Library Requirements

Table of Contents

List of Figures

1.0 Introduction

1.1 Purpose

The RASSP library management methodology enables engineering design and technical information reuse throughout the enterprise. This methodology specifies both the process for classification of reusable design data and the requirements for the underlying library management system software. It supports automated, semi-automated, and manual processes for creation and maintenance of the classification scheme, descriptive data, and physical design objects, with a rigorous class definition process to enable data normalization over a variety of sources. Three-tiered, fully distributed data access mechanisms and methods are used to ensure support for concurrent engineering in a heterogeneous, cooperative, virtual corporation environment. Sophisticated tool interoperability capabilities, both interactive and through standards-based application program interfaces (APIs), enable several levels of integration with external libraries, tools, and the RASSP enterprise environment, and limit the development cost and schedule normally associated with point-to-point integrations. The RASSP program goals for library management include lower development costs, reduced design cycle time, and increased quality due to the reliability and availability of reusable design data.

The purpose of this document is to (1) define the minimum set of requirements for the RASSP Reuse Data Manager (RRDM), (2) define an integration strategy for reusable design data, enterprise tools, and domain-specific tools in the RASSP environment, and (3) define the overall methodology for creation and maintenance of the RASSP Reuse Design Object Classification Hierarchy (RDOCH) for the RASSP program.

1.2 Scope

The Rapid Prototyping of Application-Specific Signal Processors (RASSP) is an Advanced Research Projects Agency (ARPA)/Tri-Services program designed to advance the technology and processes used in the design, manufacture, test, and procurement of digital signal processors. The program, under the direction of Lockheed Martin Advanced Technology Laboratories (LM-ATL), is explicitly designed to encourage commercialization of the technology developed.

The overriding goal of the RASSP program is to achieve an improvement in productivity (decrease the time to market) by at least a factor of four, with similar increases in quality and decreases in overall product life cycle development costs. A key factor in accomplishing this goal is the ability to find and reuse existing design technology and data in new designs. Design elements designated as "reusable" will be referred to as *Reuse Elements* (REs) throughout this document. Examples of reuse elements include signal processing algorithms, VHDL simulation models, software modules, schematics, test vectors, design specifications, product information, and so on. REs may be created and/or used in all phases of the RASSP design process, including systems definition, architecture definition, hardware design, and software design. The RASSP Reuse Data Manager (RRDM) provides the mechanism for classifying, maintaining, finding, and retrieving REs in the RASSP enterprise environment.

This document represents the composite of the *Library Management Model for RASSP - Version 2.0* [LM-ATL, 1995a], the *RASSP Reuse Data Manager (RRDM) and Reuse Strategy Requirements Specification (Draft)* [Aspect, 1995a], the *RASSP Reuse Library Integration Strategy* [Aspect, 1995b], the *RASSP Reuse Data Manager (RRDM) and Reuse Strategy Implementation Plan (Draft)* [Aspect, 1995c], the reuse section of the *RASSP Interim Report* [LM-ATL, 1995b], and subsequent research in design reuse. Unlike the documents that preceded it, this methodology is intended to be library management tool-independent. Although the requirements documented herein are focused on the RASSP digital signal processor domain, it is hoped that the overall approach applies to library management for a much broader spectrum of engineering design and test environments.

1.3 Intended Audience

This document is intended primarily for RASSP enterprise team members, library management systems developers, integration engineers, systems and database administrators, and librarians, to define the basic requirements and methodology for library management in the RASSP environment. Designers and test engineers may also find it useful as background information.

1.4 Overview

The Rapid Prototyping of Application-Specific Signal Processors (RASSP) is an Advanced Research Projects Agency (ARPA)/Tri-Service program aimed at dramatically improving the process of design, manufacture, test, and procurement of digital signal processors. The RASSP program will deliver an integrated system called the RASSP system, which integrates the CAD tools used in the RASSP design process under a framework referred to as the enterprise framework. An *enterprise framework* provides the facilities and services necessary to integrate the automated processes of an enterprise. In the RASSP system the enterprise framework provides support for workflow management, design data management, library management, computer-supported collaborative work and remote tool access. The workflow management subsystem of the RASSP enterprise system enables a RASSP system administrator to model and enforce a particular design methodology for a project. The data management subsystem of the enterprise framework provides facilities for configuration management and control of access to design data files that may reside at various sites in a computer network. Library management in the RASSP system involves cataloging, releasing, and searching for reusable design objects. The library management subsystem of the RASSP system is called the RASSP Reuse Data Manager (RRDM).

