Abstract

Over the past decade CDM Technologies, Inc. (CDM) in conjunction with the Collaborative Agent Design Research Center (CADRC) at California Polytechnic State University (Cal Poly) in San Luis Obispo, has developed a suite of information-centric software tools in support of military deployment and distribution processes. All of these tools feature agents that are capable of reasoning about data in the context provided by an internal information model. Together they represent an evolving suite of adaptive Knowledge Management Enterprise Services (KMES) that can be readily configured into a net-centric planning and decision-support toolset for a particular application domain.

As a set of KMES tools the Integrated Computerized Deployment System (ICODEES) is configured to support the movement of supplies in the military deployment and sustainment operational domain. The application focus is conveyance load-planning, including the staging of cargo in marshalling yards, assembly areas, and rail heads. However, the application area can be easily broadened through the interoperability of ICODEES with other KMES toolsets such as TRANSWAY for route planning, and the Joint Forces Collaborative Toolset (JFCT) for the planning and assessment of sea-based logistical operations.

ICODEES is an example of a new generation of information-centric military decision-support systems that feature expert agents with automatic reasoning and analysis capabilities. This is
made possible by an internal virtual representation of the load-planning environment, in terms of conveyance and cargo characteristics and the complex relationships that constitute the context within which load-planning operations are performed. ICODES agents monitor the principal determinants of cargo stowage, including: the placement and segregation requirements for hazardous cargo items; the trim, list, stress, and bending moments of ship structures; the accessibility of stow areas through ramps, cranes, elevators, hatches, and doors; the correct placement of cargo items in respect to fire lanes, no-stow areas, reserved stow areas, and inter-cargo spacing tolerances; and, the accuracy of cargo characteristics (e.g., dimensions, weight, type, and identification codes) relative to standard cargo libraries and associated reference tables.

In addition, ICODES includes the JINNI module that allows users to create staging areas and marshalling yards, giving ICODES the ability to support load-planning operations in the broader spectrum of tracking cargo through the deployment stages of assembly, staging, load-planning, and the rearrangement of load-plans during transit\(^1\).

Like all KMES components that CDM has developed over the past 12 years, the ICODES suite of planning and decision-support tools have been designed and implemented within the Integrated Cooperative Decision Making (ICDM) software environment\(^2\). ICDM is an application development framework for distributed decision-support systems incorporating software agents that collaborate with each other and human users to monitor changes (i.e., events) in the state of problem situations, generate and evaluate alternative plans, and alert human users to immediate and developing resource shortages, failures, threats, and similar adverse conditions. A core component of any ICDM-based application is a virtual representation of the real world problem (i.e., decision-making) domain. This virtual representation takes the form of an internal information model, commonly referred to as an ontology. By providing context (i.e., data plus relationships) the ontology is able to support the automated reasoning capabilities of rule-based software agents.

Principal objectives that are realized to varying degrees by the ICDM Development Toolkit include: support of an ontology-based, distributed, information-centric system environment that limits internal communications to changes in information; ability to automatically push changes in information to clients, based on individual subscription profiles that are changeable during execution; ability of clients to assign priorities to their subscription profiles; ability of clients to generate information queries in addition to their standing subscription-based requests; automatic management of object relationships (i.e., associations) during the creation, deletion and editing of objects; support for the management of internal communication transmissions through load balancing, self-diagnosis, self-association and self-healing capabilities; and, the ability to interface with external data sources through interoperability bridges and ontological facades.

This report is divided into three parts. The first part presents ICODES from an operational perspective focusing on load-planning and the staging of cargo. The second part describes the technical basis of ICODES as a suite of ICDM-based KMES components, and the third part discusses ICODES from an evolutionary perspective highlighting on-going development to expand the set of capabilities offered by the ICODES suite of tools.

\(^1\) Often required on amphibious assault ships (Navy and Marine Corps) when the Commander receives new mission orders while en route.

\(^2\) ICDM is a software development toolkit that is proprietary to CDM Technologies, Inc. and available to third parties under licensing agreements (Pohl et al. 2004).
Table of Contents

1. **ICODES as a Suite of Cargo Staging and Load-Planning Tools** .................. 5
   1.1 Operational and Technical Objectives ........................................ 6
       1.1.1 The ICODES Web Site .................................................. 7
   1.2 Essential Ship Load-Planning Capabilities .................................... 8
   1.3 Interoperability with External Systems ...................................... 10
   1.4 Agent Capabilities ..................................................................... 10
       1.4.1 The Stow Agent ................................................................. 11
       1.4.2 The Access Agent ............................................................... 12
       1.4.3 The Hazardous Material Agent ........................................... 13
       1.4.4 The Trim and Stability Agent ............................................ 14
       1.4.5 The Cargo Agent ................................................................. 15
   1.5 Thick-Client User-Interface ...................................................... 16
       1.5.1 Standard Reports .................................................................. 18
   1.6 Thin-Client User-Interface ............................................................ 19
   1.7 The ICODES Viewer ...................................................................... 20
   1.8 The ICODES Observer ................................................................... 21
   1.9 JINNI: General Space Planning Capabilities .................................... 22
   1.10 I-AIT: ICODES Automatic Identification Technology ......................... 25
       1.10.1 Multi-Modal Planning ............................................................ 28
       1.10.2 Graphical Embarkation Reports ......................................... 29
   1.11 The ICODES Rail Loader ............................................................... 30
       1.11.1 Rail Terminal Layout .............................................................. 31
       1.11.2 Trains, Spurs and Rail Lines .................................................. 32
       1.11.3 Cranes and Ramps ................................................................. 32
       1.11.4 Rail Cars ............................................................................. 33
       1.11.5 Typical Rail-Loading Functionality ...................................... 34
   1.12 Operational Impact Analysis of ICODES ........................................... 37

2. **ICODES as a Suite of Knowledge Management Enterprise Services** ............ 41
   2.1 KMES: A Net-Centric Architecture ................................................ 41
   2.2 ICDM: Philosophy and Principles .................................................. 44
       2.2.1 Collaborative-Intensive ......................................................... 45
       2.2.2 Context-Oriented Representation .......................................... 47
       2.2.3 Extensible and Adaptive ...................................................... 48
       2.2.4 Multi-Tiered and Multi-Layered ........................................... 49
2.3 The ICODES Domain Model ................................................................. 50
  2.3.1 Vessel Domain ................................................................. 50
  2.3.2 Cargo Domain ................................................................. 52
  2.3.3 Plan Domain ................................................................. 53

2.4 System Architecture ................................................................. 53
  2.4.1 Domain Tier ................................................................. 54
  2.4.2 Logic Tier ................................................................. 58
  2.4.3 Presentation Tier ........................................................ 59

2.5 The Generic Space Generator (GSG) ................................................ 64

2.6 Reference Libraries ................................................................. 67

2.7 Data Import Requirements and Options ......................................... 74

2.8 Hardware Requirements ............................................................ 75

3. ICODES as an Extensible Suite of KMES Tools .................................... 77
  3.1 The ICODES External Services Interface (IESI) ......................... 77
  3.2 The ICODES Plug-In Framework (IPIF) .................................. 78
      3.2.1 The Plug-In Design Pattern ........................................ 78
  3.3 ICODES Plug-Ins ................................................................. 79
      3.3.1 The ICODES Observer ................................................ 79
      3.3.2 The T-AVB Automated Load-Planning System (TALPS) .... 80
      3.3.3 The Generic Cargo Plug-In Framework ......................... 81
      3.3.4 The Deployable Automatic Measuring System (DACMS) ... 82
      3.3.5 ICODES Automatic Identification Technology (AIT) ..... 84
      3.3.6 The MDSS-II Interface Plug-In .................................. 85
      3.3.7 The WPS Interface Plug-In ....................................... 86
  3.4 The ICODES Stow Framework (ISF) ......................................... 88

References ......................................................................................... 91

Glossary of Acronyms ................................................................. 95

Appendix A: ICODES Data Dictionary .............................................. 99

Appendix B: ICODES Symbols Library ............................................. 107
1. ICODES as a Suite of Cargo Staging and Load-Planning Tools

The rapid deployment of military assets from CONUS to overseas locations is a complex undertaking. It involves the movement of large numbers of tracked and wheeled vehicles, rotary aircraft, weapon systems, ammunition, power generating and communication facilities, fuel, food supplies, and other equipment and goods, from military bases to the Area of Responsibility (AOR). Several modes of transportation are typically involved. Depending on the location of the military base the assets are preferably moved by road to the nearest railhead, from where they are loaded onto railcars for transportation to the Port of Embarkation (POE).

Alternatively, if rail transportation is not an option, all of the cargo must be shepherded through the public road corridor from the base to the port. At the POE the assets are briefly assembled in staging areas and then loaded onto vessels for shipment. Points of debarkation may vary widely from a commercial shipping port with fairly good facilities to an amphibious landing on a hostile shoreline under fire.

Figure 1.1: Load-planning as a complex problem

Speed and in-transit visibility are of the essence (Figure 1.1). The total time required for the loading and unloading of the conveyance is a critical factor and largely determined by the quality of the load-plan. For example, the load-planning of an ocean-going ship has many of the characteristics of a complex problem situation (Figure 1.2). First, there are continuous information changes. The vessel that arrives at the port may not be the vessel that was expected and that has been planned for. This means that the existing load-plan is no longer applicable and a new plan has to be developed. Similarly, last minute cargo changes or inoperative lifting equipment may require the existing plan to be modified or completely revised.

Second, there are several complex interrelationships. The cargo on any one ship may be destined for several Ports of Debarkation (POD), requiring careful consideration of loading and unloading sequences. However, these sequences must take into account unloading priorities that may be dictated largely by tactical mission plans. In addition, the placement of individual cargo items on
board the ship is subject to hazardous material regulations and practices. These regulations are voluminous, and complex in themselves. At times they are subject to interpretation, based on past experience and detailed knowledge of maritime risks and practices. Finally, the trim and stability characteristics of the ship must be observed throughout the planning process.

Third, there are many loading and unloading constraints. Some of these constraints are static and others are dynamic in nature. For example, depending on the regional location of the port, external ship ramps may not be operable under certain tide conditions. Local traffic conditions, such as peak hour commuter traffic and rail crossings, may seriously impact the movement of cargo into staging areas or from staging areas to the pier. While these constraints are compounded whenever loading operations occur concurrently, the general complexity of the load-planning problem is exacerbated by the number of parties involved. Each of these parties plays an important role in the success of the operation, but may have quite different objectives. Certainly, the objectives of the commercial stevedore crews that carry out the actual loading tasks are likely to differ markedly from the prevailing military objectives (e.g., rapid loading and unloading operations, safety, unit integrity, load density, documentation accuracy, and security).

1.1 Operational and Technical Objectives

Several general and specific operational and technical objectives were specified by the military users at the beginning of the development process. Foremost, it was the vision of the sponsor that ICODES should present itself to the user as a suite of collaborative and expert tools, rather than a conglomeration of predefined solution templates. Experience had shown that the problems encountered in the real world of conveyance load-planning were driven by dynamically changing factors that were often unpredictable. Accordingly, any predetermined solutions based on preconceived requirements were unlikely to adequately address the nuances of the cargo stowage problem encountered under actual operational conditions.

From an operational viewpoint, ICODES was required to be magnitudes faster than an existing DOS-based ship load-planning application\(^3\). In summary, it should allow the concurrent planning of four conveyances, provide the user with continuous assistance in the form of alerts and warnings throughout the load-planning process, incorporate an automatic cargo placement capability, link to several external systems but be capable of operating in a stand-alone mode, and offer a friendly and flexible, graphical user-interface that could be customized by the user to suit individual needs.

The general technical objectives included the requirement of an open architecture, the ability to add new and enhance existing user-assistance capabilities over the lifetime of the application, the ability to add future modules to support related functional areas such as inter-modal transportation and port planning (e.g., management of staging areas), and the ability for the user to create cargo lists and conveyances within the application if these were not available within ICODES and could not be imported from existing external systems. The degree of intelligence and flexibility required of ICODES suggested an ICDM-based software architecture with KMES components.

\(^3\) ICODES replaced the DOS-based CODES system in 1997. CODES had reduced the prevailing manual ship load-planning process to two person-days per ship. The Government’s design objectives called for ICODES to be able to generate the load-plans of four ships in two hours.
Specifically, the ICODES toolset was required to automatically alert the user of cargo placements within stow areas that are in violation of hazardous material mandates, the trim and stability requirements in the case of a ship conveyance, or a host of cargo stowage rules such as adjacency tolerances, fire lanes, boom clearances, and movement restrictions (e.g., door and hatch dimensions, crane lifting capacities and reach, ramp and elevator constraints, and stow area heights). Such agent-based analysis was to be provided in both a manual stow and an assisted stow capacity. As implemented, the latter of these modes allows the embedded agent community to develop a high quality load-plan through collaboration based on the same expert analysis applied during its manual counterpart.

For example, in the hazardous material domain these specific objectives require ICODES to be able to differentiate among the internationally recognized nine classes of hazardous material, and the sub-groupings or divisions that exist in five of these classes. In addition, interpret and apply the regulations prescribed in the following four principal reference sources:

- The 49 Code of Federal Regulations (49 CFR) that specifies segregation requirements for hazardous cargo shipments in the Continental United States (CONUS).
- The International Maritime Dangerous Goods (IMDG) library that applies to all international shipments of hazardous materials.
- The Department of Defense Identification Code (DoDIC) library that applies specifically to Class 1 hazardous items (i.e., explosives), namely munitions.
- The Dangerous Cargo Manifest National Stock Number (DCMNSN) library that is used primarily by the Marine Corps for identifying and load-planning hazardous cargo items.

Today the ICODES agent-based, decision-support system successfully addresses this entire gamut of requirements. Through ICODES the user is empowered with a decision-support tool that provides both the expert stow planner as well as the novice user with a rich collaborative set of capabilities, providing detailed visibility and intelligent planning and re-planning capabilities in a complex and highly dynamic operational environment.

### 1.1.1 The ICODES Web Site

A password protected ICODES web site was created to serve the over 2,000 military users in the field. It provides authorized users with direct access to the most up-to-date ship drawings (i.e., Vessel Library), the various reference libraries (see Section 2.6), the JINNI module (see Sections 1.9 and 2.5), documentation and training aids, component updaters for ICODES plug-ins (see Section 3.3), and descriptive information (including photographs) of most ships.

Also included on the web site is the ICODES File Share facility, which is accessible to users directly from the ICODES suite of tools. It allows users to transfer files to an on-line repository for back-up purposes, or to share data files (e.g., cargo lists and load-out plans) with other users. For example, a user is able to send a load-plan to the next POD prior to the arrival of the ship at that destination.

---

1.2 Essential Ship Load-Planning Capabilities

Over the past several years the functional scope of the ICODES suite of tools has been greatly expanded beyond ship load-planning to the staging of cargo in marshalling yards and the load-planning of multiple types of conveyances, such as barges, trucks, and railcars. However, the primary use-case that drove the initial design and development of ICODES was to support the concurrent load-planning of multiple ships in both manual and automatic modes.

The ability of the software to automatically generate load-plans called for an embedded level of intelligence that would require a high-level internal representation of context to support the necessary inferencing capabilities. Accordingly, it was decided at the outset that the system will need to be able to represent each ship as a set of expressive, context-enriched objects, to allow intelligent agents to reason about access paths, obstacles such as stanchions and bulkheads, and constraints such as fire lanes and restricted areas. This suggested the need for a repository of objectified ships. Today the ICODES Master Vessel Library includes over 300 ships.

In support of these overall operational objectives, the initial design of ICODES was required to provide the following capabilities that would allow experienced, as well as novice, users to produce accurate and efficient load-plans in a fraction of the time compared to existing practices:

**Importing and adjusting of vessel characteristics:** As an initial step in developing an effective stow plan, the ICODES user must be able to select the particular vessel(s) that will form the landscape for stowing cargo. To support agent-based reasoning, these vessel descriptions must go far beyond simple geometric data and endeavor to describe the numerous non-geometric qualities that are necessary to provide meaningful assistance in the load-planning process. Such descriptions include particular hazardous zoning considerations and available on-board facilities (i.e., cranes, hatches, doors, ramps, ballast tanks, etc.).

Once loaded, the user must be able to adjust any of this information to reflect variances exhibited by the vessel’s actual condition. Anomalies frequently encountered include the existence of unforeseen obstructions within key stowable areas, in addition to temporarily non-operational ramps, cranes, doors, or elevators, to name a few. Users must also be able to enter specific trim and stability information, including actual departure tank settings. Additionally, the user should be able to designate any area of the vessel as a no-stow zone, or modify the characteristics of any stowable area such as the allowable maximum deck stress, door heights, cargo orientation, and a host of other ship-dependent parameters.

**Importing and cleansing of cargo characteristics:** Similar to the need for rich vessel descriptions, ICODES must also be able to obtain detailed descriptions of cargo items. This has led to interface agreements with key external systems that can provide up-to-date listings of the cargo for any specific load-out operation. Once the cargo list data has been added to the system, the ICODES analytical engine must be able to review this information and compare it against several external reference libraries, generating warnings and alerts whenever discrepancies are encountered. The user should be able to

---

5 ICODES 5.4.2 – Army Automatic Identification Technology (AIT) Guide, June 2006 (www.icodesweb.com);
review the information reported by the Cargo Agent and edit the attributes of the cargo items in violation.

**Manual load-planning of cargo:** Through a robust graphical user interface (GUI), the system must provide the user with the ability to manually place, or template, cargo across target stow areas. However, the ICODES solution must constantly monitor such user-directed templating of cargo, validating its placement in terms of geometric constraints, loading and unloading accessibility, trim and stability, and hazardous material infractions. Any violations within these considerations must be presented to the user in full detail along with recommendations for corrective action.

To extend the utility of the planning tools available to the user in this manual mode, ICODES must also offer various graphical functions allowing the user to move, unstow, rotate, highlight stowed cargo based on Unit Identification Code (UIC), add comments, visually zoom into and out of the evolving plan, and both add and remove informational layers pertaining to the extensive information-oriented descriptions of vessel and cargo characteristics available within the system (see Appendix A, Cargo Data Dictionary).

**Automated load-planning of cargo:** During intense operational periods, development of effective load-plans must occur under time and resource-critical constraints. In order to accelerate the creation of plans under such conditions ICODES was required to provide a mechanism that can reason about requirements in terms of load-planning logic bounded by target cargo and vessel characteristics. This requirement called for the ability of the system to automatically generate an effective load-plan by reasoning about the physical characteristics of the problem space in terms of accessibility, trim and stability, hazardous materials, and other critical considerations. Further, to support various levels of customization and refinement, this capability must allow the user to influence such reasoning through user-supplied preferences. Such parameters were required to include cargo and stow area priorities, as well as user-specified restrictions relating to space utilization, mandatory pairing of cargo type with a designated stow area, and the co-location of cargo by either POD or military unit.

Once a load-plan has been generated, this capability must also allow an experienced load-planner to adjust any desired aspect of the automatically generated plan. Further, this editing capability should present the same look and feel to the user as the manual load-planning capability.

These initial requirements were met by the first prototype version of ICODES in the mid-1990s. Today (2006), ICODES Version 5.4.2 has greatly extended functionality beyond these initial specifications, particularly in the operational areas of marshalling yard management and multimodal conveyance loading.

In 1996, ICODES was selected as the migration system for ship load-planning by the Deputy Under Secretary of Defense (Logistics), and became the system of record with the release of Version 2 in 1997. It has been deployed by USTRANSCOM through the Surface Deployment and Distribution Command (SDDC) to the US Army since 1999 and the US Marine Corps since 2002. Other users include the US Navy and the British Army.

---

6 Previously known as the Military Traffic Management Command (MTMC) and renamed in 2004, SDDC is a component command of USTRANSCOM responsible for surface movement operations.
1.3 Interoperability with External Systems

ICODES interfaces with the World-Wide Port System (WPS), the Transportation Coordinators’ Automated Information for Movement System (TCAIMS-II), the MAGTF Deployment Support System (MDSS-II), the Integrated Booking System (IBS), TRANSWAY (for route planning); and, the Joint Forces Collaborative Toolkit (JFCT) for sea-basing operations.

Throughout load-planning operations ICODES is capable of receiving cargo lists from WPS, MDSS-II, TCAIMS-II, and IBS. However, ICODES currently has two-way connections (i.e., ability to import and export) with only the MDSS-II, WPS and TCAIMS-II systems. To facilitate the frequent updating of cargo lists during load-planning operations a Merge function identifies new and existing data items and automatically alerts the user to any anomalies, such as cargo items with duplicate TCN numbers. In 2003, ICODES was certified as Level 7 compliant with Defense Information Infrastructure Common Operating Environment (DII-COE) standards. These standards outline rigorous government requirements for software installation, configuration, as well as run-time and interoperability behavior. Development and testing is in progress to support a two-way interface with WPS. This two-way interface is scheduled to be released with the next version of ICODES at the beginning of 2007. Additionally, research is currently underway to support an interface with the Global Air Transportation Execution System (GATES), which is scheduled to replace WPS in 2008.

1.4 Agent Capabilities

The expert agents in ICODES are designed to assist the load-planner in the knowledge domains of hazardous material, trim and stability of ship conveyances, cargo access paths, cargo attribute verification, and the actual placement of cargo in stow areas. The agents continuously communicate with each other as they collaborate during their assistance activities.

When the user is developing a load-plan while operating in the manual User Stow mode, the agents will alert the user to any violations by turning the surround of the appropriate agent status window red. The user can then click on the status window to display a window with an explanation of the violation. In fact ICODES provides several different types of agent warnings:

- **Yellow** color of the agent status window and a highlighted cargo item warns the user of a situation that could lead to a potential violation (Figure 1.3).

![Figure 1.3: Typical agent warning status](image_url)

---

• **Orange** color of the agent status window and a highlighted cargo item indicates that one or more warnings have been acknowledged but still exist (Figure 1.4).

![Figure 1.4: Typical agent warning acknowledgement status](image)

• **Red** color of the agent status window and a highlighted cargo item indicates a violation and existence of a serious problem (Figure 1.5).

![Figure 1.5: Typical agent alert status](image)

• **Purple** color of the agent status window and a highlighted cargo item indicates that one or more violations have been acknowledged but still exist (Figure 1.6).

![Figure 1.6: Typical agent alert acknowledgement status](image)

If the user operates in the automated *Assisted Stow* mode the agents will collaborate to place the cargo in such a manner that there are no new violations created. Cargo items that could not be placed in any stow area without causing a violation are simply not stowed.

1.4.1 **The Stow Agent**

The Stow Agent supports manual and automatic load-planning operations, as well as a combination of both of these modes. In the case of a ship conveyance, using default settings in the automatic mode (i.e., Assisted Stow) the Stow Agent attempts to place the heaviest cargo items as low as possible on the ship without causing a violation. This results in a low center of gravity for the ship, which is desirable in most cases.
The Assisted-Stow mode provides a comprehensive set of settings. This allows the user to define exclusive and inclusive constraints and preferences in respect to both the cargo that is required to be stowed and the stow areas that have been designated as being available. The Stow Agent checks to see that the placement of a cargo item does not overlap another cargo item, a fixture of the conveyance such as a stanchion or fire lane, or if the item is not entirely within a stow area (Figure 1.7). In Assisted-Stow mode, the user can also set the front-back and side-to-side spacing requirements of a cargo item. The Stow Agent will abide by these settings and not stow within that imaginary boundary around each cargo item.

Other parameters checked by the Stow Agent include the POE and POD to ensure that they match the ports indicated in the voyage documents, and the height of each cargo item to ensure that the latter can reach its final stow position. The Stow Agent automatically adds a safety cushion (specified by the user) to the actual height, to make sure that height plus the cushion does not exceed the maximum allowable height for cargo in that stow area. Similar safety cushions can be added to the length, width and weight of a cargo item.

While in the Assisted Stow mode ICODES will ensure that the automatically generated load-plan has no violations, in manual mode (i.e., User-Stow) ICODES will allow the user to stow cargo items that are in violation.

1.4.2 The Access Agent

The Access Agent checks all paths to ensure that a cargo item can be stowed in a particular stow area. This includes openings, doors and hatches, differentiating (in the case of a ship
conveyance) between cargo that is loaded with cranes through hatches (i.e., Lift-On-Lift-Off or LOLO) and cargo that is driven or pulled into stow areas (i.e., Roll-On-Roll-Off or RORO).

Figure 1.8: Typical Access Agent violations and explanations

Under Assisted Stow conditions, if there is a violation in the stow path of a particular cargo item the Stow Agent will not place this cargo item in that stow area but will attempt to place it in another stow area. In this situation the violation is transmitted directly from the Access Agent to the Stow Agent without notification of the user.

In manual mode (User Stow), on the other hand, if a cargo item is placed in a particular stow area for which all of the possible stow paths register an access violation then the Access Agent will inform the user that the cargo item has a violation for every path to the stowed location. In addition, the Access Agent will identify for the user the shortest stow path along with the nature of violation that is associated with that path (Figure 1.8).

ICODES allows the user to edit conveyance characteristics in the case of ships, including the usability properties of the cranes and the dimensions of doors, openings and hatches. Since the Access Agent utilizes the current ship characteristics as the existing constraint conditions, these changes will be reflected in the actions of the Stow Agent in automatic mode and the alerts provided by the Access Agent in manual mode.

