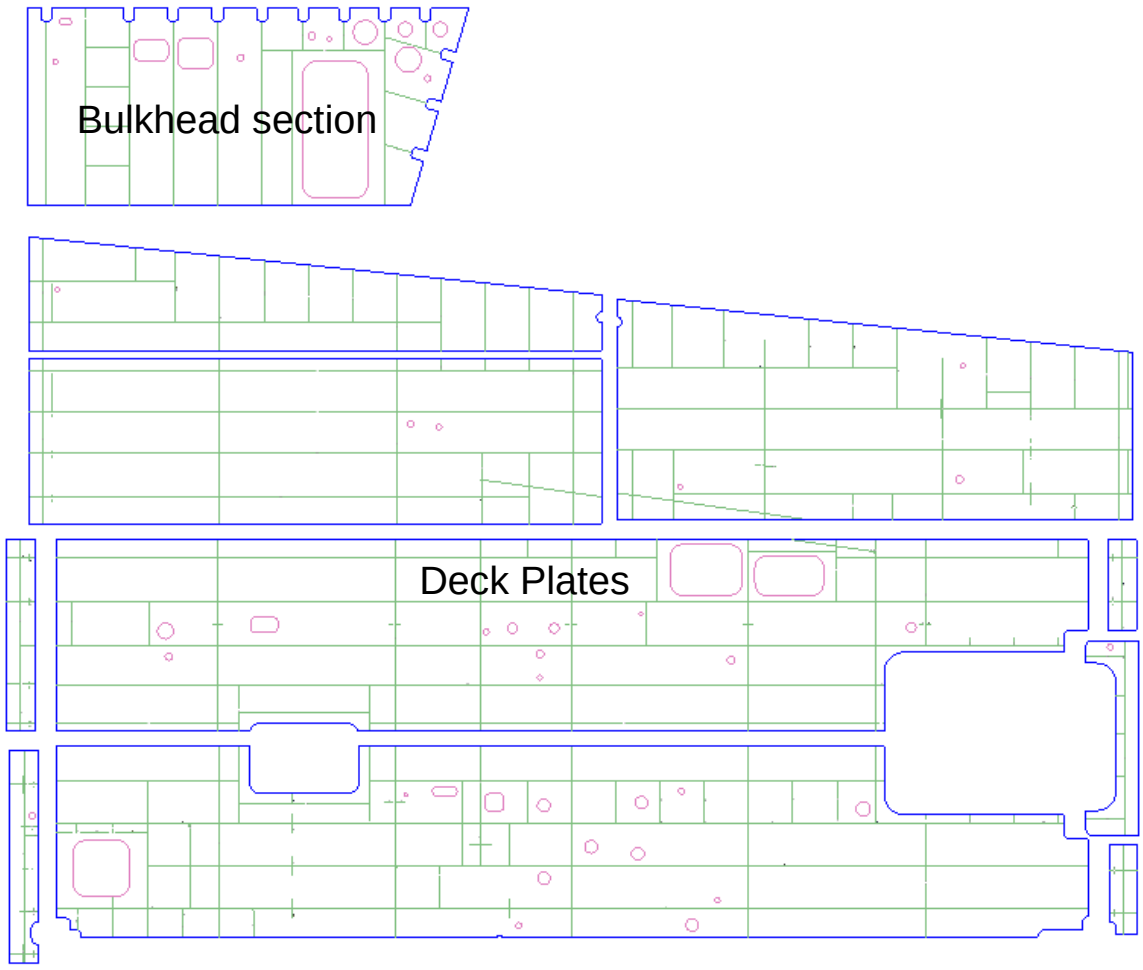


Figure 34: Test Data Set

The test data set used for building and demonstrating the Steel Processing application consists of a set of deck plates and a single bulkhead plate, each with various edge cutouts, inner cutouts, and marking lines. The content of the test data is meant to be representative of actual parts for a unit of a ship. Note that an actual data set for a unit would consist of much more information. For the purposes of the final demonstration, the test data is referenced as a TWR modification to be in-keeping with the content and flow of the demonstration. In reality however, the content of this data is more indicative of a much larger vessel.

Figure 35 shows the top level STEP AP218 entities used in the test data file. Element attributes and lower level aggregate entities were left out for clarity. Although the test data set contains references to structural profile parts only plate parts contain geometry. The structure of the parts is defined by the *assembly* entities. Assemblies contain other assemblies (sub-assemblies) and/or one or more *plates* or *profiles*. *assembly_manufacturing_part_definition* entities contain process-related information for each plate part (refer to Section 5.5.2.1). *structural_part_design_definition* entities include material characteristics for each plate. Note that the detailed design geometry has been left out of this test data set since it was not used for the processing of the rules created for the final demonstration.

structural_part_manufacturing_definition entities contain the bulk of the test data and are the primary source for the rule sets generated for the final demonstration. Each *structural_part_manufacturing_definition* contains the geometry for the plate's outer contour and, optionally, the geometry for the inner contours, layout marks, and annotations. Refer to Section 5.5.2.2 for a description of how the STEP AP218 entities for layout marks and annotations were altered for this project. Finally, interior cutout features and design definitions (*interior_cutout* and *interior_cutout_design_definition* respectively) were added to support one of the demonstration rule sets created.

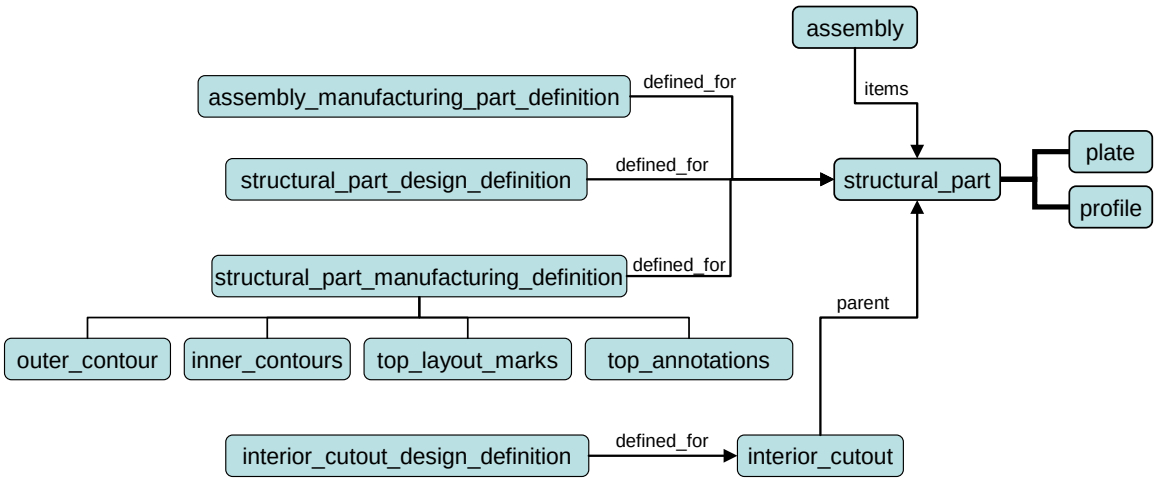


Figure 35: Top Level Test Data Entities

The following table lists the types of entities that were added to the STEP AP218 data file and were deemed necessary in order to represent a complete manufacturing work package.

Table 3 – Steel Processing Entities Added to AP218 Data Set

Entity	Description	Remarks
Piece number	Every part must have an identification number. The number is unique within a "unit" of work except for standard and like parts (where the areas are equal). - standard parts like clips, collars, chocks - like parts like brackets, non standard chocks	An attribute to include this information will be included in the added STEP AP218 entity – *assembly_manufacturing_part_definition*.
Remarks	Instructions to the shop or field to define a process or to communicate an instruction.	An attribute to include this information will be added to the current STEP AP218 entity – *assembly_manufacturing_definition*.
Drawing number	The zone number on the drawing where the part is detailed.	This information will be captured in the current STEP entity – *document_reference* with an added entity to capture the drawing detail number.
Drawing revision		
Drawing detail number	The zone number on the drawing where the part is detailed.	

Work order number	The production number used to track progress and manhour expenditures.	A *document_reference* attribute will be added to the current STEP AP218 entity - *assembly_manufacturing_definition* - *Source_type* – 'Work Order' - *Title* – work order # - *Author* – ship yard
Shop/ Construction area	Physical area where the part will be assembled.	This will be a link to a separate document that will contain information for a specific shop or construction area.
Feeds	The next higher assembly in the tree, to which the part or assembly will be sent when complete.	This is a link to the next work instruction that the part assembly will be part of.
Nest tape number	The tape number for the nest that contains the part.	A *document_reference* attribute will be included in the added STEP AP218 entity - *assembly_manufacturing_part_definition*. The Nest date will be captured using an added *date* attribute and 'Nest Tape' will be included as an enumeration for the document type.
Nest date	The date the part was nested.	
Grain requirement	If the plate grain is critical a note defining the requirements is given here.	Described as part of the description attribute for the *ship_material_property* entity
Issued date	The date the drawing for the part is issued.	An '*issued_date*' will be added to the Document entity
Manufacturing process	The detailed manufacturing process to be used on this part.	Captured as part or the *assembly_method* attribute of *assembly_manufacturing_definition*.
Catalog part number	The catalog part number for the raw material in the material system.	An attribute, for designating this, will be included to the added STEP AP218 entity – *assembly_manufacturing_part_definition*.
Machine name	The name of the burning machine.	An attribute to include this information will be added to the current STEP AP218 entity – *assembly_manufacturing_definition*.
Remnant	Denotes the utilization of a remnant piece instead of a whole plate.	A Boolean type attribute to include this information will be included in the added STEP AP218 entity – *assembly_manufacturing_part_definition*.
Part area	The area of the part in the previously defined system of measure.	A *part_area* attribute to include this information will be included in the added STEP AP218 entity – *assembly_manufacturing_part_definition*.
Stiffener plot number	The number of the stiffener sketch that defines the stiffening member located on the part.	Captured as an attribute to the added STEP AP218 entity – *assembly_manufacturing_profile_definition*.
Assembly information	Assembly hierarchy for parts	Entities within the current STEP AP218 specification used for assembly information have been augmented to capture some of the information in this table (see Section 5.5.2.1)

The rule sets developed for the demonstration were based on the content of the data. Therefore all rules pertain to steel plates, their cutouts, size and weight attributes, and the relationships between them. In general, rules developed for this system are in one of two categories: data verification and data creation.

Data verification rules are used to check the data for conformance to a set of yard-specific processes, geometry correctness (e.g., closed parts), consistency, and produce-ability. Examples of data verification rules include:

- Data integrity

  Rules that verify the content, structure, and various associations within a data file

- Annotation content and positioning

  Rules that verify the existence, content, and location of annotations and manufacturing notes/descriptions in a work package

- Geometry correctness

  Rules that verify the proper construction of geometric information that defines the physical shape and position of manufacturable parts and there associated manufacturing features

- Structural part relationships

  Rules that verify the existence and content of relationships among various entities in the data file

- Correlation of processes applied to structural data

  Rules that verify the existence and proper definition of manufacturing processes defined in the work package

- End cuts and edge preparations

  Rules that verify the existence and content structural part features based on their association/connection with other structural parts

If rules can be developed to verify any given data set, then it is natural to surmise that rules can be created to generate data as well. Data creation rules are developed in conjunction with the processing logic used to generate structural data. The rule identifies the need for such information and can rapidly provide the proper context for the added manufacturing information. Examples of data creation rules are similar to data verification rules but also encompass data extractions for postprocessing applications such as reports, manufacturing aids, and nesting input.

The rule sets generated for the final demonstration of this project were of both categories. Given that the final submission of this work was a demonstration of the software and the perceived benefit of having graphical content in a demonstration, the entities generated by the demonstration rules contained graphical/geometric content. The demonstration used the Steel Processing system in a work sharing scenario. Since yards typically have their own symbology with respect to marking lines, rule sets were created to generate thickness throw symbols for two of the teammate yards. Executing the rule thus generated the appropriate marking line geometry and be highlighted in the system's graphic display window. Also in keeping with a work sharing scenario, verification rule sets were created to check the consistency of the geometry in the data set and compare process data to the manufacturing part data that it pertains to.

5.5 Conclusions

This section documents the results of various analysis conducted throughout the project. Section 5.5.1 describes how the project has met its original goals and objectives. Section 5.5.2 describes the changes made to the current STEP AP218 specification and recommends additional enhancements to support the concept of using the specification as a framework for steel production work packages.

5.5.1 Objectives/Resolutions

The following is a list of objectives that was developed for the Steel Processing Task as part of the project's first quarter milestone deliverable – Steel Processing Requirements. The text has been modified here to show how each one was addressed throughout the completion of the project.

- *Enable efficient partnering among yards* .

  The test data set generated for the Steel Processing pilot software is representative of a work package that could be routed internally or to another yard. Although the data set is not complete, in that it doesn't contain all of the manufacturing information that would be required, it is enough to demonstrate the capabilities of the system. Efficient work share is enabled through the definition of each rule set; the application of these rules ensures that the work package can be processed by the target yard obviating the need for manual checking and/or rework.

- *Define a yard-neutral manufacturing product model*

  At the project's inception the concept of a neutral manufacturing model was defined as being a single data specification, defined using non-vendor specific and industry accepted standards, and containing *placeholders* for each set of information pertaining to yard-specific practices. The information within these placeholders would then be filled in by the application of each rule set. The 'neutral' part of this concept came not from just using a neutral data format but by also segregating all information within this format that is subject to the requirements of a specific domain. The current STEP AP218 specification, championed by the ISE Consortium, provided the neutral data format.

  Although in general the original concept is still true, the neutral manufacturing model is less explicit than what was originally conceived. Instead of using the neutral manufacturing model as a template, placeholders are defined by the patterns associated with each rule. The Steel Processing system ensures that the data associated with these patterns is valid and contains provisions for modifying and/or adding data wherever required.

- *Define a framework for defining manufacturing processes along with methods of applying these processes to the neutral manufacturing product model*

  The Steel Processing pilot application demonstrates this framework. Although that system is explained in Section 5.3 it is summarized again here –

  a. Users export their ship design data using STEP AP218 using part 28.[6]

[6] Actually, the Steel Processing system does not require that the input data be defined using STEP AP218 or any STEP-related AP. It does require that the data be XML. Rule sets are written explicitly for the format of the data used.

b. Users generate rules, and associated custom data processing modules (if required), depending on the requirements of the data.

c. Users ingest the data into the Steel Processing system and execute the rule sets

d. Users then save data file

- *Define a rules-based framework to generate yard-specific manufacturing data*

  Manufacturing data pertaining to a specific yard is generated and/or validated based on the logic contained in the rule sets. Since this logic is defined separate from any application, end users are free to define their business rules as they directly pertain to the actual data.

- *Create several real-world work exchange scenarios and document how the system will address each of them*

  Exchange scenarios are defined in Section 5.3.2

- *Analyze the AP218 for its effectiveness in defining manufacturing processes*

  See sections below for an analysis of STEP AP218 and its use within the system.

5.5.2 STEP AP218 Enhancements/Recommendations

The sections that follow highlight several areas within the current STEP AP218 specification and suggest improvements to the schema. These suggested enhancements are based on the premise that the AP218 specification will be used to capture information for a complete work package; one that can be distributed to separate manufacturing locations. Such a view therefore requires the specification to contain enough information to allow a software system to analyze the contents of a work package, determine its suitability for a specific domain, and automate the manipulation and addition of manufacturing data, wherever required, in order to transform this work package.

5.5.2.1 Assembly Hierarchy

The Steel Processing system has been developed to process data related to steel production work packages. Although most of these data requirements are met by the current STEP AP218 specification there are certain required data elements, related to part fabrication and production instructions, that are not included. These have been identified in Table 3. Figure 36 shows the extensions made to the `assembly_manufacturing_definition` STEP AP218 entity to support this information.

Given the structure of AP218 with respect to assemblies and their relationship to parts and the design/manufacturing definition entities, extending the assembly manufacturing definition entity was the most appropriate way of including support the Steel Processing data requirements. These extensions involved adding attributes to the existing `assembly_manufacturing_definition` entity and extending it to provide a structure for more specific part-related processing data.

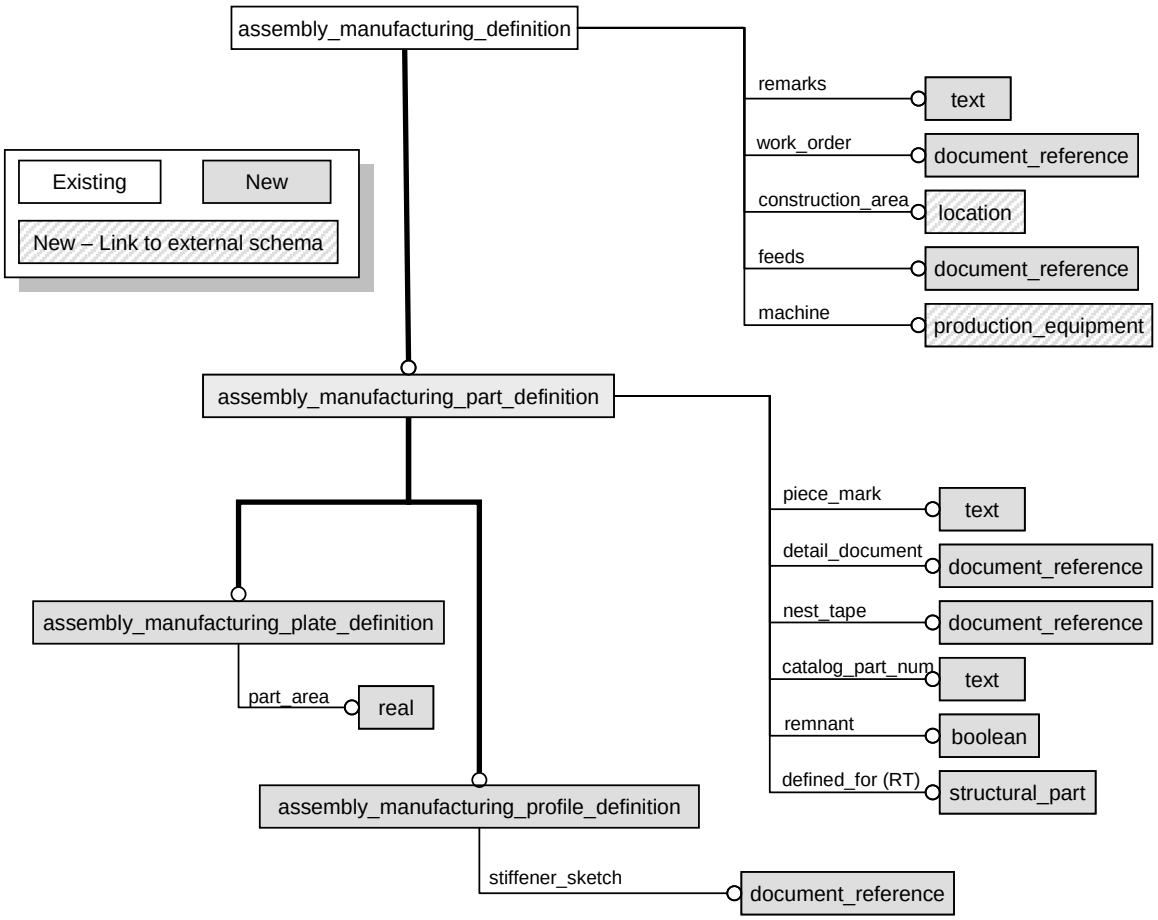


Figure 36: STEP AP218 Assembly Enhancements

assembly_manufacturing_definition was updated to include links to associated work orders, a text element to provide specific manufacturing remarks and links to the production area and equipment to be used during manufacturing. Section 5.5.2.3 provides a description of how manufacturing process-related data was included in the work package. *assembly_manufacturing_part_definition* is a derivative of *assembly_manufacturing_definition* and includes information for part-specific assembly information. This includes a unique id for all like parts (piece_mark), a link to the production detail drawings and nest tapes containing the part, the catalog part number for the raw material to be used for the part, and a distinction as to whether this material is a remnant. There's also a redefinition of the *defined_for* attribute, which is used to link the assembly to the structural part that it pertains to. *assembly_manufacturing_plate_definition* is a derivative of *assembly_manufacturing_part_definition* and includes information for plate-specific assembly information. In this case, there's only an attribute to capture the area of a given plate. Note that the test data set for the Steel Processing system consists only of plates. Therefore, this entity will be used for all elements. Finally, the *assembly_manufacturing_profile_definition* was created specifically for profile elements and is a derivative of *assembly_manufacturing_part_definition*. This entity adds an attribute for the stiffener sketch document that defines the stiffening member.

Figure 37 shows how the new assembly manufacturing definition entities will relate to parts and other assemblies. Note that this is a high-level figure with detail removed for clarity.