In today's design environments, the ability of a design engineer to maximize reuse is impaired by the fact that there is no efficient way to search for reusable design objects across multiple sources, and that many sources of reusable data are uncoupled from the design environment. Mechanisms and processes for organizing reusable design information created within a design organization and for effectively sharing that data within the organization as well as with other cooperating organizations are also lacking. The requirements described in the sections to follow specify an approach for integrating the various sources of reusable design objects to provide a single source for searching for reusable design data and enable enterprise-wide sharing of reuse data. This approach consists of (1) developing a Reuse Design Object Class Hierarchy (RDOCH) that classifies the various types of design objects in the RASSP domain and models the descriptive data associated with them, and (2) developing a commercial library management system which will implement the design object class hierarchy, providing mechanisms for searching across multiple libraries in a distributed, virtual corporation environment.

2. General Description

2.1. Enterprise System Definition

As stated in section 1.4, the RASSP Reuse Data Manager (RRDM) is a fully integrated subsystem of the enterprise framework. This framework provides support for workflow management, design data management, library management, computer-supported collaborative work, and remote tool access. The Design Methodology Manager (DMM) enables a RASSP system administrator to model and enforce a particular design methodology for a project. The Enterprise Product Data Manager (EPDM) provides facilities for configuration management and control of design files that may reside at various sites in the network. Library management entails the identification, cataloging, storage, retrieval, and maintenance of reusable design information in a wide area network (WAN) distributed database environment. The RRDM provides the facilities and services required for library management in the RASSP enterprise framework.

3. RRDM Functional Requirements

3.1. Data Model Creation and Maintenance

Reuse data management in the RRDM involves management of the descriptive data for reusable design objects, the management of the physical design objects, and querying for reusable design objects that are classified by the Reuse Design Object Classification Hierarchy (RDOCH). As described in section 2.2, the ability to classify data in an object-oriented hierarchical fashion is a requirement of the RRDM. Although a pre-defined class hierarchy will be provided (*i.e.,* the standard RDOCH, described in section 5.1), the capability to modify and extend the classification scheme based on corporate business rules and processes, or due to refinement of engineering design methodology, and to facilitate incorporation of additional classes of reusable components or additional descriptive data must be provided.

The RRDM shall provide the capability to classify data hierarchically, factor common attributes into super-classes, and support inheritance of attributes at lower levels in the hierarchy, including support for static multiple inheritance. New classes may be added to the hierarchy, and existing classes may be moved within the hierarchy or deleted from it. New attributes may be added to individual classes; existing attribute characteristics may be modified and attributes may be deleted as part of the class definition process. Destructive actions, such as deleting a class, may only be performed if no instances of the class or any of its children exist, and if no other classes are related to it. Moving a class within the hierarchy (*i.e.,* changing its parent class or classes) will be permitted only in cases where the existing parent class and new parent class are empty classes. Modifying the data type for a given attribute or deleting an attribute altogether may only be done if no instance of the class has assigned a value to that particular attribute. In order to maintain consistency with the standard RDOCH, data model maintenance privileges will be limited to authorized users.

The RRDM will provide facilities for creating instances of a class in the RDOCH, modifying the values of the individual attributes of these instances, relating classes and instances of classes to one another, and associating the physical reusable components with particular instances of classes that model the related descriptive data. These facilities shall be available through the graphical user interface, through API calls, and via a batch mechanism for loading data in bulk.

Facilities to define custom data types, and to define class attributes based on these custom data types is required. These include the following, at a minimum:

- strings of varying length
- integers
- floating point numbers, including precision and rounding algorithms
- dates
- boolean values
- structures
- collections of structures
- lists

The capability to define units of measure, associate units with class attributes, define functions for unit conversion (*e.g.,* volt to millivolt, degrees Celsius to degrees

Fahrenheit, etc.), and to provide default units based on the IEC 1360-1 standard [IEC, 1994] is also required. Support for conditional and grouped attributes, valid values checking (range and enumerated values, depending on the data type), and conditional valid values shall also be provided. These facilities are necessary in order to accurately describe the rich variety of design data available in the RASSP environment.

