7

A Reference Model for Information Infrastructures

R. Elmasri, D. Son, J. Mills and N. Kishor
The Automation & Robotics Research Institute,
The University of Texas at Arlington
7300 Jack Newell Blvd. S.
Fort Worth, TX 76118
Phone (817) 272-5900 Fax (817) 272-5952
{relmasri, dson, jmills}@arri.uta.edu

Abstract

There are a number of different approaches to information infrastructures, all of which purport to be an answer to the problem of integrating diverse heterogeneous systems. Frameworks and Reference Models are also used for similar purposes. Each has a slightly different focus and approach, yet all are essentially addressing the same problem. Several reference models have been proposed and information infrastructure systems are under development for shop floor control in the discrete manufacturing and semiconductor industries, for high level distributed interactive simulations, software engineering and for collaborative design and manufacturing. In this paper we review several of the current approaches and develop a reference architecture which is then used to categorize and compare these same approaches. Approaches and models reviewed include the ADAPTIVE Communication Environment (ACE), the Common Object Request Broker Architecture/Common Object Services Specification (CORBA/COSS), the Distributed Computing Environment (DCE), the High Level Architecture (HLA), the Simulation-Based Design (SBD) architecture, the National Industrial Information Infrastructure Protocols (NIIIP), Shipbuilding Information Infrastructure Project (SHIIP), and the Systems Integration Architecture (SIA). A brief description of the proposed reference architecture and of each system is given and the services it provides are compared with those in the reference architecture.

Keywords
Collaboration, Framework, Reference Model, Information Infrastructure

J. J. Mills et al. (eds.), *Information Infrastructure Systems for Manufacturing II*

1 INTRODUCTION

The proliferation of personal computers and desktop workstations with the advances in technology and the tremendous growth of networking and internetworking with the availability of plenty of bandwidth has changed the role of computing in institutions of all forms. Whether for business, educational, or cultural purposes, organizations are seeking to maximize their effectiveness by allowing the information needed to manage and run operations to be created, maintained, and disseminated over a growing base of interconnected desktop computers, instead of building on host-based centralized computing roots. Distributed information is the catalyst enabling organizational and business-process restructuring to be done at unprecedented speeds [14].

To lower costs, get products to market faster, and respond to changing market conditions, many organizations have turned to a flexible form of computing called client/server computing that supports the use of desktop computers as clients accessing information from various backend servers. The migration from mainframes to client/server computing has resulted largely from the emergence of the widespread use of personal computers and software, several standard graphical user interfaces, better networking, and increasingly capable and inexpensive servers.

Companies involved in intense global competition have begun flattening their organizations to avoid unnecessary layers of hierarchy. With the use of decentralized computing resources, management is able to give users more authority for local decision making, eliminate bureaucratic positions, and tap into the innovative talents of more members of the organization. Early developments of groupware like e-mail and conferencing with the wider use of advanced communication and client/server technologies is facilitating the evolution of collaborative computing that makes seamless semantic level interaction possible among users on heterogeneous systems.

With the need and wide spread acceptance of collaborative computing, various collaboration frameworks have been proposed by industry and academia. The purpose of this survey is to understand some of these frameworks and propose a general framework of collaboration that could be used to compare the features and capabilities of these frameworks. In this paper we focus on general comparison of individual frameworks features. The following frameworks have been explored:

- ADAPTIVE Communication Environment (ACE) [12]
- Common Object Request Broker Architecture/Common Object Services Specification (CORBA/COSS) [7, 13]
- Distributed Computing Environment (DCE) [2]
- High Level Architecture (HLA) [4]
- National Industrial Information Infrastructure Protocols (NIIIP) [3, 6]
- Simulation Based Design (SBD) [10]
- Shipbuilding Information Infrastructure Protocols (SHIIP) [11]
- Systems Integration Architecture (SIA) [5]

2 FRAMEWORK FOR INTEGRATION

2.1 Framework versus Architecture

The general collaboration framework that is proposed in this paper is synthesized from the common features of a multitude of frameworks or infrastructures under construction or in use by industry and academia. Some of the common technological features are:

- Client/Server Technology
- Object orientation, Object wrappers
- Components, Distributed Objects
- Component Frameworks
- Collaboration Bus
- High Level Frameworks for Collaboration

A *framework* or an *architecture* is a high-level description of the organization of functional responsibilities within a system. The goal of an architecture or framework is to convey information about the general structure of systems. Although there is no single, all encompassing architecture for computing, before developing collaborative applications, two architectures should be devised. These are a *technical architecture* and an *information architecture* [1].

A technical architecture provides a blueprint for assembling technology components. A technical architecture defines what tools and technologies will be used and how they will be used. This may include the definition of objects that encapsulate several databases, middleware and other technologies, as well as which development tools to use and how they will be integrated to provide a complete software project support environment.