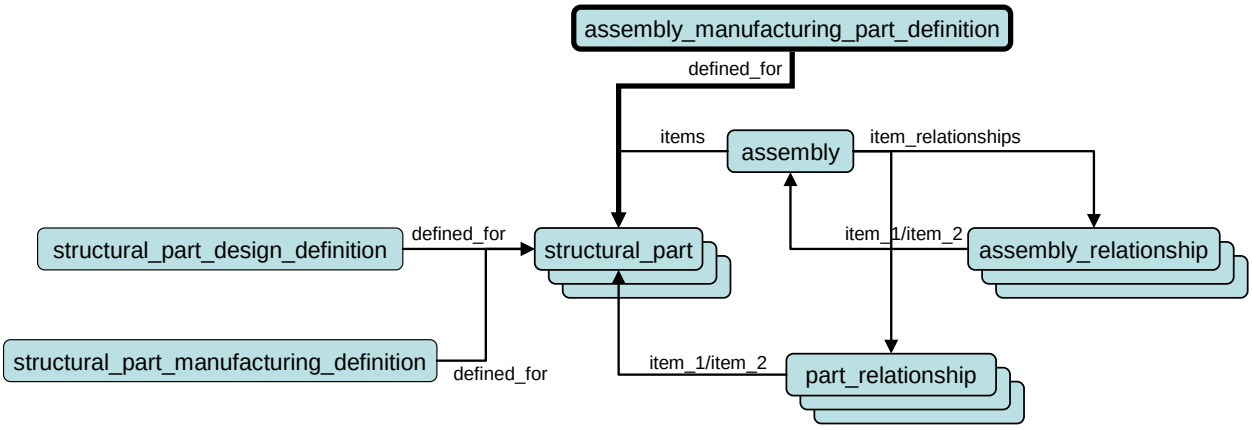


Figure 37: Assembly/Part Relationships

5.5.2.2 Labels/Layout Marks

Figure 38 shows the current support within the STEP AP218 specification for labels and layout marks. Each manufacturing definition of a part can have any number of text annotations (labels) with an optional single point specifying its location. Layout marks consists of curves only. There is no association between an annotation and a layout mark and no direct means of designating a symbol. Labels and layout marks are distinguished as being on the top or bottom surface of a part.

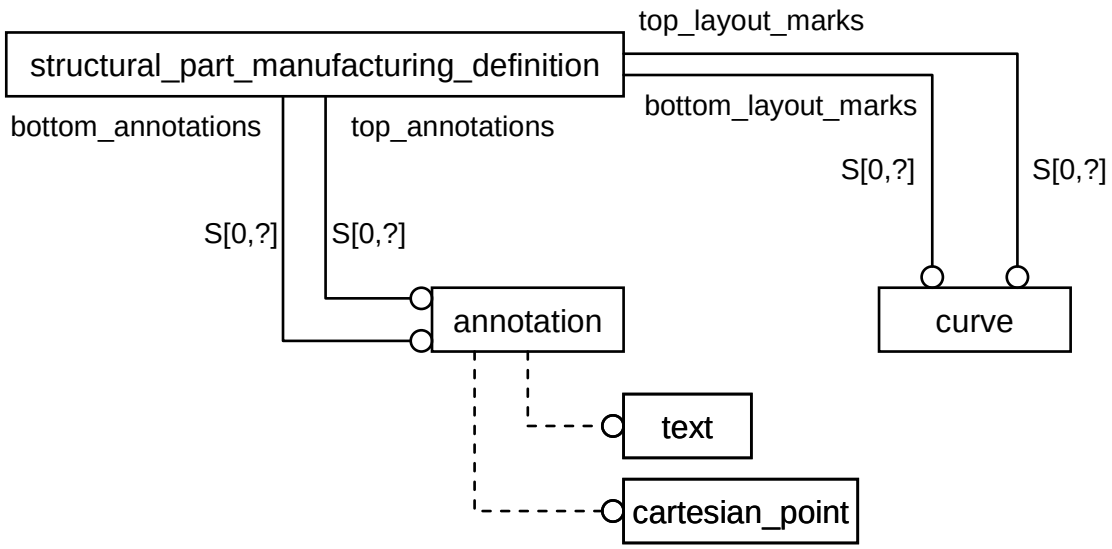


Figure 38: Current STEP AP218 Support for Labels and Layout Marks

Figure 39 shows a proposed enhancement to the AP218 specification to support labels, layout marks, and symbols.

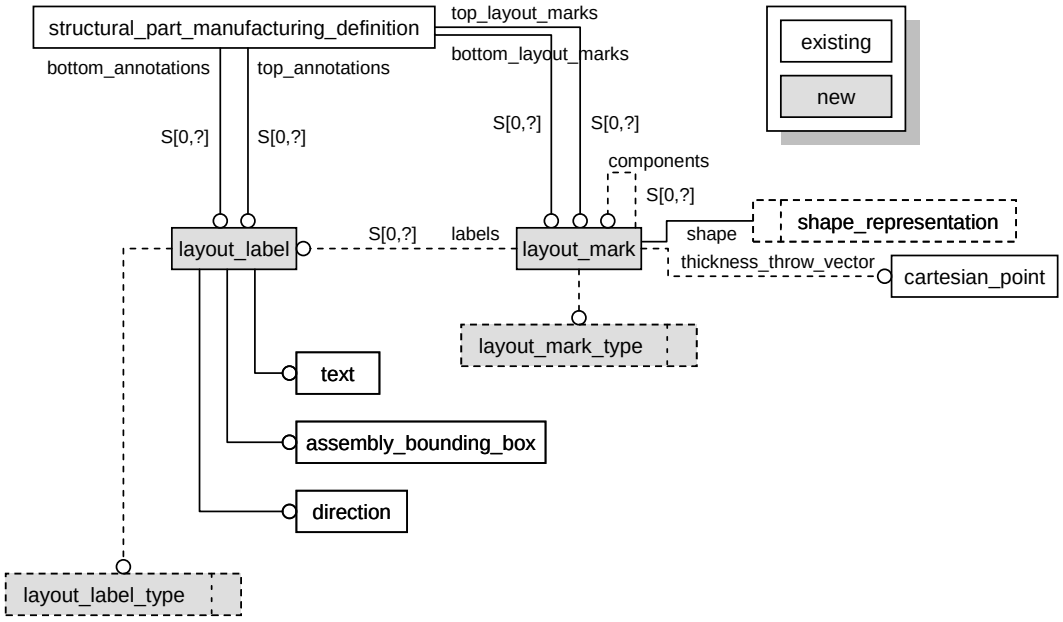


Figure 39: Proposed STEP AP218 Enhancements for Labels, Annotations, and Layout Marks

The proposed enhancements replace the *annotation* and *curve* entities with new entities *layout_label* and *layout_mark* respectively.

Layout labels consist of a text element, a bounding box, which is used to designate the text size, and a direction so the text can be oriented to something other then 0 degrees. In addition, an enumeration is included – *layout_label_type* – that is used to specify the type of the label. Layout label types can be one of the following:

Table 4 - Layout Label Types

Adjacent member id	Fit-up	Ship's Direction
Alignment Line	Frame Line	Special end-cuts
Assembly	Joint strength requirement	Stock
Assembly Sequence	Knuckle	Strength Rqmts
Attached Structure	Mass	Surfaces
Beamline (Landing Curve)	Match Marks	Tangent Line
Bending Curve	Material Type	Thickness
Bevel	Material types for adjacent parts	Thickness relative to adjacent part
Buttock	Mold location & instruction	Thickness Vector
Cant Details	Name or Id	Twist
Chamfers	No. Structural Part Attachments	Unit Number
Cutout	Orientation within the Ship	Waterline
Datum Line	Penetration	Welding
Assembly	Piece No.	Work Location
Assembly Sequence	Post Erection Cut	Unspecified
Attached Structure	Roll Line	
Beamline (Landing Curve)	Root Gaps	
Dimpling Location	Routing	
Drill Hole Center Line	Seam	
Far-Near Side	Shaping Processes	

Labels for a part manufacturing definition should only exist if they are to be etched onto the surface of the part as part of the manufacturing process. The proposed enhancements for annotations allow for more flexibility in the placement and sizing of a label and the added label type provides a means of programmatically distinguishing between different labels.

The proposed layout mark entity consists of a one or more shapes, an optional set of layout labels, and a type. The type would be one of the following:

Table 5 - Layout Mark Types

Alignment	Drill_hole_center_line	Seam
Attached Structure	Frame_line	Shell_landing
Beamline (Landing Curve)	Heating_line	Tangent_line
Buttock	Inverse_curve	Thickness_vector
Cant_details	Knuckle	Waterline
Datum_line	Neat Cut	Unspecified
Dimpling_location	Post_erection_cut	

These enhancements for layout marks provide a more structured means of defining layout line data by being able to include multiple geometric entities for a single layout mark. This also allows for the definition of symbolic annotations. Like layout labels, the label mark type allows layout marks to be categorized. Finally, the included set of layout labels allow label data to be associated with its corresponding layout marks.

The content, placement, and inclusion of marking line and annotation data are, to a large extent, driven by manufacturing rules and these rules vary between yards. Having the ability to query a manufacturing work package for this type of information therefore becomes necessary in the context of work sharing. From the standpoint of the Steel Processing System, these capabilities are important since having this information contained in the work package enables the system to automatically, and more accurately, position and categorize annotation and marking line data based on a well-defined rule set.

5.5.2.3 Mapping to Manufacturing processes and equipment

The primary purpose of the Steel Processing application is to map the information that describes steel parts with the processes needed to fabricate and assemble them into complete units. Although STEP AP218 contains the information that describes these parts and their assembly hierarchy its support for describing, or even specifying, manufacturing processes is sparse or non-existent. Also, there is no mechanism within the specification to identify the equipment to be used for manufacturing or the location where this work is to be completed. If AP218 is to be used to describe a steel production work package, as it is for the Steel Processing application, then this information needs to be included into the specification or provisions must be made to link to external sources (preferably defined using STEP) where this information can be culled.

One problem with including all of this information in a single location is that each piece of information evolves through a life of its own. Information that describes a manufacturing process is dependent on the equipment being used and the processes employed at its manufacturing location. Information describing each steel part and assembly is relatively static[7] although this can be influenced by process. Equipment specifications and facility capabilities and requirements are somewhat static as well but can change due CAM or hardware upgrades.

[7] For the purposes of this document, design changes are ignored

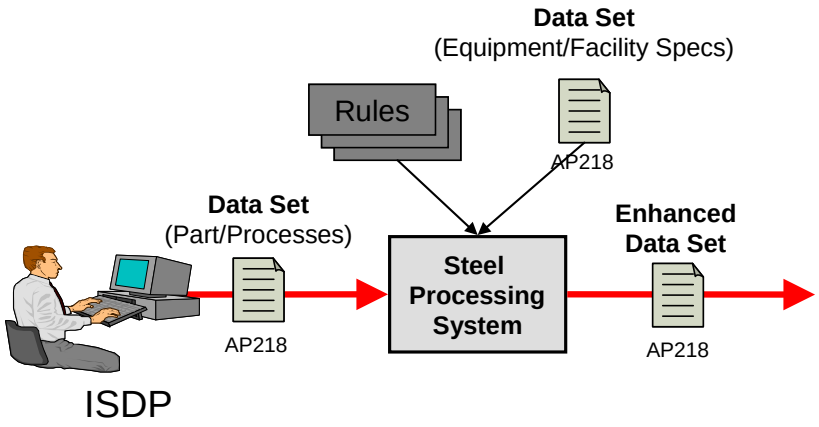


Figure 40: Steel Processing Inputs

The Steel Processing pilot application uses the STEP AP218 enhancements described in Table 3 to represent steel part and assembly data along with marginal support for manufacturing processes (see Figure 40). Equipment specifications and facility capabilities were described separately.[8] Separating the equipment and facility information from the part and manufacturing process data allows this information to be processed independent of the part data. The Steel Processing Task demonstration created two sets of equipment/facility data to reflect the two manufacturing areas the system will be used to support.

The STEP AP218 specification would be more useful to support the definition of a work package if it included manufacturing processing data either internally or through external references. Whether the format of this information is driven by manufacturing process definition schemas as defined in other STEP APs, such as AP224 or AP240, or the STEP AP218 enhancements provided by Atlantec Enterprise Solutions is yet to be determined. Defining provisions to support manufacturing processes is more than including support for the processes themselves; additional information may also be required for each structural entity as noted in the following section.

5.5.2.4 Part Relationships

This section proposes a minor addition to the current STEP AP218 specification to support the definition of the physical joint between two structural entities. This change is motivated by the need for the Steel Processing system to:

- relate raw material data with the manufacturing processes that are defined as part of a work package,
- validate the integrity of various types of connections, and
- use information contained within the work package for downstream processes.

Figure 41 shows an example of the relationship between a pair of plates and a couple of stiffeners that are attached to them. The plates are related by a seam, which is defined by one of the bounding edges for each plate. The stiffeners are related by the bottom edge of the web of the structural tee and marking

[8] The team agreed to use an existing schema provided by Atlantec Enterprise Solutions that was created to support their SBIR project entitled ‘CAD-CAM Connector Architecture’.

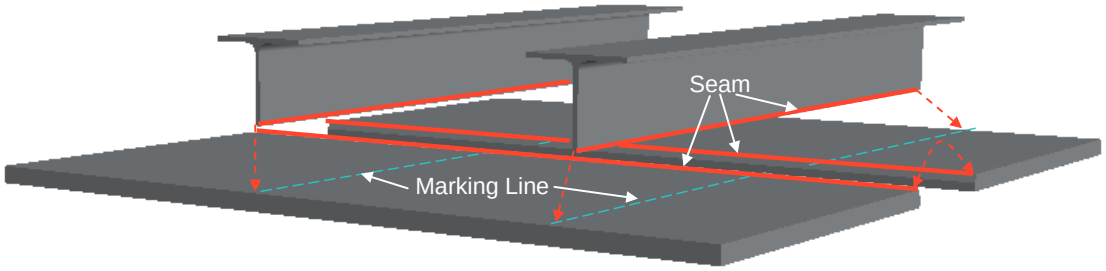


Figure 41: Plate/Stiffener Joint Relationships

lines etched on the surface of a plate. As part of the manufacturing process, the plates would first be welded together to form a larger panel system onto which the stiffeners are then attached. The geometry that defines the marking lines on the surface of the plates defines the physical location of where a stiffener would be placed. This data could also define the path for a robotic welder. The plate seams represents the physical connection point for the two plates and can also be part of the information to define the edge preparations required for the intended weld. Depending on the manufacturing processes used to etch the marking line on to the surface of the plate, the physical marking line may need to be held back from the plate seam.

If the proper information is contained in the work package, rules can be established to validate, or generate, various attributes of these part connections. STEP AP218 contains support for numerous types of relationships among various components including, features, design definitions, and manufacturing definitions. Figure 42 shows the STEP AP218 entities currently included in the specification for defining structural joints between parts.

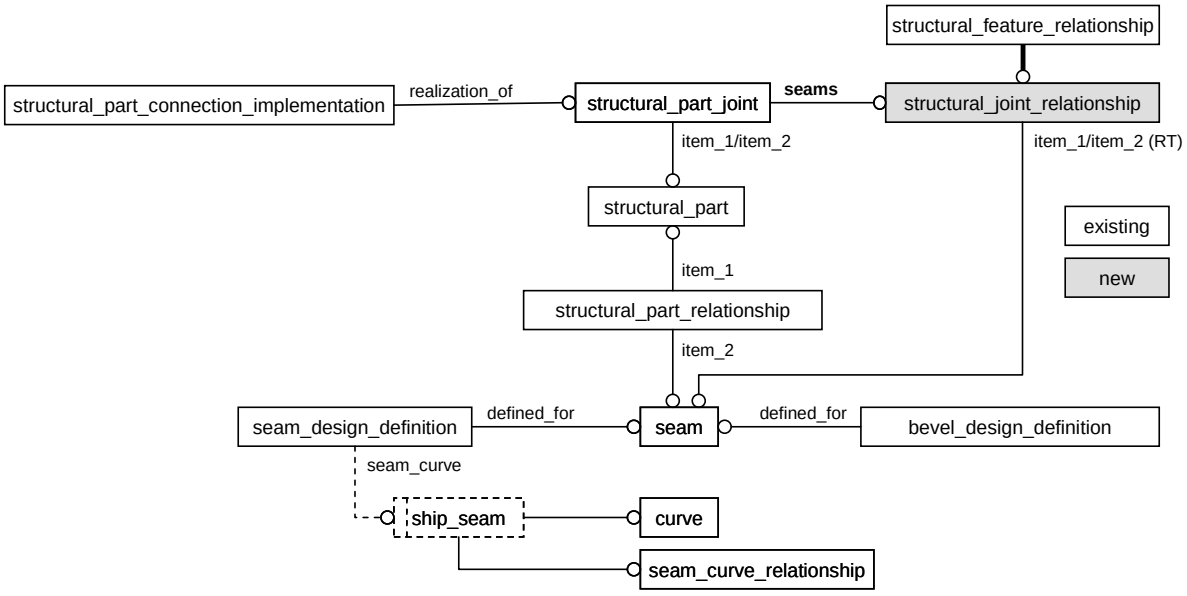


Figure 42: Structural Part Relationships in STEP AP218

Structural_part_joints identify a connection between two structural parts. The details of this connection are defined by derivatives of the *structural_part_connection_implementation* entity[9]. *Structural_parts* can then be related to a seam by means of a *structural_part_relationship*. The physical geometry for a *seam* is defined by the *seam_design_definition*, which uses the *ship_seam* to define the seam with a curve or with a component of another part using *seam_curve_relationship*. Bevel information, defined by *bevel_design_definition*, can be related to a seam using its *defined_for* attribute. This configuration allows for two parts to be related by a structural joint, using *structural_part_joint*, and allows for a seam to be related to a structural part, using the *structural_part_relationship*. However, there is no mechanism to relate a structural joint to the seams defined on or by each of the adjoining parts. *Structural_joint_relationship*, a derivative of structural_feature_relationship, is proposed to define an association between two seams. Each item in the relationship is enforced to be a seam. *Structural_part_joint* would then require an additional attribute of type *structural_joint_relationship*. An instance of a set of these entities may look like the following:

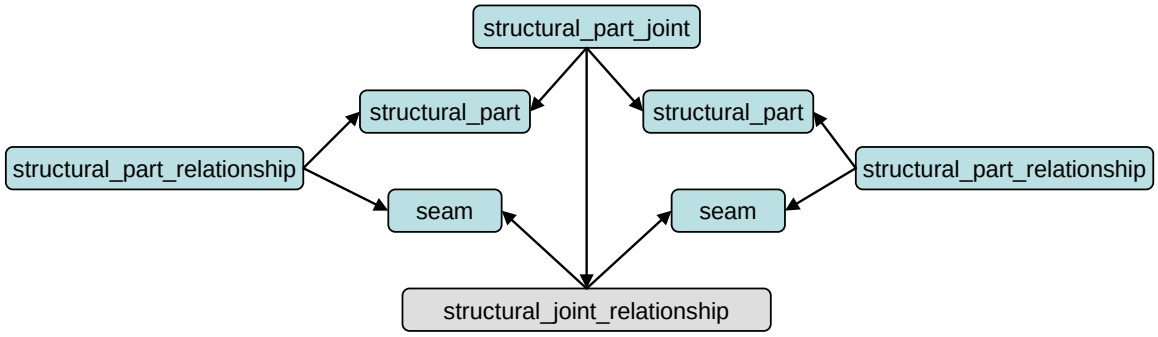


Figure 43: Structural Part Joint Relationships

[9] Such as a *weld*.

6 ENGINEERING ANALYSIS TASK

6.1 Scope and Purpose

This section contains both an overview and selected design details for a collaborative design approach to demonstrate flexible exchanges of product model data among the participating Shipyard / CAD / CAE system environments in multiple formats. The ISE-4 Engineering Analysis Task focused on facilitating engineering analyses employing Finite Element Method (FEM) analysis software at different stages of ship development (but principally in the "detailed design" stage). This task demonstrated translation methods and prototype tools to enable flexible and combined use of multiple ISO STEP Application Protocols (APs). Ship AP218 was employed for ship structural data exchange, while the AP209 protocol was used for engineering Finite Element Method (FEM) analysis. The approach, illustrated conceptually in Figure 44, is to:

- access ship geometry and material information from various ship STEP APs;
- mediate, and/or transform this data employing multiple formats (both traditional STEP Express Part 21 files and XML Part 28 files); and
- employ and demonstrate use of STEP AP209 for supporting FEM analysis.