A mechanism for specification of a complete reusable design object, and for associating that specification with a particular class in the RDOCH is also needed. Prior to release of a reusable element, given that the data has been integrated in the RRDM as a *level 2* or *level 3* integration (see section 4.2), the set of files and design objects to be released will be automatically checked against the specification to verify completeness. This specification may include:

- rules for determining whether or not the descriptive data and physical design object is complete for each class of reusable design objects
- maintenance of "state" information (*e.g.,* released, in development, obsolete)
- support for persistent collections of design objects (*e.g.,* a set, bag, ordered set)
- tool-specific encapsulation related rules

The capability to create and manage indexing information for individual class attributes shall be provided by the RRDM. Tuning mechanisms, default indexing capabilities, and related database analysis and maintenance tools shall be available to support the RRDM DBA.

Visual aids, such as a graphical representation of the classification hierarchy, shall be provided whenever feasible to assist the RRDM DBA or librarian in the class definition process. These aids will also be provided for reference to general users as they traverse the hierarchy in search of individual REs.

3.2. Query Capabilities

In order to promote the reuse of existing design information, sophisticated search, retrieval, and viewing features must be provided by the library management system. Descriptive data for REs will be maintained in the RRDM descriptive data repository, while the physical design objects may be stored in a variety of locations and formats on the enterprise network.

Physical design objects may include documents, specifications, software modules, logic symbols, VHDL simulation models, PGM graphs, schematics, netlists, advanced bills of materials, and so on. These objects may be stored in a variety of native formats (*e.g.,* Interleaf, FrameMaker, Microsoft, Word, ASCII text, TIFF Group 4, Facsimile Group 3, Postscript, or other tool-specific or standard format for documents) specific to the tool that created them, or may be in standard interchange formats. They may be maintained as physical files in a directory on the UNIX network, as text or other large objects within the descriptive data or enterprise reuse design data repository, as design objects in the product data manager, or in their native domain-specific tool environment(s).

Descriptive data (also called *meta-data*) for an individual RE must include all information required to locate, view, recreate, modify, and reuse the element itself in a new design. The search, retrieval, and viewing features of the RRDM shall provide the mechanisms necessary to enable a user to find, view, and export REs of interest, regardless of the type, physical location, or format of those particular REs. A methodology for defining the descriptive data to facilitate this process is given in section 5.1.

As stated in section 3.1, the ability to classify the data in a hierarchical classification scheme, factor common attributes into super-classes, and support inheritance of attributes and operations at lower levels in the hierarchy are requirements of the RRDM. The search and retrieval functions of the RRDM shall support this object-oriented classification methodology. RRDM users shall specify a class in the hierarchy, and have the capability to either limit the search to the specified class (*i.e.,* perform a *shallow search*), or to search the entire sub-hierarchy rooted at that class (*i.e.,* perform a *deep search*). Queries may be constructed using logical operators, such *not, and, or,* and *xor*; as well as comparison operators such as *less-than, greater-than,* and *equal-to,* depending on the data type(s) of the attributes being queried. Support for wild cards within string-valued attributes will also be provided.

In addition to the standard search capabilities, for those attributes that are defined with a set or range of valid values, queries will be checked for correctness prior to execution. Users will be given the option of selecting enumerated valid values from a list (or in the case of a valid range of values, be given the appropriate range), rather than entering the data manually. Where relationships to other classes of the RDOCH have been defined, the option to qualify a search based on attributes of the related class(es) will also be provided.

The RRDM shall provide the capability to encapsulate third-party viewing tools, (*e.g.*, word processors, spreadsheet programs, etc.), including viewers within the domain-specific tool environment from which the query was launched (depending on the level of integration- see section 4.2). Intelligent pattern matching, keyword searching, SGML and HTML searching, and general comparison capabilities for documents, image data, and other physical design objects are design goals, though not required. The RRDM shall also provide the capability to export the physical design objects for reuse by the tools that created them, or to other tools given that the data representation is compatible.

3.3. Methods

Support for three-tiered fully distributed methods on classes is required, enabling users to

- run analysis functions on a spreadsheet containing search results
- modify units of measure
- execute formulas and custom reports

in the desktop client environment (UNIX workstation, PC, Mac) rather than executing all external functions on the RRDM server. Additionally, support for distributed methods on objects is necessary to support rules-driven meta-data normalization across manufacturers. A mechanism for creating, integrating, and releasing new methods (including methods that access external databases and tools in the RASSP environment), and for handling conflicts between them (*e.g* ., when conflicting methods are released at different sites a tightly coupled federation of RRDMs - see section 3.5) will be provided.