1.4.3 The Hazardous Material Agent

The Hazard Agent verifies the proper placement of hazardous cargo items in reference to various hazardous material codes and regulations. These currently include: the 49 CFR Library (United Nations and North American hazardous material type identification); the DCMNSN Library
(hazardous material mapping to National Stock Numbers); the DoDIC\textsuperscript{8} Data Dictionary (explosives and munitions); and, the IMDG Library (International Maritime Dangerous Goods codes). It considers issues such as: Is the cargo item stowed in an acceptable location according to its stowage requirements? What are the segregation requirements for the cargo item, taking into account both the type of cargo item (e.g., break-bulk, container, vehicle) and the proximity of any other hazardous cargo items? In the case of containers, the Hazard Agent considers the hazard category of each item in the container in assessing the hazard condition of the container and its location relative to any other hazardous cargo item on the conveyance.

1.4.4 The Trim and Stability Agent

The Trim and Stability Agent currently applies to ship conveyances only. It checks the placement of cargo items on the ship to see if they violate any desired (i.e., user specified) or mandated maximum draft settings, strengths (i.e., bending of the ship) or deck stress limitations.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{trim_stability_agent.png}
\caption{Typical Trim and Stability Agent violation explanation}
\end{figure}

In automatic mode the Stow Agent will rearrange the placement of cargo during the Assisted Stow process if the placement of cargo causes the upper limits of the strengths properties of the ship to be exceeded. For example, if the predefined stow order requires the middle two stow areas of a deck to be stowed first and second, this would result in a sagging condition of the deck. Under these circumstances the Stow Agent will automatically re-define the stow order used by the Assisted-Stow process, so that the placement sequence of the cargo will begin with

\textsuperscript{8} The acronym DoDIC stands for US Department of Defense (DoD) Identification Code.
the forward and aft areas of the deck (thereby preventing the occurrence of the *sagging* condition).

ICODES calculates the effects of the exact placement of every cargo item stowed on the ship in three different planes. These planes are: forward-to-aft often referred to as the Longitudinal Center of Gravity (LCG); side-to-side or Transverse Center of Gravity (TCG); and, up-and-down or Vertical Center of Gravity (VCG). The Trim and Stability Agent takes into account the combined effects of all of the cargo items, the ballast, and the original condition of the ship to provide the user with fairly accurate estimates of the center of gravity in each of the three planes, as well as an overall assessment of the stability of the ship\(^9\) (Figure 1.9).

1.4.5 *The Cargo Agent*

The Cargo Agent checks the characteristics of each cargo item against the expected characteristics for that cargo item as they are recorded in a number of reference libraries including the Marine Equipment Characteristics File (MECF), the Joint Equipment Characteristics File (JECF), and other technical data cargo libraries. Not all cargo characteristics can be verified in this manner. While these cargo libraries currently contain more than 20,000 items, they are restricted in terms of the attributes that are provided for each cargo item. Typically, this verification process is complete and reliable only for dimensional (i.e., length, width and height) and weight attributes. If discrepancies are detected the Cargo Agent generates warnings (Figure 1.10).

---

\(^9\) ICODES 5.4.2 – Advanced Training Manual, June 2006 ([www.icodesweb.com](http://www.icodesweb.com))
1.5 Thick-Client User-Interface

Implemented in a typical Windows 2000 or XP operating system environment, the main screen of the ICODES thick-client user-interface, as shown in Figure 1.11, consists of five components or sections.

(1) The Main Menu Bar provides access to the nine principal ICODES option groups in the form of pull-down menus. The option groups are: Loadout Plan; Ships; Cargo; Stow; Access; Stability; Tools; Window; and, Help.

(2) The Graphics Window displays the conveyance drawing(s). It can accommodate multiple conveyances, with the number that are concurrently displayed limited only by the constraints of the screen size and the memory capacity of the computer.

(3) The Message Window, found at the bottom of the main screen, provides the user with messages relating to the current status of ICODES (e.g., the status of a selected option, or instructions relating to the use of a particular tool).

(4) The Agent Status Bar on the left side of the main screen provides access to agent reports and explanations of warnings and alerts.

(5) The Tool Bars on the right side of the main screen contain four groups of tools: stow tools (e.g., rotate, flip, unstow individual cargo items); view manipulation tools (e.g., zoom, pan); association tools (e.g. hitch, load onto, stack, put into, palletize, and mobile load); and, drawing tools that allow the user to

10 ICODES 5.4.2 – Quick Start Guide (Pages 1 and 2), June 2006 (www.icodesweb.com)
superimpose lines, circles, polygons, and rectangles, on a displayed conveyance drawing.

Figure 1.12: User-Stow (top) and Assisted-Stow (bottom) modes
ICODES offers a very comprehensive set of editing, saving, restoring, reporting, and special operations options (MTMC/ICODES 2002). In addition, ICODES recognizes the differences among tactical (emphasizing mission accomplishment), pre-positioning (accommodating the maintenance requirements of pre-loaded regionally positioned conveyances) and administrative (focusing on the maximum utilization of troop and cargo space) load-plans.

The preparation of a load-plan can be undertaken in either of two modes. In the **User Stow** mode the user selects a cargo item from a textual cargo list, ICODES automatically converts the selected item into the appropriate graphic cargo symbol, and once the user has placed the cargo symbol in a stow area the agents assess the impact of the cargo item in that position on both the validity of the load-plan and the condition of the conveyance in the case of a ship (Figure 1.12). The agents take into account: the path of the cargo item from the staging area to its final location on the conveyance (e.g., availability of ramps, cranes and elevators, and the dimensions of doors, hatches and openings); the segregation and other special requirements related to hazardous materials; and, the trim and stability conditions in the case of a ship.

In the **Assisted Stow** mode the user is able to define specific parameters at the cargo and conveyance levels and then request ICODES to automatically stow the cargo on one or more conveyances (Figure 1.12). Parameters include the establishment of preferences for individual stow areas, the exclusion of stow areas, the specification of spacing distances between cargo items, the orientation of cargo items, and the selection of subsets of the cargo list. Once the parameters have been specified (either by default or user selection) ICODES will automatically prepare a load-plan that does not violate any of the rules and regulations known by the agents.

1.5.1 **Standard Reports**

The **Standard Reports** window provides pre-formatted reports used for specific purposes after a load-plan for a particular conveyance has been completed. These reports are different from those generated using the **Customize** feature in ICODES.

**Reports with user input:** Some of the Standard Reports provide text boxes allowing custom information to be entered, such as the names or contact information for personnel. These are provided for reports such as Cover Pages to specific reports or pages that must be signed to approve a plan.

**Reports requiring user selections:** Other Standard Reports are pre-formatted, but provide a way for the user to select specific information that should be included in a report. For instance, the user may wish to select the equipment of a specific military unit (i.e., by UIC) for inclusion in a report. Another example is to include in the report only the hazardous material loaded in just one stow area of the conveyance.

Each of the Standard Reports is different in terms of the information it provides. Below is a listing of each report and the information it provides. As indicated, some of these reports are appropriate for ship conveyances only.

**Association Report:** This report includes all of the cargo items stowed on a given conveyance that are part of a set of associations and item identifications that the user has selected. The UIC, Compartment, Nomenclature, TAMCN, Package ID, and Association Type are displayed for each item.
Assigned Report: This report displays all of the items stowed on a given conveyance that are not displayed in the graphical representation (i.e., plan view) of the conveyance. In other words, items that have been stowed by Zone or Locker are included in this report. The UIC, NSN, Package ID, Item ID, Description, Templating Status, and Compartment are displayed for each item.

Data Recap Report: This report displays cargo grouped by JCS Cargo Category Code with one line for each JCS. The UIC, Quantity, Square Feet, Weight, M/T, and L/T are displayed for each line.

Density Listing: This report includes all of the cargo and personnel on a ship conveyance. The personnel are broken down into groups of officers, E7-E9, and enlisted personnel. Summarized cargo data are provided with UIC, Quantity, Description, and Unit of Issue displayed for each item.

Embarkation Summary: This report displays the cargo and personnel broken down by ship, with subtotals for each rank in the case of personnel. The information provided for each cargo item includes Ship Name, UIC, UPTT Code, and Quantity.

Item Identification Reports: There are two types of item identification reports for ship conveyances, each showing the same information but sorted by either Item Id or by Deck. Only cargo that is graphically displayed on the ship will be included in this report with Item ID, Description, Deck, and Count of Item Id displayed for each item.

Ship Cargo Manifest: This report shows all items stowed on the ship no matter how they have been stowed. The Ship, Compartment, Zone, Team, UIC, Landing Serial, Priority Order, UPTT, Association, Package ID/TCN, Parent Package ID, Item ID, JCS Cargo Category, Embark Category, Template Label, # Cargo, IMO Code, Length, Height, Square Feet, Cubic Feet, and Gross Weight are displayed for each item in a three tiered line fashion.

Ship Compartment Tonnage: This report shows all items stowed graphically in the plan. The Ship, Compartment, Quantity, Cubic Feet, and Weight are displayed, summarized into one line for each compartment.

Staging Report: This report provides a listing of vehicles in reverse order compared to how they have been stowed in the plan. It allows the user to determine how cargo items should be moved aboard the vessel for deployment in the proper order. The Priority Order, Landing Serial, UIC, Package ID, Description, Height, Weight, and AIT Location Code are shown for each item.

1.6 Thin-Client User-Interface

The ICODES thin-client user-interface has been provided to satisfy the need for access to the information in ICODES load-plans by persons who do not have the ICODES application installed locally. These users may need to view the cargo on in-transit conveyances, or to

---

11 Due to the high priority placed by the users on the availability of an ICODES thin-client user-interface with full interactive graphics functionality, the ICODES development team at CDM is currently exploring a promising approach for overcoming these existing technical limitations.
examine past shipments. Due to the highly interactive graphics requirements of the operational stow planner, current bandwidth and technical limitations have prevented the development of a thin-client user-interface with full capabilities.

The existing ICODES thin-client is a web-based tool implemented as an Internet Explorer™ web browser. It is also available on the ICODES Web Site, where it can be readily accessed by any authorized user. The ICODES Web Site allows authorized users to upload load-plans into a shared environment known as the ICODES File Share. This enables users to send load-plans from one POE to another, and to archive plans for future access.

The ICODES thin-client enhances this upload capability by extracting information from the load-plan and making this information available to an internal search engine. Once load-plans have been placed on the web site in the shared environment (i.e., ICODES File Share), the thin-client user-interface allows users to search for plans that contain a given ship, or include specific POEs and PODs, or were loaded within a specified period of time. Based on the results of these queries, users can select a particular load-plan for viewing.

Following the selection of a load-plan and conveyance, another web browser window opens. This window is divided into two sections. One section displays a graphical representation of the conveyance and its cargo, similar to the display of a conveyance in the ICODES thick-client. This display can be panned both horizontally and vertically, and there is a zoom function for expanding selected portions of the display. In the second section of the window there is a list of cargo items. This list can be sorted on several different attributes. Selecting a cargo item in the list highlights the symbol for that piece of cargo on the ship and vice versa, enabling users to locate specific items.

### 1.7 The ICODES Viewer

The ICODES Viewer\(^\text{12}\) was originally conceived to meet the demand of command officers to be able to look at the information in ICODES load-out plans without modifying them. It is designed to give the user access to all of the reports and views available in ICODES, but prevents the modification of data in the load-plan (Figure 1.13). In this way, persons in remote locations are able to view load-plans that have been posted on the ICODES web site, without endangering the contents of these plans to inadvertent or intentional changes that might jeopardize the integrity of the contents of the plans.

The design of the ICODES Viewer is very simple. The Viewer shares the same source code as the full ICODES application. However, when the Viewer version of the application is loaded the GUI classes activate checks that hide all of the functionality (i.e., menus and controls) that would normally allow users to modify the plan data. This simple design minimizes the software maintenance requirements by keeping the Viewer in sync with the main application as new functionality is added to the product.

\(^\text{12}\) ICODES 5.4.2 – ICODES Viewer User Guide, June 2006 ([www.icodesweb.com](http://www.icodesweb.com))
1.8 The ICODES Observer

The ICODES Observer\textsuperscript{13} is a plug-in component designed to allow users to connect remotely to the main ICODES system. The Observer provides a read-only, dynamic view of an executing instance of ICODES, including, but not limited to, cargo, violations, and reports.

Written in the Java programming language, the Observer communicates with ICODES through sockets and the ICODES External Services Interface (IESI) described later in Section 3.1. The ICODES toolset includes several plug-ins for special purpose tools. They all connect to the IESI interface, utilizing the RSA\textsuperscript{14} plug-in authentication procedure. Once connected, ICODES and the Observer communicate using XML\textsuperscript{15} messages in broadcast mode. Components in the Observer can subscribe to broadcasts for particular types of objects, corresponding to database tables in ICODES.

A goal of the Observer’s interface design was to mimic the ICODES thick-client user-interface, so that users familiar with ICODES would be able to understand the functionality and layout of the Observer with relative ease (Figure 1.14). The user has access to two toolbars, which mimic

\textsuperscript{13} ICODES 5.4.2 – Observer User Guide, June 2006 (www.icodesweb.com)

\textsuperscript{14} RSA is an encryption technology developed by RSA Data Security, Inc. The acronym stands for Rivest, Shamir, and Adelman, the inventors of the technique.

\textsuperscript{15} XML is the acronym for Extensible Markup Language (see: http://www.w3.org/XML/).
their corresponding ICODES counterparts, the Agent Toolbar and the Viewer Toolbar. The Agent Toolbar provides access to the violations maintained by each agent, while the Viewer Toolbar provides the ability to modify the viewing perspective, as well as measure areas and distances. The Observer also provides many of the same types of reports and information dialogs contained in the ICODES thick-client.

Figure 1.14: The ICODES Observer user-interface

1.9 JINNI: General Space Planning Capabilities

The JINNI module allows the user to build any kind of stow area from minimal data and simple paper sketches, for use in ICODES. Typical data requirements for a ship conveyance include: number of decks and holds; and, dimensions, shape, maximum allowable deck stress, and location of stow areas on each deck. JINNI incorporates a very powerful set of drawing and calculation tools based on the Generic Space Generator (GSG) component of the ICDM development framework that forms the core of the ICODES suite of tools (Figure 1.14). Even though not all of the information and relationships that are available in a standard ICODES ship conveyance are included in a JINNI generated drawing (e.g., certain structural and hydrostatic data may not be available to the user), many of the reasoning functions provided by the Stow Agent, Cargo Agent and Access Agent will be operative.

Figure 1.14: A vessel under construction in the JINNI module

The GSG\(^{17}\) component of ICDM was designed as a client architecture for facilitating the rapid development of graphical user-interfaces. With its ability to harness the most recently released Open Graphics Library (OpenGL) calls from Java code, GSG is able to optimize performance for both two-dimensional and three-dimensional graphics requirements. The components of GSG are extensible and reusable to promote the rapid prototyping of applications that incorporate similar concepts.

JINNI allows the user to create spaces for a variety of goods transportation and management purposes such as truck beds, railcars, staging areas, marshalling yards, and even warehouses (Figure 1.15). The current version of JINNI utilizes ship terminology to describe these spaces (e.g., deck, stow area, ramp, etc.). However, planned future extensions will add modules to allow the use of nomenclature that is appropriate for the particular type of space under consideration. The construction of a space may be undertaken in any one of three alternative ways:

A. By measuring ‘x,y’ coordinates on a scale drawing of the space and then entering these coordinates through the user-interface.

B. By measuring the size of each area within a space on a scale drawing and then constructing each space individually so that these areas can be placed in their relative positions inside the space.

\(^{17}\) GSG utilizes a custom Dynamic Link Library (DLL) in combination with Java Native Interface (JNI) technology to communicate directly with the available video card. Java JNI technology supports direct communication between Java and C++ as long as the necessary communications bridge is contained within the DLL.
C. By importing a digital picture (i.e., ‘jpeg’ or ‘gif’ image), setting the appropriate scale of the picture in JINNI, and then tracing over the image to construct the spaces.

Once spaces have been built, the following adjustments and additions can be made to these spaces:

- Adding fire lanes around the perimeter of the space at a user-specified distance (e.g., 3FT wide fire lane).
- Adding obstructions inside a constructed area such as walls, pipes, structural members, and so on.
- Adding notes and drawing shapes to further describe details of the spaces (i.e., lines, boxes, circles, and curves).
- Adding container cell to the constructed space to represent multiple levels of container stacks. Each cell allows a 20FT container or equivalent item (i.e. Tricon, Quadcon, etc) to snap into an appropriate location within the cell provided the dimensions are of a standard size.
- Adding nodes to existing shapes to adjust the number of line segments for a given shape. For example, if a given area has a cut-out due to an obstruction such as a fire hydrant, then additional nodes can be added and moved to change the boundary of the area to include the fire hydrant.
During the process of creating a space, the user has the ability to save the drawing to a file. Either the entire set of constructed spaces or an individual space can be saved for use at a later time, or for sending to another user. For example, the drawing of a previously constructed ship can serve as a template for the construction of another ship. The user can also copy and paste any number of objects to allow quick replication of similar objects. For example, lanes in a yard are often of a similar size; therefore, the drawing of one lane can be replicated to create a row of lanes.

Apart from the ICODES agents that are able to reason about load-planning concerns, JINNI itself incorporates agents that check for logical problems during the construction of a drawing. For example, these agents will warn the user if separate areas within a space have duplicate names, if such areas overlap, or if an area is not completely contained within the overall space. In addition to these agents, JINNI incorporates a number of controllers that ensure the geometric integrity of the drawing. They will attempt to correct user entry errors such as the placement of two nodes of an object in exactly the same location, as well as automatically manage the associations between different components of a drawing (e.g., between a deck and the stow areas on that deck, in the case of a ship conveyance). The final product of this process is an objectified space that can be loaded into ICODES and will be treated by the load-planning agents as if it were a ship conveyance.

1.10 I-AIT: ICODES Automatic Identification Technology

As part of an initial implementation of Automatic Identification Technology (AIT) the ICODES development team was tasked to implement an ICODES-AIT (I-AIT) capability. The primary operational focus of this initial implementation is the verification of in-place stow. From a business process point of view, I-AIT is intended to provide military cargo specialists with the ability to add current stow locations to the ICODES pre-stow plan, thereby converting this plan into a living document of the movement of cargo from the time it arrives at the port to its final location on-board ship. The business model for the verification of in-place stow can be described as sequence of operational steps, as follows:

1. A Unit Deployment List (UDL) is created in MDSS-II.
2. The UDL is used to prepare a pre-stow load-plan in ICODES.
3. Embarkers (i.e., cargo specialists) take hand-held bar-code scanners to the respective stow area to verify in-place stow locations.
4. Selected portions of the ICODES pre-stow load-plan are then downloaded via an existing wireless network to a hand-held bar-code scanner (i.e., a PDA fitted with a bar-code scanning device).
5. The embarkers confirm and/or rearrange cargo item symbols on the hand-held scanner based on their verified scanned or temporarily off-loaded location.
6. Embarkers then up-load the in-place stow information to the ICODES server via the wireless network. Several embarkers can concurrently verify in-place stow locations on multiple ships.
7. All confirmed in-place stow data is then up-loaded to an ICODES session executing on a central (normally on-board ship) server via the wireless network.

8. The emerging in-place load-plan can be observed anywhere on-board ship using the I-AIT Observer. However, only the server version of the in-place load-plan will allow user-edited functions to be executed.

The I-AIT Observer allows unlimited near real-time access to the emerging in-place load-plan. It runs on any network-connected computer on-board the vessel, and can be installed with ICODES itself. Thus, embarkers and logisticians can observe the in-place operational status, while simultaneously load-planning future missions with ICODES.

The business model for verification of in-place stow shown in Figure 1.16 illustrates the key interfaces between the hand-held PDA with scanner device (running PocketPC), the wireless network, the ICODES server, and the I-AIT Observers.

![Figure 1.16: ICODES I-AIT system diagram](image)

Once the operators have connected the AIT Scanner to the ICODES Server, they will select the respective area (stow area, staging area, pier, etc.) that they are currently working in, and download the selected stow area to their hand-held scanner (Figure 1.17). By selecting a specific cargo item with the stylus (Figure 1.18) the embarker will be able to drag the cargo item to its new location.

Once the embarkers are satisfied with the new position of the cargo item on the screen, they will press Send to initiate a wireless socket connection between the AIT Scanner and the central
shipboard ICODES Server (Figure 1.19). The hand-held scanner will then send an XML message requesting that the ICODES server stow the cargo item in the new position indicated by the embarker. If ICODES is unable to move that item for any reason, an error message will be presented on the PDA and the cargo item will revert to its previous position. Alternately, if the cargo item is successfully moved and re-stowed in ICODES, its position on the hand held PDA will be adjusted in near real time. However, it should be noted that two embarkers working on the same stow area will not be able to update the same load-plan at the same time.

Figure 1.17: User selects a vessel and a stow area on the PDA

Figure 1.18: Moving cargo item with stylus

Figure 1.19: Cargo location is up-loaded
1.10.1 Multi-Modal Planning

To enable multi-modal planning, ICODES operators will use JINNI (see Section 1.9) to represent different kinds of stowable areas such marshalling yards, staging areas and piers (Figure 1.20). While each cargo placement area represented by JINNI incorporates the standard load-planning capabilities normally associated with ICODES, the tracking of cargo items as they repeatedly transition from one modality to another requires that ICODES support the multiple ghosting of cargo items. Since each stowable space will be reused several times during a typical deployment, a Track Move capability was designed to support the following requirements:

- Vehicles should appear on only one vessel at any one time.
- Cargo in any other kinds of stowable areas may appear in multiple instances, including ghosted images.

Figure 1.20: Representing staging areas and ship decks side-by-side in JINNI

- The following conventions for ghosted cargo items should be provided:
  
  A. Whenever the user selects the real cargo item, then all ghosted copies of that item will be highlighted as well.

---

18 In their ghosted form the graphical cargo symbols are rendered with a dotted outline and a gray overlay.
B. If a *ghosted* cargo item is selected, then any related cargo items (*real* and *ghosted*) will be also highlighted.

C. The user is able to move the *ghosted* version of a cargo item independently of its *real* counterpart. However, moving a *ghosted* cargo item will have no affect on any of the other representations of that cargo item.

D. The user is able to *track change* multiple times and have the *ghosting* from each change stay on the drawing. However only one *ghost* per vessel is supported.

E. To make an individual cargo item active, the user right-clicks on the cargo item and selects *Set Active*. This will place a *ghosted* version of the cargo item in the previous active location and make the *ghosted* version active.

1.10.2 *Graphical Embarkation Reports*

The I-AIT plug-in provides graphical reports of the cargo items as they pass through the various staging areas onto the ship, as well as the violations (if any) for a given ship. To access these graphical reports, the user at the central ICODES server will identify the particular stowable area from the ship menu and select the *Set As Primary Vessel* function. This action will place the real cargo items in their locations on that ship (or stowable area) and automatically generate any violations. Any cargo items that were previously *ghosted* will appear normally on the *Primary Vessel* and be represented in their *ghosted* form in all other locations, as shown in Figure 1.21.

Figure 1.21: Planning for discharge from ship (right) to port (left)
Once a ship or stowable space has been selected as the *Primary Vessel* the user has the ability to print a graphical report of that window. In that report all of the cargo attributes will be visible, however, the graphical representation of each cargo item will be in the normal format (i.e., not *ghosted*). A ship or stowable space must be made the Primary Vessel before any graphical reports can be printed.

1.11 The ICODES Rail Loader

Rail load planning is a complex operation involving an intricate web of associations and constraints that rail load-planners deal with on a daily basis. Rail cars are associated with other cars, all of which are part of a spur. Multiple spurs are used to create a train. Trains belong to a particular section of rail line in a particular terminal, and that section of rail has to have sufficient means (e.g., ramps and cranes) to load cargo onto the rail cars. The cargo to be loaded has to fit on the designated rail car as well as negotiate over or through any intermediate cars. In addition, hazardous cargo has to be properly segregated.

The complexity of rail load-planning does not rest in the planning of one car but rather the relationships among multiple rail cars that must be taken into consideration. Rail cars are often loaded as a group of connected cars, called a *spur*, in which one end is usually butted up against a ramp or near a crane. Cargo is either driven up the ramp and down the line of cars until it reaches the last available spot, or is lifted into place by a crane. Since there are different types of rail cars, there is the added consideration that the equipment used to load any particular rail car must be compatible with all rail cars that have to be traversed during the loading operation. For example, the *constant-inflation* system of many military vehicle tires can be damaged by rail cars having side walls. This explains why rail cars with side walls must to be located farthest away from the ramp.

There are several other considerations that must be taken into account during the planning of each rail car, such as the weight capacity of the car and the distribution of the load within the car. In addition, constraints relating to the maximum widths for cargo items at a particular height must be adhered to, since train routes involve turns through constricted spaces such as tunnels. In these situations, the cargo items and rail cars tend to lean slightly making the upper corners of loaded cargo vulnerable to damage (i.e., by striking these structures).

The ICODES Rail Loader allows the user to graphically generate load-plans and print both graphical and textual reports, while automatically alerting the user to any rail loading constraint violations that may have occurred. It utilizes the communications capabilities provided by the ICODES Plug-In Framework (see Section 3.2) and the ICODES External Services Interface (see Section 3.1) to harness cargo functionality already available in the main ICODES application. Such functionality includes the Cargo Agent and the ability to interface with external systems (see Sections 1.3 and 3.1).

The load-planning of rail cars proceeds according to a well defined sequence of tasks that may be summarized as follows:

**Step 1:** Open an existing *Rail Terminal* site plan and skip to *Step 3*. However, if such a site plan does not yet exist then proceed with *Steps 2* below.