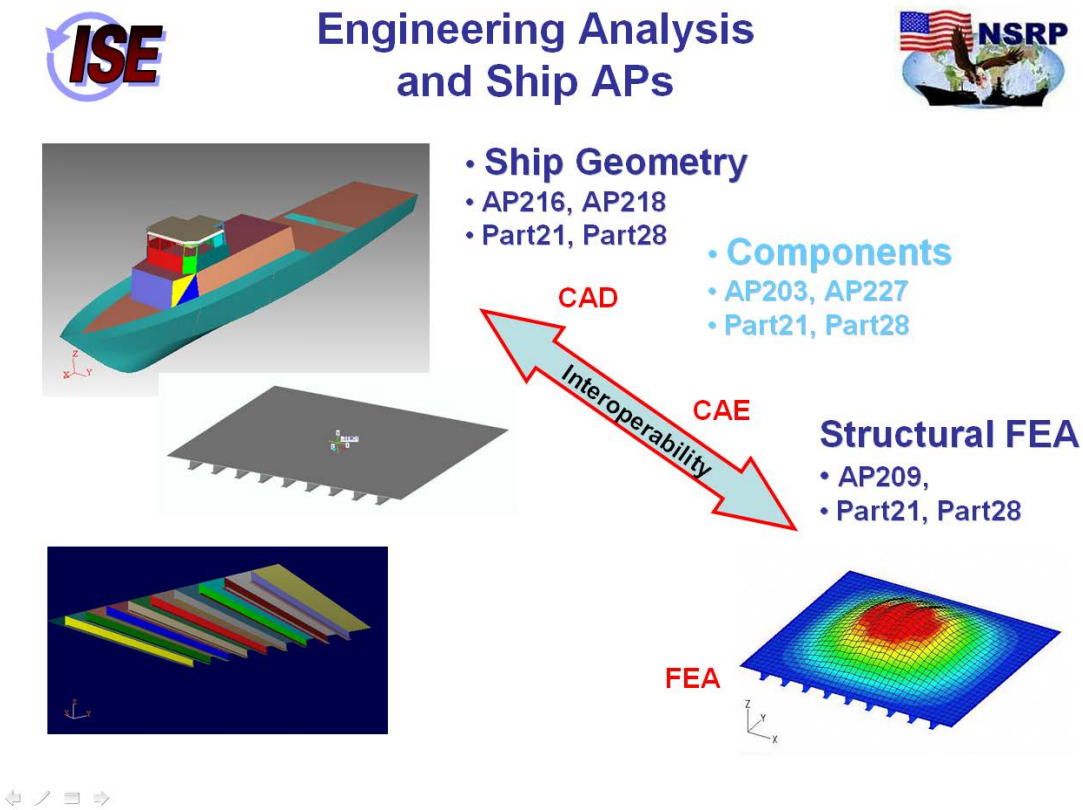


Figure 44: Schematic Task Overview

While Figure 44 is more conceptual, Figure 45 below shows the geometry region from the Torpedo Weapons Retrieval (TWR) ship actually selected for the first task demonstration. We are using structural details from the TWR engine room supporting the port propulsion engine for exchange and subsequent analysis leading to design changes for improved ship shock hardness.

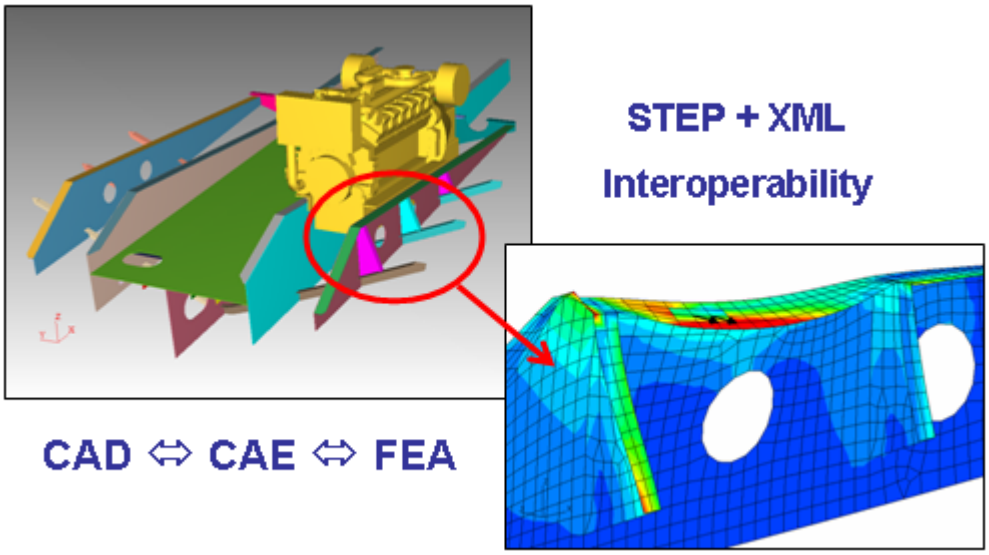


Figure 45: TWR Engine Room Structure and FEM Analysis

The ISE-4 Engineering Analysis team was led by Electric Boat Corporation (EB) with members from the Intergraph Corporation, and Northrop Grumman Ship Systems (NGSS). As appropriate, test data and STEP files supplied or created by other ISE-4 team members were employed. Also, as noted below, there is a connection outside the NSRP ISE Project to other work with ISO STEP 10303-209 (AP209).

There is a synergistic relationship between this task and several other efforts, in particular, a similarly-named PDES, Inc. Engineering Analysis Pilot Project which supports the furtherance of ISO 10303-209 (AP209) titled: "Composite and Metallic Structural Analysis and Related Design" (Reference 13). Any required updating of existing AP209-interfaced Computer-Aided Engineering (CAE) software will be conducted in that Pilot Project. Whereas, all work with the Ship AP's, related context schemas, and developed mediation/transformation tools and methods were performed under this NSRP ISE-4 EA Task.

Under other ISE-4 tasks, data exchanges via STEP files enable teaming organizations to efficiently employ multiple or different Computer-Aided Design (CAD) systems to evolve ship system designs. Under this task we presume that will also occur, and further, that during certain product life-cycle stages (particularly during the detailed design phase), there is the need to facilitate the exchange of ship design data with other groups or organizations outside the inner circle of ship design agents. Often, such organizations (e.g. shipyards) may enlist others for detailed analysis of the evolving ship concepts. Finite element and related computational methods are regularly employed for stress, vibration, shock, or acoustic analyses, across many varied industries. In the first demonstration (Section 6.5), that was, in fact, the first demo scenario.

To facilitate such detailed FEM analyses, this project strove to facilitate exchanges of ship data (primarily geometry), available in various ship APs (References 1-3) to other existing ISO STEP AP formats currently employed by the analysis community. Accordingly, a system of mediators and translators was constructed to accept (or import) explicit ship AP geometry (more about this later) in

either Express (Part 21) or XML (Part 28) forms, and merge them into an AP209 geometry file for use with several existing structural analysis codes and systems. Such a scenario was demonstrated in the end-of-project integrated ISE-4 demonstrations.

As a minor note, the ISE shipbuilding exchange community employs the terms preprocessor and postprocessor to denote creating (exporting) and reading (importing) a STEP data exchange file. In the FEA community these terms also refer to software codes (generally with interactive graphics capability) used to prepare input models for, and review the results from, a detailed FEM analysis software code.

6.2 System Design and Approach

This project was faced with utilizing an existing family of CAD and CAE software codes for analysis which already employed Part 21 (Express) for data exchange, while at the same time allowing the flexibility to accommodate the new XML translators being employed for the Ship APs (specifically AP218). Rather than demanding code vendors to implement Part 28 (XML) interfaces (for multiple STEP APs), we set out first to demonstrate with prototypes, that mediation among various file types is possible with existing XSLT engines and appropriately-created XML style sheets. An added second demonstration illustrated an approach for creating exportable explicit geometry from imported AP218 implicit or parametric geometry.

6.2.1 File Mediation

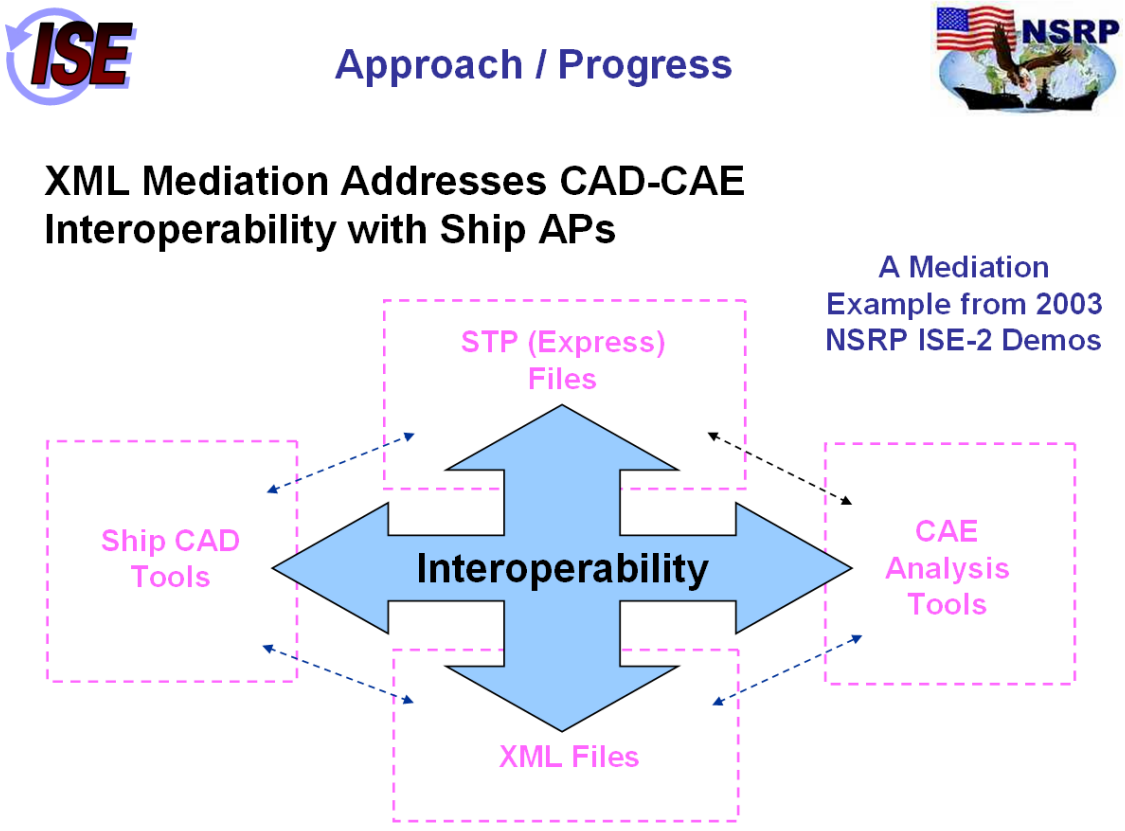


Figure 46: A Flexible Interoperability Approach

The overall mediation approach is illustrated in the following few figures which were also employed in the earlier 2003 ISE-2 demo. Figure 46 broadly shows the overall approach using XML mediation for effective interoperability. The following figure is a conceptual roadmap for XSLT mediation between different delivery forms for STEP geometry.

Using Figure 46 as a symbolic framework, we can now introduce Figure 47 showing the demo dilemma, as it were. Suppose the ship designer Northrop Grumman Ship Systems (NGSS) using Intergraph's ISDP CAD system wishes to share and exchange ship geometry with Electric Boat (EB). As seen in Figure 47, ISDP is exporting (outputting) explicit ship geometry as an AP218 ARM XML (Part 28) file (lower left in Figure 47). Whereas, EB's engineering folks are equipped with tools employing ISO STEP AP209 (and/or AP203) in AIM Express (Part 21) format. The solution or remedy as shown in Figure 48 is to employ a series of XSLT transformations.

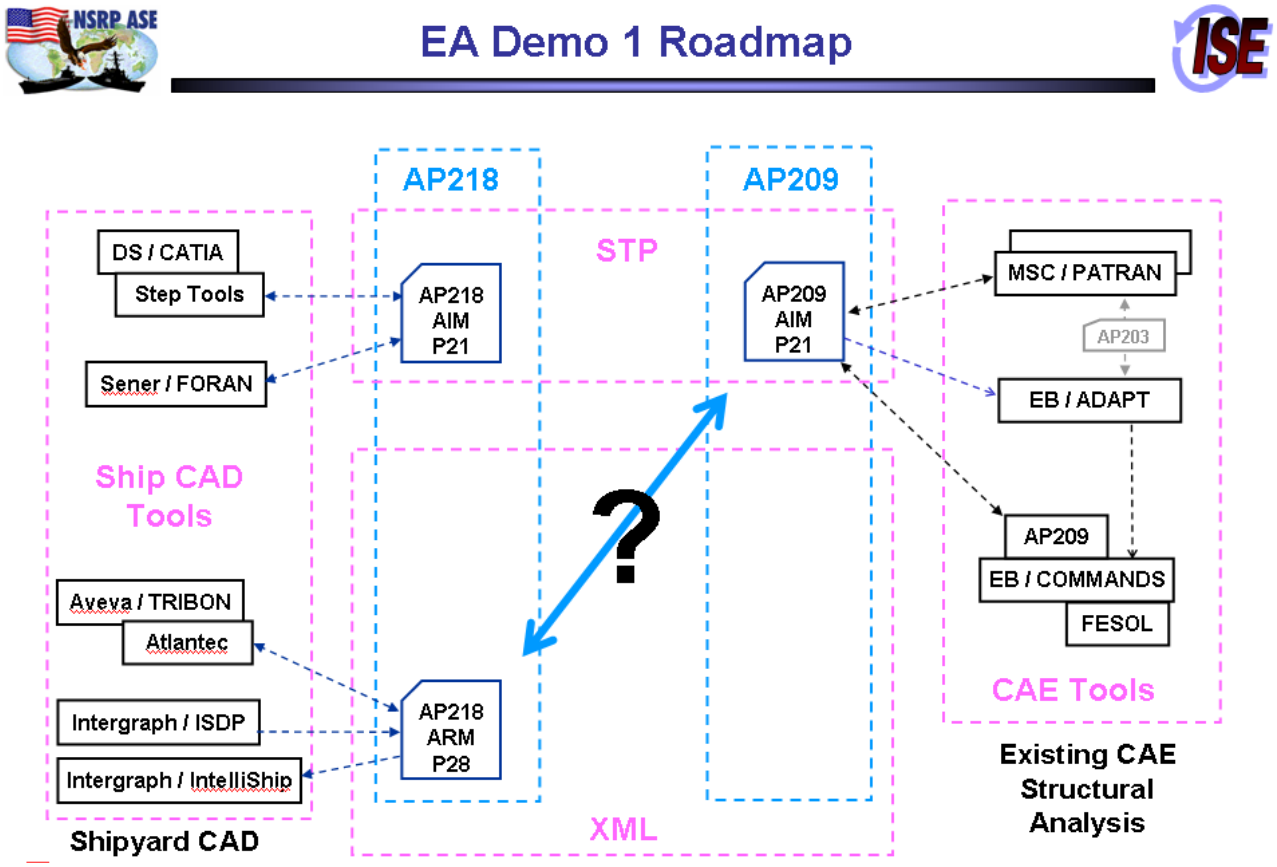


Figure 47 - The Demonstration Dilemma

The NSRP ISE data exchange solution is shown below in Figure 48. Files in one form or format can be transformed or mediated into another form. For XML files this approach uses a transformation employing a style sheet of mapping instructions. This is referred to as an XSLT.

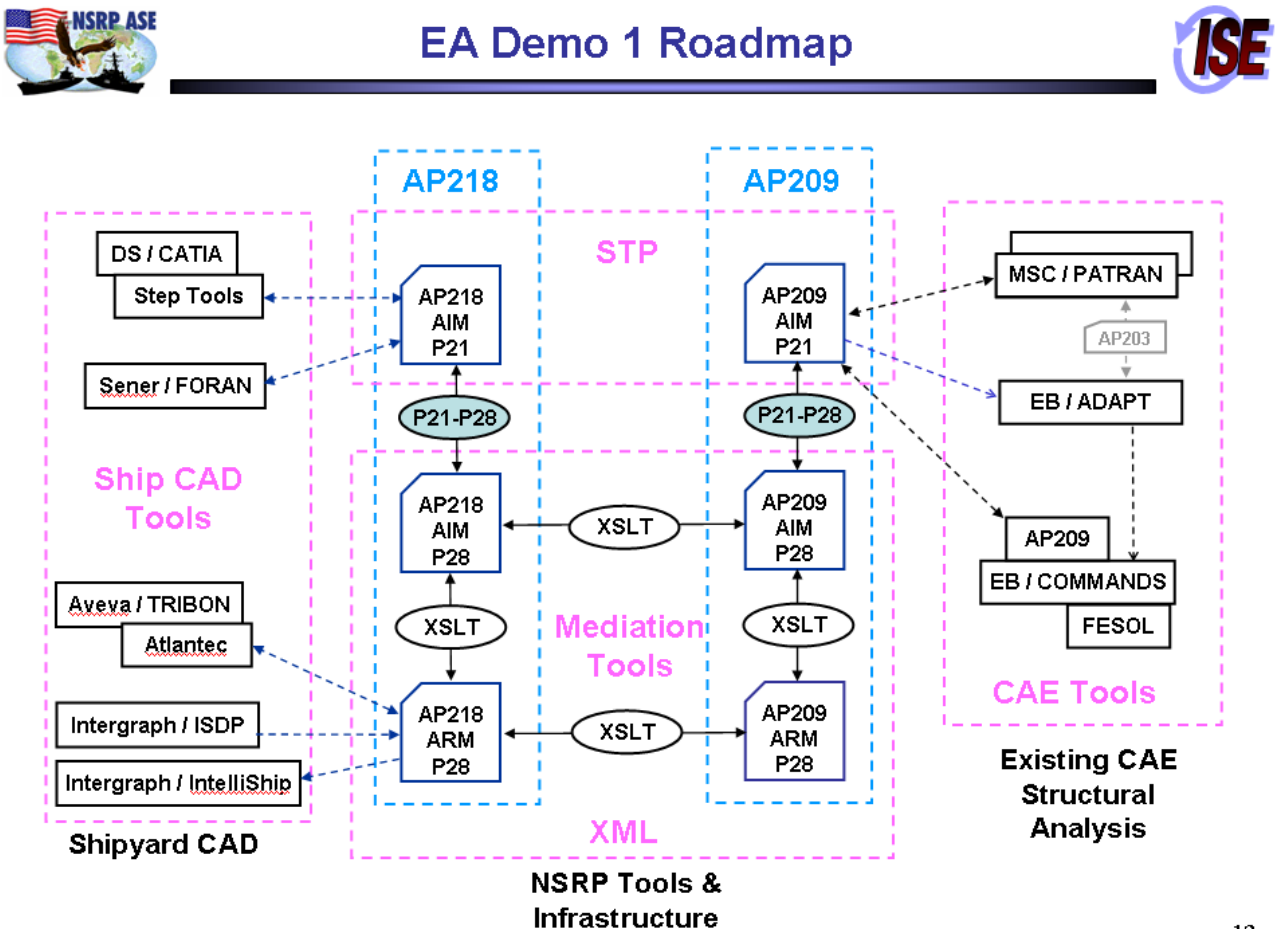


Figure 48 - The Mediation Roadmap

6.2.2 Required Mediators and Testing

To accommodate the needs of this task, mediators were required to exchange or map between:

- ARM and AIM representations of the same AP file (e.g. AP218),
- ARM or AIM representations across different AP files (AP218 and AP209), and
- between Express (Part 21) and XML (Part 28) forms.

In each of these cases slightly different types of testing were employed although all transformations (XSLTs) started with individual round-trip (or cycle) testing.

An example of the third type of round-trip is illustrated in Figure 49, where an AP209 file containing geometry, FEA mesh, and analysis results is first exported from an existing CAE code (PATRAN) in Part 21 form. It is converted to XML (Part 28) form and then cycled back to a second Part 21 file and tested for content and completeness by importing back into PATRAN. [Note - this type of cycle test has been demonstrated previously using the OSEB bound (Edition 1) XML file format, and a translator ST-XML from STEP Tools, Inc. (STI). These tests were repeated in this project using different software supporting the newer Part 28 Edition 2 schema-based XML form.]

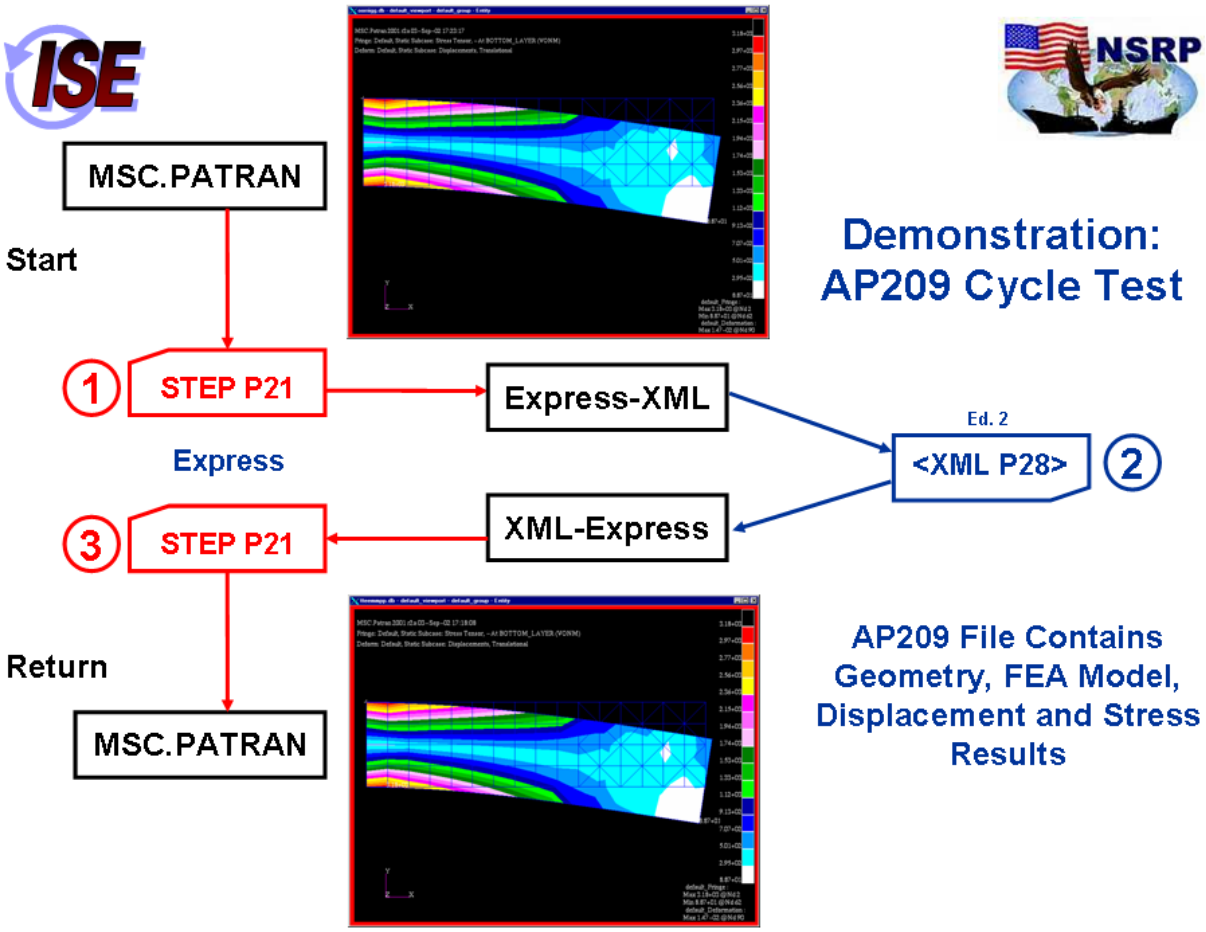


Figure 49: Express to XML and Return Cycle Test

After the creation and cycle testing of each mediator or translator, focus was shifted to capabilities required for the year-end demonstration. This involved focusing on explicit ship geometry rather than the FEA portion of AP209.