An *information architecture* describes content, behavior, and interaction of business objects. These concepts build a semantically rich model of the problem domain. The information architecture prescribes the building blocks for application development. Business objects use the services of the technical architecture objects. An information architecture provides a framework for the information components of the business, e.g., subjects, events, roles, associations, business rules etc.

In the general reference framework for collaboration shown in figure 1, the collaboration bus and all the layers below that form the technical architecture. All the layers above the collaboration bus form the information architecture.

2.2 The General Reference Framework for Collaboration

Figure 1 shows a general reference framework for building a collaborative infrastructure. This general framework is synthesized from the common features of various collaboration frameworks. Hardware, operating system, legacy and simple non-collaborative applications including various system utilities form the computing system to be integrated into a collaborative environment. The hardware includes computing (CPUs), communication (wireless, wired), and data storage/retrieval (electrical, magnetic, and optical) hardware. Traditional remote procedure call (RPC) based mechanisms or object wrappers can be used to provide a dynamically reconfigurable computing system interface to upper layers of a collaboration framework. In fact, the system-level frameworks provide an organized environment for using a collection of objects that reside on the local computing system. System-level frameworks offer simple patterns that guide the collaboration of objects on the local computing system to provide the system specific services requested through the collaboration bus.

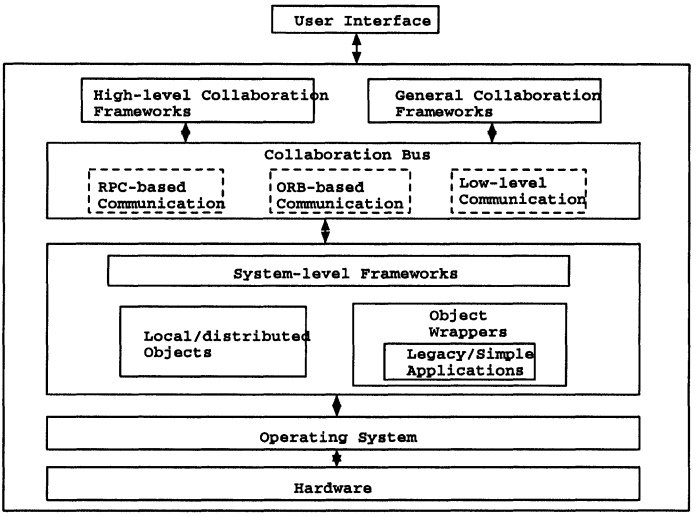

Figure 1: General Reference Framework for Collaboration

The collaboration bus is the backbone for building a collaborative infrastructure. It provides interoperability among software components residing on various computing systems. General collaboration frameworks are the collection of tools and components that use the distributed services provided through the collaboration bus to provide semantic-level interactions among distributed software components. High-level collaboration frameworks are more specialized collaboration frameworks that are designed to achieve specific business objectives. Typically a user in a collaborative environment interacts with other users and distributed information resources via a user interface. All these user interactions generated through the user interface are constrained by the high-level business

objectives. The actual semantics of interactions are resolved and put into action by the general collaboration frameworks by discovering the necessary software components via the collaboration bus.

The types of services provided by the System-level Frameworks include naming, event, time, and so on. At the General Collaboration Frameworks level, higher-level services are provided that are domain-independent, such as user interface. Finally at the High-level Collaboration Frameworks level, domain-specific services are provided for each application domain such as accounting, simulation, manufacturing, and so on. We will be using this general framework of figure 1 to assess, categorize, and compare some of the frameworks published so far.

3 Technical Frameworks

The following technical frameworks or architectures are discussed in this section.

- ADAPTIVE Communication Environment (ACE)
- Common Object Request Broker Architecture/Common Object Services Specification (CORBA/COSS)
- Distributed Computing Environment (DCE)

3.1 ADAPTIVE Communication Environment (ACE)

A Dynamically Assembled Protocol Transformation, Integration, and eValuation Environment (ADAPTIVE) Communication Environment (ACE) is a high-level C++ toolkit for writing sophisticated concurrent, parallel, and distributed applications. ACE is an object-oriented network programming toolkit for developing computer communication software.

ACE provides a standard library of distributed services that are packaged as self-contained components. These reusable components provide distributed system services like naming, event routing, logging, time synchronization, and network locking. The common communication software capabilities of ACE include [12]:

- event demultiplexing and event handler dispatching,
- service initialization,
- interprocess communication,
- shared memory management,
- message routing,
- dynamic (re)configuration of distributed services,
- concurrent execution and synchronization.