**Step 2:** Build a *Rail Terminal* site plan using the JINNI space planning toolset:
• Construct the site with the required number of stowable spaces identified.
• Overlay the rail lines on the stowable spaces.
• Export the Rail Terminal to ICODES.
• Open the Rail Terminal site plan in ICODES.
• Add the required cranes and ramps to the Rail Terminal site plan.

Step 3: Select the rail cars by serial number and add them to the site plan. If a rail car serial number cannot be found then add the rail car to the car.list file.

Step 4: Add one or more trains to the plan. These are not depicted graphically.

Step 5: Insert rail splitters into the rail lines to divide them up into spurs.

Step 6: Associate rail sections with a crane or ramp to form a spur.

Step 7: Associate spurs with a train in the proper sequence order of rail cars (i.e., from engine to caboose).

Step 8: Place the rail cars into the spurs in the appropriate sequence order from front to back.

Step 9: Place cargo items into the rail cars.

Step 10: Print the appropriate set of reports that are specific to rail loading operations.

Step 11: Print the Commercial/Government Bill of Lading based on cargo item placement.

Step 12: Send the completed Rail Plan to the receiving destination (e.g., POD).

1.11.1 Rail Terminal Layout

A Rail Terminal is similar in many respects to the representation of staging areas and marshalling yards in a Marine Terminal. However, additional features include cranes, ramps, rail lines, and spurs. Cranes, like ramps, serve as loading points for placing cargo onto rail cars. The only difference is that cranes lift cargo onto the rail cars. The weight capacity, name and spur number of each crane can be edited by the user, and once a crane object has been placed on the site plan it can be moved, rotated, or removed as needed. Cranes are attached to spurs in a manner that is very similar to the establishment of associations (e.g., a hitch association between a vehicle and a trailer) in ICODES.

The physical dimensions of ramps are defined in terms of lengths and widths for both the sloped and horizontal sections. These two sections are combined into one graphical symbol, and once placed on the site plan the symbol can be moved, rotated, and removed as needed. The sloped section has an up notation pointing to the top surface (i.e., showing the direction of the ramp) automatically added to it. Apart from the name and spur number, the user is also able to edit the weight capacity of each ramp so that agents can produce warnings and alerts in case the safe ramp capacity limit is exceeded by particular cargo items in a load-plan. Like cranes, ramps are also attached to spurs.
The graphical depiction of rail lines allows the user to determine the number of rail cars that can be accommodated in a given space. As the user draws a rail line any changes in direction are automatically smoothed into a curve that can be negotiated by rail cars. The rail lines define spur locations in terms of their stopping and starting point. Therefore, rail lines would be normally created by first selecting the ramp or crane, so that the starting point of the spur is known. Spurs are special rail lines that are associated with a ramp or a crane. While a rail yard will have many rail lines, only those rail lines that are linked directly to a ramp or crane are spurs.

There is no limit as to how many rail cars can be in a given spur, but there is a limit as to how many cars can be in a train. This limit is generally understood to be 60 to 65 rail cars. In reality, a train is made up of one or more spurs in a particular order. Individual spurs are assigned to a train in a particular order by the user. However the spurs of a train should not be in close proximity to one another in the Rail Terminal, so that they do not encumber loading operations.

1.11.2 Trains, Spurs and Rail Lines

Trains are made up of one or more spurs and each spur is made up of one or more individual rail cars. Strictly speaking there are two terms that are used to identify a set of rail cars connected together. A cut is a set of rail cars connected to each other and a spur is a section of rail line track that is used to load rail cars and is accessible by rail at one end only. In the ICODES Rail Loader these two terms have been combined under the single term spur, which refers to any group of rail cars that are to stay together in a particular order at any location along a rail line. Since a rail line may have more than one crane or ramp attached to it, the user is able to split a rail line into multiple spurs. Although, to be usable, spurs would normally be created at the location of a ramp or crane there is no requirement for this in the Rail Loader.

To add a spur, the user simply selects the start and end splitters to be used for that spur by clicking inside the two rail lines. If the splitter is inserted inside of an existing spur then the user is advised that this is not a valid action and that the existing spur must be first removed. Since the sole purpose of a splitter is to serve as a device for creating spurs they cannot be added, edited, or removed individually, but can only be manipulated through changes made to spurs.

Rail lines are created based on a node at the centerline between two parallel lines. Although the user can create rail lines that intersect each other, during load-planning operations ICODES will not allow the user to place any rail car across such intersections. Rail Lines can be modified by adding and removing nodes and then adjusting the position of those nodes.

1.11.3 Cranes and Ramps

Cranes are generally located next to a rail line. This allows cargo items to be moved alongside the crane opposite the rail car. From an operational point of view the crane lifts the cargo item up, swings it around, and places the item onto the rail car. In some cases the crane can move sideways so that access to more than one rail car is provided. In other cases, the rail cars themselves will be moved as needed until they are all loaded. Sometimes a crane is located at the end of the horizontal platform section of a ramp.

Ramps serve a similar purpose to cranes in that they define the starting point of a spur. Although operationally loading a spur using a ramp is very different from using a crane, there is not much difference between them logically. In other words, for the purposes of rail-planning from the user’s point of view, a crane is essentially the same kind of loading device as a ramp.
1.11.4 Rail Cars

Since the ICODES Rail Loader is still under development at the time of writing this report, the actual disposition of a rail car library is still under discussion. The requirements of such a library are similar to, although less complex than, the existing Vessel Library. At the same time, consideration is being given to enriching the Symbol Library (see Appendix B) with additional cargo attributes that would provide a finer granularity of context for adding intelligence to the load-planning agents (e.g., the ability of the Assisted-Stow Agent to automatically load cargo items onto vehicles during transit). Therefore, it is likely that a new Symbol and Conveyance Library will be created that will contain rail cars and symbols with a greatly enhanced set of attributes. Eventually other conveyances such as trucks (for convoy load-planning) and aircraft could be added to this library.

Regardless of the final form of the library, the objectified representation of rail cars in ICODES requires detailed drawings that accurately describe the stowable deck space on a given rail car, including the following physical features:

- Demarcation of *no-stow* areas such as tie down points and the area around the hand brake.
- Location of tie down points and tie down type(s).
- Depiction of special features such as a raised deck or a flush deck.
- Location of attachments that are specific to container cells or tie down points.
- Location of rail car undercarriage trucks under the car deck.
- Identification of the deck area that is considered to be between the rail car truck centers.
- Type of rail car (e.g., TTDX), class (e.g., 2ASF10), and serial number (i.e., unique identifier).
- Dimensional data (Figure 1.22), including the following: length over side sills (A); width over side sills (B); distance between truck centers (C); deck height from top of rail (D); length over pulling faces (E); distance from truck centers to end of car ((A – C)/2); rail car cargo weight capacity in short tons; rail car empty weight in short tons; and, number of decks (i.e., double and triple deck rail cars).
- Whether flush deck or a raised deck.
- Chain types.
- Deck type (e.g., wood or metal)

![Figure 1.22: Critical rail car dimensions](image)
When a rail car is placed on a spur it snaps into position, automatically aligning itself with the particular rail line, and then slides as far forward as possible toward the front of that spur. Similarly, when a rail car is removed from a spur the following car automatically slides forward to take up the position of the removed car. The user may also insert cars between a line of existing cars. However, if a car is added to a rail line that is not defined as a spur, then the car will automatically align itself with the rail line but not slide to the front (because it is not a spur).

The arrangement of the cars on a spur is important for several reasons. First, it is common rail load-planning practice to place any flush deck cars closest to the front of a spur leaving the raised bed cars at the end of the spur. Second, rail cars with higher cargo weight capacity are normally placed closer to the ramps (i.e., toward the front of the spur). Third, since a train may be bound for more than one destination, agents check the sequence of cars for routing order discrepancies. In addition, the ICODES Rail Loader generates several reports that are dependent on the ability of the software to understand the order of rail cars within each train and spur. For example, a report at the Rail Terminal level that lists all rail cars sorted by train, spur order, car order within each spur, and the cargo items on each car with their respective associations.

1.11.5 Typical Rail-Loading Functionality

To construct a Rail Terminal site plan the user will drag and drop objects from the Facility tab in the Objects window into the main drawing area (Figure 1.23). These objects have default behaviors and properties that may be edited by the user.

![Figure 1.23: The main screen of the Rail Facility Builder component](image-url)
The first object selected by the user will be a rectangle that represents the boundaries of the Rail Terminal site plan. These boundaries will automatically expand to accommodate any additional objects that are inserted into the rectangle. Such objects would include individual terminal areas and the operational entities within each of these areas (e.g., cranes, ramps, rail lines, and spurs). To create spurs, the user will drop splitter objects (in pairs) to denote the section of the rail line track that is to be included in the spur. The directionality of the spur will follow the directionality of the rail line.

From the user’s perspective the ICODES Rail Loader consists of two principal components, the Rail Facility Builder (for constructing the Rail Terminal site plan) and the Rail Loader (for developing a rail load-plan). A representative sample of the available functionality in each of these components is briefly described below.

**Saving a Rail Terminal site plan:** At any time during the construction of the Rail Terminal site plan the user can save the current state of the drawing by selecting the File/Save option in the menu bar. If the file has been saved previously, then the new version will overwrite the previous version. Otherwise, a dialog window will be displayed to allow the user to browse to the desired directory.

**Restoring a Rail Terminal site plan:** To restore a previously saved Rail Terminal site plan, the user will select the File/Open option in the menu bar. A dialog window will appear to allow the user to locate and open the desired file.

**Exporting a Rail Terminal site plan:** Once the Rail Terminal site plan has been completed it can be exported to the Rail Loader. However, after the user has selected the
**File/Export to Rail Loader** option the Rail Facility Builder will first verify the integrity of the site plan before exporting the file. If errors exist, a dialog window will appear to notify the user and the export will be cancelled. If no errors exist, the export will write out the rail facility to a specific location where the Rail Loader will find it and add it to its list of Rail Terminal site plans.

**Creating a rail load-plan:** To create a new load-plan the user will select the File/New option in the Rail Loader (Figure 1.24). A dialog window will prompt for the entry of the name of the load-plan and the selection of anyone of the available listed site plans.

**Importing a rail car:** To import a rail car from the library the user will open the Rail Car Report menu and select Add New from the report’s toolbar. This will display the Import Rail Car dialog window. In this window the user can select the car type and quantity of rail cars to be added to the list. The customize feature can be used to sort and filter the types of cars visible in the table.

**Creating a train using the Train Report option:** A train is created through the Train Report table. After opening this table the user will select the Add New option from the report’s toolbar to display a dialog window. This window will allow the user to enter the required train information, including the name of the train and the spurs used to create the train. Any rail cars in these spurs will be automatically associated with the train. Once this dialog has been completed, the new train will be added as a row in the train list.

The parameters of the trains in this list may be edited at any time. For example, to change the spur list of the train, the user would click on the appropriate button in the table to bring up the Spur Sub-Report. To add a new spur to the list, this would be followed by the selection of the Add New option to display the Add Spur dialog window. Selection of one of the listed spurs will immediately add this spur to the train and automatically associate any rail cars in that spur with the train.

One of the editable attributes of a spur is the rail car list associated with that spur. To edit the rail car list, the user would click on the appropriate button in the Spur Sub-Report to display the Rail Car Sub-Report. Next, to add rail cars, the user would select the Add New button in the toolbar. This will display the Add Rail Car dialog, which includes a list of all available rail cars that are not currently associated with a spur or a train. The selected rail cars will be added to the table as individual rows in the sub-report and can all be edited by the user.

**Creating a train using the drag-and-drop option:** Alternatively, starting with the Rail Car Report, the user is able to select a rail car in a textual list and drag it as a graphic symbol onto the desired rail line in the site plan. The car will automatically orient itself to point in the direction of the track and will create all appropriate associations. If the car is dropped close enough to another car it will automatically hitch itself to the other car, again, creating all appropriate associations. Throughout this operation agent warnings will alert the user to any conditions that are in violation of known constraints.

**Importing a cargo list:** To import a cargo list from the main ICODES application the user would select the Cargo/Import ICODES Cargo in the menu bar. This menu item is available only while connected to ICODES.
Load-planning with the User-Stow option: In the Cargo Report the user has the option of loading cargo items onto rail cars and/or stowable areas. Either by dragging a row, or by selecting a row and clicking on the stow button, the user can drag the selected cargo item as a graphical symbol onto the site plan and drop it anywhere. However, an agent alert will be generated if the cargo item is not dropped in a valid stow area.

Load-planning with the Assisted-Stow option: To utilize an automated load-planning function, the user would select the Cargo/Assisted-Stow option in the menu bar. A dialog window will appear to allow the user to define the cargo items to be stowed. After clicking on the Stow button, the Rail Loader will try to place the selected cargo items onto valid rail cars without generating any violations.

Generating reports: The Rail Loader component provides a rich set of reports under the Cargo/Reports menu for cargo reports and under the Train/Reports menu for train reports. As a rule these reports can be sorted and are customizable by the user.

Saving a rail load-plan: To save a rail load-plan the user will select the File/Save option in the menu bar. If the file has been previously saved, then the new version of the load-plan will overwrite the old version. If the load-plan is being saved for the first time, then a dialog window will be displayed to allow the user choose the location and name of the saved file. Alternatively, the user can also select the File/Save As option at any time to save the file directly.

Restoring a rail load-plan: To restore a previously saved rail load-plan, the user needs to select the File/Open option in the menu bar. A dialog window will be displayed to allow the user to locate and open the desired file.

1.12 Operational Impact Analysis of ICODES

During the more than eight years of ICODES delivery and wide-spread military use no quantitative metrics have been collected to compare military ship load-planning from the period prior to the availability of ICODES (i.e., prior to 1997) and after ICODES became the system of record for Army, Marine Corps, and Navy surface load-planning. However, it is generally accepted within the military load-planning community that ICODES has been responsible for a dramatic improvement in decreasing the loading time of ships and berthing costs. In addition, ICODES further proved its utility in areas for which it was not anticipated, such as ship selection for the movement of supplies, cargo in-transit visibility, historical analysis of cargo movements, and ship design. The following selected areas of military load-planning operations may serve as indicators of the improvements in operating efficiency and cost savings that have been achieved through the deployment of the ICODES suite of adaptive tools over the past several years.

Load-planning efficiency: Previous to the fielding of ICODES, the creation of a pre-stow plan would often take one load-planner using the DOS-based CODES software at least two days. Once the cargo list had been cleansed, through the laborious manual process of comparing the data pertaining to each cargo item with the official equipment library, often a day long process, the load-planner would copy-and-paste the cargo symbols on the ship deck drawings. Then other planners with expertise in hazardous cargo stowage, trim and stability, and cargo flow would check the plan, which often took
another day. This time consuming cycle would begin again for each time the cargo list was updated, often up to 30 times during the development of a pre-stow plan.

With ICODES, and in particular through its agents (i.e., Cargo, Access, Trim and Stability, Hazard, and Stow Agents), a load-planner is able to create a similar pre-stow plan in about three hours. When updated cargo lists arrive the ICODES merge function allows the same plan to be updated within minutes without re-starting the planning process.

Marine Corps cargo specialists have indicated that prior to the availability of ICODES the planning of the equipment for a Marine Expeditionary Unit (MEU) involving 10 to 14 ships would take an Operation Planning Team five to seven days. With ICODES this task has been reduced to about 14 hours.

**In-transit visibility:** An area of support that did not exist prior to ICODES is the electronic submission of cargo manifests and cargo ship placement reports to the ship personnel and to the POD staff. This capability has provided visibility of cargo to the ship to assist with in-transit issues, to the POD for off-load-planning and/or load-planning of new loads, and to military administrative personnel for tracking and historically reporting on cargo movements.

At a POD, prior to ICODES, immediately after the arrival of a vessel a cargo survey and meeting would be held to discuss cargo placement and off-loading strategies. With the availability of ICODES documentation this half-day delay is no longer necessary resulting in a significant saving of berthing costs. In addition, the off-load-planning that can now be accomplished with ICODES prior to ship arrival results in substantial labor and off-load space assignment savings.

For ships with multiple ports of loading and discharge, ICODES load-plans are now passed electronically from port to port to determine the effects of the loads and off-loads on the ship and provide a common operating picture. Beyond the port, the Army Logistic Operations Center in the Pentagon uses a database of ICODES-generated load-plans to estimate off-load times. In the past this has been a labor intensive operation, often resulting in missed deadlines.

**Trim and stability analysis:** Since the ICODES Trim and Stability Agent utilizes certified formulas for ship trim and stability calculations, the results are not only used by load-planners but also by the ship’s crew to confirm ship loading conditions. Because of the trusted quality of the validated ICODES trim and stability analysis, ships are much less prone to unsafe stow configurations and further, sail up to a day earlier than in the pre-ICODES era. The earlier departure of ships leads to fuel savings since ships are able to proceed at reduced speed and still stay on schedule. In addition, ships stowed using the precision and operational knowledge offered by the ICODES system experience decreased port costs associated with berthing and service fees.

Prior to the availability of ICODES ships were often loaded with little concern for the distribution of weight along the ship’s perpendicular axis, eventually causing several classes of ships to develop stress factures. The continuous monitoring of the condition of the ship during load-planning has led to better load distributions and the resultant reduction in costly ship repairs.
Reconciliation of planned cargo placement: Using the ICODES Automatic Information Technology (AIT) capabilities, the staging area cargo placement and the ship as-loaded plan is confirmed with hand-held Personal Digital Assistants (PDA), as opposed to manually drawn sketches and tally sheets. Using the ICODES AIT functionality, personnel costs have been reduced to about 20% of the cost of the manual process and the number of port cargo administrative personnel have been reduced by about 50%. With the increasing availability of AIT wireless communications at ports cargo locations are updated automatically to an ICODES computer in the port command center, allowing near real-time visibility of cargo to port administrative personnel and preventing the misplacement of hazardous materials

Since its first release as a system of record in 1997, the granularity of the cargo data has increased greatly as ICODES moved from Level 4 to Level 6 detail. A typical Army cargo list in 1997 seldom included more than 2,000 individual cargo items. Today (2006), ICODES is required to process Marine Corps cargo lists with more than 30,000 individual cargo items. Despite this increase in the volume of data the performance of ICODES, in terms of response time, has continued to reduce as well. The typical performance results shown in Table 1.1 are based on periodic metrics collected by CDM’s ICODES Test Group over the past eight years.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Create two-ship load-plan with 2,400 normal cargo items</td>
<td>20 min</td>
<td>8 min</td>
<td>1.5 min</td>
</tr>
<tr>
<td>Create two-ship load-plan with 1,200 hazardous cargo items</td>
<td>25 min</td>
<td>11 min</td>
<td>2.5 min</td>
</tr>
<tr>
<td>Unstow inventory of 2,400 items from two ships</td>
<td>10 min</td>
<td>5 min</td>
<td>1.0 min</td>
</tr>
</tbody>
</table>

2. ICODES as a Suite of Knowledge Management Enterprise Services

This section discusses ICODES from a technical perspective. The entire ICODES system is dissected into its individual Knowledge Management Enterprise (KMES) components and examined in terms of design methodology and exploited technologies. In addition, fundamental architectural patterns such as its Service-Oriented Architecture (SOA) and plug-in-oriented extensibility are described along with key component interaction models. However, before embarking on a technical journey into the inner-workings of the ICODES agent-based decision-support system, a discussion of the underlying software design philosophy on which ICODES is founded is in order. Collectively, these qualities are derived from the KMES approach and the ICDM-based principles employed by both CDM and the CADRC Center in the design and development of agent-based, decision-support systems.

2.1 KMES: A Net-Centric Architecture

Knowledge Management Enterprise Services (KMES) are self-contained software modules with powerful functional capabilities and clearly defined interface specifications. They are designed to be platform independent with the ability to automatically configure themselves to a particular execution environment. However, most importantly they are reusable as components of any system that has a need for their capabilities. Some of these services may have quite narrow capabilities such as the mapping of cargo data imported from an external source to an internal information model or ontology, while others will incorporate larger functional domains such as the optimum routing of goods from multiple origins along alternative routes to multiple destinations.

The KMES approach to software systems incorporates intelligent agent technology to provide an internal staff of software agents. These agents analyze and categorize incoming signals and data, and then issue warnings and alerts as appropriate. The agents manipulate the incoming data within an internal information-centric representation framework to publish statements of implication, and if so empowered, proceed to develop plans for appropriate action. Legacy data-centric systems can become clients of such an agent-based KMES software environment through the use of interoperability bridges that map the data model in one system to the information model of the other and allow a two-way exchange.

Conceptually, an intelligent net-centric software environment calls for the seamless merging of a KMES-based information management facility with existing data sources. This can be achieved with an information-centric architecture that consists essentially of two components (Figure 2.1): a data-centric Data Capture and Integration Layer that incorporates linkages to existing data sources; and, an Intelligent Information Management Layer that resides on top of the data layer and utilizes software agents with automatic reasoning capabilities, serving as decision-support tools.

The Intelligent Information Management Layer architecture (Figure 2.1) utilizes intelligent software agents capable of collaborating with each other and human operators in tactical and logistical command and control environments. Typically such intelligent systems are based on software development frameworks, such as the ICDM software development environment used extensively by CDM during the past 12 years for the development of military and commercial systems, including the ICODES suite of tools (Pohl et al. 2004).
Data Capture and Integration Layer: The bottom layer of the system takes the form of an operational data store and/or Data Warehouse, implemented within a commercial off-the-shelf relational database management system (RDBMS). This data categorization facility integrates data extracted on a periodic basis from several external sources into a common data schema. The design of the data schema is closely modeled on the structure of the ontology of the Intelligent Information Management Layer to minimize the required data-to-information and information-to-data mappings between these two system layers.

In conformity with normal enterprise data management practices the Data Capture and Integration Layer incorporates the following four characteristics:

- It is subject-oriented to the specific business processes and data domains relevant to the application area (e.g., shipment of supplies from the CONUS to a military theater).
- It is integrated so that it can relate data from multiple domains as it serves the data needs of the analysis functions performed by collaborative agents in the Intelligent Information Management Layer.
- It is periodically updated through its linkages to external data sources.
- It is time-based to support the performance of analyses over time, for the discovery of patterns and trends.
A multi-tier architecture is used to logically separate the necessary components of the data layer into levels. The first tier is the RDBMS, which ensures the persistence of the data level and provides the necessary search capabilities. The second tier is the service level, which provides the interface to the data level and at the same time supports the data access requests that pass through the mapping interface from the Intelligent Information Management Layer to the Data Capture and Integration Layer. It is designed to support request, response, subscribe, and publish functionality. The third tier is the control level, which routes information layer and user requests to the service level for the update, storage and retrieval of data. Finally, a view layer representing the fourth tier serves as a user-interface for the Data Capture and Integration Layer.

**Information Management Layer:** The Intelligent Information Management Layer consists of KMES components in the form of a group of loosely coupled and seamlessly integrated decision-support tools. The core element of each KMES component is an ontology that provides a relationship-rich model of the particular decision-support domain. Typically, KMES components are based on a three-tiered architecture incorporating technologies, such as distributed-object servers and inference engines, to provide a framework for collaborative, agent-based decision-support that offers developmental efficiency and architectural extensibility. The three-tiered architecture clearly distinguishes between information, logic, and presentation. Most commonly an information tier consists of a collection of information management servers (i.e., information server, subscription server, etc.), while a logic tier incorporates an agent engine, and a presentation tier is responsible for providing interfaces to human operators and external systems.

The notion of service-oriented is represented as much in the elements of each of these tiers as it is in the functional capabilities of each KMES. Therefore, even the internal elements of a KMES communicate through standard interfaces as they provide services to each other. They are, in essence, decoupled software modules that can be replaced with improved modules as the technology advances. Each of these modules functions in an integrated fashion to form a comprehensive agent-based decision-support execution framework.

The underlying ontology of a KMES is normally divided into several related domains. While some of these domains describe application-specific events and information (e.g., goods movement transactions, shipping routes, and so on) others describe more general, abstract notions (e.g., event, threat, view, privacy). The goal in developing such an ontology is to abstract general, cross-domain notions into high-level domain models. As such, these descriptions can be applied across several application sub-domains. More domain-specific, concrete notions can then be described as extensions of these high level models.

Commonly, a KMES ontology includes several primary meta-characteristics. Through inheritance, these meta-characteristics can be propagated to more specific ontological components. For example, one of these meta-characteristics may be the property of being trackable. If this characteristic is introduced at the physical.Mobile level then, through inheritance, any entity that is kind of physical.Mobile automatically receives the property of being trackable. Taking this example further, a second meta-characteristic may relate to the dispensability of an item. If this property is represented at the physical.Item level then, similar to the trackable characteristic, anything that is a kind of a physical.Item automatically receives the...
quality of being dispersible or suppliable. In addition, as an extension of physical.Mobile, such suppliable items are also trackable. It can be readily seen that together these two meta-characteristics provide an effective foundation for assigning meaningful context to data through such an ontology representation.

While meta-characteristics are only implicitly represented in an ontology, other notions may be represented explicitly in the form of object classes. For example, two such notions in the logistical application domain might be Container and Empowerable. A Container holds (i.e., contains) components. In fact, a Container can be thought of as a dynamic set of components. There is no inherent order imposed. However, while the contents of Containers may be very different all Container objects have a common derivation. An Empowerable object is one that can be dynamically embodied, or enhanced with additional information, knowledge, or capabilities. This is intentionally a very open-ended notion allowing for unconstrained exploration into various potential powers that can be imparted to an object.