A previous document (Reference 14) described the planned test examples, starting with simple profiles and plates then extending to assemblages of TWR structure such as bulkheads and engine room support structure. As actual testing of mediators progressed, we added additional test cases starting with a simple cubic box (advanced_brep_shape_representation), and single plate structures with and without cutout holes. These latter cases were both solid (brep) and surface (manifold_surface_shape_representation) configurations.

For the actual demonstration, TWR engine room ship geometry representing the outer support girder for the port engine was exported from ISDP (AP218 ARM XML Part 28 file), mediated in real time with the GUI (Graphical User Interface) shown later, and imported into EB's ADAPT software (as an AP209 AIM Part 21 file). The demo geometry consisted of an assemblage of five advanced brep solids.

6.2.3 STEP Part 21 and Part 28

Traditionally when STEP has been used for product model data exchange, the translators have produced a neutral data representation (known as a STEP physical file) in the Express language. This is a Part 21 file. Some of the ISE-4 ship preprocessor translators produce files in this Express-encoded format. However, with the growing use of the internet and the World Wide Web (WWW), a new STEP format has been developed, called Part 28, which utilizes XML structures to encapsulate the STEP data. (In this task, unless noted in reference to earlier OSEB forms, all references to XML Part 28 refer to Edition 2 of Part 28, a schema-based form.)

Processors have been developed during the ISE (Integrated Shipbuilding Environment) Project to convert the Part 21 files into XML-based Part 28 files to enable interactive exchanges of Engineering Analysis data. Mediator style sheets have also been developed to go in the opposite direction, converting XML-based Part 28 files into the more traditional Part 21 format. Thus, some of the ISE-4 Ship translators will produce files in Part 21 format while others will create Part 28 files, but the publicly available tools developed under the ISE Project will enable conversion of these files between Part 28 and Part 21.

6.2.4 Preprocessors and Postprocessors

A preprocessor reads the model in the native CAD or CAE system and generates a STEP neutral file. The preprocessors developed for ISE-4 Engineering Analysis were validated by careful analysis of the Part 21 or Part 28 file created to see if the required entities and attributes were properly captured in this file. Checklists were created and employed to verify the accuracy of the created STEP files.

A postprocessor reads the neutral STEP file and converts it to a model on the receiving CAD or CAE system. The postprocessors were validated by checking the attributes and functionality of the product model on the receiving system.

[As noted earlier, in the FEA community the terms preprocessor and postprocessor also refer to software codes (generally with interactive graphics capability) used to prepare input models for, and review the results from, a detailed FEM analysis software code.]

6.2.5 Conformance Classes

The ISO STEP Application Protocols employed in this project have different Conformance Classes (CCs). We are primarily concerned in this task with the explicit geometry contained in STEP files as shape representations.

Several geometry types (shape representations) were employed. In the ship APs, this is referred to as "explicit" geometry, as opposed to "implicit" geometry. In this task we focused initially on two shape representations (non-manifold surface and advanced B-rep solids). These are the two most commonly used shape representations employed in AP218. The non-manifold surface shape representation is used in both AP215 and AP216.

6.3 Geometry Shape Representations

Table 6 below shows the usage of explicit shape representations in the Application Protocols of interest in ISE and this task in particular. AP209 (and AP203, not shown) employ and allow 7 different shape reps for wire frame, surface, and solid geometry. Whereas, the ship APs employ fewer shape representations. AP218 has Conformance Classes using 4 shape reps.

For this task we focused initially on two shape representations (non-manifold surface and advanced B-rep solids). These are the two most commonly used shape representations employed in AP218, and the non-manifold surface shape representation is also used in both AP215 and AP216.

Table 6 - Geometry AICs

Geometry AICs Used	AP215	AP216	AP218	AP209
Part 501: AIC: Edge-based wireframe		●	●	●
Part 502: AIC: Shell-based wireframe				●
Part 507: AIC: Geometrically bounded surface				●
Part 508: AIC: Non-manifold surface	●	●	●	
Part 509: AIC: Manifold surface				●
Part 510: AIC: Geometrically bounded wireframe			●	●
Part 512: AIC: Faceted boundary representation				●
Part 514: AIC: Advanced boundary representation			●	●

Non-manifold surface shape representation is employed in the ship APs - AP215, AP216, and AP218, whereas, AP209 and AP203 employ the manifold surface shape representation (which is a subset of non-manifold). We have discussed updating AP209 to include non-manifold shape, and this will be pursued in Edition 2 of AP209, which is outside the scope of this task.

For this task we will map non-manifold surfaces from the ship APs to manifold surfaces in AP209. In practical terms, this appears to be of only minor (academic) concern since all the non-manifold surfaces in test cases examined, are actually exported as manifold surfaces.

6.4 Team Translator Environments

6.4.1 ISDP/IntelliShip (Intergraph and NGSS)

For this task effort, the primary source of AP218 and AP216 files was Intergraph's Integrated Ship Design & Production (ISDP) suite of CAD products. This was employed at both Intergraph and at Northrop Grumman Ship Systems (NGSS) Avondale Operations site.

6.4.1.1 Overview

The AP216 and AP218 STEP translators are designed to export and import Part 21 or Part 28 STEP files. The preprocessor is implemented in Intergraph's ISDP tool. The postprocessor is implemented in Intergraph's IntelliShip shipbuilding CAD product. Both preprocessor and postprocessor extend the ship AP translators which were originally developed during the former ISE-2 Project.

6.4.1.2 Interfaces

ISDP Preprocessor (Export)

A command is provided that allows the user to enter the name and location of the STEP file to be created or to accept a default name and location.

IntelliShip Postprocessor (Import)

A command is provided that allows the user select a file from the list of files in the default location or to enter the name and location of the STEP file to be translated.

6.4.1.3 Hardware and Software Environment

Target Platform

ISDP Preprocessor: Solaris x86

IntelliShip Postprocessor: Microsoft Windows

Development Tools:

Intergraph is using ST-Developer by STEP Tools, Inc for the postprocessor.

Operations:

1. ISDP Preprocessor (Export) The user will do the following:
   - Select the Export to STEP command
   - Select AP216 or AP218 from the list of AP's that can be exported
   - Enter the name and location of the output STEP file or accept the defaults
   - All AP216 or AP218 explicit geometry data within the file will be translated into the selected STEP file.

2. IntelliShip Postprocessor (Import) The user will do the following:
   - Select the Data Exchange Task
   - Select the Import command from the DataExchange Vertical Toolbar
   - Select AP216 or AP218 from the list of AP's that can be exported
   - Select the input file from a list or Enter the name and location of the STEP file
   - All AP216 or AP218 explicit geometry data within the STEP file will be translated into the IntelliShip workspace and stored in the database.

6.4.2 Various CAE Systems (Electric Boat)

6.4.2.1 Overview

Electric Boat's role in this task is two-fold. First, EB implemented and tested a series of mediators and translators (to accommodate the various APs and formats), and second, EB conducted the engineering analysis portion of the demonstration (illustrating and employing AP209). In both cases EB was working with Window's Based PC hardware, and various EB-developed and third-party commercial-of-the-shelf (COTS) software.

6.4.2.2 Mediator Translator Development

Electric Boat (with the assistance of other team members) created and tested the mediators and translators required for this task. EB also constructed a convenient Graphical User Interface (GUI) to assemble and execute the assemblage of mediators for the demonstration.

6.4.2.3 Finite Element Analysis (FEA)

Electric Boat currently employs a variety of in-house and COTS CAE tools for finite element structural stress, shock, vibration, and acoustic analysis. The most-frequently employed analysis codes are FESOL, VIBES, and ABAQUS. The first two were developed and are maintained by EB. The most-used CAE preprocessors for FEA model development are EB's COMMANDS code and PATRAN, while versions of ABAQUS/CAE, FEMAP, and HyperMesh are also available on our CAEN system. The last four are

COTS tools. On the design side, we are primarily employing the CATIA V4 CAD system, but will likely migrate to one or more newer CAD tools for increased flexibility, and to accommodate our emerging Product Lifecycle Management (PLM) system.

As an electronic CAD-CAE electronic exchange mechanism, EB currently employs the STEP AP203 format, and also has generated software such as ADAPT (Automated Design to Analysis Process Transformation) to facilitate editing and de-featuring CAD geometry for use in analysis. All the aforementioned CAE preprocessors accept (import) geometry via an AP203 format. With COMMANDS, PATRAN, and CATIA we have also implemented a set of AP209 interfaces. These have been demoed in prior PDES meetings.

The selection of AP209 (over AP203) for this effort is based on the more complete and extensive geometry representation in AP209. In particular, this includes the ability to associate material properties with geometry and analysis models, and the assigning of attributes to 1D (wire) and 2D (surface) geometry. These are not possible nor complete in AP203 or AP214). Figure 50 graphically illustrates the contents of an AP209 file, including the ability to associate FEA models (and linear analysis results) with the structural geometry and product structure information. In this task we employed all parts of AP209 except for the Composite Constituent Shape (CCS) portion. We worked only with metallic parts.

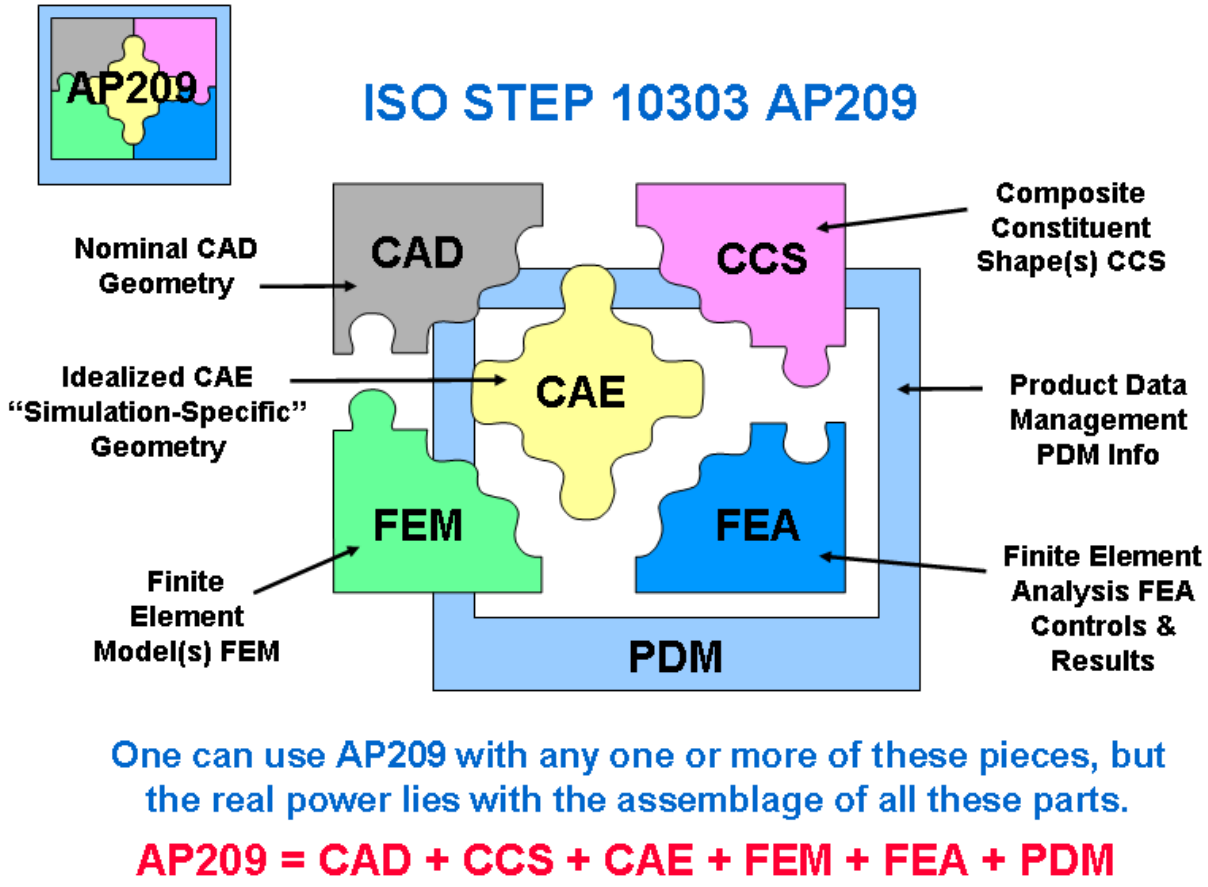


Figure 50: AP209 File Contents

6.5 Demonstration Results

Near the end of the ISE-4 Project, as with prior NSRP ISE Projects, a joint set of demonstrations was held. Planning for this event took place during the year. In ISE-4 this Engineering Analysis Task was included along with three other projects focused on Ship Arrangements, Electrical, and Steel Plate Manufacturing.

6.5.1 Overall ISE-4 Demonstration Outline

- Provide an overall ISE-1, -2, -3, -4 theme and lead-in!
- The general time-line backdrop is Ship Lifecycle, from Design (Concept, Preliminary, Detailed), Manufacturing, to Maintenance and Support, plus Navy Oversight.
- There are multiple, collaborating shipyards (Co-Design Yards).
- Partnering shipyards employ different CAx (CAD/CAE/CAM) systems.
- Moreover, different CAx systems have chosen different means for exchanging digital design data.
- All are employing Ship STEP standards (ISO 10303-215, 216, and 218.
- However, delivery mechanisms vary, some use Express (Part21 AIM form) while some use XML (Part28 ARM form).
- This sets the stage for various mediation examples.
- Task demos illustrate different versions of collaboration and mediation.
- All the illustrations employ publicly-released real ship design data (drawings and product models) of the Torpedo Weapons Retrieval (TWR) vessel.

6.5.2 EA Demonstration 1

6.5.2.1 Engineering Analysis Scenario

- During detailed design, 3D CAD product models are captured and in time become increasingly more mature.
- Suppose during this, another "world" is heard from - the Department of Homeland Security has interest in the TWR ship design, but only if some modest improvements in shock hardness can be designed into it. (The TWR is a "non-combatant", lightweight construction, relatively inexpensive ship, but is potentially available for other harbor or near-shore security protection/assessment duties.)
- The partnering shipyard doing detailed design in the overall scenario (NGSS in this scenario) receives the Homeland Security request for potential design changes or improvements and decides to hire a "consulting firm" specializing in shock design and analysis (Electric Boat in this scenario).
- The consulting firm works across multiple industries (including, but not limited to ships) and is STEP literate and capable.
- However, the shock consultant (EB) uses the more general STEP APs - AP203 and AP209 (for analysis). Moreover, has many existing tools and capabilities which employ Express (Part21 AIM form) digital data exchanges.
- But the detailed design shipyard (NGSS, employing Intergraph's ISDP CAD system) is exporting its digital product model data in XML (Part28 ARM form).
- This is no problem and sets the stage for our NSRP ISE infrastructure mediation demo.

6.5.2.2 More Detailed Engineering Analysis Scenario

- Other ISE-4 tasks are illustrating mediation, but largely exchanges between Express (Part21 AIM form) and XML (Part28 ARM form) within a single application protocol (e.g. AP215, or AP218).
- The Engineering Analysis Task will illustrate and demonstrate mediation not only between XML ARM and Express AIM, but also across different APs (AP218 to AP209), albeit only for the commonly employed explicit shape representations.
- The actual TWR focus area for detailed engineering shock assessment with Finite Element Analysis (FEA) will be the support structure under the main propulsion engines.
- The demo will progressively zoom in (with slides) from the overall TWR product model, to the engine room showing parallel Caterpillar diesel engines, to inboard and outboard girders supporting the port engine.
- This latter explicit product model geometry will be exported from ISDP via one or more AP218 files in XML Part28 ARM form.
- An automated sequence of steps will mediate from this AP218 form to a starting AP209 file (in Express Part21 AIM form) for use with existing CAE tools (FEA and FEM preprocessing codes).
- Shock analyses will be performed using the outboard girder, initially as transferred from ISDP, and then after adding more intermediate brackets near the forward engine mount (and over hull frames 17 and 19). Other potential design changes may include eliminating the lightening hole and/or changing plate thicknesses.
- Recommendations for shock design improvements would then be passed back to NGSS.
- Explicit geometry files containing just the added or modified pieces could also potentially be passed back through the mediators to ISDP's (IntelliShip).

6.5.2.3 Demonstration GUI

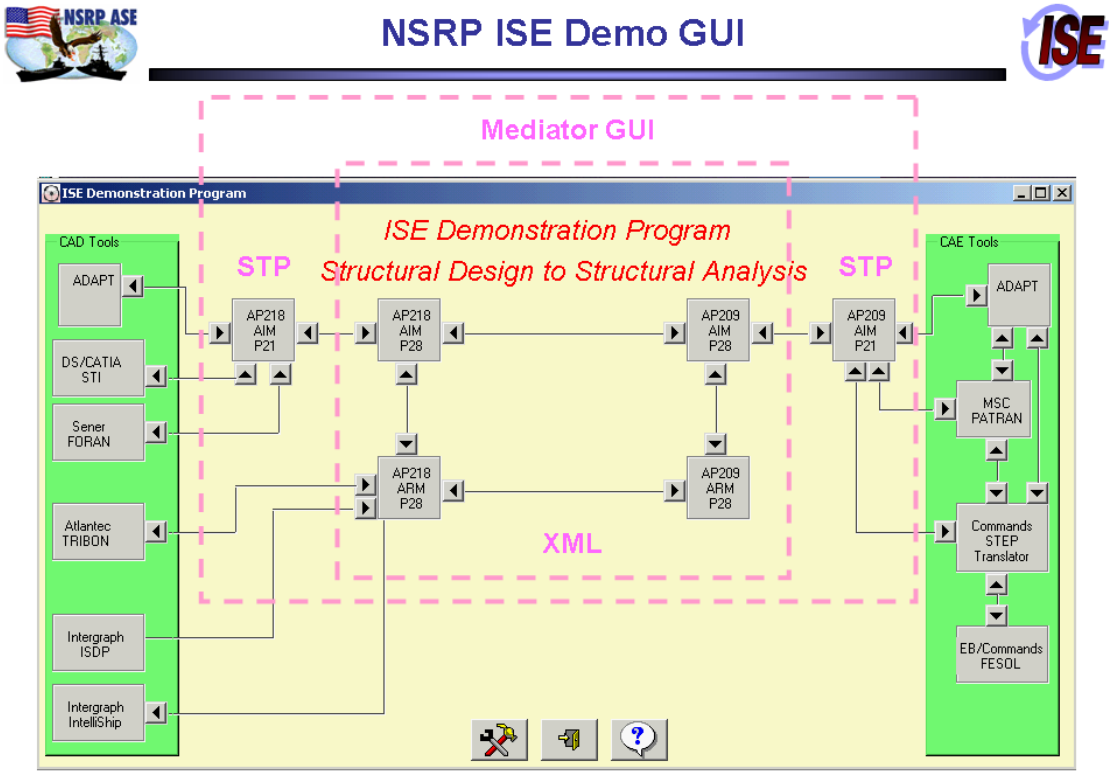


Figure 51: EA Demo 1 GUI

To facilitate the actual demo, we built a Graphical User Interface (GUI) - Figure 51. This GUI is basically patterned after the data flow shown earlier in the Figure 48 for the Mediation Roadmap. The ISDP exported file (AP218 ARM XML) is imported (lower left) and automatically processed through 3 mediators to create the AP209 AIM Part 21 (Express) file in the upper right. File symbols turn green in real time as processing proceeds along. At the end, the EB ADAPT code is opened automatically and the transferred geometry is displayed.

6.5.3 EA Demonstration 2

During the conduct of this project as we built and tested mediators for exchanging the explicit geometry, a second activity was initiated - to explore creation of the explicit geometry from the AP218 parametric implicit geometry. This added effort was initiated to evaluate the ability to manipulate geometry outside of a ship CAD tool, and to experiment with the power of parametric geometry representation for rapid design change.