3.1.1 ACE and General Reference Framework for Collaboration

ACE provides a rich set of reusable C++ wrappers and framework components that perform common communication software tasks across a range of operating system platforms. So ACE provides the object wrappers and some system-level frameworks, mainly dealing with communications, in the General Reference Framework shown in figure 1. These are eventually used to build a collaboration

bus. Specifically, ACE has been used to build a real-time implementation of CORBA called TAO.

3.2 Common Object Request Broker Architecture/ Common Object Services Specification (CORBA/COSS)

Figure 2 shows OMG's Object Management Architecture. CORBA/COSS refers to Object Request Broker and Common Object Services in the architecture. The bottom 4 levels in figure 2 provide domain-independent services, whereas the top 2 levels (Vertical Market Facilities, Collaborative Applications) are domain specific.

3.2.1 Object Request Broker

The Object Request Brokers [7] provide the basic communication channel through which objects on various systems interact. ORBs enable objects to transparently make and receive requests and responses in a distributed environment. It is the foundation for building applications from distributed objects and for interoperability between applications in hetero- and homogeneous environments.

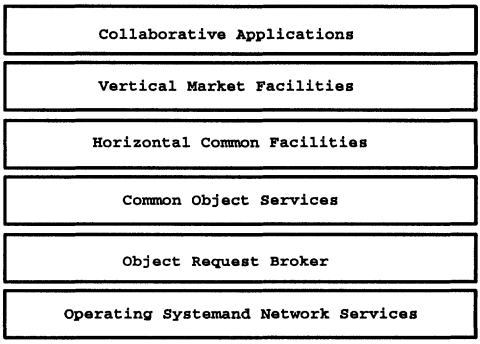

Figure 2: OMG's Object Management Architecture

3.2.2 Common Object Services

Common Object Services are a collection of services (interfaces and objects) that support basic functions for using and implementing objects. They are collections of system-level services packaged as components with IDL-specified interfaces. They represent the essential interfaces needed to create an object, introduce it into its environment, use and modify its features, and so forth. OMG has currently defined standards for fifteen object services which are naming, event, life cycle, persistent object, transaction, concurrency control, relationship, externalization, query, licensing, property, time, security, object trader and collection services [7, 8].

3.2.3 CORBA/COSS and General Reference Framework for Collaboration

With respect to the General Reference Framework in figure 1, CORBA/COSS provides object wrappers or object adapters, distributed objects or components, many system-level frameworks and the collaboration bus. . Furthermore, OMA's horizontal common facilities provide the necessary building blocks to build general collaboration frameworks. The vertical market facilities are the high-level or specialized collaboration frameworks for specific domain.

3.3 Distributed Computing Environment (DCE)

The Open Group's Distributed Computing Environment (DCE) [2, 9] is a suite of integrated software services that is part of a computing system's infrastructure and enables the development of distributed applications across heterogeneous systems. To support the development of distributed applications DCE provides the following key distributed services like Remote Procedure Call, Security service, Cell and Global Directory Services, Threads service, Distributed Time Service, and Distributed File Service.

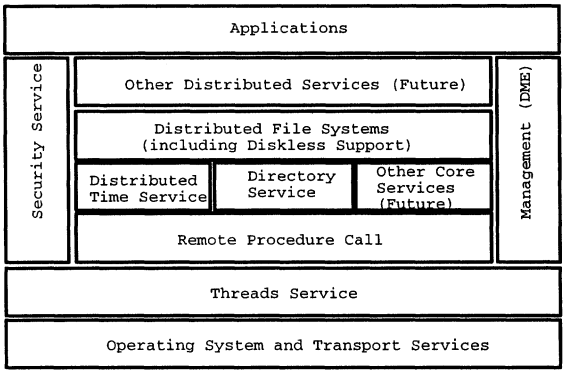

Figure 3: DCE Architectural Framework

Figure 3 shows the DCE architectural framework and components. It is designed to be independent of hardware, operating systems, and networks, allowing DCE distributed applications to run on a wide variety of computers simultaneously. Communication transparency is incorporated into the infrastructure for the RPC. Network security and locating distributed resources are solved by security and directory services. Threads service enable RPC servers to handle multiple requests concurrently.

3.3.1 DCE components

The functionality of the components in the DCE architectural framework in Figure 3 can be summarized as follows:

- *Remote Procedure Call (RPC)* - mechanism by which clients invoke procedures in servers. The RPC mechanism insulates clients from details of where servers are located on the network, the types of hardware and operating system platforms on which they execute, differences in data representations between client and server platforms, and the particular network transports in use.
- *Directory Services* - support local DCE administration domains called cells and inter-cell name resolution.
- *Security Services* - support authentication, authorization, and privacy.
- *Distributed Time Services (DTS)* - synchronize all clocks in a DCE cell, as well as between cells.
- *Threads Service* - supports the creation and management of multiple threads of control within a client or server.
- *Distributed File Service* - provides a uniform namespace, and file location transparency.