3.4. Reuse Design Data Management

As described in section 3.1, reuse elements may consist of a virtually unlimited number of file types or objects, ranging from ASCII text to complex image information. They may be stored in a variety of formats, including those that are native to the design tools, native to specific work processing programs, or in standard formats that may be processed by several tools, such as CFI-DR [CFI, 1994] and STEP [ISO, 1993]. They may be maintained as physical files in a directory on the network, as physical objects in the either the descriptive data or enterprise reuse design data repository, as part of a version tree managed by DM 2.0, or as design objects maintained in the domain-specific tool environment.

Given that the descriptive data for an individual RE includes sufficient detail necessary to locate, view, and enable modification and reuse of the element itself, the RRDM must also manage descriptive data related to multiple versions of these design objects. Additionally, for the REs that are maintained as text or large objects by the RRDM, version management and access control are also the responsibility of the RRDM. Version management requirements are addressed in section 3.6. Authorization and user authentication requirements are given in section 3.7.

In cases where the physical design objects are maintained (1) as physical files in the network, (2) by the product data manager, or (3) in the domain-specific tool environment, the RRDM is responsible for storage of descriptive data and for providing access to that data only. The RRDM will not be held accountable for maintaining the integrity of the design files themselves, in other words.

For those cases where the physical design files are managed (1) as text objects in the descriptive data repository, (2) as binary large objects in the descriptive data repository, or (3) by the enterprise reuse design data repository, several additional issues must be addressed. These include:

- WAN client/server access to the enterprise reuse design data repository for storage and retrieval of the design objects
- the capability to maintain multiple enterprise reuse design data repository servers at one or more nodes anywhere in the WAN environment, (*i.e.,* not necessarily collocated with the descriptive data repository)
- compression/decompression of physical design data, as needed for increased network throughput

The mechanisms necessary for version management of the files maintained in the enterprise reuse design data repository and/or descriptive data repository will be provided, including basic check-in/check-out capabilities for creation and modification of those design objects. Users will also have the ability to 'undo' a given check-out action. The RRDM will provide a mechanism for maintaining consistency between the physical design objects and related descriptive data. Complete change history for all physical design objects will be maintained as a part of this process. For physical design data to be managed within RRDM, the import/export features of the GUI, APIs, and batch data management utilities must support text and binary large objects.

Within the RRDM classification hierarchy and descriptive data repository, mechanisms for relating the descriptive data objects to the REs are required. Individual instances of classes in the RDOCH may have multiple physical design objects of various formats associated with them. A single physical design object may be applicable to multiple instances of the classes of the RDOCH (*e.g.,* an individual VHDL network architecture model for a daughter card that is also described as a subset of the performance model for a particular architecture). The capability to maintain physical design objects in a folder hierarchy is required, where the related descriptive data may point to either a node (or nodes) in the hierarchy or to a specific design object(s). Multiple instances of classes in the RDOCH may point to the same node in the folder hierarchy, and a given instance may have one or more nodes in that hierarchy associated with it.

Finally, the back-up and recovery processes and procedures for the RRDM must ensure that there is no loss of data in case of an outage under reasonable operation conditions, and support standard practices for transaction logging and accounting in all cases.

3.5. Distributed Architecture

The RASSP system supports a *virtual enterprise,* where the different organizations within the enterprise may be distributed across multiple sites in a wide-area network, heterogeneous environment. As a result, the RRDM architecture must support multiple descriptive data and enterprise reuse design data stores across the WAN, enabling users to query in a directory services fashion.

Individual sites may or may not have a complete instantiation of the RASSP system, including a suite of CAD tools, the RRDM, and the product data manager. At a minimum, the REs generated at a given site will be cataloged in the RRDM at that site.

The RASSP system will support two paradigms for interaction between RRDMs at the various sites in a wide-area network -- a tightly-coupled federation of RRDMs and a loosely-coupled federation of RRDMs. All RRDMs in a *tightly-coupled federation* must maintain the same class hierarchy and class definitions. Any updates done to the class hierarchy at one site will be propagated to the other sites in the federation. The reusable design objects maintained in the RRDM at a given site s_1 may or may not be replicated at the other sites, as determined by the site librarian(s), database administrator, and corporate business processes. If a particular RE residing at a site s_1 (and not currently replicated at site s_2) is required for use in a design at s_2, it may be copied to s_2, as needed. In this case, the descriptive data associated with the reusable design object is made part of the RRDM at site s_2 as well. Queries of the descriptive data across RRDMs within a tightly-coupled federation shall be transparent to the user.