Building upon some other non-NSRP work, particularly some sponsored by the Center for Naval Ship Technology, we used some existing Java modular software created to support robotic welding of heavy steel plate. To that some AP218 specific routines were added to create a demonstrator which is shown schematically in Figure 52 below.

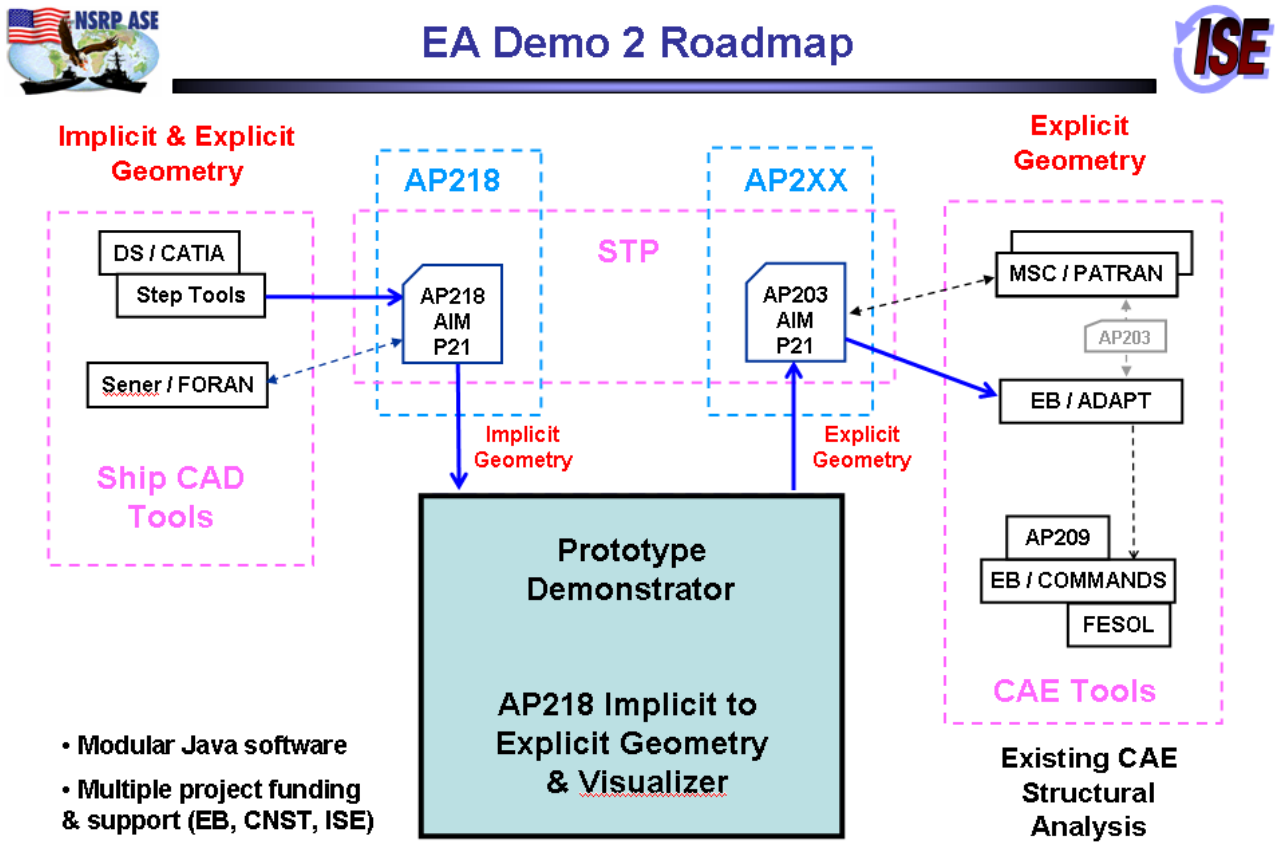


Figure 52: EA Demo 2 Roadmap

This prototype tool was created to read AP218 implicit parametric geometry, create and display the explicit form, and to allow on-the-fly changes to the design parameters. At any time an AP203 export file could be created for transfer of the modified geometry to other CAD or CAE tools. During the live demo,

the bulkhead geometry was imported and changes to stiffener properties were demonstrated. The exaggerated angle stiffener geometry (Figure 53 below) was then exported and shown also in EB's ADAPT code. Several additional slides were employed to illustrate other types of parametric AP218 features such as plate cut-outs.

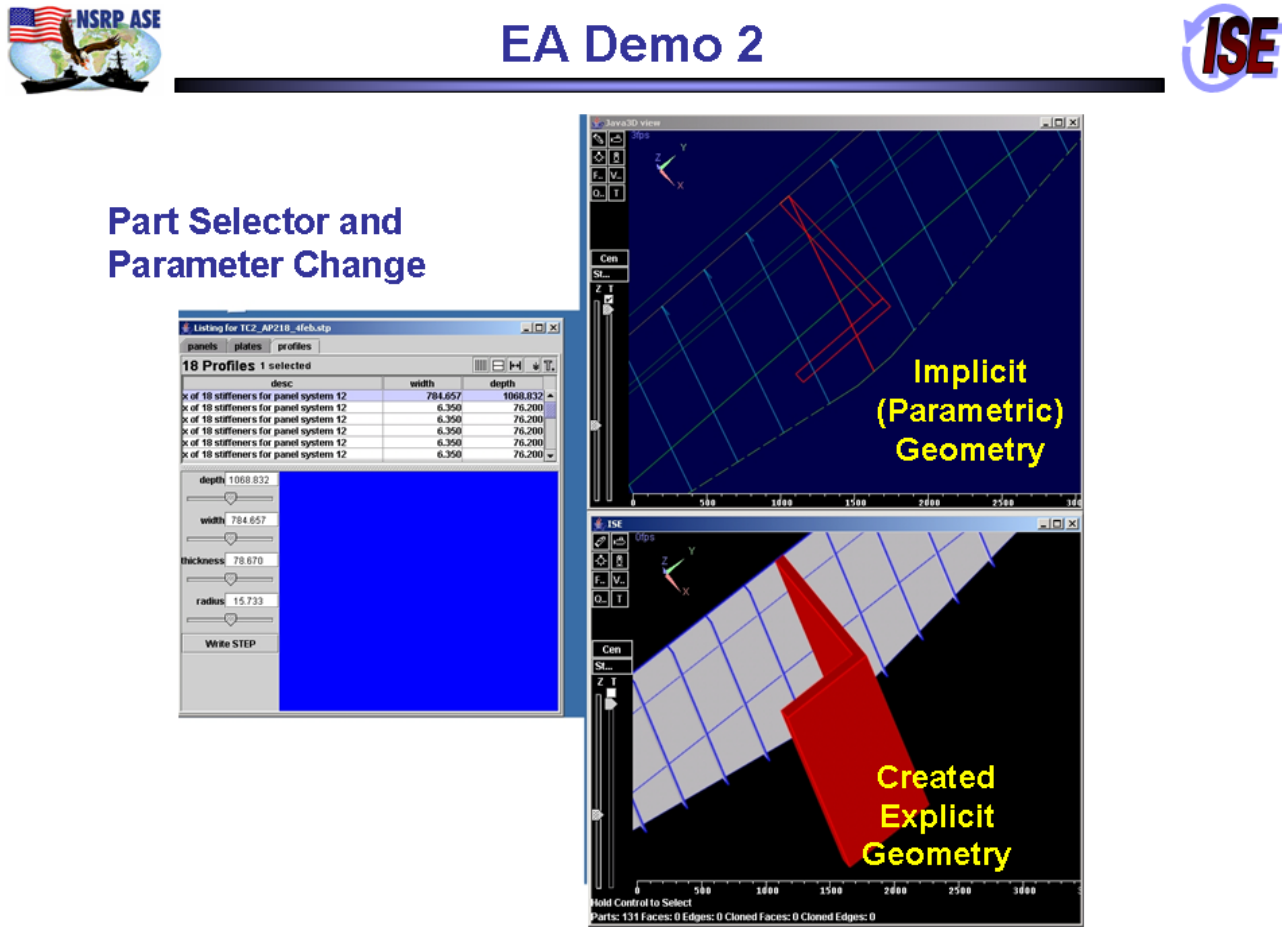


Figure 53: Dynamic Change to Stiffener

7 Electrotechnical Task

7.1 Scope

The NSRP ISE-4 Electrotechnical Task is developing an easy way for engineers to enter the data that defines an electrotechnical system in the areas of the design, fabrication and maintenance of ships and of on-shore naval support facilities. The input stage uses KSS KnowledgeManager to make a model of the system that is hierarchically organized as a tree-structure of entities. Each entity may be assigned attributes and both the entities and attributes may be annotated with comments and may be associated to relevant data- and informational-files. This project is providing a tool as part of KSS KnowledgeManager to transform that data into AP212 ARM-XML representation (Part 28 format) (see Reference 15), and ultimately to the AIM representation. The input stage, using KSS KnowledgeManager, is expected to be more intuitive and convenient to the design engineers than writing the XML directly.

Important antecedent work providing guidance for the development of AP212 implementations, especially in ship building and ship maintenance environments, are the SEASPRITE Project report "*Ship Product Model"*,(Reference 16) *(Ship common model),* and the usage guide for AP212 in ship environments, *"AP212 Ship Electrical Usage Guide"*(Reference 17).

7.1.1 Input

Most of the existing examples of the exchange of ship data, in other application protocols, are heavily weighted with geometric or geometry-based data and are made between different CAD systems. The engineering and design data of electrical systems depends less on geometrical description than that of most of the other ship system AP's. This project is demonstrating the collection and transfer of electrical system engineering data and will defer the collection and exchange of any electrical system geometric data. For example, the interconnections of pieces of equipment or other electrical devices is modeled and exchanged but the actual routing of the connecting cables or wires will not. The engineering design data of electrical systems requires different methods of storage, display, and manipulation than those available from CAD systems. The KSS KnowledgeManager application will be used for those tasks and will provide the translation of the data to AP212 XML output.

The KSS KnowledgeManager's hierarchical data structure is made up of completely general *KMEntities* and lists of *KMAttributes*. They are given particular significance in any context or design environment by means of their names and their associated comments and data files.

7.1.2 Output Format

Output from the KSS KnowledgeManager translator will be in XML and conforms to AP212 schemas prepared by EB.

The AP212 application protocol defines abstract devices and their connections but does not contain the definition of any particular electrotechnical equipment. This project utilizes the Common Parts Catalog to define specific equipment. The Common Parts Catalog, (CPC), is a library of commonly used equipment and devices developed by a consortium of U. S. shipyards. The CPC data has an XML form consistent with the AP212 ARM-XML (ISO 10303, Part 28) format.

7.1.3 Installations

The use cases addressed in this project and the actual development of the implementing translator and schemas will be confined to dealing with a representative selection of equipment from the following three types of facilities:

- Shore based communications facility: C4ISR equipment.
- Ship: Main power distribution equipment.
- Ship: Lighting power distribution equipment.

7.1.4 Data

The particular equipment that will be used in these categories will be:

- C4ISR equipment: Ethernet Optical Hub, Fiber Optic Distribution Panel, 16 Port Communications Server, 24 Port Ethernet Switch, DS3 Feedthru Panel, TNX-1100, DISA TNX-1100, KVM 2 Port Switch, Sun Server, Linux Server, KG-75, PSAX-1250, PSAX-2300, Client PC Workstation. Cables: Fiber Optic, CAT-5 RJ45, Coaxial, Video and PS-2 with applicable jacks and plugs.
- Power equipment: Diesel generators, circuit breakers, distribution panels, power cable connectors, and power distribution cables.
- Lighting equipment: Fixtures, controls and switches, wiring.

7.2 Design Details

7.2.1 KSS KnowledgeManager

7.2.1.1 KM tree structure

The KSS KnowledgeManager's hierarchical data structure is made up of completely general *KMEntities* and lists of *KMAttributes*. They are given particular significance in any context or design environment by means of their names and their associated comments and data files. In the specification of electrotechnical systems in KSS KnowledgeManager the high level structure of the tree-data will be arranged as is shown in the next two figures. Following this template or standard for data arrangement will allow the translation program to recognize the data types that will be transferred to particular AP212 entities and data structures.

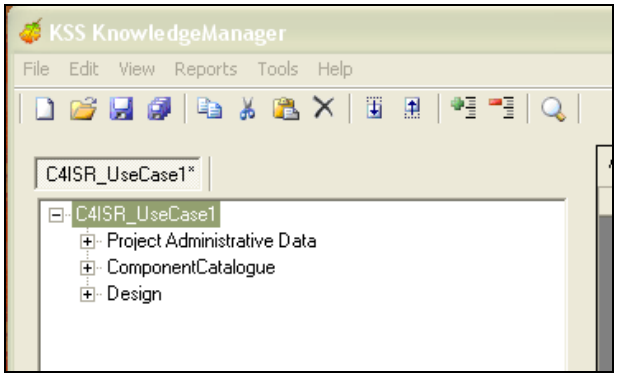


Figure 54: High level data structure for system design

At the highest level of data storage there will be the three classifications, (Figure 54), *Project & Administrative Data* for project-wide information, *Component Catalog* for definitions of equipment available for use in the system, and *Design* containing the equipment instances and relationships of the system design.

The specification of connected components in KSS KnowledgeManager will be done in terms of three high level entities: *Electrotechnical Equipment, Electrotechnical Cable*, and *Electrotechnical Connectors*. These items will always appear first as definitions in a *Component Catalog*, and instances of them will then appear in the *Design*. The catalog information will include general information, version specific information, and partial definitions having some of the components' parametrically defined data given particular values. The instances of the components in a design will have all required parametric data filled in, in particular, values for individual names, positions and connections, (see the next Section 7.2.1.2 and Figure 56.).

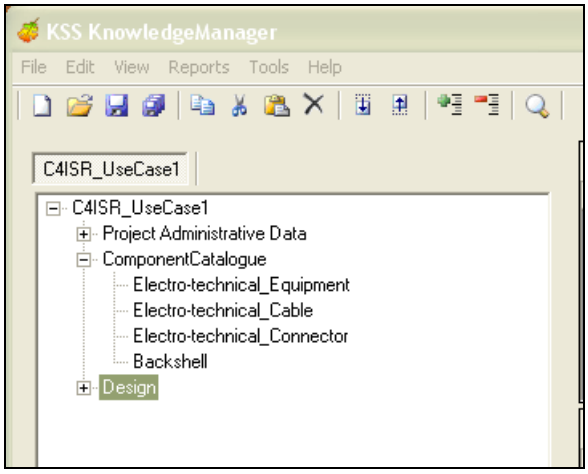


Figure 55: *Component Catalog* structure for an electrotechnical system design

7.2.1.2 Design Data Structure: Equipment, Connectors and Connections

Under the Design data category will be two levels of equipment specifications, first the instances of the equipment and cables and next, the connectors that belong to each of them. The entity names that appear in the entity tree are not constrained and may be assigned by a designer according to project needs. The attribute *named "Subtype"* should contain the name of the equipment's definition entity as specified in the *Component Catalog* section.

The connection association between two connectors will be managed by the specification of a derived attribute, see Attribute: *Connection pointer* and the derivation expression below it in Figure 56. This method will allow KSS KnowledgeManager to keep the connection information up to date.

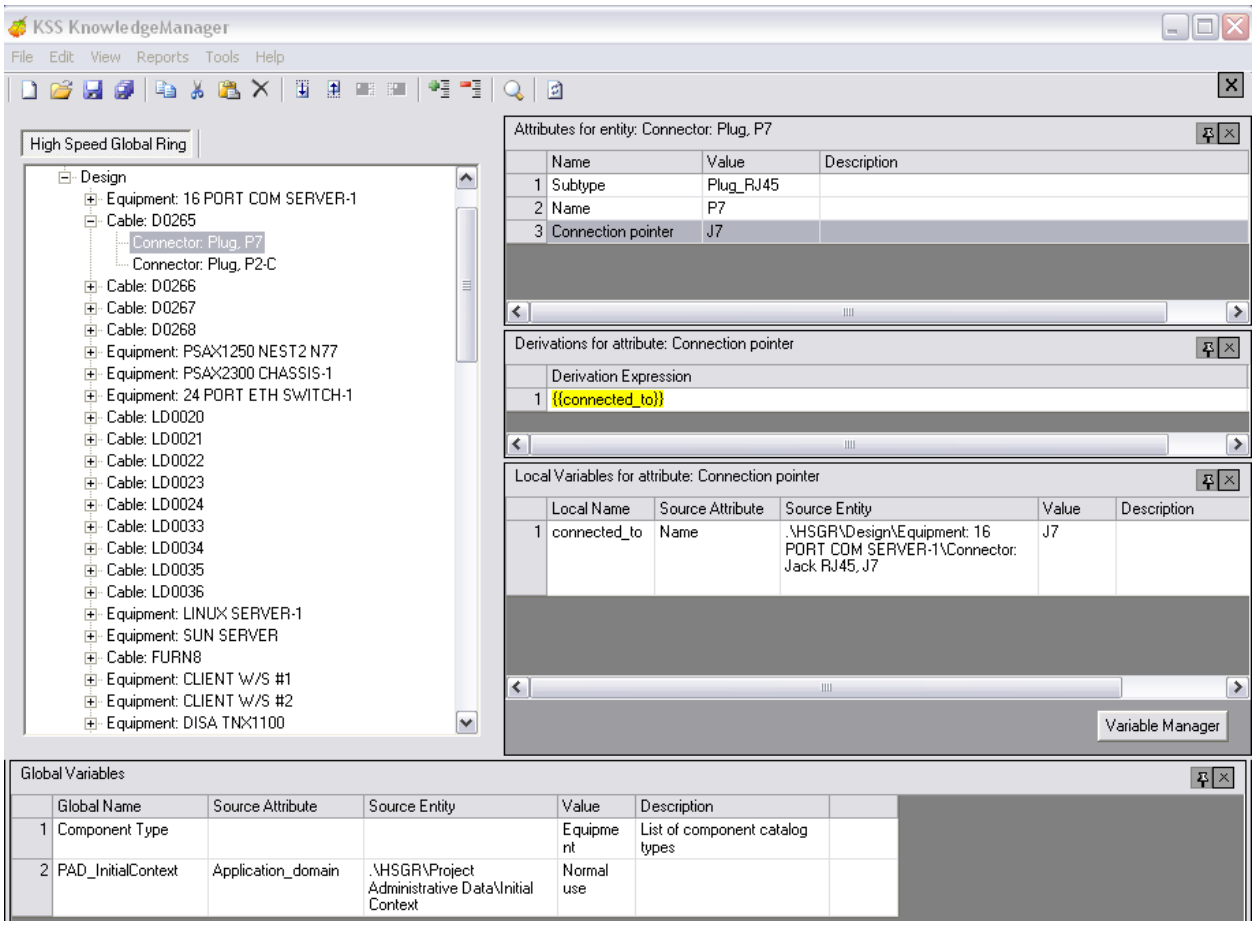


Figure 56: KM data structure for the *Design* section and specification of a *Connection*

7.2.1.3 Link to the Component Parts Catalog Information

Since the AP212 application protocol defines abstract devices and their connections but does not contain the definition of any particular electrotechnical equipment, this project will utilize the Common Parts Catalog to define specific equipment. The Common Parts Catalog, (CPC), is a library of commonly used equipment and devices developed by a consortium of U.S. shipyards. The attribute: called "Subtype" should contain the name of the CPC item that corresponds to the equipment instance. See the highlighted attribute below in Figure 57.