3.3.2 DCE and General Reference Framework for Collaboration

With respect to the General Reference Framework, some system-level frameworks and the collaboration bus are what DCE provides.

4 TECHNICAL FRAMEWORKS

The following information frameworks or architectures are discussed in this section.

- High Level Architecture (HLA)
- National Industrial Information Infrastructure Protocols (NIIIP)
- Shipbuilding Information Infrastructure Project (SHIIP)
- Simulation Based Design (SBD)
- Systems Integration Architecture (SIA)

4.1 High Level Architecture (HLA)

The Department of Defense (DoD) Modeling and Simulation Master Plan defines the objective of developing a Common Technical Framework for Modeling and Simulation. The Common Technical Framework [4] consists of three components for simulation development and interaction: the High Level Architecture (HLA), Conceptual Model of the Mission Space (CMMS), and Data Sandardization (DS). Among the three components HLA is the highest priority.

4.1.1 Overview of HLA

The fundamental goal of HLA is to establish a common high-level architecture to facilitate the interoperability of all types of models and simulations among themselves and with C4I systems as well as to facilitate the reuse of Modeling and Simulation (M&S) components. The *HLA federation* is the set of HLA-compliant interoperating simulations. A simulation that is a member of a HLA federation is called a *federate*. Figure 4 illustrates the functional view of the HLA. HLA is defined by the following components [4]:

- *HLA rules* - A set of rules which must be followed to achieve proper interaction of simulations in a federation. These rules describe and cleanly separate the responsibilities that are placed on both the federates and on the runtime infrastructure (RTI). The RTI is a set of software components that implement the services specified by the HLA Interface Specification. The RTI provides the common interface services for the execution of an HLA federation.
- *Interface Specification* - The set of HLA interfaces that provide each of the services between the RTI and a federate. These services are organized into the following distributed simulation management categories: Federation Management, Object Management, Time Management, Declaration Management, Ownership Management, and Data Distribution Management.
- *Object Model Template* - The prescribed common method for recording the information contained in the required HLA Object Model for each federate and federation. Each federate is required to define its own Simulation Object Model (SOM). The SOM defines the kinds of interaction messages that the simulation is capable of sending or receiving. Each federation is required to define its particular Federation Object Model (FOM) which standardize the way objects are represented between the federates, what their attributes are, and the kinds of interactions that will be supported between federates.

4.1.2 HLA and General Reference Framework for Collaboration

HLA provides the high level collaboration frameworks for the simulation domain in the General Reference Framework for Collaboration shown in figure 1. HLA is a modeling and simulation vertical market facility in the OMG's Object Management Architecture shown in figure 2. HLA can be used by simulation system developers and policy makers because it provides a systematic and consistent basis for addressing simulation system design and implementation issues in a collaborative environment.

4.2 National Industrial Information Infrastructure Protocols (NIIIP)

National Industrial Information Infrastructure Protocols (NIIIP) [6] are being developed by a consortium of 18 organizations to enable Virtual Enterprises (VE)

to collaborate and share engineering and manufacturing information. NIIIP defines a virtual enterprise as a temporary consortium of organizations, often crossing company boundaries, that come together to exploit fast-changing opportunities. The NIIIP consortium is developing a reference architecture and reference implementation for the technologies needed from industrial Virtual Enterprises. Basically it seeks to satisfy the information needs of the Virtual Enterprise by integrating and building upon the work of a number of selected standards. It consolidates, harmonizes, integrates, and extends existing protocols.

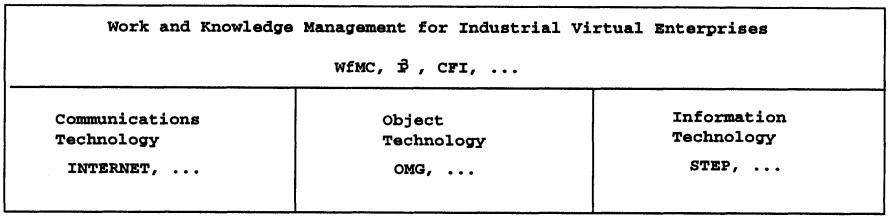

Figure 5: NIIIP Reference Architecture

4.2.1 NIIIP Reference Architecture

Figure 5 shows NIIIP Reference Architecture. At the highest level, there are four technology requirements for industrial Virtual Enterprises [6]:

- Common communication protocols
- Uniform object technology for system and application interoperability
- Common information model specification and exchange
- Cooperative management of integrated Virtual Enterprise processes

NIIIP consortium uses core technologies such as: Internet and related communications facilities and services; Object Management Group (OMG) and related object technologies; and Standard for the Exchange of Product Model Data (STEP) and related information modeling technologies. Since these core technologies are not yet well integrated, NIIIP focuses on extending and integrating these technologies to enable cooperative work.