2.2 ICDM: Philosophy and Principles

For the past two decades the CADRC Center at Cal Poly (San Luis Obispo) and more recently CDM Technologies have pursued the design and development of agent-based decision-support systems\textsuperscript{20}. Throughout this journey the CADRC and CDM have relied on the Integrated Cooperative Decision-Making (ICDM) software development toolkit (Figure 2.2) to assist them in the creation and management of such systems.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{ICDM.png}
\caption{The ICDM development toolkit}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{ICDM_principles.png}
\caption{ICDM design principles}
\end{figure}

Not only does ICDM function as an accelerator (i.e., rapid development) and stabilizer (i.e., built-in robustness and fault tolerance) in the development of decision-support systems, but it also provides a concrete vehicle for representing the key concepts and philosophies that the

\textsuperscript{20} CDM Technologies, Inc. is the commercial arm of the Collaborative Agent Design Research Center (CADRC) at California Polytechnic State University (Cal Poly), San Luis Obispo.
CADRC Center and CDM have found to be useful for the success of KMES-based systems (Pohl et. al. 2000, Pohl 1997). The key design principles on which ICDM is founded are collaboration-intensive, context-based representation, flexibility and adaptability, multi-tiered and multi-layered, within the framework of a service-oriented, distributable architecture (Figure 2.3).

Although no formalized reference model currently exists to concretely define a service-oriented architecture (SOA) the software engineering community is in basic agreement as to the concepts associated with a SOA approach (Newcomer and Lomow 2004). It is also generally acknowledged that under current circumstances not all of these concepts need to be comprehensively present in order to qualify as a SOA, thus highlighting the need for further standardization. Nonetheless, the following concepts are generally consistent among most SOA interpretations:

- **Services**: An encapsulated body of behavior or capability that can be contracted by a requestor in an effort to satisfy some need.

- **Service Descriptions**: A detailed description of an available service, among other information, describing the input, output, and general contract for employment.

- **Advertising and Discovery**: Ability of a service to advertise its capabilities in an accessible manner (i.e., push or pull fashion) to prospective consumers.

To fulfill their contracts, services may transparently employ other services thereby effectively obscuring the line between service providers and service consumers. Further, SOA-based systems lend themselves to flexible distribution across the available computing resources. This characteristic promotes considerably more flexible load-balancing and scaling capabilities than are offered by more monolithic architectures. In addition, because of their discrete nature, systems designed around a SOA methodology benefit from a low degree of coupling among their components (i.e., consumers depend on selected services and their associated contracts, not necessarily specific service instances or implementations). Due to this decoupled nature, capabilities within a SOA-based environment can be dynamically added, modified, or removed with little or no impact on service consumers as long as existing contracts are still supported and new or replacement services are discoverable through accessible registries.

### 2.2.1 Collaboration-Intensive

Certainly in the real world, collaboration among decision-makers and experts is a critical ingredient in making educated and effective decisions. This is especially true when operating across an extensive and varied set of domains. Through years of research in collaborative design the CADRC has found that this same quality extends to the realm of agent-based decision-support systems. Conceptually, the systems developed by the CADRC and CDM consist of dynamic collections of collaborators (both human and software-based) each playing a role in the collective analysis of a problem or situation and the consequential decision-making assistance required in formulating an accurate assessment and/or solution (Figure 2.4).

Whether human or software-based, collaboration within an ICDM-based system occurs in terms of a descriptive ontology (Chandrasekaran et. al., 1999). Until recently these ontologies were limited to describing information and knowledge that represents various aspects of the domain(s) over which the system is to operate. For example, in the domain of architectural design the applicable ontology would describe such notions as spaces, walls, accessibility, appropriate lighting, and so on. Although effective and certainly a fundamental element of an information-
centric system a considerable portion of the system still remains in a form not necessarily supportive of highly collaborative environments. Further, these non-ontology based components require separate and dedicated interfaces along with specialized management. A number of the services that collaborators within an ICDM-based system interact with (e.g., time, query, subscription, execution, reasoning, etc.) were still presented as client-side adjunct-based interfaces requiring additional management to support collaboration. For example, if two clients wish to share or discuss the same subscription profile, a separate mechanism for identifying and referencing the collection of interests is required. In this case, the interface would be the client-side Application Programming Interface (API) maintained by the subscription service itself. Although certainly possible, supporting such specialized functionality requires the particular services (i.e., the subscription service in this case) to present and manage a specific API to expose or match global references. Although subtle in nature, complexity such as this can easily escalate when considering the high degree of collaboration inherent in multi-agent KMES components.

This limitation may be overcome by extending an ontology to include behavioral aspects of the decision-support system in the form of object-based constructs representing the services within the KMES component (i.e., the services themselves are represented in the collaborative ontology in the same manner as information and knowledge). The only difference is that these distributed and shareable objects offer behavior in addition to information. As a result collaborators are able to interact with these services through the same distributed object operations that they would perform on the information and knowledge objects. Any constraints identified in the behavior are enforced by the standard ontology management facility. The operations that can be performed on these ontology-based objects consist of the basic creation, deletion, and modification functionality. To support the aforementioned example in which two collaborators wish to reference and discuss aspects of the same subscription profile, the two collaborators would treat the profile in question as just another set of multi-faceted, shareable distributed object. In other words, similar to the manner in which rich information models, or ontologies are used as a basis for collaboration this notion is extended to include interaction and collaboration across the
services that constitute the system itself. The effect is essentially that interaction with, and collaboration across, information, and now behavior (e.g., services), is reduced to a basic set of object manipulation capabilities. In this sense, the fundamental subject of all operations, whether in the form of access or manipulation, is an object. The only difference is that some distributed, shareable objects offer information and some offer behavior. The client-side portion of the ontology replaces the need for specialized client-side functionality.

### 2.2.2 Context-Oriented Representation

Prior to the commencement of the ICODES project the Collaborative Agent Design Research Center at Cal Poly (San Luis Obispo, California) had developed several decision-support applications utilizing (under license) the Integrated Cooperative Decision Model (ICDM) development toolkit for multi-agent systems (Pohl and Myers 1992; Myers et al. 1993; Pohl, Myers and Chapman 1998). An ICDM-based application is based on an information-centric premise, in the sense that it incorporates an internal information model of objects, their characteristics, and the relationships that associate these objects to each other and the functional capabilities of the application (Myers and Pohl, 1994; Pohl et al. 1992; Pohl 2002).

The term information-centric refers to the representation of information, as it is available to software modules, not to the way it is actually stored in a digital machine. This distinction between representation and storage is important, and relevant far beyond the realm of computers. When we write a note with a pencil on a sheet of paper, the content (i.e., meaning) of the note is unrelated to the storage device. A sheet of paper is designed to be a very efficient storage medium that can be easily stacked in sets of hundreds, filed in folders, folded, bound into volumes, and so on. As such, representation can exist at varying levels of abstraction. The lowest level of representation considered in Figure 2.5 is wrapped data. Wrapped data consists of low-level data, for example a textual e-mail message that is placed inside some sort of an e-mail message object. While it could be argued that the e-mail message is thereby objectified it is clear that the only objectification resides in the shell that contains the data and not the e-mail content. The message is still in a data form offering a limited opportunity for interpretation by software components.

A higher level of representation endeavors to describe aspects of a domain as collections of interrelated, constrained objects. This level of representation is commonly referred to as an information-centric ontology. At this level of representation context can begin to be captured and represented in a manner supportive of software-based reasoning. This level of representation (i.e., context) is by far the most empowering design principle on which ICDM is based. Further, as mentioned in the previous section portions of this context may be extended to exhibit behavior. In addition to services, however, distributed behavioral objects can also be employed as a mechanism for supporting the notion of facades.

Existing as one of the fundamental design patterns employed in object-oriented design, facades provide a level of derivation attained from the particular representation or ontology on which they are based (Pohl K. 2001). In the case of ICDM and the kinds of ontologies it manages, facades offer a method of supporting and managing an alternative perspective from that modeled in the ontology from which they are derived. In other words, ICDM-based facades allow the perspective inherent in a particular model of a domain to be augmented, or in some way altered to support a more appropriate (i.e., to the façade user) representation of the concepts, notion, and entities over which that user is operating (Figure 2.6). Note that user in this sense refers to any
accessing component. While certainly useful in systems supporting multiple perspectives caution must be employed in preventing abuse by introducing inconsistency and unnecessary duplication.

![Objective: To support component replacement, addition, and reuse, through a low degree of coupling.](image)

- All communications via distributed objects:
  - A client has no knowledge of other clients.
  - An agent has no knowledge of other agents.
- Support interoperability with external systems through the concept of hybrid facades serving as perspective filters of ontologies:
  - A facade may provide a local perspective for one client.
  - A facade may be shared and provide a collaborative environment for multiple clients.
  - As a behavioral entity, a facade provides a transparent interface with capabilities ranging from:
    - Changing the value of a client attribute
    - Modifying object constraints
    - Extending and overwriting an existing ontology
    - Performing diverse management functions

Figure 2.6: Extensible and Adaptable

Facades can also be utilized to support real-time calculations. In this sense, the façade derivation would involve a calculation or algorithm perhaps based on one or more attributes of the base object(s). For example, consider an architectural space exhibiting length, width, and height described in American pound/foot units which is to be accessed by a design system that only understands Metric kilogram/meter units and also requires space volumes. Utilizing ontology-based facades a model could be developed in which, not only the length, width, and height, but also the volume of the space could be calculated and presented to the design system in terms of Metric units. Although there are a number of approaches to supporting calculated attributes in the case where an alternative perspective is to be supported, the façade approach permits an extensible (i.e., one perspective extended from another) and encapsulated (i.e., easily maintainable) solution.

### 2.2.3 Extensible and Adaptive

One of ICDM’s primary goals is to support a high degree of flexibility in respect to the configuration of its components both at the development and execution levels. ICDM supports the addition, replacement, and reuse of software components in the context of agent-based, decision-support systems, and achieves this goal by reducing inter-component coupling to an absolute minimum (Figure 2.6). There are two key ICDM properties that permit this flexibility. First, all collaboration between clients takes place via, and in terms of the informational ontology (i.e., distributed objects). No direct communication exists between collaborators. The result is a collaborative environment in which client identities are essentially irrelevant in respect to this process.
The second property deals with the manner in which clients access and interact with the ontology. ICDM offers a standard interface component known as the Object Management Layer (OML) which both shields accessors from the complexity of ontology management as well as provides an abstracted view of the ontology. Clients of OML interact with the ontology via object wrappers (POW) based on a set of corresponding ontology-specific templates. Promoting the notion of adaptability, these templates are discovered by OML as a runtime activity. The resulting support for dynamic definition permits elements of the ontology to be extended, eliminated, or even redefined during the course of a runtime session.

Apart from the ability to adapt to an evolving definition of a domain, adaptability is also supported in interaction with external systems. This level of adaptability functions in conjunction with the concept of façades mentioned earlier. Replacing the classical approach of building a dedicated and separate translation bridge between collaborating systems, ICDM promotes the incorporation of such translation into the ontology itself. In other words, using ICDM’s support for ontology-based façades, translation or derivation of each system’s perspective can be encapsulated and managed solely within façade objects. The resulting translation facility exists as a set of behavioral façade objects accessed and manipulated in a manner no different than is applied to other ontology objects. The result is an elegant design where support for translation-based communication between disparate systems is seamlessly incorporated as part of the ontology.

### 2.2.4 Multi-Tiered and Multi-Layered

The forth design principle to which ICDM adheres addresses the architectural organization of ICDM-based systems. More specifically, this principle identifies distinct separations between areas of functionality at both the conceptual (i.e., tier) level and the more concrete (i.e., layer) level. Conceptually, the architecture of an ICDM-based decision-support system is divided into three distinct tiers namely, information, logic, and presentation. To manage its particular domain each tier contains a number of logical layers that work in sequence (Figure 2.7).

As the name suggests the information tier houses both the information and knowledge (i.e., ontology) being operated on in addition to all of the mechanisms needed to support management, transport, and access. The information is further delineated into layers. The first of these is the Object Management Layer (OML) described in an earlier section. Below the OML resides the Object Access Layer (OAL) responsible for managing access to the information tier. The OAL exists as a level of abstraction below OML and interfaces directly with the Object Transport Layer (OTL). Based on the CORBA specification (Mowbray and Zahavi 1995) the OTL is responsible for communicating the various requests and subsequent replies for distributed information and behavior issued through the OAL throughout the system. The OTL is the only layer that forms a dependency on an underlying communication protocol. As such, support for alternative communication facilities can be implemented with minimal impact on either the OAL or the OML. This is an excellent example of the benefits of a layered architecture in supporting component reuse and replacement.

The Logic Tier contains the business rules (i.e., agents) and analysis facilities by which these rules are managed. Although extensible to include other forms of reasoning the current version of ICDM focuses on opportunistic rule-based analysis. Regardless of which form of reasoning is employed this capability is supported by two layers namely, the Business Rule Layer (BRL) and the Business Engine Layer (BEL). The BRL is primarily system-specific and contains the agent-
based analysis facilities resident in the system. Execution of agents is in turn managed by the BEL. To integrate the Logic Tier with the Information Tier the BEL interfaces with OML permitting the agents to both access and contribute to the ontology.

The final tier is the Presentation Tier. This tier is responsible for interfacing with the various users of the system. In this sense a user may be a human operator or an external system. In the case of a human operator support is provided through a Graphical User Interface Layer (GUIL) that presents and promotes interaction with the contents of the Information Tier. In the case of an external system, support takes the form of a Translation Layer (TL) that manages the mapping of representations between systems. Like the GUIL, access to and from the Information Tier is supported by OML.

2.3 The ICODES Domain Model

The ICODES Domain Model in its entirety encompasses the concepts and entities that essentially comprise the view that ICODES has of the world of stow planning. In other words, the ICODES Domain Model provides expressive, context-oriented descriptions of both the tangible entities (e.g., vessels in terms of stowable areas, on-board facilities, etc., and cargo in terms of geometry, weight, etc.), as well as the intangible concepts (e.g., hazardous constraints, mobility, preference, accessibility, sequencing, etc.) necessary to support the high level of decision-support offered by the ICODES suite of tools.

The Model contains a number of sub-domains, including the Vessel Domain, Cargo Domain, and a domain describing the logistics of space planning in general. The following sections discuss each of these three domains in detail.

2.3.1 Vessel Domain

As the name implies, the ICODES Vessel Domain model (Figure 2.8) provides a logistically biased view of a vessel. This representation has evolved substantially since the first version of ICODES was released in 1997. Both the advances in maritime technology, as well as the increasing demands from the user-base to support different types of load operations and cargo types, have triggered the evolution of the Vessel Domain model. A clear example of this need for enrichment of the Vessel model is the increase use of containers in the world of maritime transportation over the past 10 years. This requirement has resulted in model extensions to support container cells and tiers, as well as different container cell numbering systems such as the Baplie and MILSTAMP conventions. The following is a description of each of the primary elements comprising the Vessel model:

Stow Areas and Zones: The stowable space in the vessel is represented by StowAreas that keep track of the weight and area occupied by the cargo stowed in them. Further, to facilitate trim and stability calculations, StowAreas also record their center of gravity. Existing as a less formalized stowable area, Zones allow the user to represent a subsection of a StowArea. Among numerous other attributes, Zones record their maximum cargo height and maximum deck stress. They can be used to represent a variety of stowing conditions, each imparting their distinct semantics. For example, NoStowZones can be used to demarcate areas of the vessel where cargo should not be stowed. Likewise, OffLoadZones can be used to represent temporary stow locations external to the vessel where cargo is to be off-loaded.
Figure 2.8: The Vessel Domain model

**Access Entities:** The Vessel model supports many different types of access entities, as those parts of the vessel that have to be traversed by cargo items as they move to their final stow location on the vessel. Examples of such access entities include booms, bulkheads, doors, elevators, hatches, openings, and ramps. These transition points must be richly described within the vessel domain in order to allow analytical agents to both identify and resolve potential accessibility issues.

**Trim and Stability:** Most of the trim and stability information in the vessel domain is derived from a standard Ship Data file (i.e., SDA file) provided by the Maritime Administration (MARAD)^21. The vessel model contains expressive descriptions of available tanks, ballast, hydrostatic properties, bonjean curves, strength, and draft marks critical for accurate trim and stability analysis.

---

^21 MARAD regularly surveys the ships used by SDDC for the transportation of military supplies and provides CDM with updated ship drawings and characteristics files. CDM as the custodian of the Master Vessel Library objectifies the updated ship information and makes it available to authorized users through the ICODES web-site.
Container Representation: The Vessel model employs the notion of container tiers to support vessels that are equipped to transport containers. Each container tier has a collection of container cells that represent the locations where individual containers can be stowed. These container tiers provide the user with a top down view of the locations where container can potentially be stowed. Container cells can also be grouped into container stacks and container bays that provide a cross-sectional representation of where containers can be stowed. Supporting referential standards, these container cells can be identified by either Baplie or MILSTAMP numbers.

Ship Gear: The Vessel Domain also supports the notion of gear that is part of a particular vessel used and can be used in support of stow operations. Examples of such ship gear include forklifts, pallets, sweepers, and scissor trucks. As an integral part of the ICODES trim and stability analysis, the Vessel model representation of such on-board gear includes both weight and dimensional information.

2.3.2 Cargo Domain

The ICODES Cargo Domain model encompasses data and relationships directly related to the placement and tracking of cargo items (Figure 2.9). The representation of cargo within this domain includes not only the dimensional parameters of each cargo item, but also descriptions of hazardous material constraints (if any) and other logistical information required to effectively support the agent-based decision-support capabilities housed within the ICODES space-planning environment.

Figure 2.9: The Cargo Domain model

Each cargo item records information about its location to support the precise tracking of cargo as it moves from staging area to berth and through the ship to its final location. As such, items may contain multiple locations able to convey not only their current position but also where they have come from and where they are going in terms of stow or staging locations. Within the user
interface, ICODES uses ghost images of cargo to provide users with an easily discernable view of this dynamic locality information.

As mentioned earlier, cargo items include one or more hazardous information objects as integral parts of their description. Each of these objects represents information about the particular hazardous material(s) that comprise them.

Although ICODES allows the user to create a complete cargo list from inception, it is a more normal procedure for the cargo list to be imported from an external system such as TCAIM-II, MDSS-II, or WPS (see Section 1.3). Each of the external systems that ICODES supports may have several attributes unique to that system. Through various formal interface agreements ICODES supports such attributes and is capable of displaying the information in the same fashion as native attributes. Since these external attributes are considered pass-through data from one system to another ICODES only provides a limited ability for the user to alter these values.

### 2.3.3 Plan Domain

The ICODES Plan Domain essentially aggregates all of the information relating to a stow plan into a cohesive Plan (Figure 2.10). Some of the information contained within an ICODES Plan includes relevant vessels, cargo, and outstanding agent reports including violations. With the exception of agent reports, which are represented within this domain, each of these elements is described in terms of the Vessel and Cargo models discussed above.

![Figure 2.10: The Plan Domain model](image)

### 2.4 System Architecture

As a KMES-based system, designed according to SOA principles and implemented within the ICDM development environment, ICODES incorporates a three-tier architecture that draws a clear distinction between representation, logic and presentation. It is a multi-agent system based on a knowledge management premise, in the sense that it incorporates an expressive information model consisting of context-oriented objects, their detailed characteristics, and the relationships that associate these objects to each other and the functional capabilities of the application.
This internal information model provides the necessary context to enable reasoning agents to collaborate with each other and the human user to collectively evaluate events and generate warnings and alerts. Figure 2.11 provides a view of the ICODES conceptual architecture. Among other aspects, this diagram clearly illustrates the attention to the separation of concerns (i.e., representation, logic, and presentation) that is embedded in the design. Figure 2.12 provides an additional view of ICODES, presented in the Unified Modeling Language (UML) symbology (Fowler 1997), describing the significant components along with their inter-dependencies. This alternative view also emphasizes the attention given to inter-component visibility and the objective of a decoupled architecture.

### 2.4.1 Domain Tier

The domain tier, also referred to as the semantic network, of the ICODES multi-tiered architecture is essentially comprised of a set of services that govern access to, and general lifecycle management of, the contextual objects representing the space-planning domain. The scope of this contextual description goes beyond representation of the physical elements that comprise the load-planning environment (i.e., vessels, cargo, etc.) but also include the more intangible, and sometimes much more subtle, concepts and notions that are vital to telling the complete story of the problem space and evolving solution(s). For example, included in the domain model and housed within this tier are expressive descriptions of the notions of restrictions, violations, recommendations, and accessibility. These are the intangible concepts that empower the ICODES agents with the ability to grasp a deeper understanding of the space-planning activity than would be possible with a data-centric representational paradigm.
The following sections discuss the ICODES domain tier in terms of both the service-oriented components that comprise it as well as the reasoning-enabling context model it manages.

**Semantic Network Service:** The primary service within this tier is the *Semantic Network Service*, which has five related roles. First, it acts as a repository for objects and certain metadata in the semantic network. Second, it provides transaction management for any actions operating over this network of objects. Third, it arranges for the maintenance of the persistent storage of objects. Fourth, it stores dependencies of objects or tagged data on objects housed within the semantic network. Finally, it arranges for notification of interested components (i.e., agents, plug-ins, etc.) when objects in the semantic network change.

Objects are added to, or removed from, the semantic network either through the actions of the user via the user-interface or by software components (i.e., agents, etc.) within the system. The *Semantic Network Service* records sufficient information concerning new objects to allow for the later retrieval of these objects. In addition, this service assigns an identifier, or *SNID* (Semantic Network ID), to each object that it receives. Special
provisions are made for *singleton* objects, which are permitted to use pre-assigned identifiers within a special range. These identifiers are unique within an ICODES load-plan. Upon removal of an object from the semantic network, the manager will automatically remove any objects or tagged data that are dependent on that object.

Every type of object and agent in the semantic network also has a related logical description. This meta-level description contains information about each attribute of the object, such as its name and type. The *Semantic Network Service* collects this information for each object that is present in the system. During start-up, the *Semantic Network Service* also discovers and interrogates descriptions of objects that were not known to the ICODES program when it was created. This allows components (i.e., agents, plug-ins, etc.) to create additional objects in the semantic network that are otherwise indistinguishable from the pre-determined types of available objects. Instances of these run-time defined objects are given lifecycle management by the *Semantic Network Service* and are therefore accessible to external components.

All objects that are part of the semantic network are considered *immutable* except through calls to the *Semantic Network Service*. Since this service has no knowledge of the attributes of the objects in its care, with the exception of their identity, an additional mechanism must be present in order to make changes to semantic network objects. This is achieved through the notion of *shadow objects*. A shadow object exists as a copy of a semantic network object that is a candidate for modification. The primary difference between a standard copy of a semantic network object, made via the copy constructor or assignment operator, and a shadow object is that the *Semantic Network Service* is informed that a shadow is being created and can therefore enforce the rule that only one shadow object exists at any particular time for a given semantic network object. Objects must be checked-out from the semantic network before they can be modified. Checking out an object will create a shadow object that can then be modified. Following modification, shadow objects must be formally checked-in to effectively commit the changes.

**Subscription Service:** The *Semantic Network Service* engages another service that is present within the Domain Tier, namely the *Subscription Service*. This service provides a robust mechanism permitting semantic network clients to register interests in events occurring within the semantic network. The *Subscription Service* allows subscribers to be notified of events including object creation (i.e., objects being added to the semantic network), object destruction, and object modification. Employing a call-back design pattern, the subscriber provides a method or function to be called when the particular event of interest occurs. When the *Subscription Service* determines that the specific event has occurred, appropriate notification is sent to all interested parties. As an added means of efficiency, such notification may include state information regarding the event, thus obviating the need for the receiver to issue a set of follow-on queries to examine the details of the event.

**Transaction Service:** The Domain Tier also houses a *Transaction Service* that maintains the overall integrity and consistency of the ICODES information environment. At the heart of this service is the notion of a *transaction*, as taken from the world of database systems. Any change to the semantic network occurs within the context of a transaction, although that transaction is sometimes provided implicitly. Multiple transactions may be
active simultaneously, although no object may take part in more than one transaction at any given time.

As with database transactions, no change to the system is recorded permanently until the transaction is complete. Before the transaction is complete, portions of the system that are not involved in the transaction are unaware of any changes to the objects taking part in the transaction. It is possible for an object to be removed from a transaction before the transaction has been completed, in which case the object returns to its former state. Since objects are also added to and removed from the semantic network in the context of a transaction, removing an object from a transaction can sometimes mean that the restoration of an object to its former state causes it to disappear from, or reappear within, the semantic network (or rather, the view of the Semantic Network as seen from the perspective of the transaction).

**Persistence Service:** Upon completion of a transaction, the Transaction Service employs the Persistence Service, which is responsible for reflecting the affects of this transaction within the persistent form of the semantic network. This involves the addition of new objects to storage, the updating of persistent storage to reflect changes made in modified objects, and the removal of objects that have been removed from the semantic network. Although not restricted to any particular vendor, ICODES currently uses the Microsoft SQL Desktop Engine database for its persistence needs. Further, ICODES is flexible enough to actually use more than one form of storage concurrently.