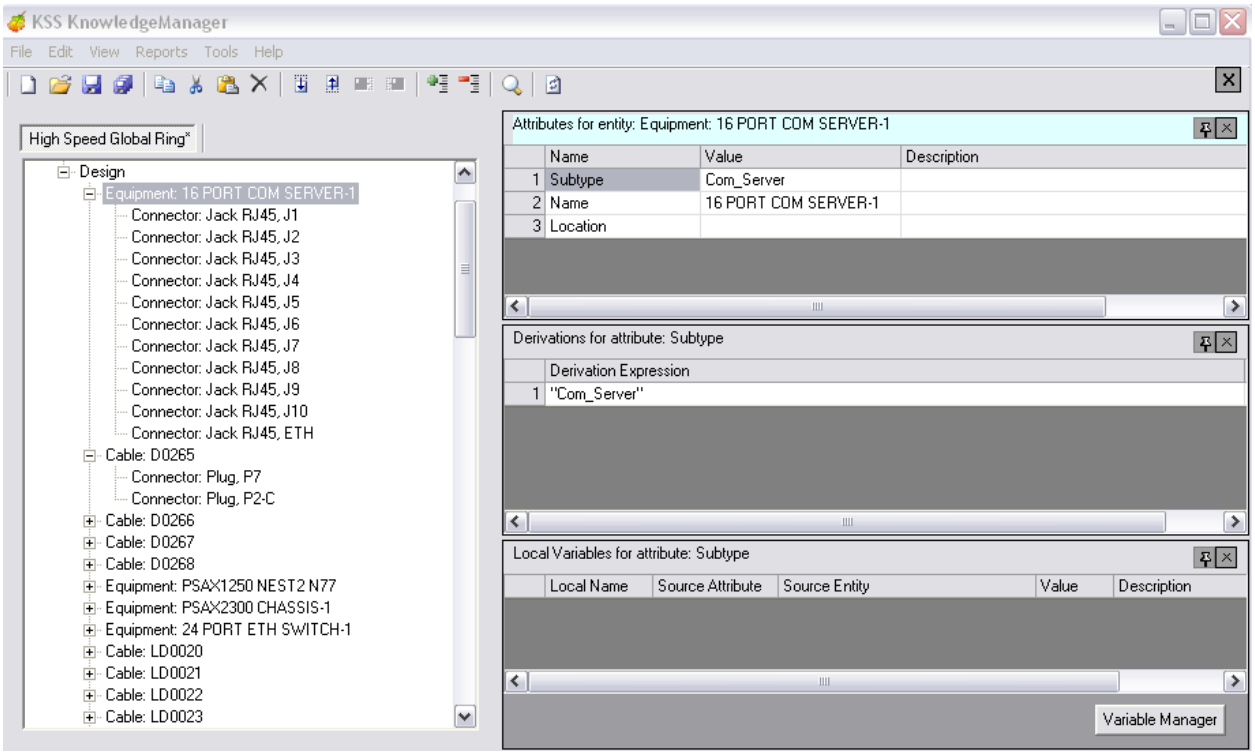


Figure 57: Specification of equipment's *Common Parts Catalog* name

7.2.2 Translator function

7.2.2.1 High Level View

The items in the *Component Catalog* section of the knowledge model will be translated to definition type items in AP212, *Design_discipline_item_definition* for equipment and cable and *Interface_terminal* for connectors. In order to facilitate a linkage in the ARM XML data structure between the generic definition entity, *Design_discipline_item_definition*, and the various equipment items in the Common Parts Catalog, the AP212 schema file has been augmented with sub-entities of *Design_discipline_item_definition*, one for each CPC item. The sub-entities, although located in the AP212 namespace of the AP212 schema file, don't constitute an addition to the AP212 protocol, but serve as a tight link to the equipment classification system embodied in the Common Parts Catalog.

Items in the *Design* section of the knowledge model will be translated into the AP212 entities, *Single_device*, *Terminal*, and *Connection* and their AP212 defined relationships.

These two types of translation and their relationships according to AP212 are shown in Figure 58. Here is shown one piece of equipment connected to one cable.

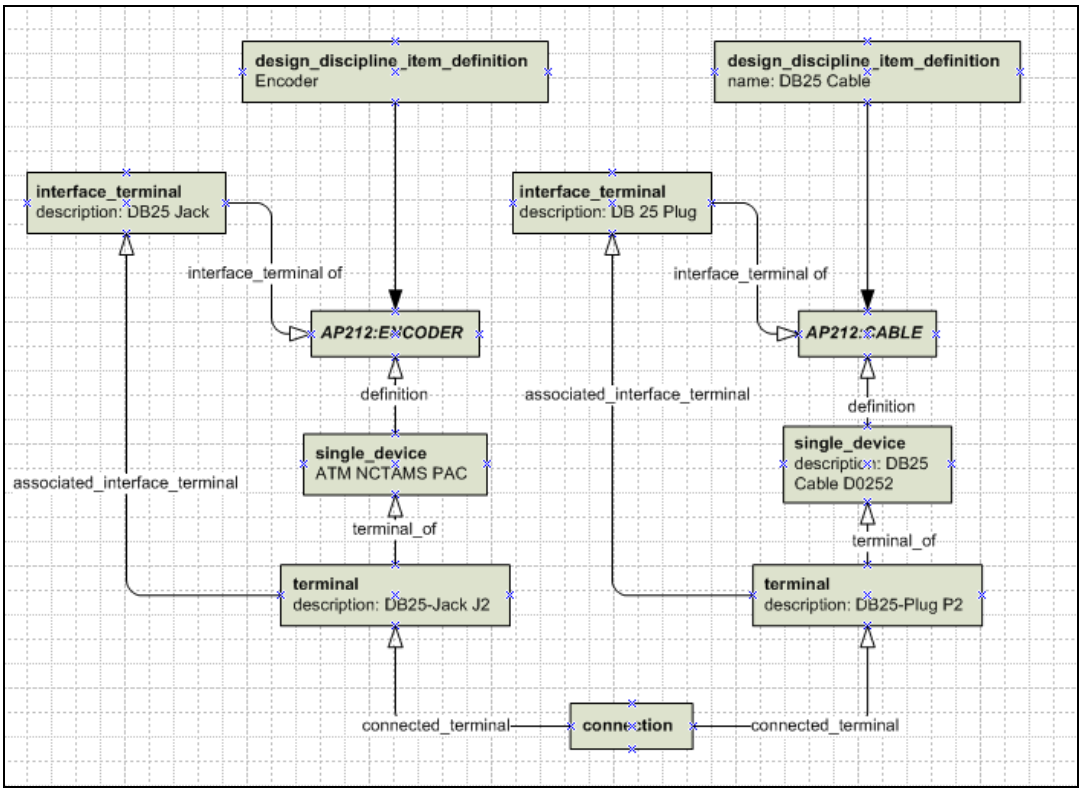


Figure 58: Example of translated AP212 structure

7.2.2.2 File Driven - Schema Driven

The translation process will be driven by the schema file in the sense that any equipment, cable, or connector, the name of which is found in the schema file, can be translated. The KSS KnowledgeManager program does not limit the items that can be translated. The process is shown below in Figure 59. Here the file, *KSS XML Output*, is the normal or native output storage for knowledge models.

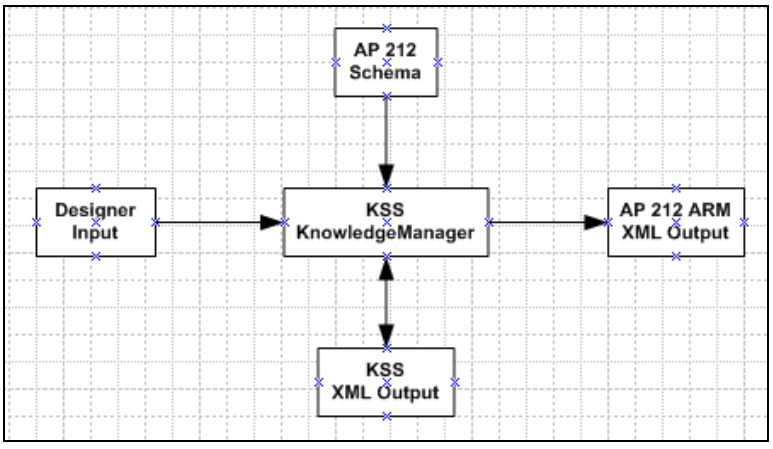


Figure 59: Schema File guides the translation

7.2.2.3 Dependence on CPC

The association between the AP212 definition entity and the CPC catalog-item and its data is accomplished through a new association, *Is_cataloged_as* established in the AP212 namespace of the AP212 schema file similarly to the introduction of the *Design_discipline_item_definition* subclasses. That association is shown below in Figure 60. The use of that combined schema to drive the complete translation process is shown in Figure 61.

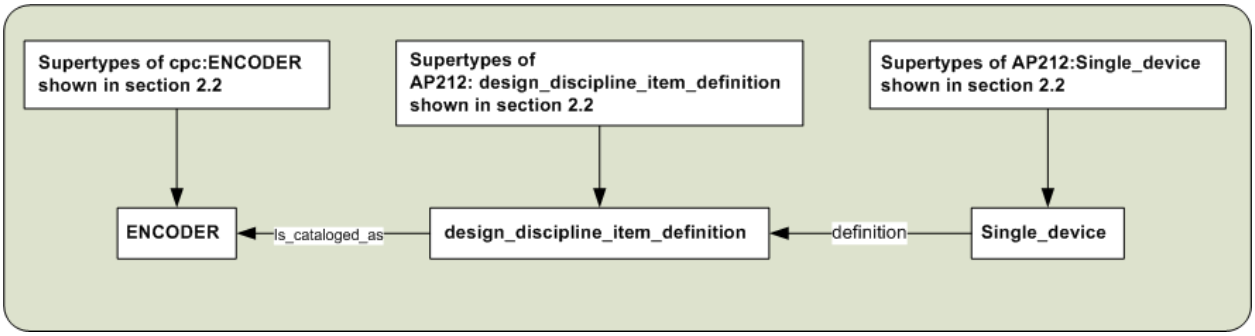


Figure 60: Introduction of the CPC to the AP212 schema

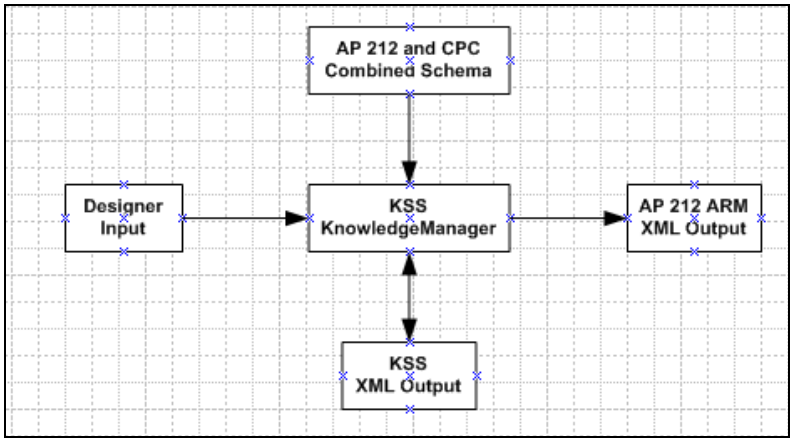


Figure 61: The translation guided by the combined schema

7.2.3 ARM XML Output

7.2.3.1 Entities, Associations and their subclasses

The AP212 entities used in the project include:

- Design_discipline_item_definition
- Interface_terminal
- Single_device
- Terminal
- Connection
- Document

The AP212 associations used include:

- interface_terminal_of
- definition
- associated_interface_terminal
- terminal_of
- connected_terminal
- implements
- document_assignment
- assembly_component_relationship

The CPC Entities adjoined to the ARM XML schema for the first two use-cases of this project are listed in Section 7.1.4. Others will be added as necessary for the third as the use-cases are executed.

7.2.3.2 AP212 Documents and Assemblies

7.2.3.2.1 Document Assignments

Each entity in the Knowledge model may be associated to relevant data- and informational-files. These files will be translated into AP212 format as per Figure 62.

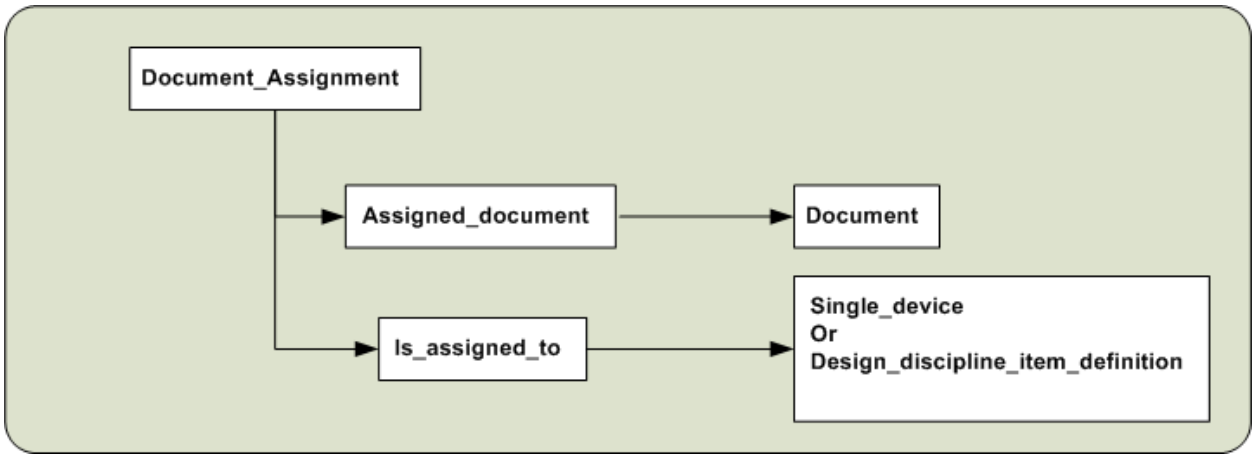


Figure 62: The Document_Assignment implementation

7.2.3.2.2 Assembly Relationships

Knowledge model assembly relationship between two equipment entities will be managed as follows: each Equipment entity will be assigned a value of Single Part or Assembly in the attribute named *Composition*. The Composition value in addition with the value in the attribute named *Container* and the derivation expression below it will allow KSS KnowledgeManager to keep the assembly relationship information up to date. Each Equipment entity, classified as either Single Part or Assembly, can identify

a Container through the derivation *contained_in* if it is part of an Assembly entity. The Container entity must be classified as Assembly. An entity classified as Single Part may not serve as a Container.

In Figure 63 below Equipment: EPCP Bus A has a value of "Single part" in the attribute *Composition.* It has a value of "EPCP (P400)" for attribute named "Container", thus establishing a relationship that states "EPCP(P400) is an assembly that contains Equipment: EPCP Bus A".

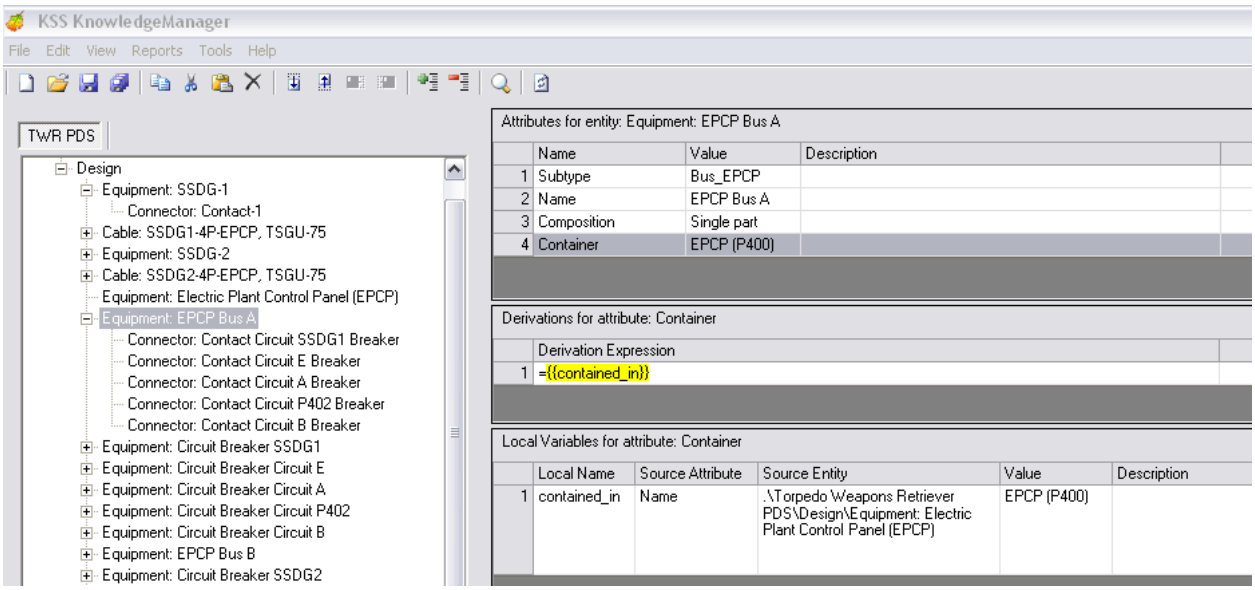


Figure 63: KM data structure for the *Design* section and specification of an Assembly Relationship

The AP212 Assembly_Component_Relationship implementation is shown below in Figure 64.

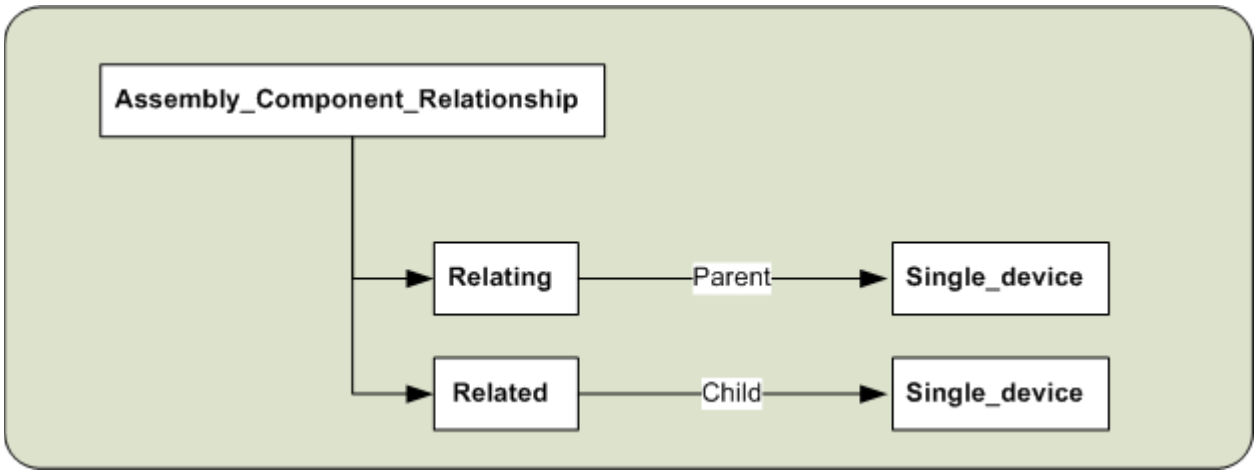


Figure 64: Assembly_Component_Relationship implementation

7.2.3.3 Idiosyncrasies of the ARM XML Schema

AP212 defines the relationship between *Design_discipline_item_definition* and *Interface_terminal* as from the *Interface_terminal* to the *Design_discipline_item_definition* and named *Interface_terminal_of*. The Part-28 schema for AP212 defines the same relationship as from *Design_discipline_item_definition* to *Interface_terminal* and named *Interface_terminal*.

There is a similar reversal of direction and change of name found in the definitions of the relationship between *Single_device* and *Terminal*.

The translator follows the schema file.

7.2.4 Subsequent Output

Transformation of the ARM XML data to AP212 AIM format, (part of this project), and possibly to AP212 EXPRESS format, (as will eventually be needed), will be done by programs and files collectively referred to as *Mediators*. These are the responsibility of EB, and for the most part, already exist.

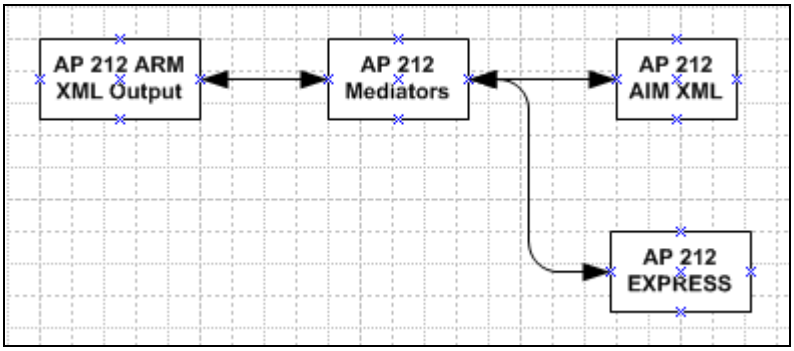


Figure 65: Further transformation of the output beyond ARM XML

7.3 Implementation Agreements

The subsequent sections detail the agreements that where reached among the ISE-4 team members when implementing ISO10303.212 Edition 1 (Reference 18).

7.3.1 XML and Translations between Part 28 and Part 21 Files

Traditionally when STEP is used for product data exchange, the translators produce a neutral data representation known as a STEP physical file, or a Part 21 file. This is specified in an International Standard known as: ISO 10303-21 - "Implementation Methods: Clear Text Encoding of Exchange Structure" (Reference 19). This remains the "official" format for the exchange of STEP files and is supported by the ISE-4 – Electrotechnical exchanges.

However, with the growing use of the internet and the World Wide Web, a new STEP format, called Part 28, has been developed which utilizes XML structures to encapsulate the STEP data. This is specified in a Technical Specification known as: ISO 10303-28 - "Implementation Methods: XML Representation of EXPRESS Schema and Data" (Reference 15), and is the preferred exchange format for several of the ISE Team members.

In order to accommodate exchanges using this new XML standard, the ISE Team has developed "Mediator" software that enables a Part 21 file to be transformed into a Part 28 file and vice-versa. Thus, the STEP test files can potentially be written in either Part 21 (standard STEP neutral file format) or Part 28 (XML format) and still be exchanged successfully among ISE-4 participants.

This project is producing Part 28 files from KSS KnowledgeManager files.

7.3.2 Library Components

KSS has established a parts library of Electrotechnical components for each of the test cases which is being used for the exchanges of the Electrotechnical Data. Complete descriptions of the Electrotechnical Library Components (including their material and engineering description) are provided with the AP212 and CPC schema documents and Knowledge models issued by this ISE-4 Project.

7.3.2.1 Common Parts Catalog (CPC)

All project data entities must exist in the CPC if their catalog data is to be used by the translator. Any data entities that don't exist in the CPC may possibly be translated using User Defined elements into the ARM XML file, subject to future agreements.