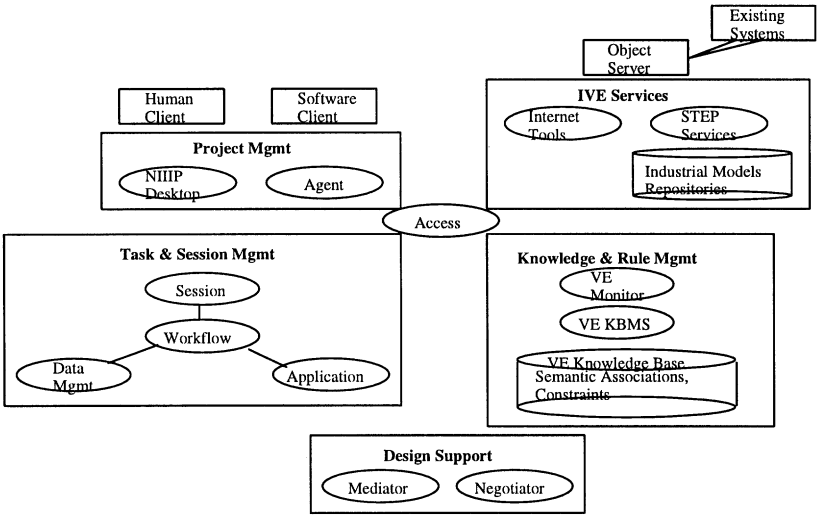

Figure 6: NIIIP Components

As shown in figure 6, the NIIIP architecture currently defines 13 components (shown in ovals in figure 6) and these in turn are grouped into five subsystems. The components of the NIIIP architecture are candidates for Common Facilities within OMG's architecture.

4.2.2 NIIIP components

The 13 NIIIP components are services needed to create and manage industrial Virtual Enterprises and can be summarize as follows: [3]

- *Desktop* - Human interface to multiple VEs and VE resources; VE administration
- *Access* - CORBA and non-CORBA mediated object access; replication, caching; interface to DFS, SHTTP, TCP/IP disguised initially as a CORBA request but then "trapped out" (intercepted)
- *Agent* - Non-human interface to VE organizational structures, resource ownership, settlement; "proxy objects" acting as transducers
- *STEP Services* - Behaviors to support the STEP Data Access Interface (SDAI)
- *Data Management* - Shared data management; change management; configuration management; View/schema/encryption translations between source and target
- *Workflow* - Cross-product workflow management
- *Session* - Authentication, uses, context (tasks and sessions), resource allocation, transaction logging; simulation; long transaction control

- *Application* - Application invocation setup; moving data to application, application to data
- *Mediator* - Resolving semantic ambiguity; summarizing, abstracting, validating data; localized knowledge; maintenance of local ontologies; data translation between source and target application objects
- *Negotiator* - Manage multi-party negotiations impartially
- *VE KBMS* - Active semantic network database for VE
- *Internet Tools* - Access to the services of the Internet; conferencing, WWW, control point for the secure VE
- *VE Monitor* - Generate *before* and *after* method bindings to NIIIP protocol anchor and end-user application objects

4.2.3 NIIIP and General Reference Framework for Collaboration

With respect to the General Reference Framework for Collaboration, NIIIP provides general collaboration frameworks and high-level collaboration frameworks for the Virtual Enterprise domain.

4.3 Simulation Based Design (SBD)

Simulation Based Design [10] is an open, multi-agent, distributed, collaborative, virtual development environment aimed at revolutionizing the acquisition and support of complex systems throughout their entire life cycle. SBD is a technology component of DARPA's Simulation Based Acquisition (SBA) Program. SBD provides geographically distributed enterprises with a synthetic environment for planning, developing and optimizing a product through Collaborative Virtual Prototyping (CVP). SBD supports multiple virtual products operating in physics based synthetic environments and their Multi-Disciplinary Analysis (MDA) and Multi-Disciplinary Optimization (MDO) via a detailed Smart Product Model (SPM) stored in a database. SBD uses HLA and CORBA as its interface standards.

SBD can be seen as a common database, consisting of smart product model, smart product catalog and simulations, surrounded by various tools like requirement tools, legacy engineering tools, workflow tools, commercial tools (CAD etc.), 3D visualization tools, cost tools etc. A layered architecture for SBD is shown in figure 7.