![Figure 2.13: ICODES Data Access Layer (IDAL)](image)

The primary interface offered to Persistence Service clients takes the form of the Independent Data Access Layer (IDAL). This interface is used for both saving an object’s state to a data store as well as sharing objects across processes. IDAL provides a handler
class that defines the interface for persistence of data throughout the ICODES system. Several IDAL handlers can be registered concurrently supporting multiple persistence stores to be employed at the same time. Currently, ICODES employs two distinct IDAL handlers (Figure 2.13). The first is the DAL handler used to persist data in a relational database management system (RDBMS). The second handler is the broadcast handler used to allow ICODES to communicate with external applications through the ICODES External Services Interface (IESI), described in a later section.

**Attachments:** The Semantic Network Manager is capable of attaching tagged data and/or objects to any existing object in the semantic network. The attached data benefits from the normal lifecycle management offered by the semantic network and are therefore available to any internal or external component (e.g., agent, plug-in, etc.).

The first of these two types of attachable elements, tagged data, is textual in nature and although it imparts no contextual information, has presumably a degree of meaning to the contributing component. The tagged data is given a name by the contributor requesting the housing of the attachment. Once named, any component may then retrieve the attached element by providing the name and the identifier of the semantic network object to which the element is attached.

However, far more powerful than attachment of meaningless text is the ability to attach a contextual object. Since an attached object becomes dependent on the object it is attached to (i.e., is owned by), the Semantic Network Manager will automatically remove the attached object if the object it is attached to is removed from the semantic network. This allows a component to create objects in the semantic network that are to be integrated with a load-plan without being concerned about cleaning up objects that are no longer relevant. This ability to add external, non-native types of content to the semantic network is yet another mechanism empowering ICODES with the ability to support the evolving needs of the user with minimal to no impact on the existing code base.

**2.4.2 Logic Tier**

Originally, ICODES was designed to execute on a UNIX operating system platform. However, as the Microsoft Windows operating system became both more capable and more popular the decision was made to port the entire ICODES system to a Windows platform. This fairly significant venture also opened the door to refactor the original ICODES architecture in favor of the three-tier design described in this document. At the time it was felt that while a substantial effort, this more current and flexible design would provide significant opportunities to be ported to additional platforms if this need should arise in the future, as well as offering increased deployment options in terms of distribution across a network of computing resources.

The Logic Tier portion of the ICODES three-tier architecture not only houses the standard application-level logic typically found in such a tier but also contains the community of decision-support agents that provide the analytical depth empowering the ICODES space planning environment.

The agents described in Section 1.4 collaborate in both a direct and indirect fashion to assist in the development of efficient and correct stow plans. A brief discussion of how this community of agents works together to formulate violation-free load-plans in both the assisted stow and the manual stow modes, follows.
Assisted Stow: The main function of the agent community operating within the Assisted Stow mode is to automatically find a valid stow location (i.e., free of any issues relating to accessibility, hazardous materials, trim and stability, etc.) for cargo items within some space. This is achieved through a round-robin style of agent collaboration.

Once configured with user preferences and restrictions, the Assisted Stow capability takes advantage of the Cargo Agent by filtering out cargo items that are not deemed valid for stow. In most cases cargo items are filtered out because of missing information relating to dimension or weight. The next step is to locate an empty space on the vessel where the cargo item can be placed without overlapping other cargo items. However, the search for an empty space is constrained by factors such as the weight of the cargo item. For example, the heaviest items in the cargo list should be placed near the bottom of the ship for reasons of overall stability.

This candidate location is then presented for evaluation to both the Access Agent and the Hazardous Material Agent. In accordance with their particular domain expertise, if either agent finds an issue with the candidate placement the Assisted Stow capability nominates an alternative location and the evaluation repeats. This process is repeated until a violation-free location is found or the expanse of possible stow locations is exhausted. In either case, the Assisted Stow capability continues onto the next cargo item to be stowed.

Manual Stow: In contrast to the collaborative assessment model applied in Assisted Stow, the Manual Stow mode of operation instructs the agents to function independently of one another. In other words, as the user places, or templates, a cargo item within a stow space each agent reacts concurrently identifying any outstanding issues incurred by such placement in accordance with their individual domain of expertise. Any such violations or warning are presented to the user in the form of agent reports comprised of a concise depiction of the issue along with any possible resolutions the agent may be able to offer. However, regardless of the severity of the issue, the user makes the final determination of how, and even if, the issue is to be resolved.

2.4.3 Presentation Tier

Like most industry-standard applications, ICODES offers a graphical user-interface (GUI) as the primary means of exchanging information with the user. Because ICODES is designed to run on a Windows operating system platform its user-interface adheres to the standard Windows logo compliance design pattern. ICODES offers its presentation capabilities in two forms, standalone and web-based. The functional capabilities of these user-interface alternatives are described in Sections 1.4, 1.5, 1.6, 1.7, and 1.8, and will therefore not be repeated here. However, some of the technical aspects of both the thick-client stand-alone user-interface and the web-based Viewer warrant further explanation.

The design of the ICODES standalone GUI is comprised of several internally developed as well as off-the-shelf components. Each of these components operates in unison to provide a robust means for interacting with the user. Following is a brief discussion of several of these components.

Vessel Display: The vessel display portion of the ICODES stand-alone user-interface is based on the Open Inventor low level graphics library (Wernecke 1994). Open Inventor is a commercial object-oriented three-dimensional (3D) graphical toolkit built on top of
OpenGL (Shreiner 2000). This toolkit uses a programming model based on a 3D Scene database that dramatically simplifies graphics programming. Open Inventor offers the software developer a rich set of objects including cubes, polygons, text, materials, cameras, lights, trackballs, handle boxes, 3D viewers, and editors. Together these elements allow for the development of robust, interactive graphical software applications.

Using the power of Open Inventor, ICODES offers a Vessel Display capability that provides an interactive rendering of the load-planning workspace. More specifically, the Vessel Display component provides a graphical representation of vessel and cargo information housed within the ICODES Domain Tier. Using this capability the user is able to both visualize and manipulate the deck plan of the vessel as well as the location of cargo stowed within the vessel.

**G2D Viewer:** The Graphics 2D (G2D) Viewer is a cross-platform, C++ implementation of a set of two-dimensional graphics tools. G2D displays its graphical elements in a layered manner. The viewer tool also supports drag-and-drop operations as well as the drawing of graphic primitives, including multi-polygons and Bezier curves. The G2D Viewer also supports the vector and matrix mathematics functionality that is required for the use of transforms.

The fundamental G2D viewer architecture essentially consists of three components: the G2D surface that forms the conceptual container in which all displayed information resides; multiple graphical layers displaying the graphical elements comprising the scene; and, the user-interface layer where items are drawn during drag or drawing operations (Figure 2.14). ICODES employs the G2D Viewer to provide the container bay, side-profile, and trim and stability views of the vessel.

![Diagram of G2D Viewer architecture](image)

**Figure 2.14:** Fundamental G2D Viewer architecture (UI layer not shown)

---

22 Developed by Pierre Bézier in the 1970s for CAD/CAM operations, a Bezier curve is a cubic equation that can be used in computer graphics for the construction of non-linear shapes.
Web-Based User-Interface (Thin-Client): ICODES load-plans are described as documents that represent all of the information available to the ICODES application during the preparation of a load-plan. When a load-planner uploads a load-plan to the ICODES File Share, the ICODES Thin Client uses standard XML parsers along with other tools to extract information from the file and place it in a database, thereby making that information available for later use in the user-interface.

The Thin Client uses a standard SAX parser to process the load-plan file. Instead of building a tree representation of an entire XML file in memory as a DOM parser would do, a SAX parser identifies the individual parts of an XML document as it reads the file and immediately passes those parts to an object that implements the org.xml.sax.ContentHandler interface. When the parser identifies, for example, the start of an XML element and processes its attribute list, the parser will call the ContentHandler.startElement method, passing the element’s name and universal identification (if applicable), and the list of attributes as name/value pairs. SAX parsers eliminate the need to parse the entire document before processing can begin, which is important when dealing with notably large XML documents such as ICODES load-plans.

The particular component of the Thin Client that inserts load-plan information into the database uses a number of custom classes that implement the ContentHandler interface. Each such class is responsible for processing a specific part of a load-plan, resulting in a highly focused and modular design.

The handler objects are connected in a pipeline arrangement, so that all the information from the parser flows through all of the handlers. Each handler selects the parts of the document in which it is interested, and then passes that information to the next object in the pipeline. All handlers are completely independent of each other (i.e., they are unaware of what objects might occur before or after themselves within the pipeline). This design has proven to be very flexible, resulting in the ability to add new functionality to the load-plan processor on an as-needed basis, while keeping code changes to a minimum.

Embedded in each load-plan are a number of graphics files in the SVG format. There is one such file for each ship in the load-plan and one for each unique cargo symbol (i.e., cargo items are matched to symbols based on model name, allowing many items to use copies of the same symbol). For display in the Thin Client, the cargo symbols must be added to the ship graphic files so that they appear in the location and orientation in which they occur in the original load-plan. The process of adding cargo symbols to the graphical representation of a ship is detailed in the next few paragraphs.

There are two phases involved in processing a load-plan for use in the Thin Client: extracting information for the database; and, creating the necessary graphical display files. During the first phase, two actions are taken that relate to preparing the graphical display. First, ship and cargo graphics are read from the load-plan and saved as individual files. Second, stowed cargo information is read from the load-plan and stored internally in lists indexed by ship names.

---

23 SAX is an acronym for Simple API for XML (see: [http://www.saxproject.org/](http://www.saxproject.org/)).

24 DOM is an acronym for Document Object Model (see: [http://www.w3.org/DOM/](http://www.w3.org/DOM/)).

25 SVG is an acronym for Scalable Vector Graphics (see: [http://www.w3.org/Graphics/SVG/](http://www.w3.org/Graphics/SVG/)).
Once the load-plan file has been completely processed, the second phase combines ship graphics and cargo symbols, arranging for the cargo symbols to appear in the correct location on the ship. During the second phase, each ship graphics file is opened and read using a SAX parser. The various elements of the file are passed through an intermediate handler, whose purpose is described below, to a final handler that is responsible for writing the complete graphics definition to another file.

As the parser passes elements to the intermediate handler, the handler examines each one, looking for two kinds of specific elements. First, the intermediate handler looks for a type of SVG element that can serve as the parent for elements that define named shapes. When the intermediate handler receives this element, processing of the ship graphics file temporarily stops. The intermediate handler then uses the stowed cargo information gathered in the first pass to determine what symbols are needed to represent the cargo on this ship. For each such symbol, the handler creates another parser that reads the symbol file and passes the contents to the writing handler, effectively inserting the contents of the symbol file into the output graphics file. Once all of the cargo symbols have been inserted, processing of the ship graphics file continues.

The intermediate handler also looks for the elements that represent stow areas on-board the ship. When one is found, the handler finds the list of cargo items that are stowed in that stow area. For each one, the handler creates elements that refer to the cargo symbols inserted earlier, along with other elements that describe the position at which the symbol should be displayed. These elements are then written to the output file.

The result of this processing is a set of files, one for each ship in the load-plan. Each file describes the graphical representation of the ship with symbols for the cargo located appropriately at the position of each cargo item on the ship. At this point, the cargo information from the load-plan is available in the Thin Client database, and the graphical representation of ships and cargo are saved as files in SVG format. The load-plan is now ready to be displayed to the user. The technical aspects of the user-interface are discussed in the next section.

**Technical User-Interface Considerations:** The ICODES web application is implemented using Struts\textsuperscript{26}, which uses the Model-View-Controller (MVC) design pattern. This pattern separates an application into three primary areas of responsibility: every object in the application is either part of the Model; a View object; or, the Controller.

For web applications, the Model is a group of interrelated classes that together represent the business logic of the application. In most cases, the model objects interact with back-end processes such as databases and other servers to retrieve information. As a result, it could be suggested that there are two models involved in a web application: the model of the application; and, the model of the domain of the application, which is composed of information received from the user and retrieved from a database or other sources. In this section, the application model will be referred to as the Model, since this is what is meant by the Model-View-Controller pattern, while the model of the domain will be referred to as the domain model.

\textsuperscript{26} Struts is an open source framework for building Servlet/JSP (JavaServer Pages) based web applications based on the Model-View-Controller (MVC) design paradigm (see: (http://struts.apache.org/ and http://java.sun.com/blueprints/patterns/MVC.html ).
In contrast to *Model* objects, Views are classes or other artifacts whose responsibility is to present information to a user and to enable user interaction with the application. To reduce complexity, Views are generally completely decoupled from the *Model*. In other words, a View does not know where its information comes from or how to modify that information, but only how to present that information in a specific manner.

The connection between *Model* objects and Views is the Controller. In a web application, the Controller receives input from remote clients (generally web browsers). Based on the input and the current state of the application, the Controller passes the input to instances of classes that are part of the *Model*. These classes process the input, either by retrieving information from, or by making changes to, the *domain model*. The results of this processing are returned to the Controller.

Combining the results returned by the *Model* objects with information about the flow of control between Views, the Controller locates the object that will provide the next page of the application. The Controller then passes the *Model* output to the next View object, which is responsible for constructing a web page that includes the *Model* output. The web page is then returned to the user’s browser for display.

The Struts web application development framework specializes this pattern in a number of ways that tend to simplify the creation of complex applications. Struts provides a Controller object that serves as infrastructure for managing the navigation of a web application. It is also possible to extend the base Struts controller, as the ICODES Thin-Client does, to provide application-specific functionality.

The flow of control in a Struts application is defined by a number of configuration files. These files describe the actions that the Controller should take given input from a particular View (i.e., from the web page created by that View). These action descriptions include the class of the *Model* object that will receive the input, and the class of the View that will process the output of the *Model* for that input. All the connections between Controller, *Model*, and View are externalized to the configuration files, resulting in a highly modular, decoupled architecture.

In Struts, the application *Model* classes are known as Action classes, since each of them defines one or more actions that the application can perform. For example, in the Thin-Client there is an Action for finding a load-plan based on criteria selected by the user. This Action class uses a domain model object to mediate between the Action and the database, confining database access code to a small number of classes that have a limited purpose. At the end of each Action method, there is a *forwarding* operation to a specific View, but the View is identified only by name, and not by class. This allows, indeed requires, a configuration file to declare a mapping between a view name and a View class. Again, the purpose of this approach is to decouple the *Model* classes from the Views that define the display of information.

Specifically, in the Thin-Client the Action class for the load-plan search ends with a forwarding to a View named *loadoutPlanSearchList*. This View is defined in a configuration file as follows:

```xml
<forward name="loadoutPlanSearchList"
         path="/viewLoadoutPlanSearchList.icw" />
```

This declares that the view named *loadoutPlanSearchList* will be found at the Uniform Resource Locator (URL) /viewLoadoutPlanSearchList.icw. The mapping from URL to a
specific View class is found in another configuration file, and is of the following elided form:

```xml
<page id="viewLoadoutPlanSearchList">
<uri>viewLoadoutPlanSearchList.icw</uri>
<bodyPage>loadoutPlanSearchList.jsp</bodyPage>
<title>Loadout Plan Search List</title>
<actionType>com.cdmtech.icodesweb.struts.action.IcodesBaseAction</actionType>
<className>com.cdmtech.icodesweb.struts.IcodesActionMapping</className>
</page>
```

This declaration maps the Universal Resource Identifier (URI) to a specific JavaServer Pages (JSP) (http://java.sun.com/products/jsp/) file that creates a web page to display the information provided by the Model classes.

The multiple levels of indirection in this process produce a highly decoupled system with the ability to be extended and modified without extensive changes to existing code. Directing the output of the load-plan search to another page, for instance, would be as simple as writing the new page and changing the value of the `<bodyPage>` element in the above XML file. After restarting the web application, the page flow would be altered without editing the code of either the Controller or any of the Model classes. The flexibility of a Struts application allows the developers of the ICODES Thin-Client to respond quickly to requests for new and modified functionality.

### 2.5 The Generic Space Generator (GSG)

The Generic Space Generator (GSG) is part of the ICDM development framework and is provided to facilitate the rapid development of graphical user-interfaces. With its ability to harness the most recently released Open Graphics Library (OpenGL) functionality from within Java code, GSG was designed to optimize performance for both two-dimensional and three-dimensional graphics requirements (Wright and Sweet 2000). The components of GSG are extensible and reusable to promote the rapid prototyping of applications that incorporate similar concepts. It was used for the development of the JINNI module of the ICODES suite of tools, described in Section 1.9.

GSG utilizes a custom Dynamic Link Library (DLL) in combination with Java Native Interface (JNI) technology to communicate directly with the available video card. Java JNI technology supports direct communication between Java and C++ as long as the necessary communications bridge is contained within the DLL. As shown in Figure 2.15, GSG currently invokes most JNI calls through the `NativeCanvas` class.

At the time of development, the most recent releases of Java included the ability to pass arrays to C++ in the form of `ByteBuffer` that contain a pointer to the array, as well as information such as length. The ability to pass arrays efficiently is of consequence since many graphical functions require large arrays of vertex and line information to be passed to the video card. It is therefore planned that future releases of GSG will use the now available Java bindings for OpenGL (JOGL) to further increase efficiency and performance.
Figure 2.15: GSG Java Native Interface (JNI) invocation

Figure 2.16: GSG component architecture
The principal components of the GSG architecture together with their relationships are depicted in the system architecture diagram shown in Figure 2.16. Following are brief descriptions of each of the major components comprising this architecture.

- **GUI Components**: Reusable components that are responsible for the primary GSG display capabilities.

- **Window Manager**: Main window or frame that is constantly visible. The Window Manager is responsible for the layout and the docking of inner windows. Every GSG application has a single Window Manager.

- **Viewer**: Displays the graphical entities housed within the GSG Domain Model.

- **Entities**: Objects that adhere to the GUI Java Bean programming model providing functions to get and manipulate properties as well as notification of property changes through `PropertyChangeEvent`s.

- **Domain View(s)**: Usually in a tree layout, provides visual representation of the objects in the GSG Domain Model. Objects can usually be modified or edited from this view.

- **Domain Model**: Contains all graphical and non-graphical entities the GSG is configured to manage. Provides non-fiction if an entity is added or removed. Components such as the Viewer listen for changes to the GSG Domain Model in order to determine the existence of Java Bean-based entities.

- **Domain**: Describes a set of entities and reusable components. Captures a log of changes that allows for rollback capabilities.

- **Domain XML**: An XML description of a Domain as well as the layout and menus of the various GUI components.

- **Domain Loader**: Transforms the Domain XML into a Domain object. Initializes any Viewer(s) as well as the Window Manager.

- **Local Controls**: Both notify operators of potential problems, provide assistance within the graphics domain, and register for notification of property changes on entities or components of interest.

- **Graphical**: Entities that know how to draw themselves in the Viewer. Properties of these entities can be viewed and edited.

- **Logical**: Entities that cannot be displayed in the Viewer. Properties of these entities can be viewed and edited.

- **Interactions**: User mouse interaction handlers for the Viewer. Reacts to events such as movement, placing and modification of objects in the Viewer.

- **Actions**: Reusable units of code that perform a service reacting to certain events (e.g., user interactions, etc.).
2.6 Reference Libraries

The ICODES reference libraries provide access to existing reference data sets that catalog the static characteristics of identifiable data entities (i.e., encyclopedic data). These include data sets to characterize: standardized pieces of Department of Defense (DoD) equipment and supplies, hazardous material, explosives, and munitions. ICODES provides access to the reference libraries to allow:

- Users to browse, query, and print the data contained in a library from within ICODES.
- Users or agents to import data into the Cargo Semantic Network from a standardized reference source.
- Agents to validate data entered by the user, or imported from an external system, by comparison to the standardized data contained within the libraries.

For performance reasons the data contained in the libraries are not used directly by the Cargo Semantic Network. Instead, the libraries are used as sources of data that may be imported into the Cargo Semantic Network by the user or by agents. This allows large sets of reference data to be stored in relatively slow, but memory efficient, disk-based data structures, while the object representation resides entirely in memory for relatively fast access by users and agents alike.

The ICODES reference libraries are constructed from data sets made available by external organizations that have custodianship responsibilities. The construction process makes no modifications to the data values provided, with the exception of values that do not pass the validation criteria specified for the corresponding ICODES library attribute. Invalid values are supplied with an ICODES defined default value (essentially a null) of the appropriate type. The validation criteria used by ICODES depend on the attribute in use. This can range from a simple enumeration of all valid values to a regular expression describing the valid characters permitted in each position in the sequence of characters that make up the value. Attribute values that are considered incorrect by the users of ICODES, will not be modified during the construction process, nor will the latter add data records considered to be missing. These types of modifications must be made by the agency responsible for the data set, and will not be reflected in the ICODES reference libraries until after the source data used to construct the corresponding ICODES library have been updated. Efforts are under consideration to provide updates to the reference libraries on a more frequent basis than releases of ICODES.

The following reference libraries are currently supported by ICODES:

49 CFR Library: Provides access to hazardous material reference data that have been cataloged by both the United Nations and the North American hazardous type identifiers. Since the source data are intended to present information to human users through a browser, they are not ideal for automated access or processing by computers. Therefore tools have been developed to convert the data into a format that is compatible with the ICODES toolset. It includes the following data elements (Table 2.1):

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UN Code</td>
<td>string 6</td>
<td>UN or NA hazardous type identifier</td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Packing Group</td>
<td>string 4</td>
<td>Hazardous degree of danger</td>
</tr>
<tr>
<td>Class</td>
<td>string</td>
<td>Hazardous classification code</td>
</tr>
<tr>
<td>Division</td>
<td>string</td>
<td>Hazardous division</td>
</tr>
<tr>
<td>IMO Code</td>
<td>string</td>
<td>International Maritime danger code, as derived from the Class, Division, and Compatibility Group attributes.</td>
</tr>
<tr>
<td>Label</td>
<td>string</td>
<td>Hazardous label</td>
</tr>
<tr>
<td>Vessel Stowage Code</td>
<td>character</td>
<td>Indicates permissible stowage locations</td>
</tr>
<tr>
<td>Compatibility Group</td>
<td>character</td>
<td>Hazardous compatibility group</td>
</tr>
<tr>
<td>Proper Shipping Name</td>
<td>string</td>
<td>Hazardous proper shipping name</td>
</tr>
<tr>
<td>Other Provisions</td>
<td>string</td>
<td>Hazardous stowage requirements</td>
</tr>
</tbody>
</table>

**DCMNSN Library:** The DCMNSN Library provides access to reference data for mapping between hazardous material information and National Stock Numbers (NSN). It includes the following data elements (Table 2.2):

Table 2.2: Reference Library DCMNSN Data Dictionary

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSN</td>
<td>string 13</td>
<td>The National Stock Number</td>
</tr>
<tr>
<td>Proper Shipping Name</td>
<td>string</td>
<td>Hazardous proper shipping name</td>
</tr>
<tr>
<td>DoDIC</td>
<td>string</td>
<td>DoD munitions identifier</td>
</tr>
<tr>
<td>Hazard Class</td>
<td>string</td>
<td>unknown</td>
</tr>
<tr>
<td>IMO Code</td>
<td>string</td>
<td>International Maritime danger code, as derived from the Class, Division, and Compatibility Group attributes.</td>
</tr>
<tr>
<td>Class</td>
<td>string</td>
<td>Hazardous material class</td>
</tr>
<tr>
<td>Division</td>
<td>string</td>
<td>Hazardous material division</td>
</tr>
<tr>
<td>Compat Grp</td>
<td>character</td>
<td>Hazard compatibility group</td>
</tr>
<tr>
<td>UN Code</td>
<td>string 6</td>
<td>UN or NA hazardous type identifier</td>
</tr>
<tr>
<td>Pack Group</td>
<td>string</td>
<td>Hazardous degree of danger</td>
</tr>
<tr>
<td>Label</td>
<td>string</td>
<td>Hazardous label</td>
</tr>
<tr>
<td>NEW/Round</td>
<td>string</td>
<td>The net explosive weight, per unit of issue, in pounds</td>
</tr>
</tbody>
</table>
**DoDIC Library:** The DoDIC Library provides access to reference data for explosives and munitions that have been cataloged by the DoD Identification Code (DoDIC). It includes the following data elements (Table 2.3):

Table 2.3: Reference Library DoDIC Data Dictionary

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DoDIC</td>
<td>string 4</td>
<td>DoD munitions identifier</td>
</tr>
<tr>
<td>UN Code</td>
<td>string 6</td>
<td>UN or NA hazardous type identifier</td>
</tr>
<tr>
<td>Class</td>
<td>character</td>
<td>Hazardous material class</td>
</tr>
<tr>
<td>Division</td>
<td>character</td>
<td>Hazardous material division</td>
</tr>
<tr>
<td>Compat Grp</td>
<td>character</td>
<td>Hazard compatibility group</td>
</tr>
<tr>
<td>NEW/Round</td>
<td>float 8</td>
<td>The net explosive weight, per unit of issue, in pounds</td>
</tr>
<tr>
<td>Nomenclature</td>
<td>string</td>
<td>The English text description corresponding to a particular DoDIC</td>
</tr>
<tr>
<td>Ammunitio n Group</td>
<td>string</td>
<td>Type of ammunition (fireworks or substance)</td>
</tr>
<tr>
<td>IMO Code</td>
<td>string</td>
<td>International Maritime danger code, as derived from the Class, Division, and Compatibility Group attributes.</td>
</tr>
</tbody>
</table>

**ECF Library:** The ECF Library provides access to reference data for military vehicles and equipment that have been cataloged by Model Number and Line Item Number (LIN). It includes the following data elements (Table 2.4):