7.3.3 Agreements for ISE-4 – Electrotechnical Implementations

This section specifies the Implementation Agreements considered mandatory for translators developed and tested in the ISE-4 program based upon ISO-10303 AP212 Ed. 1 Exchanging Electrotechnical information. These Implementation Agreements follow the common framework adopted for ISE-2 and ISE-3 ESTEP translators by extending the ESTEP Implementation Agreements that were made for AP216, AP218 and AP227. The agreements are either specific to the ISE Project (labeled "ISE-4 – Electrotechnical Program") or recommended practice ("Recommended").

7.3.3.1 Electrotechnical Specific Implementation Agreements

The following rules and guidelines will be followed by all the ISE-4 – Electrotechnical AP212 implementations:

7.3.3.1.1 ISE-4 - Electrotechnical Task Specific

1. The minimal set of required entities and attributes will be specified for each Test Case in the AP212 Electrotechnical Schema Analysis File (Reference 20).

7.3.3.2 Common AP212 Ed. 1 Implementation Agreements

The following rules and guidelines will be followed by all the ISE-4 – Electrotechnical AP212 implementations:

7.3.3.2.1 Recommended Practice

1. All ISE-4 - Electrotechnical transfers should include Name and Description information, even if it is optional in the AP.

2. The use of GUIDs is required for all ISE-4 - Electrotechnical transfers.

7.3.3.3 Common ISE-4 – Electrotechnical Implementation Agreements

The following rules and guidelines will be followed by all the ISE-4 - Electrotechnical implementations:

7.3.3.3.1 Nomenclature Entities

In the construct of the Knowledge models of Electrotechnical data, the devices should be named in conformance with the nomenclature of the AP212 schema file in order for the translation program to recognize them and their data.

7.3.3.3.2 User Defined Elements

A user_defined element will eventually be required in the AP212 schema file. Unless the CPC contains every possible electrotechnical piece of equipment in a ship, a user_defined element must be available to store data that is not mapped elsewhere. Although generic, this element must validate against the AP212 schema to ensure proper structure and context.

7.3.3.3.3 GUIDs Stored and Preserved (ISE-4 - Electrotechnical Task)

Preprocessor/Postprocessor: The native system shall store and preserve a unique, immutable, and persistent globally unambiguous identifier (GUID) associated with each definition subtype instance. This GUID is implemented by the id entity in AP212.

- The native system shall use the GUID for the two following purposes in ISE-4 – Electrotechnical exchanges:
  1) to identify an unchanged part
  2) to provide a unique identifier for each part to enable it to be referenced from a Part 28 file for a different transfer
- The receiving software system shall store the GUID of all parts received with the exception of those which the team has agreed not to preserve. If a part is modified, such that it will affect translation at other systems, then a new GUID shall be assigned, corresponding to the site where the modification is made. Otherwise, the GUID of a part is preserved.

7.3.3.3.4 Globally Unambiguous Identifier usage (Recommended)

The following section defines the format and use of Globally Unambiguous Identifiers in the ISE Project. Globally Unambiguous Identifiers (GUID) have been incorporated as part of the Ship Common Model and are used by all the Shipbuilding Application Protocols. They are used to support the following incremental exchange scenarios:

- Separate exchanges of items, definitions, and representations within an AP while preserving the relationships between these application objects. For example, the first exchange may only identify the single_device, by sending only single_device entities, while the second exchange may contain the specific properties and representations, by sending single_device and encoder (design_definition) entities.
- Separate exchanges of a single ship model or shore facility model using AP212 while preserving relationships between selected AP212 application objects. For example equipment that is connected to other equipment in a different compartment.
- Separate exchanges of a single model using several shipbuilding AP's while preserving relationships between application objects in different AP's. For example, the first exchange may include an encoder entity in AP212 and the second may include the entire compartment arrangement from AP215 whose compartment location identifier apply to the electrotechnical equipment.

Globally unambiguous identifiers and incremental exchanges are important for shipbuilding for the following reasons:

- Volume of data in a ship or shore-based design. It is simply not practical to exchange an entire ship or shore-based design in one exchange. Typical data exchanges focus on specific areas of the ship, facility, or specific systems. It is important that the result of these multiple exchanges results in a model which preserves the basic relationships between shipbuilding, facility, and electrotechnical elements.
- Long design and production time. A ship design is developed over an extended period, as long as several years for military ships. Similarly, the production information is also developed over a similar period. It is important to exchange data at any point in the design or production cycle while preserving the basic relationships between shipbuilding, facility, and electrotechnical elements.
- Instance based design. A ship or facility design is composed of individual occurrences of parts at specific locations. It is important to uniquely identify and track each individual instance through design, production, and eventually to operation.
- Ship Class. A class of ships may be designed similar to a single ship design, with specific exceptions called out. It is composed of individual occurrences of parts at specific locations on the ships. It is important to uniquely identify and track each individual instance in the ship class through design, production, and eventually to operation.
- Concurrent design and production. Typically, production of a ship begins before the entire design is complete. Both the design and production information is assembled incrementally. Further, in order to manage changes, it is important to uniquely identify each individual instance on the ship through design and production.
- Co-design. Several design agents may collaborate on the design of a ship, ship class, or facility. This is feasible only if each company can identify the incremental changes made by the other partner.
- Co-production. Several shipbuilders may collaborate on the production of a ship, ship class or facility. This is feasible only if each company can identify the incremental changes made by the other partner.
- Collaboration. A number of companies may collaborate in the design and production of a ship, ship class, or facility. This is feasible only if each company can identify the incremental changes made by each company.

7.3.3.3.4.1 Functional requirements

Since GUIDs are used to support incremental exchanges, it is not possible to fully specify the functional requirements for GUIDs in the Shipbuilding Application Protocols. The functional requirements for GUIDs are as follows:

- The GUID must be globally unique and persistent, within a ship design, across ship designs, across facilities, and across companies. It must point to the instance that it designates or to nothing at all (as when the instance has been deleted).
- Each software system must assign persistent GUIDs to each application object created or updated.
- Each software system must support access to the application object via the GUID.
- Each software system must store the GUID of all application objects.

- Each software system must return the same GUID as it received, if the application object was not changed.

7.3.3.3.4.2 Implementation constraints

A number of different implementation schemes and technologies can be used to support this notion of GUID. However, there are several implementation constraints, which should be considered:

- Must support interoperability across companies, applications/systems, and technologies. (The identifier should not be opaque but must expose the essential data elements needed for persistent identification.
- Must support systems that use value-based keys as well as systems that use system-generated object identifiers (should not impose the requirement on value-based systems to store an additional attribute for system-generated identifiers.)
- Instance identifiers must contain globally unique type identifiers but also have additional requirements. Consequently, instance identifiers carry more information than type identifiers.
- Identifier should be character-based and the syntax should be specified so that industry standard tools and algorithms can be used.
- Identifier should support existing Part 28 XML usage.
- Identifiers should be based on unique identification of single data objects in the software application. The Ship Common Model typically uses four or more objects to describe a single data object: item, Design_definition, Functional_definition, and Manufacturing_definition. While it is important to uniquely identify each STEP application object, it is also important to the software application to uniquely identify the data object, which generates these STEP application objects.

7.3.3.3.4.3 ISE recommended GUID format

The Integrated Shipbuilding Environment (ISE) Project has adopted and is implementing translators based upon the following GUID format.

The ARM attribute Global_id.id (AIM attribute applied_classification_assignment.id) is written as the string in the following format:

‘<XX_Key x_href=YY Id=WW x-owner=ZZ x-rev=AA />’

Argument definition:

XX_Key - XX is name of related ARM entity, e.g. Design_discipline_item_definition. This argument is required. The first letter of application object is upper case, rest lower case. Suffix = "_Key".

x-href=YY - YY is the URL of SOAP server and/or database. This argument is required.

Id=WW - WW is a string which identifies object. Id is required only for definable_objects. The Id is one of two types:

Value (transparent usage) - a unique value which identifies this instance
GUID (opaque usage) - a system generated identifier. Must be prefixed by "guid:"

x-owner=ZZ - ZZ is a string which identifies the object. This argument is required only for Definition objects. Same string as Id of related definable_object.

x-rev=AA - AA is revision number. This argument is required only for Definition objects. Same string as Definition.version_id attribute.

The combination of (XX_key, x-href, Id) is unique for definable_objects.
The combination of (XX_key, x-href, x-owner, x-rev) is unique for Definitions.

EXAMPLE. The following example illustrates this usage for the related Encoder entity in AP212:

<Encoder_key x-href="urn:iso10303-28:schema/ap212" id="guid:E3ec96fc5-56fe-4b6d-be60-c81ea39ad3d3" x-owner="guid:Ab7d98fce-5f77-401f-b24c-b8cc0c7d2b61" x-rev="A"/>.

7.3.4 AP212 Schema Analysis

A schema analysis is included as Reference 20. This document outlines the agreed upon schema elements and attributes between EB and KSS. The ARM XML file produced by each test case has been validated against this schema. The schema is a work in progress file and will be updated or modified as required to fulfill future test case requirements.

7.4 Test Data for ISE-4 Demonstration of Electrotechnical Exchange

The Electrotechnical Task has developed an easy way for engineers to enter the data that defines an electrotechnical system. The input stage uses KSS KnowledgeManager to make a model of the system that is hierarchically organized as a tree-structure of entities. Each entity may be assigned attributes and both the entities and attributes may be annotated with comments and may be associated to relevant data- and informational-files. The data is then transformed into an AP212 ARM-XML representation and ultimately to the AIM representation. The input stage, using KSS KnowledgeManager, is more intuitive and convenient to the design engineers than writing the XML directly.

The process described was demonstrated and validated in this project through three test cases:

1. C4ISR Test Case
2. Ship Main Power Distribution Test Case
3. Lighting Power Distribution Test Case

Each test case is judged a Pass or Fail depending on the following validation criteria:

1. The output ARM XML file conforms to its schema. That includes checking the XML data structure, data typing and presence of mandatory fields.
2. By manual inspection to make sure that all the applicable data in the Knowledge model has been transferred into the ARM XML file.

7.4.1 C4ISR Test Case

This case is divided into three sub-cases to facilitate our development process.

1. Minimum Use Case: In order to develop and test the concepts of the translation process from the KSS KnowledgeManager output in XML format to an XML file that satisfies the STEP AP212 ARM-XML schema, we worked at first with a knowledge model containing a single piece of equipment. The equipment is an ATM, an Asynchronous Transfer Module, a type of data encoding device.

The design and implementation tasks for the minimum use case are divided into the following four categories:

   a. Identification of the needed KSS KnowledgeManager XML entities and Attributes.

   b. Identification of the needed AP212 ARM-XML entities and attributes.

   c. Identification of correspondences between the two XML schemas (KSS and AP212): entity to entity and data-item to data-item.

   d. The KnowledgeManager XML file will be translated into an ARM-XML file and validated against the AP212 schema.

2. Simple Use Case: continues the emphasis on process extending to a small number of interconnected equipment. In this simple use case four additional pieces of electrical equipment are introduced: feedthru, cable, jack and plug; as well as the relations between this equipment. This use case implements elements and attributes required to identify connections from related equipment terminal to terminal and the terminal's properties.

The connections are listed below:

   a. Encoder to Cable (jack to plug)

   b. Cable to Feedthru (plug to jack)

   c. Feedthru to Cable (jack to plug)

   d. Cable to Encoder (plug to jack).

3. Extended Use Case: containing approximately twenty additional pieces of equipment demonstrates the full capability of the translator in a C4ISR environment.

7.4.2 Ship Main Power Distribution Test Case

The test case involves the TWR's Electrical Power Distribution System. The data represented shows connectivity of components and distribution of power. The Electrical Power Distribution System was modeled from two SSDGs (Ship's Service Diesel Generator) with related 24VDC Starting Batteries Bus via the Electric Plant Control Panel (EPCB-400), Engine Room Power Panel (P402), Engine Room Power Panel (P401) and 120V Main Distribution Panel (L-100) to their end line user equipment.

7.4.3 Lighting Power Distribution Test Case

This data was taken from the TWR ship lighting systems data, including Pilot House, Galley and Engine Room lighting.

7.4.4 Sources of Test Data

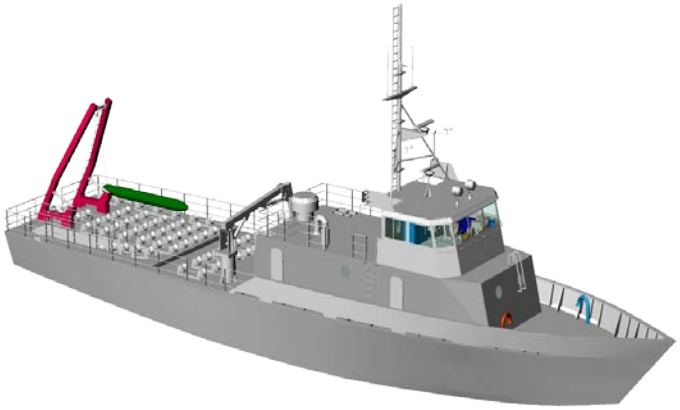


Figure 66: Torpedo Weapons Retriever (TWR)

The test data for Test Case one (C4ISR) was provided by Space and Naval Warfare Systems Center, San Diego (SPAWAR).

The test data for Test Cases two and three were derived from the data for a U.S. Navy ship - a 120 foot Torpedo Weapons Retriever (TWR), built in 1985. The drawings for this ship have been cleared for public release and have been provided to the ISE-4 – Electrotechnical team by NSWCCD. This project has produced test files based upon this model and posted them to the ISE web host. The test files, a short narrative, related data, and wherever possible the associated native files shall be made publicly available on the TWR product model repository at NSWCCD. The intent is to produce a set of standardized test files that are representative of tools, techniques, and standards specific to each shipyard environment. These test files are also expected to provide the basis for STEP AP usage guides and abstract test suites.

7.5 Electrotechnical Task Results

The ISE-4 Electrotechnical Task has developed a component design framework for data translation and archival using STEP AP212. This has emphasized both the Knowledge Management (knowledge & rules) and Engineering Data (results) aspects of the problem.

In solving this problem, the project has created reusable STEP toolsets (mediator stylesheets) to be placed in the DONXML Repository.

The project has tested AP212 with diverse electrotechnical test cases and demonstrated the joint use of STEP and XML with new programming resources.

These efforts will be continued as part of a follow-on project awarded by NSRP, which is called ISE-5. In ISE-4, the project team prepared translators for working with AP212 format data. However, this is only part of the complete data set required by designers and engineers on an electrotechnical project. The full data set includes not only engineering data (AP212), but also geometric data (AP227, AP203, AP214). Therefore, in ISE-5, KSS will further the work begun in ISE-4 by using the KSS framework to translate a more complete set of electrotechnical data. The use cases defined for ISE-4 will be expanded to include not only engineering data and calculations (AP212), but also 2D and 3D geometry (AP227, AP203, AP214) representative of a more complete set of documentation. In addition, KSS will support joint service implementations under other projects that will enhance the benefits from the electrotechnical efforts under ISE-5.

8 ROI ANALYSIS

As part of the ISE-4 Project, a Return on Investment (ROI) analysis was performed on one of the design processes identified for the project. The task selected was the one for electrotechnical systems which was described in Section 7 above. There were two major recommendations made as a result of this ROI analysis.

- Based on the anticipated benefits of interoperability, the Navy and shipbuilders should continue to be actively engaged in the process of creating tools that enable interoperability.
- Efforts should be made to commercialize the translators funded by NSRP in order to minimize the costs of implementation and maximize the benefit received from interoperability.

The complete ROI analysis has been published as a deliverable of the ISE-4 Project, and is entitled "ISE-4 ROI Task - Analysis and Recommendations" (Reference 21). The following is a summary of the results of that report:

The ROI analysis was conducted by:

- Selecting a task from the ISE-4 interoperability projects (electrotechnical interoperable processes were selected)
- Performing a lifecycle process analysis on the selected task
- Creating a lifecycle analysis framework (assumptions, constraints, process and method) using the selected task
- Creating a baseline framework using data from the ISE-4 Project
- Identifying costs and cost savings from interoperability
- Performing cash flow analysis, Net Present Value (NPV), and ROI calculations using the baseline and interoperability cost and cost savings data

The ROI calculations presented in this report are speculative in nature, having been derived from assumptions and estimates rather than actual results. A more exhaustive benchmark could be performed in the future using the identified framework if desired.

Because the calculations are based upon estimates rather than hard data, three cases have been formulated, conservative, most likely, and best case. Assumptions for each of these cases are identified allowing the reader to examine and evaluate the analysis performed.

The selected task for analysis was the electrotechnical design and engineering process. A lifecycle approach was used in the analysis focusing on the power distribution system, the lighting system, and C4ISR systems. This includes concept design, preliminary design, production design, production, and operation (including system re-fit design and engineering). Documents and data were taken from the torpedo weapons retriever (TWR) power and lighting systems and the SPAWAR C4ISR test cases used during the ISE-4 Electrotechnical Task.

Based on the data, assumptions, and estimates the ROI analysis identifies a 20% to 40% benefit to be gained from interoperable electrotechnical processes where data and documents are re-used rather than re-created. More than half that benefit is realized during the operating life of the ship by re-using rather than re-creating documents during system re-fit design and engineering efforts.

Based on the findings it is recommended that the shipbuilding supply chain continue to be actively engaged in the process of creating tools that enable interoperability and that these tools be commercialized to maximize the benefits received by the stakeholders.

9 TECHNOLOGY TRANSFER

Technology Transfer has remained a major thrust of the ISE-4 Project. The most important activity was the Final Joint Project Demonstration which was held in Washington, D.C. in April 2006. This demonstration involved all four ISE-4 Tasks, and was presented to a diverse audience of shipyard and Navy personnel.

ISE-4 Project presentations were given at several major international events. These included:

- A paper entitled “Improving Interoperability through use of ISO 10303” (Reference 22) was presented at the 12th International Conference on Computer Applications in Shipbuilding in Busan, Korea in August 2005
- A paper entitled “Enhancing Interoperability Throughout the Design & Manufacturing Process” (Reference 23) was presented at the Ship Production Symposium in Houston, Texas in October 2005
- A presentation entitled “Developing an Integrated Shipbuilding Environment for the Ship's Life Cycle” was presented at SHIPTECH 2006 in Panama City, Florida in January 2006
- A presentation entitled "STEP Shipbuilding Demonstration Highlights Integrated Shipbuilding Environment 4 (ISE-4)" was presented at the Open Technical Forum of the ISO TC 184/SC4 Meeting in Toulouse, France in June 2006

In addition, ISE-4 Project presentations were given at three NSRP Panel Meetings to joint meetings of several panels. These included: Providence in May 2005, San Diego in September 2005, and Seattle in May 2006.

Tools, implementers agreements and other documentation are publicly available on the ISE Tools Website (www.isetools.org).

Modifications and enhancements required to the ISO STEP Application Protocols also have been filed and are listed in Appendix A.

.

10 SUMMARY

One key to improving productivity in U.S. shipyards is to develop interoperability between yards and processes throughout the ship's life cycle. Information interoperability is the sharing of information across system, application, and organization boundaries. The next step toward the NAVSEA vision of ONE SHIPYARD, via the next generation PLM-based IPDE work, should be the establishment of a consensus regarding what constitutes the digital design deliverable. Today design deliverables are conveyed via drawings (which are human but not computer interpretable). These paper-based deliverables are specified to the minutiae (down to the shape and color of arrowheads and leader lines on schematics). No such consensus exists for the digital product model (even though it drives the entire acquisition process). We need an implementable and enforceable yet open specification of the digital product model for naval shipbuilding.

It should be emphasized that information interoperability is the vehicle by which applications interact within the IPDE system itself and, consequently, to other applications within the IPDE enterprise. Only secondarily is it the vehicle for data interchange between companies (different IPDE's).

The ISE architecture (and prototypes) have defined how to create the formal documentation of the requirements for the digital product model deliverable. This is an exacting and demanding process, but it should be the cornerstone of any Navy policy that seeks to sustain the ONE SHIPYARD vision. The goal of this activity is to specify the digital product model for shipbuilding, to a level of detail and aligned with current Web capabilities, that enables the production-worthy sharing of the product model between and within various IPDE deployments.

The ISE-4 Project has successfully tackled this interoperability problem by implementing standards-based solutions in four disciplines. The solutions ranged from implementing new application protocols that had not yet been tested (e.g. AP215 for Ship Arrangements) to using established application protocols in unique ways to support shipbuilding requirements (e.g. AP209 for Engineering Analysis, AP212 for Electrical, and AP218 for both Steel Processing and Engineering Analysis). Thus, the ISE-4 Project represents a major step in completing the Information Interoperability Roadmap (Figure 4) by addressing four separate disciplines and developing interoperability tools that assist shipbuilding from early stage design through manufacturing and even for the life cycle of the ship.

APPENDIX A – ISO ENHANCEMENTS

During the course of this project, it became evident that some enhancements were required to the ISO STEP Standards. These error fixes and required improvements were revealed during the implementations of AP215 (during the Ship Arrangements Task), AP218 (during the Steel Processing Task), and AP212 (during the Electrotechnical Task).

The required enhancements have been identified to ISO and submitted as SEDS (STEP Enhancement and Discrepancy Reports) to be resolved and included in the next updates (Technical Corrigenda, Amendments, or new Editions) of the appropriate application protocols. Those enhancements are listed below for reference purposes.

A.1 AP215

The following STEP Enhancement and Discrepancy Reports (SEDS) for AP215 were submitted as a Result of ISE-4 Ship Arrangements Task.