4.3.1 SBD Architecture

The layered architecture for SBD shown in figure 7 hides details of the underlying hardware and network by supporting a collection of higher level agents which provide direct, broad based support for the generic user. Related agents are grouped into user services, reflecting the principal capabilities which will be supported by the SBD system. The user services are [10]:

- *Data Services* - deal with product and process representation and manipulation. It uses an object oriented approach to modeling data, allowing

component descriptions and behaviors to be linked in an active system called the Smart Product Model (SPM).

- *Integration Services* - support collaboration and interoperation of tools. It includes a shared electronic notebook for human communication and systems for assembling collections of tools into integrated megaprograms.
- *Interaction Services* - provide advanced visualization, immersive and collaborative means for spatially manipulating products and processes. It also provides for operator-in-the-loop simulation.
- *Application Services* - manage the more or less static (mostly Commercial Off The Shelf (COTS)) applications which perform specific roles for SBD users.

Individual services are comprised of multiple agents, and agents across the system communicate and interoperate in very flexible ways. The intelligent information bus is the underlying layered structure below the user services. Its major role is to support the communications needs of the higher level agents. This bus comprises three main components [10]:

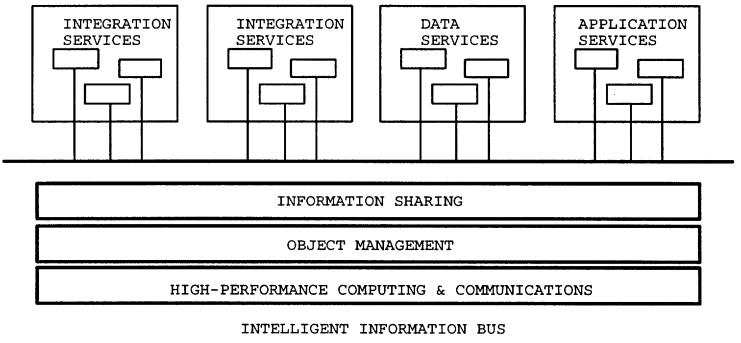

Figure 7: Simulation Based Design Architecture

- *Information Sharing layer* - is concerned with the higher level communications needs between entities in the system. It supports notions such as publication and subscription of interest in specific objects, attributes, or events.
- *Object Management layer* - hides the complexity of communication in a distributed, heterogeneous computing environment from users and applications.
- *High Performance Computing and Communications (HPCC) network interface layer* - isolates details of the underlying hardware and communications at the network level from the Object Management Layer and other, higher level agents.

4.3.2 SBD and General Reference Framework for Collaboration

SBD provides the collaboration bus, general collaboration framework, and a high level collaboration framework for CVP in the general reference framework for collaboration shown in figure 1. SBD is a vertical market facility for CVP in the OMG's object management architecture.

4.4 Shipbuilding Information Infrastructure Project (SHIIP)

The goal of the SHIIP project is to develop, deploy and standardize a new shipbuilding methodology for the US shipbuilding industry [11]. The advanced information infrastructure comprises the component of the new shipbuilding methodology. The goal of the SHIIP information infrastructure is to deliver enterprise-wide information to the shipbuilding work force.

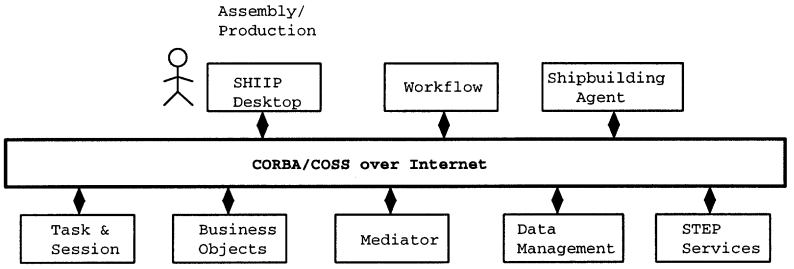

Figure 8: SHIIP Reference Architecture

4.4.1 SHIIP Reference Architecture

The SHIIP Reference Architecture follows the lead of the NIIIP architecture in that it seeks to satisfy the information needs of the shipbuilding enterprise by integrating and building upon the work of a number of selected standards. These standards include the Object Management Group (OMG) for distributed object technology; the Internet Standards (including the WWW standards such as HTML and VRML) for communications; the STEP standards for product information sharing; and the Workflow Management Coalition standards for workflow.

Figure 8 shows SHIIP reference architecture at the top level. The SHIIP reference architecture contains a subset of the Components Objects from the NIIIP reference architecture. This subset consists of services selected based on their low-risk nature and applicability to the needs of the shipbuilding enterprise. The Workflow, Task/Session, and STEP services have been substantially defined and prototyped within the NIIIP project and will be deployed in the SHIIP project. The Desktop, Agent, and Data Management Components will be specialized from the NIIIP protocols to suit shipbuilding requirement. The Business Objects (and associated Mediator) represent a deployment of work under way within OMG in its Business Object Domain Task Force. These objects have not been explicitly

specified in the NIIIP protocols but are interoperable with the NIIIP objects through the CORBA framework.