Table 2.4: Reference Library ECF Data Dictionary

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Number</td>
<td>string 12</td>
<td>A standardized identifier for DoD equipment</td>
</tr>
<tr>
<td>NSN</td>
<td>string 13</td>
<td>The National Stock Number</td>
</tr>
<tr>
<td>LIN</td>
<td>string 6</td>
<td>ECF Line Item Number; an ECF specific key used to locate the data records for a particular model</td>
</tr>
<tr>
<td>LIN Index</td>
<td>string 2</td>
<td>ECF Line Item Index; an ECF specific key used to locate a specific data record from the set of data records identified by the LIN</td>
</tr>
<tr>
<td>Type Eqpt</td>
<td>character</td>
<td>A standardized coded identifier used to categorize equipment by type</td>
</tr>
<tr>
<td>Length</td>
<td>unit 8 (distance)</td>
<td>The bare (unloaded) length in inches</td>
</tr>
<tr>
<td>Height</td>
<td>unit 8 (distance)</td>
<td>The bare (unloaded) height in inches</td>
</tr>
<tr>
<td>Width</td>
<td>unit 8</td>
<td>The bare (unloaded) width in inches</td>
</tr>
</tbody>
</table>

69
**IMDG Library:** The IMDG library provides access to the reference data for the International Maritime Dangerous Goods codes that have been collected by the International Maritime Organization (IMO). UN Code normally indexes this library. It includes the following data elements (Table 2.5):

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UN Code</td>
<td>string 6</td>
<td>UN or NA hazardous type identifier</td>
</tr>
<tr>
<td>Class</td>
<td>string</td>
<td>Hazardous classification code</td>
</tr>
<tr>
<td>Division</td>
<td>string</td>
<td>Hazardous division</td>
</tr>
<tr>
<td>Proper Shipping Name</td>
<td>string</td>
<td>Hazardous proper shipping name</td>
</tr>
<tr>
<td>Flashpoint</td>
<td>string</td>
<td>Range of flashpoints</td>
</tr>
<tr>
<td>Pack Group</td>
<td>string</td>
<td>Hazardous degree of danger</td>
</tr>
<tr>
<td>Stowage Codes</td>
<td>string</td>
<td>Codes representing English stowage instructions. These codes are the keys in the IMDG Stow Codes Library.</td>
</tr>
<tr>
<td>Compat Grp</td>
<td>character</td>
<td>Hazard compatibility group</td>
</tr>
<tr>
<td>Marine Pollutant</td>
<td>string</td>
<td>Indication of whether or not this material is a marine pollutant</td>
</tr>
<tr>
<td>Vessel Stowage</td>
<td>string 2</td>
<td>Indicates permissible stowage locations</td>
</tr>
<tr>
<td>IMO Code</td>
<td>string</td>
<td>International Maritime danger code, as derived from the Class, Division, and Compatibility Group attributes.</td>
</tr>
</tbody>
</table>

**MECF Library:** The Marine Equipment Characteristics Library (MECF) provides access to reference data for US Marine Corps vehicles and equipment that have been cataloged by Model Number and Table of Authorized Material Control Number (TAMCN). It includes the following data elements (Table 2.6):
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item Id</td>
<td>string 6</td>
<td>Table of Authorized Material and Equipment Control Number (TAMCN); a standardized USMC specific key used to identify a particular model</td>
</tr>
<tr>
<td>TAM Index</td>
<td>string 2</td>
<td>MECF Line Item Index; an MECF specific key used to locate a specific data record from the set of data records identified by the Item Id</td>
</tr>
<tr>
<td>Supply Class</td>
<td>character</td>
<td>JCS Supply Class Code; a standardized code which identifies a particular category for supply</td>
</tr>
<tr>
<td>Description</td>
<td>string 50</td>
<td>The English text description which corresponds to a particular model</td>
</tr>
<tr>
<td>Logical Set</td>
<td>character</td>
<td>Identifies if an entry is part of a logical set</td>
</tr>
<tr>
<td>NSN</td>
<td>string 13</td>
<td>The National Stock Number; a unique identifier to a particular type of DoD supply item</td>
</tr>
<tr>
<td>Shp Config</td>
<td>character</td>
<td>A coded MECF identifier for a particular shipping configuration</td>
</tr>
<tr>
<td>Type Eqpt</td>
<td>character</td>
<td>A standardized coded identifier used to categorize equipment by type</td>
</tr>
<tr>
<td>Length</td>
<td>unit 8 (distance)</td>
<td>The bare (unloaded) length in inches</td>
</tr>
<tr>
<td>Width</td>
<td>unit 8 (distance)</td>
<td>The bare (unloaded) width in inches</td>
</tr>
<tr>
<td>Height</td>
<td>unit 8 (distance)</td>
<td>The bare (unloaded) height in inches</td>
</tr>
<tr>
<td>Weight</td>
<td>unit 8 (weight)</td>
<td>The bare (unloaded) weight in inches</td>
</tr>
<tr>
<td>JCS Cargo Category</td>
<td>string 3</td>
<td>The JCS Cargo Category Code</td>
</tr>
<tr>
<td>Model Number</td>
<td>string 14</td>
<td>A standardized identifier for DoD equipment</td>
</tr>
<tr>
<td>Load Length</td>
<td>unit 8 (distance)</td>
<td>Length of the cargo bed in inches</td>
</tr>
<tr>
<td>Load Width</td>
<td>unit 8 (distance)</td>
<td>Width of the cargo bed in inches</td>
</tr>
<tr>
<td>Load Height</td>
<td>unit 8 (distance)</td>
<td>Height of the cargo bed in inches</td>
</tr>
<tr>
<td>Load Weight</td>
<td>unit 8 (weight)</td>
<td>Maximum weight of cargo which may be loaded in pounds</td>
</tr>
</tbody>
</table>

**Tech Data Library:** The Tech Data Library provides access to reference data that have been cataloged by the National Stock Number (NSN) for use by the LOGAIS family of systems and TCAIMS-II. It includes the following data elements (Table 2.7):
Table 2.7: Reference Library *Tech Data* Data Dictionary

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSN</td>
<td>string 13</td>
<td>National Stock Number; a unique identifier to a particular type of DoD supply item</td>
</tr>
<tr>
<td>NSN Configuration</td>
<td>string 15</td>
<td>English text which describes a particular shipping configuration, and corresponds to a particular Configuration Code</td>
</tr>
<tr>
<td>DoDIC</td>
<td>string</td>
<td>The DoD Identification Code for munitions and explosives</td>
</tr>
<tr>
<td>Nomenclature</td>
<td>string</td>
<td>The English text description corresponding to a particular DoDIC.</td>
</tr>
<tr>
<td>Weight</td>
<td>unit 8 (weight)</td>
<td>The bare (unloaded) weight in pounds</td>
</tr>
<tr>
<td>Length</td>
<td>unit 8 (distance)</td>
<td>The bare (unloaded) length in inches</td>
</tr>
<tr>
<td>Width</td>
<td>unit 8 (distance)</td>
<td>The bare (unloaded) width in inches</td>
</tr>
<tr>
<td>Height</td>
<td>unit 8 (distance)</td>
<td>The bare (unloaded) height in inches</td>
</tr>
<tr>
<td>Load Weight</td>
<td>unit 8 (weight)</td>
<td>Maximum weight of cargo which may be loaded</td>
</tr>
<tr>
<td>Supply Class</td>
<td>character</td>
<td>JCS Supply Class Code; a standardized code which identifies a particular category for supply</td>
</tr>
<tr>
<td>Unit Of Issue</td>
<td>string 2</td>
<td>A code which indicates the unit of issue</td>
</tr>
<tr>
<td>JCS Cargo Category</td>
<td>string 3</td>
<td>The JCS Cargo Category Code</td>
</tr>
<tr>
<td>Quantity Per Cargo</td>
<td>string 5</td>
<td>The number of separate components from which an item is comprised</td>
</tr>
<tr>
<td>Model Number</td>
<td>string 14</td>
<td>A standardized identifier for DoD equipment</td>
</tr>
<tr>
<td>UPTT Code</td>
<td>string 2</td>
<td>A code which characterizes an item within the Unit Personnel and Tonnage Table.</td>
</tr>
<tr>
<td>Trn Radius</td>
<td>integer 4</td>
<td>The minimum turning radius of a vehicle in inches</td>
</tr>
<tr>
<td>UN Code</td>
<td>string</td>
<td>UN hazardous type identifier</td>
</tr>
<tr>
<td>NEW/Round</td>
<td>unit 8 (weight)</td>
<td>The net explosive weight of a munitions, per unit of issue, in pounds</td>
</tr>
<tr>
<td>IMO Code</td>
<td>string 4</td>
<td>International Maritime danger code</td>
</tr>
<tr>
<td>COMM</td>
<td>string 3</td>
<td>The MILSTAMP commodity code for this item</td>
</tr>
<tr>
<td>Hdlg</td>
<td>character</td>
<td>The MILSTAMP special handling code</td>
</tr>
</tbody>
</table>
Symbol Lookup Library: The Symbol Lookup Library provides a mapping between cargo types and the symbols used to visually represent them. While the ICODES toolset displays these symbols in its graphical load-plan displays, there are currently no facilities available for viewing the textual and numerical content of this library. The attributes listed below are the attributes that would be visible if such a facility were available (Table 2.8).

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NSN</td>
<td>string</td>
<td>National Stock Number; a unique identifier to a particular type of DoD supply item</td>
</tr>
<tr>
<td>Model Number</td>
<td>string</td>
<td>Model Number attribute from the ECF Library. (Model Numbers listed in the MECF and Tech Data Libraries are not present in the Symbol Lookup Library.)</td>
</tr>
<tr>
<td>LIN</td>
<td>string</td>
<td>ECF Line Item Number; an ECF specific key used to locate the data records for a particular model in the ECF Library</td>
</tr>
<tr>
<td>Item Id</td>
<td>string</td>
<td>Table of Authorized Material and Equipment Control Number (TAMCN); a standardized USMC specific key used to identify a particular model</td>
</tr>
<tr>
<td>TAM Index</td>
<td>string</td>
<td>MECF Line Item Index; an MECF specific key used to locate a specific data record from the set of data records identified by the Item Id</td>
</tr>
</tbody>
</table>

Symbol Library: The Symbols Library is a graphical library that provides objectified 2D scale symbols of military cargo that are used in the templating of cargo during load-planning. The Symbols Library consists of 777 cargo items and is available in both AutoCAD and OpenInventor file formats (see Appendix B).

Vessel Library: The Vessel Library is a graphical library that provides objectified ship drawings and associated files for all ships commonly used in the military load-planning community. There are currently over 320 ships in the Vessel Library. CDM maintains the Vessel Library under contract to USTRANSCOM. It is accessible to authorized users over the Internet, so that military personnel can down-load the latest version of any ship for load-planning purposes. The ships are periodically surveyed by MARAD and changes transmitted to CDM Technologies for implementation.
Water Ports Reference Data Table: The latest Water Port Facility Text File in the USTRANSCOM Table Management Distribution System is ported over to the ICODES toolset on a weekly basis. The file is received via e-mail and then reformatted for automated comparison with the previous version currently resident in ICODES. Only if differences are detected between the two versions is the existing version replaced by the new version, in ICODES.

2.7 Data Import Requirements and Options

The ICODES toolset currently supports 178 attributes for each cargo item (Appendix A). However, only a subset of these attributes is typically included in the cargo lists that ICODES receives from external systems. The precise list of data elements that are exchanged with each external system is defined in the formal Interface Agreement for that particular system.

The minimum set of cargo attributes required for ICODES to be able to process a cargo list is different for Army and Marine Corps cargo lists, because the Marine Corps does not recognize TCN\(^{27}\) as a cargo attribute. Since the Marine Corps does not have an equivalent single cargo item identifier, ICODES requires a combination of three data elements (i.e., UIC, NSN, and PKG ID) to establish the uniqueness of any particular Marine Corps cargo item. For either service additional attributes are required if hazardous material considerations are to be taken into account by the load-planning agents.

**Army Basic Minimum:** TCN, UIC, Bumper Number, Container Number, Container Owner, Model Number, Description, Length, Width, Height, Weight, Type Pack, POE, and POD.

**Army Hazardous Material Minimum:** UN Code, Pack Group, IMO Code, DoDIC, Rounds, Type Explosive, and Limited Quantity.

**Army Optional Attributes:** Booking Number, Voyage Document Number, Bed Height, Consignee, LIN, LIN Index, Remarks, and "Destination" (e.g., Fort Bliss for cargo being offloaded for onward movement).

**Marine Corps Basic Minimum:** UIC, NSN, PKG ID, Serial Number, NSN Configuration, Association, Parent NSN, Parent Package ID, Parent UIC, Container Number, Item ID, Description, Length, Width, Height, Weight, Pack Type (same as Type Pack), Geoloc Code, Mission Number, Team Name, Supply Class, UPTT Code, and Unit of Issue.

**Marine Corps Hazardous Material Minimum:** UN Code, Pack Group, IMO Code, DoDIC, Ammunition Group, Rounds, Type Explosive, and Limited Quantity.

**Marine Corps Optional Attributes:** Cap Set, Embark Category, ISO Container Number, LTI Code, JCS Cargo Category, Landing Serial, Offload Geoloc, Onload Geoloc, Priority Order, RUC, SUC, SL3, Tag, and Tag Id.

Strictly speaking the sets of Basic Minimum attributes listed above for the Army and the Marine Corps are practical minimum requirements based on operational considerations. Theoretically, the absolute minimum requirements would be satisfied by an attribute set that uniquely identifies the cargo item and describes its physical weight and dimensions. For the Army this minimum

---

\(^{27}\) The 17-character Transportation Control Number (TCN) is the unique identifier for all in-transit Army cargo.
requirement is satisfied by five attributes, namely: TCN; Weight; Length; Width; and, Height. As mentioned previously, in the case of the Marine Corps three attributes are required for unique identification purposes (i.e., UIC, NSN, and PKG ID) for a total of seven attributes.

2.8 Hardware Requirements

The recommended system software and hardware requirements for the ICODES toolset include the Windows 2000 or Windows XP Professional Operating System executing on:

- Single processor CPU operating at a minimum of 2 GHz (clock speed).
- Minimum of 1 GB of RAM (2 GB is preferable).
- Graphics card with at least 32 MB RAM.
- 40 GB disk drive (with a minimum 5 GB of available free space).
- 2X CD-ROM drive.

In addition, the ICODES AIT application module is currently (2006) released with Intermec CK31 scanners mounted on PDAs.
3. ICODES as an Extensible Suite of KMES Tools

A critical requirement for the ICODES stow-planning system is the ability to grow to meet increasing needs. This evolving need can take the form of increasing the scope of the ICODES space planning toolset to include support for shipping by rail and trucks, airlift, warehousing, staging, and other domains that require space planning and consequential visibility. Beyond this, within a wider scope of domains, ICODES must also be architecturally ready to integrate additional capabilities or services, such as viewers tailored to specific operational needs, critical data feeds from external sources, and newly available capabilities such as smart tags and other emerging technologies. Integration of such capabilities into the ICODES tool suite must have little to no impact on the existing design and functionality. To accomplish this feat the ICODES development team has incorporated a rich programmatic interface based on the well-established plug-in design pattern. The following sections discuss not only this means of integrating new capabilities, but also provide descriptions of some of the plug-ins that have already been successfully integrated into the ICODES suite of tools.

3.1 The ICODES External Services Interface (IESI)

As the user-base of ICODES increased so did the number of requests to support specialized problems and application domains that were not considered in the original design of the ICODES toolset. Examples of these non-native domains include: car carriers, logistics support vessels; rail planning; and, yard management. While the original version of the ICODES Domain Model could be readily extended, it was understood that this would eventually lead to an unacceptable level of complexity. In any case, although the anticipated additional capabilities were likely to be similar in nature to the existing capabilities they were bound to require new features and logic changes significant enough to warrant separate applications.

Based on the KMES architectural principles discussed previously in Section 2.1, the solution that the ICODES development team decided upon was to create the ICODES External Services Interface (IESI). This component provides the means for ICODES to interact with an external application (e.g., plug-in, etc.) by providing access to the ICODES Domain Model, expert agents, and application logic. The information exchanged between the main application and these external applications is transmitted using wrapped XML packets. The solution allows for the rapid development of external applications that can augment the ICODES Domain Model in the necessary areas and operate within their own application logic, without affecting the primary ICODES application. The IESI facility can be broken down into three primary components, as follows:

IESI Core: The core of the IESI capability consists of a request handler together with a broadcast path that allows bi-directional communication between the main ICODES system and the various external applications. The broadcast path utilizes a uni-directional socket through which any system state changes are broadcast to the external service. Within ICODES, such state changes originate within the semantic network. The request handler, on the other hand, utilizes a bi-directional socket through which the plug-in

---

28 Ships specifically designed for the transportation of automobiles. These ships normally feature hoistable decks that can be adjusted to variable heights in sections.
application makes requests to the primary ICODES system. Any applicable responses are then delivered to the requesting external party.

IESI Connections: The IESI Connections component consists of one or more wrappers placed around standard Windows sockets. Although these wrappers are typically placed around the sockets offered by the Windows platform, any inter-process communications facility can be used. This is possible because the IESI Connections component supports the transmission of binary data and can detect new data without blocking.

IESI Service Wrappers: The Service Wrappers component performs the marshalling and un-marshallling activities required for an external application to invoke functions and methods within the ICODES environment via the request socket. These wrappers provide a sufficient description of the function or method being wrapped to support the passing of actual parameter information. Such support encompasses input-only, output-only, and input-output types of parameters, as well as the remote construction and destruction of objects that are then usable by the external application via the appropriate connection.

3.2 The ICODES Plug-In Framework (IPIF)

Building on the support provided by the IESI interoperability bridge, the ICODES Plug-In Framework (IPIF) is a Java-based harness for the rapid development of plug-in style external applications. A Plug-in in this sense is an external process or KMES that requires access to information available from within the ICODES semantic network. To support this requirement any IPIF-based plug-in communicates with the main ICODES environment through the IESI interoperability bridge.

The IPIF facility is comprised of four packages: Communications; GUI; Translator; and, Registry or REG (Figure 3.1). The communications package provides socket-based communication services for connecting and communicating with the IESI interoperability bridge. The GUI package provides a common set of graphical user-interface components including windows, dialog boxes, and reports. The translator package provides an XML-based functional language for extending the base XML schema to add functionality (e.g., conditional branching and calculated types), as well as providing various conversion tools. Finally, the REG or registry package provides tools for interacting with the main Windows Registry. Together these packages provide a robust framework for developing capable plug-ins that seamlessly interface with the main ICODES environment.

3.2.1 The Plug-In Design Pattern

The Plug-In Design Pattern is a well-established approach to extensibility that borrows significantly from the Separated Interface Design Pattern (Larman 1998) to provide a fairly decoupled method of adapting new capabilities and behavior to a standard interaction pathway. The Plug-In Design Pattern also successfully obviates issues associated with supporting multiple versions of behavior whose particular rendition cannot be determined until runtime. Further, such dynamic coupling is performed without the need for recompilation or externally managed configuration.

IESI exposes key ICODES functionality via an XML-based message/object-passing system using sockets.
As the plug-in metaphor implies, this pattern consists of two parts. The first is a standardized interface that acts as a quasi socket, which effectively exposes certain functionality to external behavior. This behavior, or plug, is the second part of the equation. By essentially plugging into the socket the behavior is effectively adapted to the originally exposed interface. All configuration needs are managed internally by the plug-in. The result is a mechanism through which additional capabilities and services can be integrated into the main ICODES system dynamically during execution.

3.3 ICODES Plug-Ins

The follow sections briefly discuss the technical aspects of several plug-in KMES modules that are currently available in the ICODES suite of tools. The functional capabilities of some of these plug-in extensions (e.g., the ICODES Observer and the I-AIT module) have been discussed previously in the first part of this report.

3.3.1 The ICODES Observer

As discussed previously in Section 1.8, the ICODES Observer plug-in component allows users to connect remotely to the main ICODES system. The Observer provides a read only, dynamic view of a running instance of ICODES, including, but not limited to, cargo, violations, and reports.

The Observer is implemented in the Java language and communicates to ICODES through the IESI interoperability bridge (Figure 3.2). It utilizes the RSA encryption methodology over secure sockets to authenticate any connection. Once connected, ICODES and the Observer plug-in are able to exchange data in the form of XML messages referred to as broadcasts. Component
services within the Observer (e.g., StowableItemListManager, CargoListManager, ShipGearManager, and DrawObjectManager) are able to utilize subscription services to subscribe to broadcasts of particular types of objects, corresponding to database tables in ICODES. The dual concepts of broadcasts and subscriptions are essential to the ability of the Observer to monitor events of interest that are reflected in changes occurring in the ICODES semantic network.

In the future, it is planned that the Observer will take advantage of the ICODES Stow Framework (ISF) facility described in Section 3.4 to synthesize functionality among other plug-ins. In this role its current capabilities will be expanded to support multiple ICODES connections, enabling the user to monitor several instances of ICODES simultaneously.

3.3.2 The T-AVB Automated Load-Planning System (TALPS)

The US Marine Corps performs aircraft maintenance using mobile repair facilities installed inside standard 20-foot containers. These container-based stations are referred to as Mobile Facilities and are deployed on two sister ships, the Wright and the Curtiss. Both of these ships belong to the T-AVB vessel class.

The TALPS plug-in was developed in direct response to requests from T-AVB operators for the extension of ICODES to support the load-planning requirements and constraints unique to deploying these operational Mobile Facilities on-board ship. These facilities must obey the entire gamut of rules associated with ship load-planning in addition to being stowed in conjunction with their individual fairly complex electric power requirements. There are three different power circuits on a T-AVB ship: 400Hz-100Amp; 60Hz-100Amp; and, 60Hz-200Amp. Different Mobile Facilities require different combinations of these electric power alternatives (e.g., 400Hz power is used to drive welders). Therefore, the load-plan must balance the deployment locality of any given Mobile Facility based on electric power requirements and availability. In addition, various dimensional requirements must be taken into account. For example, sufficient open space must be allowed at one end of a Mobile Facility to allow for the movement of large repair equipment and parts in and out of the facility.

Written completely in Java, TALPS employs the mature Java3D technology (Selman 2002) to perform the rendering required for the various graphical views of the target vessel. These include the top-down view, bay view, side profile view, and 3D perspective view. As a plug-in, TALPS communicates with ICODES using sockets administered by the IPIF CommManager (see Section 3.2) to make remote ICODES calls using the IESI interoperability bridge (see Section 3.1). Figure 3.3 illustrates the various interactions between the TALPS plug-in and the main ICODES system, notably decoupled via the IPIF interface. As shown in Figure 3.3, these interactions range from the transfer of cargo data to the storage of load-plans developed within the TALPS environment. This interaction is critical because TALPS relies on ICODES for persistence, agents, stow interactions, vessel data, and cargo data.

To illustrate, a typical TALPS session may involve interacting with ICODES in the following manner:

Upon startup TALPS retrieves the vessel information and the current cargo list from ICODES, any cargo already stowed in the plan is shown in the TALPS display. At this stage all agent messages are also synchronized. As the user manipulates cargo using TALPS the modifications are automatically communicated to ICODES, where they are
persisted and examined by the ICODES agents for potential violations or inconsistencies. Conversely, any changes made directly in ICODES will be automatically transmitted to TALPS providing a seamless level of integration. This level of interoperability is largely due to the effectiveness of the IPIF *CommManager* operating in tandem with the IESI interoperability bridge.

![Diagram](image)

**Figure 3.3:** Interactions between the TALPS plug-in and the main ICODES system

TALPS is empowered by the *Mobile Facility Agent*, which provides the necessary logic for this KMES component. The primary function of the Mobile Facility Agent is to monitor the electrical connections of the *Mobile Facilities* stowed on the ship. In the case where an operational *Mobile Facility* does not have the required electric power connections readily available, the agent will attempt to plan connections to the appropriate power source on-board the vessel. In certain circumstances this may include daisy-chaining *Mobile Facilities* together in order to accommodate multiple needs with limited resources. Once candidate connections have been established the agent will proceed to analyze the power requirements of each facility to determine if any of the available connection configurations are suitable. In cases where the required electric power feeds are not available, the agent will notify the user. In future versions of the TALPS plug-in, with the availability of operational data feeds, the Mobile Facility Agent can be readily extended to provide richer decision-support in the form of detailed plans for mitigating this problem.