SEDS #215-002

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 274, Clause 5.1.5.2.4
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: In paragraph 5.1.5.2.4 it seems that /PDR_NAME('compartment property') should not be operative since a new 'name' is substituted depending on the considered property.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): This Path should not set requirement for the PDR name; the name is set in the mappings of the Compartment_property subtypes.

SEDS #215-003

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 294, Clause 5.1.6.16.2
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: In paragraph 5.1.6.16.2 'numeric_measure' has been substituted by 'count_measure'
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Numeric_measure is not included in the AP215 AIM schema.
  - Mapping should be changed to use count_measure instead.

SEDS #215-004

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 391, Clause 5.1.16.1.4
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: In 5.1.16.1.4 Space_product_structure.version_id maps to /VERSION_ID(product_definition)/.

  - Similar structure in AP216 (Ship_moulded_form.version_id) maps to /VERSION_ID(group)/.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Use same mapping as AP216: /VERSION_ID(group)/.

SEDS #215-005

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 426, Clause 5.1.20.2.5
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: In 5.1.20.2.5 Compartment_functional_definition to Compartment (as defined_for).
  - Add {/CLASS_ID(product_definition, 'compartment'/} to end of Mapping Path.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Add {/CLASS_ID(product_definition, 'compartment'/} to end of Mapping Path.

SEDS #215-006

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 275, Clause 5.1.5.2.8
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: In 5.1.5.2.8 Path should be to named_unit rather than derived_unit.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Path should be to named_unit rather than derived_unit.

SEDS #215-007

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 428, Clause 5.1.20.4.5
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: In 5.1.20.4.5 Deck_zone_functional_definition to Deck_zone (as defined_for).
  - Add {/CLASS_ID(product_definition, 'deck zone'/} to end of Mapping Path.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Add {/CLASS_ID(product_definition, 'deck zone'/} to end of Mapping Path.

SEDS #215-008

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 431, Clause 5.1.20.7.5
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: In 5.1.20.7.5 Zone_functional_definition to Zone (as defined_for).
  - Add {/CLASS_ID(product_definition, 'zone'/} to end of Mapping Path.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Add {/CLASS_ID(product_definition, 'zone'/} to end of Mapping Path.

SEDS #215-009

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 279, Clause 5.1.5.4.3
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: In 5.1.5.4.3 Delete extraneous colon at end of Path
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Delete extraneous colon at end of Path

SEDS #215-010

- Part and Clause where Issue Originates: WG3N1230_IS215, ARM EXPRESS
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: Add superstructure_zone to Pre_defined_zone_function Enumeration TYPE.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Add superstructure_zone to Pre_defined_zone_function Enumeration TYPE.

SEDS #215-011

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 275, Clause 5.1.5.2.6
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: Mapping to Shape_aspect in conjunction with external_instance_reference in AP215 document is incorrect.
  - Replace with Product_definition_relationship mapping.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): In 5.1.5.2.6 compartment_design_definition to external_instance_reference (as boundaries), replace Mapping Path in its entirety to:

  5.1.5.2.6 compartment_design_definition to external_instance_reference (as boundaries)
  AIM element: PATH
  Reference path: product_definition_shape =>
  property_definition
  property_definition.definition->
  characterized_definition = characterized_product_definition
  characterized_product_definition = product_definition
  product_definition
  {/CLASS_ID(product_definition, 'compartment'/}
  product_definition <-
  product_definition_relationship.relating_product_definition
  product_definition_relationship
  {product_definition_relationship.name = ‘compartment boundary’}
  product_definition_relationship.related_product_definition->
  product_definition
  {([/CLASS_ID(product_definition, 'moulded form')/]
  [/EXT_INST_REF(product_definition, 'ship moulded form schema', 'moulded form')/])

([/CLASS_ID(product_definition, 'structural system')/]
[/EXT_INST_REF(product_definition, 'ship structures schema', 'structural system')/])}

SEDS #215-012

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 280, Clause 5.1.5.4.6
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: Mapping to Shape_aspect in conjunction with external_instance_reference in AP215 document is incorrect.
  - Replace with Product_definition_relationship mapping.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): In 5.1.5.4.6 zone_design_definition to external_instance_reference (as boundaries), replace Mapping Path in its entirety to:

  5.1.5.4.6 zone_design_definition to external_instance_reference (as boundaries)
  AIM element: PATH
  Reference path: product_definition_shape =>
  property_definition
  property_definition.definition->
  characterized_definition = characterized_product_definition
  characterized_product_definition = product_definition
  product_definition
  {/CLASS_ID(product_definition, 'zone'/}
  product_definition <-
  product_definition_relationship.relating_product_definition
  product_definition_relationship
  {product_definition_relationship.name = ‘zone boundary’}
  product_definition_relationship.related_product_definition->
  product_definition
  {([/CLASS_ID(product_definition, 'moulded form')/]
  [/EXT_INST_REF(product_definition, 'ship moulded form schema', 'moulded form')/])
  ([/CLASS_ID(product_definition, 'structural system')/]
  [/EXT_INST_REF(product_definition, 'ship structures schema', 'structural system')/])}

SEDS #215-013

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 278, Clause 5.1.5.3.9
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: Mapping to Shape_aspect in conjunction with external_instance_reference in AP215 document is incorrect.
  - Replace with Product_definition_relationship mapping.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): 5.1.5.3.9 deck_zone_design_definition to external_instance_reference (as constituent_compartments), replace Mapping Path in its entirety to:

  5.1.5.3.9 deck_zone_design_definition to external_instance_reference (as constituent_compartments)
  AIM element: PATH

Reference path: product_definition_shape =>
property_definition
property_definition.definition->
characterized_definition = characterized_product_definition
characterized_product_definition = product_definition
product_definition
{/CLASS_ID(product_definition, 'deck zone'/}
product_definition <-
product_definition_relationship.relating_product_definition
product_definition_relationship
{product_definition_relationship.name = ‘compartment in deck zone’}
product_definition_relationship.related_product_definition->
product_definition
{[/CLASS_ID(product_definition, 'compartment')/]
[/EXT_INST_REF(product_definition, 'ship arrangement schema', 'compartment')/]}

SEDS #215-014

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 278, Clause 5.1.5.3.10
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: Mapping to Shape_aspect in conjunction with external_instance_reference in AP215 document is incorrect.
  - Replace with Product_definition_relationship mapping.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): 5.1.5.3.10 deck_zone_design_definition to external_instance_reference (as deck_for_zone), replace Mapping Path in its entirety to:

  5.1.5.3.10 deck_zone_design_definition to external_instance_reference (as deck_for_zone)
  AIM element: PATH
  Reference path: product_definition_shape =>
  property_definition
  property_definition.definition->
  characterized_definition = characterized_product_definition
  characterized_product_definition = product_definition
  product_definition
  {/CLASS_ID(product_definition, 'deck zone'/}
  product_definition <-
  product_definition_relationship.relating_product_definition
  product_definition_relationship
  {product_definition_relationship.name = ‘deck for deck zone’}
  product_definition_relationship.related_product_definition->
  product_definition
  {([/CLASS_ID(product_definition, 'moulded form')/]
  [/EXT_INST_REF(product_definition, 'ship moulded form schema', 'moulded form')/])
  ([/CLASS_ID(product_definition, 'structural system')/]
  [/EXT_INST_REF(product_definition, 'ship structures schema', 'structural system')/])}

SEDS #215-015

- Part and Clause where Issue Originates: WG3N1231_IS215, Page 446, Clause 5.2.2.2.1 and Page 970 AIM Express G Figure H.28.
- Other Parts Affected by the Issue: WGN1231_IS215 (AIM SF Express) and WG3N1232_IS215 (AIM LF Express)
- Related Issues: 215-011, 215-012, 215-013, 215-014.
- Summary/Abstract/Keywords:
- Problem Description: In 5.2.2.2.1, remove Shape_aspect from external_identification_item Select as no longer needed due to SEDS 215-011 through 014.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): remove Shape_aspect from external_identification_item Select

SEDS #215-016

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 218, Clause 5.1.2.4.4
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: AP227 Ed2 placed Global_id assignment on planned_physical_plant_item application object rather than on plant_item.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): In 5.1.2.4.4 space_connection_relationship to external_instance_reference (as connecting_system), change mapping path reference to AP227 object to:

  [/EXT_INST_REF(product_definition, 'plant spatial configuration', 'planned physical plant item')/]}

SEDS #215-017

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 391, Clause 5.1.16.1.6
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: AP227 Ed2 placed Global_id assignment on planned_physical_plant_item application object rather than on plant_item. Revised AP218 references to plate and profile application objects replace single reference to abstract parent entity (structural part).
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): In 5.1.16.1.6 space_product_structure to external_instance_reference (as external_items), change mapping path to:

  AIM element: PATH
  Reference path: group <-
  /GROUPS(product_definition, ’item structure’)/
  {([/CLASS_ID(product_definition, 'plate')/]
  [/EXT_INST_REF(product_definition, 'ship structures schema', 'plate')/])
  ([/CLASS_ID(product_definition, 'profile')/]
  [/EXT_INST_REF(product_definition, 'ship structures schema', 'profile')/])
  ([/CLASS_ID(product_definition, 'planned physical plant
  item')/]

[/EXT_INST_REF(product_definition, 'plant spatial configuration', 'planned physical plant item')/])
([/CLASS_ID(product_definition, 'compartment')/]
[/EXT_INST_REF(product_definition, 'ship arrangement schema','compartment')/])}

SEDS #215-018

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 390, Clause 5.1.16.1.3
- Other Parts Affected by the Issue: none
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: In 5.1.16.1.3 product_structure_type, change mapping to Group.description.
  - Group.name attribute already assigned in the LINK_TO_GROUP mapping template as 'item and item_structure'.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Revised mapping path:
  AIM element: group.description
  Source: ISO 10303-41
  Reference path:
  group
  group.description
  {(group.description = ‘compartments in arrangement’)
  (group.description = ‘items in compartment’)}

SEDS #215-019

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 441, Clause 5.2, Annex A.
- Other Parts Affected by the Issue: IS216, IS218, AIM SF Express, and AIM LF Express
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: Add measure_with_units for use in conversion_based_units for exchange of non-SI.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Add measure_with_units and subtypes for measures used in each AP for use in conversion_based_units for exchange of non-SI.

SEDS #215-020

- Part and Clause where Issue Originates: WG3N1226_IS215, Clause 5.2, Annex A, and Annex H.
- Other Parts Affected by the Issue: AIM SF Express, and AIM LF Express
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: Add action_relationship, product_category_relationship,
- acyclic_product_category_relationship to AIM SF, AIM LF as used in mapping tables.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Add action_relationship, product_category_relationship, acyclic_product_category_relationship to AIM SF, AIM LF as used in mapping tables.

SEDS #215-021

- Part and Clause where Issue Originates: WG3N1226_IS215, Page 447, Clause 5.2.2.2.3 and Page 970, Figure H.28
- Other Parts Affected by the Issue: AIM SF Express, and AIM LF Express
- Related Issues: none
- Summary/Abstract/Keywords:
- Problem Description: Add compound_representation_item to Identification_item SELECT type.
- Conditions Under Which the Issue Was Discovered: NSRP ISE4 AP215 translator development
- Proposed Solution (Optional): Add compound_representation_item to Identification_item SELECT type.

A.2 AP218

This section highlights several areas within the current STEP AP218 specification and suggests improvements to the schema. They were developed during the Steel Processing Task under the ISE-4 Project. These suggested enhancements are based on the premise that the AP218 specification will be used to capture information for a complete work package; one that can be distributed to separate manufacturing locations. Such a view therefore requires the specification to contain enough information to allow a software system to analyze the contents of a work package, determine its suitability for a specific domain, and automate the manipulation and addition of manufacturing data, wherever required, in order to transform this work package. These enhancements were previously described in Section 5 of this report, and are repeated below for completeness of the Appendix. The figures and tables that are referenced appear in Section 5 of this report.

A.2.1 Assembly Hierarchy

The Steel Processing system has been developed to process data related to steel production work packages. Although most of these data requirements are met by the current STEP AP218 specification there are certain required data elements, related to part fabrication and production instructions, that are not included. These have been identified in Table 3. Figure 36 shows the extensions made to the `assembly_manufacturing_definition` STEP AP218 entity to support this information.

Given the structure of AP218 with respect to assemblies and their relationship to parts and the design/manufacturing definition entities, extending the assembly manufacturing definition entity was the most appropriate way of including support the Steel Processing data requirements. These extensions involved adding attributes to the existing `assembly_manufacturing_definition` entity and extending it to provide a structure for more specific part-related processing data.

`assembly_manufacturing_definition` was updated to include links to associated work orders, a text element to provide specific manufacturing remarks and links to the production area and equipment to be used during manufacturing. Section A.2.3 provides a description of how manufacturing process-related data was included in the work package. `assembly_manufacturing_part_definition` is a derivative of `assembly_manufacturing_definition` and includes information for part-specific assembly information. This includes a unique id for all like parts (piece_mark), a link to the production detail drawings and nest tapes containing the part, the catalog part number for the raw material to be used for the part, and a distinction as to whether this material is a remnant. There's also a redefinition of the `defined_for` attribute, which is used to link the assembly to the structural part that it pertains to.

assembly_manufacturing_plate_definition is a derivative of *assembly_manufacturing_part_definition* and includes information for plate-specific assembly information. In this case, there's only an attribute to capture the area of a given plate. Note that the test data set for the Steel Processing system consists only of plates. Therefore, this entity will be used for all elements. Finally, the *assembly_manufacturing_profile_definition* was created specifically for profile elements and is a derivative of *assembly_manufacturing_part_definition*. This entity adds an attribute for the stiffener sketch document that defines the stiffening member.

Figure 37 shows how the new assembly manufacturing definition entities will relate to parts and other assemblies. Note that this is a high-level figure with detail removed for clarity.

A.2.2 Labels/Layout Marks

Figure 38 shows the current support within the STEP AP218 specification for labels and layout marks. Each manufacturing definition of a part can have any number of text annotations (labels) with an optional single point specifying its location. Layout marks consists of curves only. There is no association between an annotation and a layout mark and no direct means of designating a symbol. Labels and layout marks are distinguished as being on the top or bottom surface of a part.

Figure 39 shows a proposed enhancement to the AP218 specification to support labels, layout marks, and symbols. The proposed enhancements replace the *annotation* and *curve* entities with new entities *layout_label* and *layout_mark* respectively.

Layout labels consist of a text element, a bounding box, which is used to designate the text size, and a direction so the text can be oriented to something other then 0 degrees. In addition, an enumeration is included – *layout_label_type* – that is used to specify the type of the label. The options for Layout Label Types were presented previously in Table 4.

Labels for a part manufacturing definition should only exist if they are to be etched onto the surface of the part as part of the manufacturing process. The proposed enhancements for annotations allow for more flexibility in the placement and sizing of a label and the added label type provides a means of programmatically distinguishing between different labels.

The proposed layout mark entity consists of a one or more shapes, an optional set of layout labels, and a type. The Layout Mark Types were described previously in Table 5.

These enhancements for layout marks provide a more structured means of defining layout line data by being able to include multiple geometric entities for a single layout mark. This also allows for the definition of symbolic annotations. Like layout labels, the label mark type allows layout marks to be categorized. Finally, the included set of layout labels allow label data to be associated with its corresponding layout marks.

The content, placement, and inclusion of marking line and annotation data are, to a large extent, driven by manufacturing rules and these rules vary between yards. Having the ability to query a manufacturing work package for this type of information therefore becomes necessary in the context of work sharing. From the standpoint of the Steel Processing System, these capabilities are important since having this information contained in the work package enables the system to automatically, and more accurately, position and categorize annotation and marking line data based on a well-defined rule set.

A.2.3 Mapping to Manufacturing processes and equipment

The primary purpose of the Steel Processing application is to map the information that describes steel parts with the processes needed to fabricate and assemble them into complete units. Although STEP AP218 contains the information that describes these parts and their assembly hierarchy its support for describing, or even specifying, manufacturing processes is sparse or non-existent. Also, there is no mechanism within the specification to identify the equipment to be used for manufacturing or the location where this work is to be completed. If AP218 is to be used to describe a steel production work package, as it is for the Steel Processing application, then this information needs to be included into the specification or provisions must be made to link to external sources (preferably defined using STEP) where this information can be culled.

One problem with including all of this information in a single location is that each piece of information evolves through a life of its own. Information that describes a manufacturing process is dependent on the equipment being used and the processes employed at its manufacturing location. Information describing each steel part and assembly is relatively static although this can be influenced by process. Equipment specifications and facility capabilities and requirements are somewhat static as well but can change due CAM or hardware upgrades.

The Steel Processing pilot application uses the STEP AP218 enhancements described in Table 3 to represent steel part and assembly data along with marginal support for manufacturing processes (see Figure 40). Equipment specifications and facility capabilities were described separately. Separating the equipment and facility information from the part and manufacturing process data allows this information to be processed independent of the part data. The Steel Processing Task demonstration created two sets of equipment/facility data to reflect the two manufacturing areas the system will be used to support.

The STEP AP218 specification would be more useful to support the definition of a work package if it included manufacturing processing data either internally or through external references. Whether the format of this information is driven by manufacturing process definition schemas as defined in other STEP APs, such as AP224 or AP240, or the STEP AP218 enhancements provided by Atlantec Enterprise Solutions is yet to be determined. Defining provisions to support manufacturing processes is more than including support for the processes themselves; additional information may also be required for each structural entity as noted in the following section.

A.2.4 Part Relationships

This section proposes a minor addition to the current STEP AP218 specification to support the definition of the physical joint between two structural entities. This change is motivated by the need for the Steel Processing system to:

- relate raw material data with the manufacturing processes that are defined as part of a work package,
- validate the integrity of various types of connections, and
- use information contained within the work package for downstream processes.

Figure 41 shows an example of the relationship between a pair of plates and a couple of stiffeners that are attached to them. The plates are related by a seam, which is defined by one of the bounding edges for each plate. The stiffeners are related by the bottom edge of the web of the structural tee and marking lines etched on the surface of a plate. As part of the manufacturing process, the plates would first be welded together to form a larger panel system onto which the stiffeners are then attached. The geometry

that defines the marking lines on the surface of the plates defines the physical location of where a stiffener would be placed. This data could also define the path for a robotic welder. The plate seams represents the physical connection point for the two plates and can also be part of the information to define the edge preparations required for the intended weld. Depending on the manufacturing processes used to etch the marking line on to the surface of the plate, the physical marking line may need to be held back from the plate seam.

If the proper information is contained in the work package, rules can be established to validate, or generate, various attributes of these part connections. STEP AP218 contains support for numerous types of relationships among various components including, features, design definitions, and manufacturing definitions. Figure 42 shows the STEP AP218 entities currently included in the specification for defining structural joints between parts.

Structural_part_joints identify a connection between two structural parts. The details of this connection are defined by derivatives of the *structural_part_connection_implementation* entity. *Structural_parts* can then be related to a seam by means of a *structural_part_relationship*. The physical geometry for a *seam* is defined by the *seam_design_definition*, which uses the *ship_seam* to define the seam with a curve or with a component of another part using *seam_curve_relationship*. Bevel information, defined by *bevel_design_definition*, can be related to a seam using its *defined_for* attribute. This configuration allows for two parts to be related by a structural joint, using *structural_part_joint*, and allows for a seam to be related to a structural part, using the *structural_part_relationship*. However, there is no mechanism to relate a structural joint to the seams defined on or by each of the adjoining parts. *Structural_joint_relationship*, a derivative of structural_feature_relationship, is proposed to define an association between two seams. Each item in the relationship is enforced to be a seam. *Structural_part_joint* would then require an additional attribute of type *structural_joint_relationship*. An instance of a set of these entities was shown previously in Figure 43.

A.3 AP212

During the Electrotechnical Task, KSS analyzed the STEP Standards, in particular AP212. As they became familiar with the STEP standard, they realized it is lacking in parameterization capabilities. STEP is mostly written to function with static values for entities. It is recommended that an element be added to the STEP standard to write to and retrieve values from at run-time.

KSS KnowledgeManager design uses both static and dynamic variables in the entity attribute values. This critical data must be stored in AP212 if we are to translate data from KnowledgeManager to AP212.

It is thus recommended that the following elements and attributes be added to the STEP AP212 standard to implement variables:

Element name
Variable

Element Attributes
name – specifies a speaking designation of the variable
source_attribute – the attribute that contains the data that this variable references.
source_entity – the entity that contains the source_attribute

value – string or numeric value
description – an alphanumerical string containing human-interpretable text that gives further details about the variable.

Element name
Variable_association
Element Attributes
assigned_variable – specifies the variable(s)
is_assigned_to – specifies the item with which the variable is associated
role – describes the relationship between the variable and the associated item. The value is either user defined or predefined.

The current STEP standard identifies one method of associating one entity to another - an Assembly_Component_Relationship. However, there is no element to establish a hierarchy (parent-child) positional relationship between two entities that do not meet the criteria for an Assembly_Component_Relationship, yet are related in an Electrical Systems model. It is recommended that the following element and attributes be added to retain this parent-child positional relationship information:

Element name
Device_relationship

Element Attributes:
related - specifies the single_device that corresponds to the "parent" entity
relating - specifies the single_device that corresponds to the "child" entity
relation_type - describes the relationship between the related and relating entities. The value is either user defined or predefined.