4.4.2 SHIIP and General Reference Framework for Collaboration

With respect to the General Reference Framework for Collaboration, SHIIP provides general collaboration frameworks and high-level collaboration frameworks for the ship building application domain.

4.5 Systems Integration Architecture (SIA)

Systems Integration Architecture (SIA) [5], developed at the Agile Aerospace Manufacturing Research Center (AAMRC) of the Automation & Robotics Research Institute, is an agile information infrastructure that integrates legacy systems and provides a basis model to build virtual enterprises. The basis model for SIA is a functional programming approach to systems development. It is based on the concept that all information processing is a series of transformations of data sets called aspects. The transformations are done by modules called Functional Transformation Agents (FTAs). FTAs are considered as black boxes whose interior can be developed using any software or hardware.

Figure 9 shows a high level block diagram of SIA. The architecture has three modules to provide the necessary services to create and control FTAs and aspects in a heterogeneous distributed environment. The three modules are [5]:

- *Executive Module* - provides the front end of SIA using a GUI. It makes all the necessary communications required in a distributed environment transparent to the user. It uses the communications module to achieve this transparency. Executive module essentially controls the user access to the system and its resources. It allows any degree of business control from a workflow management approach. The executive can start, stop, and pause/resume processes as well as request status information.
- *Librarian Module* - is the central module in SIA. It is a kind of meta data dictionary whose basic function is to manage the references and associations among users, processes, applications, and data sets, and to maintain information about the existence and format of all data in the system.
- *Communications Module* - now uses CORBA as the reference architecture to create a heterogeneous and distributed communications interface. The executive module and the librarian use this communication interface via high level network services to make the network communications transparent to the end user.
- *Function (or Launcher) Module* - allows users to launch local/remote applications using local/remote data sets, while using a local user interface.

4.5.1 SIA and General Reference Framework for Collaboration

With respect to the General Reference Framework for Collaboration, SIA provides general collaboration framework which facilitates the development of high-level collaboration frameworks. For instance, *Aeroweb* which is a high-level collaboration framework for supply-chain management has been developed on top of SIA.

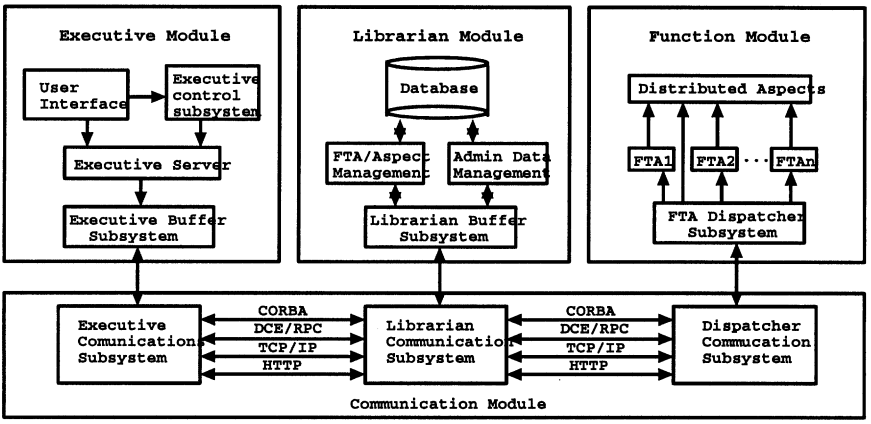

Figure 9: Systems Integration Architecture

5 CONCLUSIONS

We have overviewed 8 existing frameworks and categorized the existing collaboration frameworks into technical and information frameworks. We also proposed a general reference framework for collaboration and used it to compare the features and capabilities of those frameworks. Figure 10 illustrates the features and capabilities of those frameworks.

CORBA/COSS and DCE are the basic technical frameworks for building a heterogeneous distributed system. Since these are meant to be standards and specific implementations from various vendors are available. ACE provides a reconfigurable framework for basic network services required to build a distributed system. ACE is used to build a real time CORBA implementation called TAO.

Most of the information frameworks explored use CORBA/COSS to provide collaboration bus. HLA is a high priority effort by DMSO to facilitate all types of DoD simulations. Since HLA provides an object oriented information framework for simulations, it may be standardized to be a vertical market facility for simulations by OMG. DARPA's SBD is a robust framework for collaborative and distributed design. SHIIP seems to be the first and largest industry effort to use the existing technological base and new standards like NIIIP, HLA, SBD, and CORBA/COSS to build an enterprise wide collaborative virtual prototyping

system. SIA provides flexible general collaboration framework to facilitate the rapid development of high-level collaboration frameworks on top of it. We also have to mention that there are other collaboration frameworks which are being developed, but not covered in this paper due to space limitation.