### 3.3.3 The Generic Cargo Plug-In Framework

The Generic Cargo Plug-in Framework is designed to act as a simplified interface for remotely accessing and editing ICODES cargo attributes from external components. The result is a standardized mechanism allowing third party developers to easily interface with the cargo information housed within the ICODES semantic network. There are two interfacing
mechanisms available, as follows: from within the same Java virtual machine; or, remotely through a socket communication node. Specifications of the key method calls that are available to clients of the Generic Cargo Plug-in Framework are listed below.

```java
public boolean shakeHands(String username, String userPass, String serverName, String appName, String appPass, String appPerms, File userPrivKey, File appPrivKey);
```

This method will connect to the instance of ICODES running on the machine `serverName`. User name, password, and key, as well as application name, password, and key, are used by the Communications Manager to negotiate the connection, while `appPerms` refers to the application's permissions (e.g., internal, writable, extend). For example, an application that wishes to change cargo attributes will need `writable` permission to do so.

```java
public String getItems(Map<String, String> keys, ArrayList<String> fields, boolean agg);
```

This method performs a search of the ICODES cargo list based on the keys passed in. These keys are contained as name-value pairs in the `Map<String, String>` `keys`. If the flag `agg` is true then the search will return a match for any cargo item that matches any of the keys. If `agg` is false then the search will return matches for only those cargo items that match every key. An XML string containing all matching cargo items along with the attributes contained in the `keys` map, as well as all attributes contained in the `fields` list, will be returned by the method. Attributes that are included in the `fields` list are returned to the user for viewing and changing, but are not factored into the search.

```java
public String getAllItems(ArrayList<String> fields);
```

A call to `getItems` with an empty list of keys or a null value for `keys` will result in a call to this method, which returns an XML string with a list of all cargo items in the ICODES cargo list along with the attributes whose names are contained in the `fields` list. The `snid`, as always, is also returned.

```java
public void setItems(String xmlStr) throws InvalidSNIDException;
```

Submitting any changed item(s) to ICODES requires a call to `setItems` This function takes a XML string that describes the changes being made. The `snid` is used as a unique identifier to indicate to ICODES which cargo item is required to be changed. In the event that the ICODES load-plan has changed and the `snid` is no longer valid, an `InvalidSNIDException` will be returned.

3.3.4 The Deployable Automatic Measuring System (DACMS)

With the concomitant desires for increased cargo data accuracy and reduction in the personnel required to manage the movement of cargo, a number of automated systems have become commercially available. The objective of these systems is to measure the physical dimensions and weight of larger cargo items (i.e., particularly mobile cargo items such as trucks and trailers) as they move through a measuring station. The data collected is then automatically uploaded, mostly via wireless network, to an external database or other application. One such system is the Deployable Automatic Measuring System (DACMS) marketed by Intercomp Corporation (3839 County Road 116, Minneapolis, MN 55340-9342). The ICODES development team
implemented a plug-in to interface with the DACMS equipment. However, based on KMES principles the DACMS plug-in was designed in a generic manner as a module that can be easily customized to interface with any typical automatic measuring facility.

Succinctly stated the DACMS plug-in is designed to allow automatically captured dimensional data to flow into ICODES for use by agent and the operator in the generation of accurate pre-stow and as-loaded load-plans. This capability obviates the need for personnel to manually measure cargo items, record these measurements on paper and then enter the data into the computer. This is a particularly laborious and time consuming process because the marshalling yard where the cargo items are typically staged and measured is normally located some considerable distance from the office of the load-planners.

Developed completely in Java, the DACMS plug-in features a graphical user-interface and a simple interoperability bridge that monitors a measurements file that is exported by the measurement equipment. Whenever the measurements file has been updated the plug-in
scrutinizes the data alerting the user to anomalies such as dimensional values of zero and duplicate TCN numbers. The DACMS plug-in is socket-enabled by virtue of its relationship with the IPIF/IESI framework and is therefore able operate in the marshalling yard and send updates to a remotely located ICODES system via a wireless network.

Figure 3.4 presents a class-based view of the DACMS architecture. The arrows represent the directionality of the relationships and the consequential visibility between object classes. The measuring equipment typically writes out a fixed-width text document as items are measured. The plug-in continuously monitors this file for subsequent measurement data updates. When updates are detected the plug-in analyzes the dimensional data and alerts the user those values that require adjustment. Once all measurement data have been adjusted and verified, the DACMS plug-in can be connected to the main ICODES system using a TCP/IP connection. Using this connection, the ICODES user can import the adjusted list of cargo from the DACMS plug-in into ICODES having the option of either merging the data with an existing ICODES cargo list or overwriting the existing data.

### 3.3.5 ICODES Automatic Identification Technology (AIT)

The ICODES Automatic Identification Technology (I-AIT) plug-in was developed in direct response to military (i.e., Marine Corps and Army) requirements to verify that the location and identity of stowed cargo on a vessel correlates with the original load-plan information. Implementation of this capability was significantly influenced by the mandate of the US Secretary of Defense that calls for the wide-spread application of Radio Frequency Identification (RFID) tags to ensure the in-transit visibility (ITV) of military cargo by 2009.

Prior to development of the I-AIT plug-in the creation of an as-loaded plan (i.e., showing the actual location of each cargo item on-board ship) required a person to manually draw the cargo symbols on deck drawings printed by ICODES. These drawings were then used by the ICODES load-planner to create the as-loaded plan. Apart from being highly labor intensive, the manual method was also prone to inaccuracies. Therefore, with the availability of on-board wireless networks and more powerful PDAs with both barcode scanning and wireless capabilities, the priority for the development of an AIT capability was elevated. The existing I-AIT plug-in is seen as a precursor to a comprehensive radio frequency identification (RFID) capability that is planned for the near future.

Like all ICODES plug-ins, the I-AIT software connects to ICODES through the IESI interoperability bridge. Once connected, the I-AIT plug-in retrieves the user-requested vessel and cargo information from the ICODES semantic network. To provide as seamless an interface as possible, all actions occurring via the I-AIT plug-in that are executed on the PDA are immediately reflected within the main ICODES environment. The user-interface component of the I-AIT plug-in allows the operator to stow, move and edit cargo items in accordance with what is actually encountered in the field. The operator is also able to display, select and stow individual cargo items, as well as graphically hitch and stack cargo. The I-AIT plug-in also supports creation, display, and movement of customizable stow notes by the operator.

---

30 Even though the precise location of individual cargo items within each stow area is indicated graphically in ICODES load-plans, there is no guarantee that these locations will be honored by the non-military personnel who are often responsible for loading the cargo. The situation is exacerbated by the absence of any grid or other reference demarcations in the stow areas on-board most (if not all) vessels.
From the design standpoint (Figure 3.5), the graphics portion of the I-AIT software is provided by a PDA-compatible version of the G2D tool described earlier (see Section 2.4.3). The core of the plug-in consists of a `BaseMainFrm` class that can be extended to provide service-specific functionality. Each of these services provides its own `ScanMessageHandler` class that allows access to the hardware-specific barcode reading functionality. This handler-based decoupling also supports the rapid porting of the I-AIT plug-in software to other hardware platforms. To date the I-AIT plug-in capability has been verified on the Symbol 8146, TDS Recon, TDS Ranger, iPAQ, and Intermec CK-31 platforms. All of these PDAs support various versions of the Pocket PC operating system (Krell 2002).

### 3.3.6 The MDSS-II Interface Plug-In

Prior to the development of the MDSS-II plug-in, communication between the MAGTF Deployment Support System (MDSS-II) and ICODES was restricted to a character-delimited, flat file format. Using this primitive mechanism, cargo lists would need to be exported and imported to each system whenever updates were made. The Marine Corps users required a more direct, less user interaction intensive mechanism for automatically transferring cargo data changes between these two systems. The MDSS-II plug-in was developed to satisfy this interoperability requirement. The plug-in communicates with the MDSS-II application via its `ICODES Listener`, which interfaces with a shared file space used for transferring cargo information (Figure 3.6). Each time a cargo item is updated in either system, the event is exported to an XML file that is read by the other system, keeping the cargo lists in both MDSS-II and ICODES completely synchronized. The MDSS-II plug-in consists of the following principal software components:
• **IcodesCargoItem**: Stores cargo items in XML format understandable by ICODES. This class also contains logic to map from the ICODES semantic network model to the representation schema used in the MDSS-II plug-in.

![Conceptual MDSS-II integration architecture](image)

Figure 3.6: Conceptual MDSS-II integration architecture

• **IcodesCargoListAdapter**: Contains an in-house XML parser that closely resembles an Extended SAX Parser. The parser interprets the XML file sent from ICODES to the MDSS-II plug-in.

• **IcodesCargoListManager**: Configures the communication between ICODES and the MDSS-II plug-in. This component receives broadcast data updates sent via the IPIF communication channel. The IcodesCargoListManager is also responsible for organizing IcodesCargoItems and adding, removing and updating these entities as necessary.

• **MDSSIICargoItem**: Stores a cargo item in a XML format that is understandable by MDSS-II. MDSSIICargoItem also contains a link to the corresponding IcodesCargoItem. Unlike the IcodesCargoItem, the MDSSIICargoItem is relatively short-lived since there is no need to retain the original MDSSIICargoItem after translation to an IcodesCargoItem.

• **MDSSIICargoListAdapter**: Contains an in-house XML parser that closely resembles an Extended SAX Parser (i.e., similar to IcodesCargoListAdapter above). The parser interprets the XML cargo item description sent from the ICODES agent and stores it as an MDSSIICargoItem for subsequent translation to an IcodesCargoItem contained within the target cargo list.

### 3.3.7 The WPS Interface Plug-In

Since the first release of ICODES in 1997 the World-wide Port System (WPS) has been the principal source of cargo lists for ICODES. The WPS plug-in allows the ICODES user to electronically import container and breakbulk data from WPS and load it into ICODES either by
generating an .icl file, or by passing the data directly to ICODES through a socket connection. In addition to retrieving cargo information from WPS, the plug-in also provides the ability for the ICODES user to update cargo information in the original WPS cargo records. In the case of an update of the WPS data, the user can browse to an .icl file or import the data directly from ICODES, review what data will be sent within a tabular report, and then electronically post the updated data directly to the WPS machine.\(^{31}\)

![Figure 3.7: Conceptual WPS plug-in integration architecture](image)

The WPS plug-in can be started from within ICODES or as a stand-alone application. From the simple interface, the user can initiate a pull operation from WPS, view the data in a tabular report, and write out an .icl file for loading into ICODES. Alternatively, the user can utilize the IPIF communications component (see Section 3.2) to import the data directly from the plug-in via a socket connection. As shown in Figure 3.7, the WPS plug-in employs various IPIF and core Java technologies. First, it requires the IPIF communications and IESI packages to handle the socket transfers between the plug-in and ICODES. The IPIF GUI and IXML packages are used for internal data storage, XML-based save and restore functions, and the tabular cargo reports. The plug-in also makes use of a Java Data Base Connectivity (JDBC) interface for creating direct connections to the RDBMS within WPS, for purposes of retrieving and/or updating data.

At the time of the writing of this report, development is underway to significantly extend the interoperability between ICODES and WPS, by providing the means for ICODES to export the final as-loaded data to WPS. This recent change in business processes recognizes that the final as-loaded plan in ICODES represents the most accurate version of the data pertaining to this particular shipment.

\(^{31}\) WPS is planned to be combined with the existing GATES system into a single air and surface cargo transportation transaction system by 2008. At that time ICODES will need to interface with GATES.
3.4 The ICODES Stow Framework (ISF)

In a continuing effort to improve and expand the tools available to the logistics community, the ICODES toolset continues to be enhanced with added capabilities. Paramount among current efforts in this area is the development of the ICODES Stow Framework (ISF). The increased productivity and accuracy experienced using the ICODES space planning system brought with it a growing desire to apply ICODES to additional operational domains (i.e., rail planning, yard management, etc.). While the underlying service-oriented architecture of ICODES, together with the embedded KMES principles (see Section 2.1), allow such extensions to be readily implemented, it was decided to formalize the configuration of these extensions within the ICODES architecture.

![Conceptual ICODES Stow Framework (ISF) architecture](image)

Figure 3.8: Conceptual ICODES Stow Framework (ISF) architecture

With this in mind, the following set of goals was established for the ISF capability:

- Rapid development of ICODES-like tools.
- Accurate modeling of the specific types of spaces to be stowed, as opposed to modeling a single *universal* space.
- Support for collaborative planning.
- Scalability to very large problem spaces that could require the ICODES suite of tools to process more than a million objects.
The ISF design provides a basic set of semantics that can be extended to meet more specific needs. These semantics include the following generalized notions:

- A StowableItem is any object that can be placed at a location in a domain.
- A StowablePlace is any object where a StowableItem can be placed.
- An AccessEntity is any object that is part of a path used by a StowableItem to reach a stowable place (i.e. door, ramp, etc.).

Internally, these objects are represented as Hybrid Report Objects that combine IPIF schema-based object models (see Section 3.2) with the GSG bean model (see Section 2.5) in order to generate bean-compliant classes from schema definitions. As shown in Figure 3.8, conceptually, the architecture of ISF consists of three base components as follows: the BaseDomain, which provides basic domain functionality; the MainGUI, which provides basic user-interface components supporting the presentation of specific domains; and, the BaseViewer component, which provides basic graphical viewer capabilities. All three of these components extend their GSG equivalent components. The result is a generic space-planning foundation to facilitate the seamless interoperability of numerous domain-specific KMES tools.
References


Cerkez, P.S. (2002); ‘TALPS: The T-AVB Automated Load-Planning System’; *AI Magazine*, Summer (pp.77-87).


  Quickstart Guide Page 1 and Page 2
  Basic Training Manual
  JINNI User Guide
  Advanced Training Manual
  WPS Users Guide
  ICODES Viewer User Guide
  System Administration and Installation Guide
  Army Automatic Identification Technology (AIT) Guide
  USMC Automatic Identification Technology (AIT) Guide
  Observer User Guide


Newcomer E. and G. Lomow (2004); ‘Understanding SOA with Web Services’; Addison-Wesley, Menlo Park, California.


Pohl K. (2001); ‘Perspective Filters as a Means for Interoperability Among Information-Centric Decision-Support Systems’; Office of Naval Research (ONR) Workshop hosted by the Collaborative Agent Design Research Center (CADRC), Cal Poly (San Luis Obispo, CA) in Quantico, Virginia, June 5-7.


Wernecke J. (1994); ‘The Inventor Mentor: Programming Object-Oriented 3D Graphics with Open Inventor’; Addison-Wesley, Menlo Park, California.

Wooldridge M. and N. Jennings (1995); ‘Intelligent Agents: Theory and Practice’; The Knowledge Engineering Review, 10(2) (pp. 115-152).

Glossary of Acronyms

2D Two-Dimensional
3D Three-Dimensional
AALPS Automated Air Load Planning System
AIT Automatic Identification Technology
AOR Area of Responsibility
API Application Programming Interface
BEL Business Engine Layer
BRL Business Rule Layer
CAD Computer-Aided Design
CADRC Collaborative Agent Design Research Center (California Polytechnic State University (Cal Poly), San Luis Obispo, California)
CAM Computer-Aided Manufacturing
CD-ROM Compact Disk – Read Only Memory
CDM CDM Technologies, Inc., San Luis Obispo, California
CFR Code of Federal Regulations (hazardous materials)
CODES Computerized Deployment System
CONUS Continental United States
CPU Central Processing Unit (computer)
DACMS Deployable Automatic Measuring System
DCMNSN Dangerous Cargo Manifest National Stock Number (hazardous materials)
DII-COE Defense Information Infrastructure – Common Operating Environment
DLL Dynamic Link Library
DoD Department of Defense
DoDIC Department of Defense Identification Code (hazardous materials)
DOM Document Object Model
DOS Disk Operating System
E7 military rank: Chief Petty Officer (Navy); Gunnery Sergeant (Marine Corps); Sergeant First Class (Army); and, Master Sergeant (Air Force)
E9 military rank: Master Chief Petty Officer (Navy); Sergeant Major (Marine Corps and Army); and, Chief Master Sergeant (Air Force)
ECF Equipment Characteristics File
G2D Graphics 2D
GATES  Global Air Transportation Execution System
GB   Giga Bytes (billion bytes)
GHz  Giga Hertz (billion cycles per second)
GSG  Generic Space Generator
GUI  Graphical User Interface
GUIL Graphical User Interface Layer
Hz   Hertz (cycles per second)
I-AIT ICODES - Automatic Identification Technology
IBS  Integrated Booking System
ICDM Integrated Cooperative Decision Making (software development framework)
ICODES Integrated Computerized Deployment System
ID   Identification
IDAL Independent Data Access Layer
IESI ICODES External Services Interface
IMDG International Maritime Dangerous Goods (hazardous materials)
IMO International Maritime Organization
IPIF ICODES Plug-In Framework
ISF  ICODES Stow Framework
ISO International Standards Organization
ITV  In-Transit Visibility
JCS  Joint Chiefs of Staff
JDBC Java Data-Base Connectivity
JECF Joint Equipment Characteristics File
JFCT Joint Forces Collaborative Toolkit
JNI Java Native Interface
JOGL Java bindings for OpenGL
JSP  JavaServer Pages
KMES Knowledge Management Enterprise Services
L/T  Long Ton (2,240 lb)
LCG Longitudinal Center of Gravity
LIN  Line Item Number
LOGAIS Logistics Automated Information System
LOLO  Lift-On-Lift-Off
LTI   Limited Technical Inspection (code)
M/T  Measurement Ton (40 cubic feet)
MAGTF Marine Air and Ground Task Force
MARAD Maritime Administration
MDSS-II MAGTF Deployment Support System
MECF Marine Equipment Characteristics File
MEU  Marine Expeditionary Unit
MTMC Military Traffic Management Command (renamed: Surface Distribution and Deployment Command)
MVC Model-View-Controller
NSN  National Stock Number
OAL  Object Access Layer
OML  Object Management Layer
OTL  Object Transport Layer
OpenGL Open Graphics Library
PC   Personal Computer
PDA  Personal Data Assistant
PKG  Package
POD  Port of Debarkation
POE  Port of Embarkation
POW  Promoting Object Wrapper
RAM  Random Access Memory
RDBMS Relational Data-Base Management System
REG  Register
RFID Radio Frequency Identification
RORO Roll-On-Roll-Off
SAX  Simple API for XML
RSA  Rivest, Shamir, and Adelman (encryption technology)
SDA  Ship Data Access file
SDDC Surface Deployment and Distribution Command (previously: Military Traffic Management Command)
SNID Semantic Network Identifier
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOA</td>
<td>Service-Oriented Architecture</td>
</tr>
<tr>
<td>SQL</td>
<td>Standard Query Language</td>
</tr>
<tr>
<td>SVG</td>
<td>Scalable Vector Graphics</td>
</tr>
<tr>
<td>T-AVB</td>
<td>class of ship (aviation logistics support ship)</td>
</tr>
<tr>
<td>TAMCN</td>
<td>Table of Authorized Material Control Number</td>
</tr>
<tr>
<td>TALPS</td>
<td>T-AVB Automated Load-Planning System</td>
</tr>
<tr>
<td>TCP/IP</td>
<td>Transmission Control Protocol / Internet Protocol</td>
</tr>
<tr>
<td>TCAIMS-II</td>
<td>Transportation Coordinators’ Automated Information for Movement System</td>
</tr>
<tr>
<td>TCG</td>
<td>Transverse Center of Gravity</td>
</tr>
<tr>
<td>TCN</td>
<td>Transportation Control Number</td>
</tr>
<tr>
<td>TDS</td>
<td>Transaction Distribution System</td>
</tr>
<tr>
<td>TL</td>
<td>Translation Layer</td>
</tr>
<tr>
<td>UDL</td>
<td>Unit Deployment List</td>
</tr>
<tr>
<td>UI</td>
<td>User-Interface</td>
</tr>
<tr>
<td>UIC</td>
<td>Unit Identification Code</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>UN</td>
<td>United Nations</td>
</tr>
<tr>
<td>UPTT</td>
<td>Unit Personnel and Tonnage Table (code)</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>US</td>
<td>United States of America</td>
</tr>
<tr>
<td>USMC</td>
<td>United States Marine Corps</td>
</tr>
<tr>
<td>USTRANSCOM</td>
<td>United States Transportation Command</td>
</tr>
<tr>
<td>VCG</td>
<td>Vertical Center of Gravity</td>
</tr>
<tr>
<td>WIM</td>
<td>Weigh In Motion</td>
</tr>
<tr>
<td>WPS</td>
<td>World-Wide Port System</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
## Appendix A: ICODES Data Dictionary

<table>
<thead>
<tr>
<th>ICODES Name</th>
<th>ICODES Type</th>
<th>ICODES Data Domain</th>
<th>ICODES Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledged</td>
<td>Boolean (uneditable)</td>
<td>True/False</td>
<td>Indicates whether user has acknowledged a cargo item’s violation or warning in an agent report since it was generated, Yes or No.</td>
</tr>
<tr>
<td>AIT Location Code</td>
<td>String (editable)</td>
<td>Any string</td>
<td>Automated Information Technology Location code. This code is generated by LOGMARS Labels.</td>
</tr>
<tr>
<td>Ammunition Group</td>
<td>String (editable)</td>
<td>Any string</td>
<td>Code used to indicate what Ammunition Group a given DoDIC belongs to (i.e., AA, BB, CC, DD, FF, GG, Inert or any combination.</td>
</tr>
<tr>
<td>Area</td>
<td>Double</td>
<td>Lower Limit = 0, Precision = 0</td>
<td>The area of the rectangle bounding the footprint of the item.</td>
</tr>
<tr>
<td>Asset</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>The type of carrier or asset of cargo items such as a ship or helicopter.</td>
</tr>
<tr>
<td>Assisted Stow Area</td>
<td>String (fixed length,[4])</td>
<td>(ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890)</td>
<td>Used for identifying the stow area or compartment that a particular cargo item has been assigned to. Also used for importing stow area information from WPS to do an As Loaded Plan.</td>
</tr>
<tr>
<td>Assisted Stow Ship</td>
<td>String</td>
<td>Any string</td>
<td>Used for identifying the ship that a particular cargo item has been assigned to.</td>
</tr>
<tr>
<td>Assisted Stow Zone</td>
<td>String</td>
<td>Any string</td>
<td>Used for identifying the zone that a particular cargo item has been assigned to.</td>
</tr>
<tr>
<td>Assoc. Height</td>
<td>Float (editable)</td>
<td>Lower Limit = 0, Upper Limit 9999, Precision = 0</td>
<td>The adjusted height of the parent for a particular association as a result of taking the dimensions of the children.</td>
</tr>
<tr>
<td>Assoc. L/T</td>
<td>Float</td>
<td>Precision = 2</td>
<td>Long Tons calculated based on the Assoc. Weight of the cargo item.</td>
</tr>
<tr>
<td>Assoc. Length</td>
<td>Float (editable)</td>
<td>Lower Limit=0, Upper Limit 9999, Precision=0</td>
<td>The adjusted length of the parent for a particular association as a result of taking the dimensions of the children.</td>
</tr>
<tr>
<td>Assoc. M/T</td>
<td>Double</td>
<td>Precision=2</td>
<td>Measurement Tons calculated based on the Assoc. Length, Assoc. Width, and Assoc. Height of the cargo item.</td>
</tr>
<tr>
<td>Assoc. Weight</td>
<td>float (editable)</td>
<td>Lower Limit=0, Upper Limit 999999, Precision=0</td>
<td>The adjusted weight of the parent for a particular association as a result of taking on the dimensions of the children.</td>
</tr>
<tr>
<td>Assoc. Width</td>
<td>float (editable)</td>
<td>Lower Limit=0, Upper Limit 999999, Precision=0</td>
<td>The adjusted width of the parent for a particular association as a result of taking on the dimensions fo the children.</td>
</tr>
<tr>
<td>Association</td>
<td>Character (static)</td>
<td>Hitched, Inventoried, Load Onto, Mobile Loaded, Palletized Put Into, Set, Stacked, Parent, None,</td>
<td>A link made between two or more cargo items, known as the children and another cargo item, known as the parent.</td>
</tr>
<tr>
<td>Baplie Cell Number</td>
<td>String (uneditable)</td>
<td>Any String</td>
<td>Container Cell Identification as defined in the UN/EDIFACT Bayplan Message v3.1, section c517 .e3225 per the ISO-format, as provided by MARAD. This format defines container cells using a Bay/Row/Tier convention with 3 positions for the Bay/Row and 2 positions for the Tier (BBBRRRTTT). When a Bay Number is less than 3 characters leading zeroes are provided (i.e., 0340210). Where athwart ship containers are present on a vessel the BAPLIE enumeration will define the first digit of a stack (i.e., this would be 5 in a Baplie Cell Number of 5340210).</td>
</tr>
<tr>
<td>Bed Height</td>
<td>Float</td>
<td>Lower Limit=0, Upper Limit=999999, Precision=0</td>
<td>The height of the bed floor surface of a truck or trailer floor surface of a trailer.</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Booking #</td>
<td>String (editable,[4])</td>
<td>Identifies a group of items intended to be stowed on the same ship.</td>
<td></td>
</tr>
<tr>
<td>Bumper #</td>
<td>String (editable)</td>
<td>Vehicle bumper number stenciled onto vehicle bumpers. Determined locally.</td>
<td></td>
</tr>
<tr>
<td>Cap Set</td>
<td>String (uneditable)</td>
<td>No description available.</td>
<td></td>
</tr>
<tr>
<td>Cargo Type</td>
<td>Character (static)</td>
<td>An ICODES generated code based on TCN, Type Pack and JCS Cargo Category Code to determine type of cargo a given item is (i.e. break-bulk, container, vehicle)</td>
<td></td>
</tr>
<tr>
<td>CIIC</td>
<td>Character [CJPSU12345678]</td>
<td>Control item inventory code.</td>
<td></td>
</tr>
<tr>
<td>Class</td>
<td>Short</td>
<td>United Nations hazardous cargo class used in conjunction with Division.</td>
<td></td>
</tr>
<tr>
<td>COMM</td>
<td>String (editable)</td>
<td>MILSTAMP commodity code.</td>
<td></td>
</tr>
<tr>
<td>Compartement</td>
<td>String (uneditable)</td>
<td>A stowable area based on the deck/hold combination and always matches the attribute Stow Area.</td>
<td></td>
</tr>
<tr>
<td>Compat Grp</td>
<td>Character (uneditable)</td>
<td>Ammunition compatibility group code.</td>
<td></td>
</tr>
<tr>
<td>Consignee</td>
<td>String (editable, fixed length[6])</td>
<td>A Unit Identifier Code used for cargo items placed on a Preposition Ship as they do not belong to a specific unity within DoD.</td>
<td></td>
</tr>
<tr>
<td>Container #</td>
<td>String (editable)</td>
<td>MILSTAMP container number. A numbered code used to identify specific containers, utilized pallets or RORO trailers.</td>
<td></td>
</tr>
<tr>
<td>Container Owner</td>
<td>String (editable, fixed length)</td>
<td>The owner of a container regardless of the ocean carrier that moves it.</td>
<td></td>
</tr>
<tr>
<td>Coverage</td>
<td>enumerated type (editable)</td>
<td>A hazardous attribute identifying the type of coverage for a particular cargo item (i.e., Unknown, N/A, Open or Closed.</td>
<td></td>
</tr>
<tr>
<td>Cube</td>
<td>Integer (editable)</td>
<td>The number of cubic inches or cubic centimeters for a particular cargo item.</td>
<td></td>
</tr>
<tr>
<td>Date and Time Group</td>
<td>String (uneditable)</td>
<td>Indicates when an action occurred by date and time.</td>
<td></td>
</tr>
<tr>
<td>Density</td>
<td>Float</td>
<td>Density of the item, calculated by weight divided by volume.</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>String (editable)</td>
<td>Text description of a Model Number or Item ID.</td>
<td></td>
</tr>
<tr>
<td>Division</td>
<td>Short</td>
<td>United Nations hazardous division used in conjunction with Class.</td>
<td></td>
</tr>
<tr>
<td>DoDIC</td>
<td>String (editable)</td>
<td>DoD Identification Code.</td>
<td></td>
</tr>
<tr>
<td>Dot Class</td>
<td>Character (editable,1)</td>
<td>Department of Transportation Class for a particular cargo item.</td>
<td></td>
</tr>
<tr>
<td>Embark Category</td>
<td>Character (editable, 1)</td>
<td>A code used to identify the Cargo Embarkation Category for a particular item.</td>
<td></td>
</tr>
<tr>
<td>Exp Date</td>
<td>Unsigned Long</td>
<td>A date used to identify the end of the usable life for a particular cargo item.</td>
<td></td>
</tr>
<tr>
<td>Flash Point</td>
<td>String</td>
<td>The temperature where a given material can be ignited.</td>
<td></td>
</tr>
<tr>
<td>Geoloc Code</td>
<td>String (fixed length[4], editable)</td>
<td>Geographical Location Code used to indicate a location on the globe.</td>
<td></td>
</tr>
<tr>
<td>Haz Vol</td>
<td>Double</td>
<td>The volume of a specific hazardous material within the cargo item, in quarts or liters.</td>
<td></td>
</tr>
<tr>
<td>Haz Wt</td>
<td>Double</td>
<td>The weight of a specific hazardous material within the cargo item, in pounds or kilograms.</td>
<td></td>
</tr>
<tr>
<td>Hdig</td>
<td>Character (editable)</td>
<td>MILSTAMP special handling requirement.</td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>Type</td>
<td>Constraints</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------</td>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Height</td>
<td>Float</td>
<td>Lower Limit=0, Upper Limit=9999, Precision=0</td>
<td>The height of an item in inches or centimeters.</td>
</tr>
<tr>
<td>IMDG Page No.</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>The page number in the IMDG where the information for a particular UN Code can be found.</td>
</tr>
<tr>
<td>IMO Code</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>The code used to identify the International Maritime Organization Class, Division and Compatibility Group for a particular UN Code.</td>
</tr>
<tr>
<td>IMO Code List</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>A series of IMO Codes indicating each unique IMO Code inside of a multiple hazard cargo item in ICODES. This attribute is not editable and is only used for cargo items with more than one UN Code attached to them.</td>
</tr>
<tr>
<td>ISO Container #</td>
<td>String (editable)</td>
<td>Any string</td>
<td>A number given to a container build according to International Standard Organization specification.</td>
</tr>
<tr>
<td>Item #</td>
<td>String</td>
<td>Any string</td>
<td>User-defined data for grouping break-bulk cargo.</td>
</tr>
<tr>
<td>Item Count</td>
<td>Integer (editable)</td>
<td>Lower Limit=1</td>
<td>The number of accountable items represented by this item.</td>
</tr>
<tr>
<td>Item Id</td>
<td>String (editable)</td>
<td>Any string</td>
<td>A code used to identify a particular type of cargo item in the USMC in a manner similar to how the U.S. Army uses model number.</td>
</tr>
<tr>
<td>JCS Cargo Category</td>
<td>String (fixed length)[3], editable</td>
<td>Any string, 3 char</td>
<td>Joint Chiefs of Staff Cargo Category Code.</td>
</tr>
<tr>
<td>L/T</td>
<td>Float</td>
<td>Precision=2</td>
<td>The weight of an item in long tons, regardless of measurement system (1 long ton = 2240 pounds)</td>
</tr>
<tr>
<td>Label</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>Hazardous warning label(s)</td>
</tr>
<tr>
<td>Landing Serial</td>
<td>String ([4], editable)</td>
<td>Any string, &lt;= 4 char</td>
<td>A unique four digit code that is assigned to the personnel and cargo items for a particular unit for landing.</td>
</tr>
<tr>
<td>Length</td>
<td>Float</td>
<td>Lower Limit=0, Upper Limit=999999, Precision=0</td>
<td>Maximum length of the item in inches or centimeters.</td>
</tr>
<tr>
<td>Limited Qty</td>
<td>Boolean (editable)</td>
<td>True/False</td>
<td>A number of pounds or kilograms for a particular hazardous item. When the total number of pounds or kilograms is less than the Limited Qty, then that item is not considered for hazardous segregations.</td>
</tr>
<tr>
<td>LIN</td>
<td>String (editable)</td>
<td>(AF</td>
<td>CB</td>
</tr>
<tr>
<td>LIN Index</td>
<td>String (editable)</td>
<td>Any string</td>
<td>Used in conjunction with LIN as a unique key for identifying an entry in the model library.</td>
</tr>
<tr>
<td>Load Height</td>
<td>Float</td>
<td>Lower Limit=0, Upper Limit=999999, Precision=0</td>
<td>The maximum loading height of the equipment in inches or centimeters.</td>
</tr>
<tr>
<td>Load Length</td>
<td>Float</td>
<td>Lower Limit=0, Upper Limit=999999, Precision=0</td>
<td>The maximum loading length of the equipment in inches or centimeters.</td>
</tr>
<tr>
<td>Load Weight</td>
<td>Double</td>
<td>Lower Limit=0, Upper Limit=999999, Precision=0</td>
<td>The maximum loading weight of the equipment in pounds or kilograms.</td>
</tr>
<tr>
<td>Load Width</td>
<td>Float</td>
<td>Lower Limit=0, Upper Limit=999999, Precision=0</td>
<td>The maximum loading width of the equipment in inches or centimeters.</td>
</tr>
<tr>
<td>Logical Set</td>
<td>String (editable)</td>
<td>Any string</td>
<td>An indicator or name given to a group of cargo items which stay together.</td>
</tr>
<tr>
<td>LTI Code</td>
<td>Character (editable,[1])</td>
<td>(ABCDEFGHIJKLMNOPQRSTUVWXYZ[0-9])</td>
<td>The Limited Technical Inspection Code used for a particular cargo item.</td>
</tr>
<tr>
<td>M.T.</td>
<td>Double</td>
<td>Precision=2</td>
<td>Weight of an item in metric tons, regardless of the measurement system.</td>
</tr>
<tr>
<td>M/T</td>
<td>Double</td>
<td>Lower Limit=0, Precision=2</td>
<td>Volume of the item in measurement tons (40 cubic feet), regardless of the measurement system.</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------------------</td>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Manufacture Code</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Manufacture Date</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Marine Pollutant</td>
<td>String</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>MILSTAMP Cell Number</td>
<td>String (editable)</td>
<td>Container Cell Identification as defined in the Defense Transportation Regulations, Appendix V, Vessel Stowage Location Codes, as provided by MARAD. This format defines container cells using a Hatch/Bay/ Row/Tier (HBRT) convention with one position for each identifier. When the count of hatch, bay, row, or tier exceeds 9 , letters are used beyond this point (i.e., &quot;B&quot; means 11).</td>
<td></td>
</tr>
<tr>
<td>Mission Number</td>
<td>String (editable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Mode</td>
<td>Character (editable,[1])</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Model Number</td>
<td>String (editable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>MSE</td>
<td>String (editable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Mstat</td>
<td>Character (editable,[1])</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>NEW/Round</td>
<td>Float</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Nomenclature</td>
<td>String</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>NSN</td>
<td>String (fixed length,[13],editable)</td>
<td>The NSN consists of the applicable four digit federal supply classification code (FCS) and nine digit serial number that fixes the identity to the particular item of supply.</td>
<td></td>
</tr>
<tr>
<td>NSN Configuration</td>
<td>String (editable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Offload Geoloc</td>
<td>String (fixed length[4],editable)</td>
<td>The Geographical Location Code for the location where cargo is to be offloaded.</td>
<td></td>
</tr>
<tr>
<td>On Prty</td>
<td>Integer (editable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Onload Geoloc</td>
<td>String (fixed length[4],editable)</td>
<td>The Geographical Location Code for the location where cargo is to be on-loaded.</td>
<td></td>
</tr>
<tr>
<td>Other Provisions</td>
<td>String</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Overlap</td>
<td>Float</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>PTCN</td>
<td>String (fixed length)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Pack Group</td>
<td>enumerated type (editable)</td>
<td>None, I, II, III</td>
<td></td>
</tr>
<tr>
<td>Package Lot Number</td>
<td>String (editable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Packing Certificate</td>
<td>String (editable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Parent Pkg Id</td>
<td>String</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Parent Pkg NSN</td>
<td>String (fixed length[13],editable)</td>
<td>The NSN of the Parent for a particular Association is entered by ICODES into this attribute for all children of that parent.</td>
<td></td>
</tr>
<tr>
<td>Parent Pkg UIC</td>
<td>String (editable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Pkg Id</td>
<td>String (editable)</td>
<td>Any string</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Placement Status</td>
<td>Boolean</td>
<td>True/False Used in the Cargo Break-bulk window to indicate if a cargo item has been graphically placed inside of a break-bulk group. Has a value of Yes when the item has been placed and a value of No when the item has not been placed.</td>
<td></td>
</tr>
<tr>
<td>Plan Id</td>
<td>String (editable)</td>
<td>Any string This is the code used by LOGAIS in the database identifying all of the records belonging to a particular plan.</td>
<td></td>
</tr>
<tr>
<td>POD</td>
<td>String (fixed length[3], editable)</td>
<td>Any string, 3 char Point of Debarkation, destination port.</td>
<td></td>
</tr>
<tr>
<td>POE</td>
<td>String (fixed length[3], editable)</td>
<td>Any string, 3 char Point of Embarkation.</td>
<td></td>
</tr>
<tr>
<td>Primary Explosives</td>
<td></td>
<td>This is a calculated value for items containing only class 1 hazardous data to display the &quot;most hazardous&quot; or &quot;highest hazard&quot; information for that item.</td>
<td></td>
</tr>
<tr>
<td>Pre Ship</td>
<td>String (editable)</td>
<td>Any string Used for identifying the planned ship that a particular cargo item has been assigned to.</td>
<td></td>
</tr>
<tr>
<td>Pre Stow Area</td>
<td>String (fixed length[4], editable)</td>
<td>Any string, 4 char Used for identifying the stow area that a particular cargo item has been assigned to. Also used for importing stow area information from WPS to do an As Loaded Plan.</td>
<td></td>
</tr>
<tr>
<td>Pre Zone</td>
<td>String (editable)</td>
<td>Any string Used for identifying the zone that a particular cargo item has been assigned to.</td>
<td></td>
</tr>
<tr>
<td>Priority Order</td>
<td>Integer (editable)</td>
<td>Lower Limit=0 Code used to identify what priority order a particular cargo item has in the offload sequence.</td>
<td></td>
</tr>
<tr>
<td>Problem</td>
<td>String (uneditable)</td>
<td>Any string Indicates the item’s severity of conflict in an agent report, warning or violation.</td>
<td></td>
</tr>
<tr>
<td>Proper Shipping Name</td>
<td>String (editable)</td>
<td>Any string MILSTAMP proper shipping name, a standardized name given to a specific hazardous material.</td>
<td></td>
</tr>
<tr>
<td>Quantity</td>
<td>Integer</td>
<td>Lower Limit = 0 The number of items.</td>
<td></td>
</tr>
<tr>
<td>QuantityPer Cargo</td>
<td></td>
<td>This is the same thing as Item Count and indicates how many of a given component are found within a cargo item.</td>
<td></td>
</tr>
<tr>
<td>Rail Head</td>
<td>String (editable)</td>
<td>Any string Shipping origin.</td>
<td></td>
</tr>
<tr>
<td>Registration #</td>
<td>String (editable)</td>
<td>Any string Vehicle registration number.</td>
<td></td>
</tr>
<tr>
<td>Remarks</td>
<td>String (editable)</td>
<td>Any string Information that is entered by the ICODES user or imported from WPS.</td>
<td></td>
</tr>
<tr>
<td>Rounds</td>
<td>Integer (editable)</td>
<td>Lower Limit=0 The number of rounds in an ammunition shipment unit.</td>
<td></td>
</tr>
<tr>
<td>Rstat</td>
<td>Character (editable, [1])</td>
<td>[1456789] Shipment unit status code.</td>
<td></td>
</tr>
<tr>
<td>RUC</td>
<td>String (uneditable)</td>
<td>Any string Reporting Unit Code. This is the UIC that a given unit is reporting to.</td>
<td></td>
</tr>
<tr>
<td>Seal #</td>
<td>String (editable)</td>
<td>Any string A code used to identify the transportation equipment seal used for a particular item.</td>
<td></td>
</tr>
<tr>
<td>Secondary Class</td>
<td>Short</td>
<td>None, [1-9] United Nations hazardous cargo class used in conjunction with Division.</td>
<td></td>
</tr>
<tr>
<td>Secondary Division</td>
<td>Short</td>
<td>None,1,2,3,4,5,6 United Nations hazardous division used in conjunction with Class.</td>
<td></td>
</tr>
<tr>
<td>Secondary IMO Code</td>
<td>String (uneditable)</td>
<td>Any string The code used to identify the International Maritime Organization Class, Division and Compatibility Group for a particular UN Code.</td>
<td></td>
</tr>
<tr>
<td>Section</td>
<td>String (fixed length[3], editable)</td>
<td>Any string, 3 char A code used to identify a section within a particular unit.</td>
<td></td>
</tr>
<tr>
<td>Sequence #</td>
<td>Integer (editable)</td>
<td>Lower Limit=0 A code used to identify where a particular cargo item fits into a particular sequence of cargo items or events.</td>
<td></td>
</tr>
<tr>
<td>Serial Number</td>
<td>String (editable)</td>
<td>Any string Vehicle serial number or user defined identities.</td>
<td></td>
</tr>
<tr>
<td>Service</td>
<td>String (fixed length[2], editable)</td>
<td>Any string, 2 char An organization code identifying a specific armed-service organization within DoD.</td>
<td></td>
</tr>
<tr>
<td>Column Name</td>
<td>Data Type</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-----------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>Shelf Life Code</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>Specific code used to indicate how long a given item will last when stored.</td>
</tr>
<tr>
<td>Ship</td>
<td>String</td>
<td>Any string</td>
<td>Name of ship or barge intended for use.</td>
</tr>
<tr>
<td>Ship Cell Number</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>A ship specific cell number obtained from MARAD data.</td>
</tr>
<tr>
<td>Ship Config</td>
<td>String (editable)</td>
<td>Any string</td>
<td>Shipping configuration code.</td>
</tr>
<tr>
<td>SL3</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>Stock Listing 3 code used to identify a type of item or material.</td>
</tr>
<tr>
<td>Spcl Hdlg Type Water</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>No description available.</td>
</tr>
<tr>
<td>Spcl Hdlg Water</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>No description available.</td>
</tr>
<tr>
<td>Spcl Provisions</td>
<td>String</td>
<td>Any string</td>
<td>HazMat special provisions for a particular UN Code.</td>
</tr>
<tr>
<td>Spcl Stowage</td>
<td>String (editable)</td>
<td>Any string</td>
<td>HazMat special stowage provisions for a particular UN Code.</td>
</tr>
<tr>
<td>Stack Limit</td>
<td>Integer (editable)</td>
<td>Lower Limit=0, Upper Limit=99</td>
<td>The maximum number of cargo items that can be stacked on top of one another.</td>
</tr>
<tr>
<td>Stow Area</td>
<td>String (fixed length, 4)</td>
<td>{ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890}</td>
<td>MILSTAMP code of stow area.</td>
</tr>
<tr>
<td>Stow Height</td>
<td>Float</td>
<td>Lower Limit=0, Upper Limit=9999, Precision=0</td>
<td>The height of a particular cargo item when it is stowed.</td>
</tr>
<tr>
<td>Stow Weight</td>
<td>Float</td>
<td>Lower Limit=0, Upper Limit=9999999, Precision=0</td>
<td>The weight of a particular item when it is stowed.</td>
</tr>
<tr>
<td>Stowable</td>
<td>Boolean</td>
<td>True/False</td>
<td>Indicates whether the item is stowable according to the Cargo Agent, either Yes or No. If any of the dimensions or weight equal zero, Stowable will be No.</td>
</tr>
<tr>
<td>Stowage Codes</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>CODES-specific codes used to reference the Stowage Codes found in the IMDG.</td>
</tr>
<tr>
<td>Stowed</td>
<td>Boolean</td>
<td>True/False</td>
<td>Whether the item has been stowed or not, either Yes or No.</td>
</tr>
<tr>
<td>SUC</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>Supporting Unit Code. This is the UIC that a given unit is supporting.</td>
</tr>
<tr>
<td>Supply Class</td>
<td>Character (editable, 1)</td>
<td>{ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890}</td>
<td>Identifies the Supply Class Code (i.e., A, 2, or 7).</td>
</tr>
<tr>
<td>Symbol Name</td>
<td>String</td>
<td>Any string</td>
<td>Indicates the name of the symbol that is used to graphically represent a stowed cargo item. For example, if you want a stowed item to look like a trailer, you could key in M101 and that item will appear as a M101 trailer in the Graphics Window.</td>
</tr>
<tr>
<td>Symbol Type</td>
<td>Symbol, Box</td>
<td>Indicates the type of symbol that is used to graphically depict a stowed cargo item. Symbol means there is a detailed symbol while Box indicates a box is used.</td>
<td></td>
</tr>
<tr>
<td>Tag</td>
<td>Character (editable, 17)</td>
<td>{ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890}</td>
<td>USMC attribute related to AIT.</td>
</tr>
<tr>
<td>Tag Id</td>
<td>Integer (editable)</td>
<td>Lower Limit=0, Upper Limit=9999</td>
<td>USMC attribute related to AIT.</td>
</tr>
<tr>
<td>TAM Control Number</td>
<td>String (editable)</td>
<td>Any string</td>
<td>No description available.</td>
</tr>
<tr>
<td>TAM Index</td>
<td>String (editable)</td>
<td>Any string</td>
<td>Table of Authorized Material Index Code (i.e., 01, 03 or 07).</td>
</tr>
<tr>
<td>Target Utilization</td>
<td>Float</td>
<td>Percent</td>
<td>Percentage describing how much the break-bulk group volume to use.</td>
</tr>
<tr>
<td>TCN</td>
<td>String (fixed length, 17, editable)</td>
<td>Any string, 17 char</td>
<td>MILSTAMP transportation control number.</td>
</tr>
<tr>
<td>Team Name</td>
<td>String (editable)</td>
<td>Any string</td>
<td>Name given to all cargo assigned to a particular Amphibious Assault Ship.</td>
</tr>
<tr>
<td><strong>Templating Status</strong></td>
<td>Enumerated type Assigned, Templated, None</td>
<td>Used to indicate how a particular cargo item has been stowed aboard a ship (i.e. Templated, Assigned, or None)</td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Tertiary Class</strong></td>
<td>Short None, [1-9]</td>
<td>United Nations hazardous cargo class used in conjunction with Division.</td>
<td></td>
</tr>
<tr>
<td><strong>Tertiary Division</strong></td>
<td>Short None,1,2,3,4,5,6</td>
<td>United Nations hazardous division used in conjunction with Class.</td>
<td></td>
</tr>
<tr>
<td><strong>Tertiary IMO Code</strong></td>
<td>String (uneditable) Any string</td>
<td>The code used to identify the International Maritime Organization Class, Division and Compatibility Group for a particular UN Code.</td>
<td></td>
</tr>
<tr>
<td><strong>Time Stamp</strong></td>
<td>Date MM/DD/YY 00:00:00</td>
<td>Date and Time that a particular record was created.</td>
<td></td>
</tr>
<tr>
<td><strong>Total NEW</strong></td>
<td>Float Precision=6</td>
<td>Net explosive weight in pounds or kilograms.</td>
<td></td>
</tr>
<tr>
<td><strong>Total Rounds</strong></td>
<td>Integer Lower Limit=0</td>
<td>A user-defined total number of rounds for a particular cargo item.</td>
<td></td>
</tr>
<tr>
<td><strong>Trn Radius</strong></td>
<td>Integer (editable) Lower Limit=0 Upper Limit=9999</td>
<td>The turning radius required for a vehicle.</td>
<td></td>
</tr>
<tr>
<td><strong>Type Cargo</strong></td>
<td>Character (editable) {ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789}</td>
<td>Cargo type code, unique identifier that specifies the hazardous cargo classification.</td>
<td></td>
</tr>
<tr>
<td><strong>Type Eqpt</strong></td>
<td>Character (editable,[1]) {CDEFGHJKLMNPRSTUVWXYZ}</td>
<td>ECF Model Library type equipment code.</td>
<td></td>
</tr>
<tr>
<td><strong>Type Expl</strong></td>
<td>String (editable) Any string</td>
<td>Type of ammunition, None, Fireworks or Substance.</td>
<td></td>
</tr>
<tr>
<td><strong>Type Pack</strong></td>
<td>String (fixed length[2], editable) Any string, 2 char</td>
<td>MILSTAMP type pack code.</td>
<td></td>
</tr>
<tr>
<td><strong>UIC</strong></td>
<td>String (editable) Any string</td>
<td>Unit Identification code.</td>
<td></td>
</tr>
<tr>
<td><strong>UIC Noun Name</strong></td>
<td>String Any String</td>
<td>This is a long description for a given UIC.</td>
<td></td>
</tr>
<tr>
<td><strong>ULN</strong></td>
<td>String (fixed length [7], editable) Any string, 7 char</td>
<td>Unit Line Number for a particular cargo item.</td>
<td></td>
</tr>
<tr>
<td><strong>UN Code</strong></td>
<td>Short [UN</td>
<td>NA]+UNUM(char 2, fixed int 4)</td>
<td>A United Nations or North American hazardous identification number.</td>
</tr>
<tr>
<td><strong>UN Code List</strong></td>
<td>String (uneditable) Any string</td>
<td>A series of UN Codes indicating each unique UN Code inside of a multiple hazards cargo item in ICODES. This attribute is not editable and is only used for cargo items with more than one UN Code attached to them.</td>
<td></td>
</tr>
<tr>
<td><strong>Unit Of Issue</strong></td>
<td>String (fixed length [2], editable) Any string, 2 char</td>
<td>Measurement Units used when item was first issued (i.e., feet, inches or pounds)</td>
<td></td>
</tr>
<tr>
<td><strong>UPTT Code</strong></td>
<td>String (fixed length [2], editable) Any string, 2 char</td>
<td>Unit Personnel and Tonnage Table Code</td>
<td></td>
</tr>
<tr>
<td><strong>Vessel Stowage</strong></td>
<td>Character (editable,[1]) {ABCDE}</td>
<td>Code specifying the authorized stowage locations for specific cargo.</td>
<td></td>
</tr>
<tr>
<td><strong>Volume</strong></td>
<td>Double Lower Limit=0, Precision=0</td>
<td>Volume of the minimum box which encloses the item, in cubic feet or cubic meters.</td>
<td></td>
</tr>
<tr>
<td><strong>Weight</strong></td>
<td>Float Lower Limit=0, Upper Limit=999999999, Precision=0</td>
<td>Weight of the item, in pounds or kilograms</td>
<td></td>
</tr>
<tr>
<td><strong>Width</strong></td>
<td>Float Lower Limit=0, Upper Limit=999999999, Precision=0</td>
<td>Maximum width of the item, in inches or centimeters.</td>
<td></td>
</tr>
<tr>
<td><strong>WPS Record #</strong></td>
<td>Integer (uneditable) Any integer</td>
<td>WPS record number.</td>
<td></td>
</tr>
<tr>
<td><strong>WPS Stow Area</strong></td>
<td>String (editable)</td>
<td>This attribute will be filled in when a network pull is done from WPS to ICODES. The information found here will match the STOW field in WPS for that cargo item. This attribute is provided for use during both the As Loaded process as well as for comparison purposes with the ICODES Stow Area attribute.</td>
<td></td>
</tr>
<tr>
<td>Zone Identifier</td>
<td>String (uneditable)</td>
<td>Any string</td>
<td>Used to indicate what zone a particular cargo item was placed on a ship.</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
Appendix B: The ICODES Symbols Library