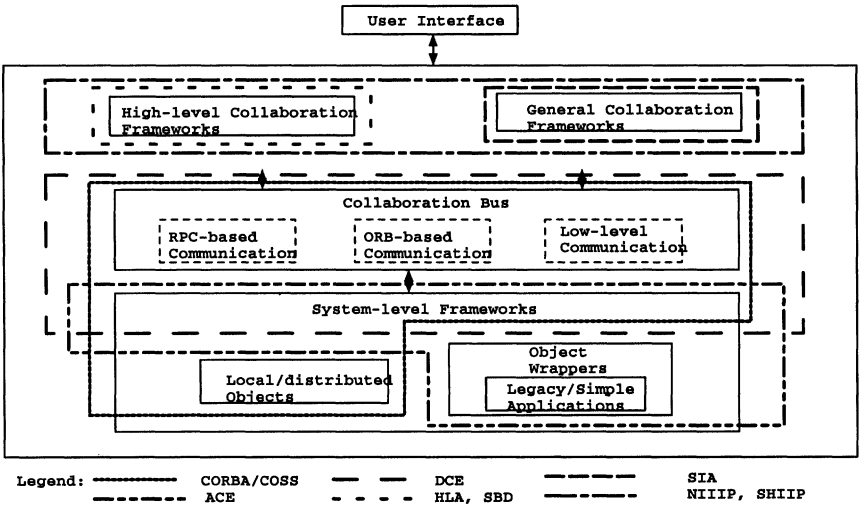

Fig 10: Frameworks and General Reference Framework

6 REFERENCES

[1] Clarke, J., Stikeleather, J., and Fingar, P. (1996) Distributed Objects for Business, Next Generation Computing, SIGS Books & Multimedia, NY.

[2] DCE DCE 1.1 documentation, available at http://www.transarc.com/afs/transarc.com/public/www/Public/Documentation/dce/1.1/index.html

[3] Goldschmidt, A. (1996) NIIIP Protocol Decomposition: an Approach to Internet Collaboration, IEEE WET ICE '96 NIIIP Position Paper.

[4] HLA (1997) Introduction to the HLA, available at http://www.dmso.mil, Simulation Interoperability WorkShop (SIW), Fall 1997.

[5] Mills, J.J., Elmasri, R., Khans, K. Miriyala, S. and Subramanian. K. (1995) The Systems Integration Architecture: an Agile Information Structure, Informations Systems Development for Decentralized Organizations, Arne Solvberg et. Al., Eds., 1995, Chapman & Hall, London.

[6] NIIIP (1996) The NIIIP Reference Architecture (Revision 6), available at http://www.niiip.org.

[7] OMG (1997) Common Object Services Specification, Revised March 1995, Updated March 1997, available at ftp://ftp.omg.org/pub/

[8] Orfali, R., Harkey, D., and Edwards, J. (1996) The Essential Distributed Objects Survival Guide, John Wiley & Sons Inc., NY.
[9] Orfali, R., Harkey, D., and Edwards, J. (1996a) The Essential Client/Server Survival Guide, John Wiley & Sons Inc., NY.
[10] SBD SBD Architecture Paper, available at http://sbdhost.parl.com/publicDocs/pubIndex.html.
[11] SHIIP (1996) SHIIP Definition Document Ver. 1.0, available at http://shiip.npo.org.
[12] Shmidt, D. ACE Documentation, available at http://www.cs.wustl.edul/schmidt/ACE-documentation.html
[13] Soley, R. M. and Stone, C. (1995) Object Management Architecture Guide, Revision 3.0, 3rd ed., John Wiley & Sons Inc., NY.
[14] Williams, D. and O'Brien, T. (1995) Software Without Borders: Applications That Collaborate, The Future of Software, Derek Leebaert, ed., MIT Press, MA.

7 BIOGRAPHY

Ramez Elmasri is a professor in the department of Computer Science and Engineering at the University of Texas at Arlington. He has worked with the Information Technology Group at ARRI (Automation & Robotics Research Institute) for several years on Systems Integration Infrastructures. He is co-author of the textbook "Fundamentals of Database Systems", and has over 60 research publications.

Daeweon Son is a graduate student in the department of Computer Science and Engineering at the University of Texas at Arlington. He recently joined the Information Technology Group at ARRI.

John J. Mills is the director of ARRI and AAMRI (Agile Aerospace Manufacturing Research Institute). He also is Fort Worth Chamber Foundation Chair in Automation & Robotics. He is a member of Industry Steering Board, Technology Enabling Agile Manufacturing, DOE and Board of the Academic Coalition for Intelligent Manufacturing Systems.