The RRDMs in a *loosely-coupled federation* do not share a common class hierarchy or common class definitions. A designer at site s_n will not be able to query across all of the RRDMs in the federation at once in this case, though the facility to query them individually will be provided. Any RE resident at site s_2 that is required for use in a design at s_1 will be replicated at s_1. Descriptive data associated with the reusable design object will necessarily be modified by the librarian at site s_1 prior to insertion to resolve any differences in the descriptive schemas between the two sites, by executing the workflow shown in Figure 2-2.

3.6. Release Management

Release Management in the RRDM involves two distinct areas of functionality: (1) the generation and maintenance of new versions of a reusable design object, including maintenance of the descriptive data about that RE, and (2) notification of the users (subscribers) of the design object about the new version.

3.6.1. Usage Lists

When identifying an existing RE for potential use in a new design, the question, *"Where has this been used previously, and how was it used in that case?"* must be answered. in the same context, it is important to note when and for what purpose a particular modification was made: was a bug fixed, or was the functionality modified to meet new requirements? In the case of the former, the modification should be propagated to any and all other designs that have incorporated the same RE (in a user controlled manner); in the latter situation, that judgment must be made on a case by case basis.

In response to these issues, individual REs will have one or more *"usage lists"* associated with them, storing information about each instance where the design object has been used. These usage lists contain information such as the site identifier, the project, contact person, an identifier for the design, the relevant date, and so on, for each design in which a particular RE is used. An expiration date or obsolescence code, indicating when a particular reference may be removed from the list, may also be added by the designer. The RASSP librarian may optionally specify upper and lower bounds for the expected life cycle of a particular usage list element which overrides the entries made by individual design engineers. Full query capabilities against the usage lists will be provided by the RRDM.

When a particular RE is modified and a new version is created, a report indicating that the RE has been changed will be sent to any designer whose name appears in the subscription list for that element, as well as to the librarian. The individual designers may then determine whether or not that same change is required in their own designs.

3.6.2. Version Management

Configuration Management (CM) in the RASSP environment is defined as management of the versioning of design objects (REs). It involves the creation, approval, and release of a particular version of a design object, as well as organizing those versions and assembling compatible versions of design objects to form a complete release of a given product. In order to minimize the incompatibilities between different CM mechanisms across the various tools that comprise the RASSP environment, a common model for configuration management has been proposed [LM-ATL, 1994b].

The majority of the features of this common model do not apply to version management of REs, as the RRDM is intended to manage *released* versions only. The mechanisms implemented for managing multiple versions of design objects in the RRDM must be compatible with those defined in this model, however. In particular, mechanisms for transferring version information between the RRDM and product data manager, and between the RRDM and domain-specific tools must be clearly defined, and consistent with this common model where possible. In other words, the RRDM should be capable of accepting version information that is modeled as per the CM conformance class of the STEP (Standard for the Exchange of Product Model Data) standard AP203 [ISO, 1993], and of producing it when exporting CM information to the product data manager or domain-specific tools, given that it is feasible to do so.

Design engineers and librarians may submit new versions of a reusable design object for inclusion in an RRDM, as needed. These new versions may correct problems that exist in older releases of the same element, or may be different implementations of or upgrades to existing versions of a particular RE. In the RRDM, multiple releases of a given RE will be assigned the same object identifier with unique revision (version) ids. The notions of *global* , *shared* , and *private workspaces* , and of *released* , *working* , and *transient* versions of individual design objects, as defined in [LM-ATL, 1994b], do not apply to the RRDM per se. All of the physical design objects managed by the RRDM will be *released* versions, by definition. For REs identified prior to release, whose physical design objects are maintained by the product data manager or domain-specific tool, CM status information may be included in the descriptive data for reference purposes.

A particular version of a reusable design object may be deleted from an RRDM with the following constraint: if the usage list for that RE is non-empty, then it is marked for deletion but will not actually be deleted until the usage list becomes empty. If a particular RE has been marked for deletion, it shall not appear in the list of results returned by any query unless that query is performed by the RRDM DBA.

3.7. Authorizations and User Authentication

As stated in [LM-ATL, 1994a], the various CAD and enterprise tools that make up the RASSP environment use diverse and incompatible models and mechanisms for handling authorization information. In order to facilitate the transfer of authorization information between tools and limit the overhead involved in maintaining this information, a common, generic approach to the authorization problem has been proposed for RASSP. This approach provides a logically consistent framework for handling authorizations in an object-oriented environment.

Given that many of the tools that will be integrated into the RASSP enterprise framework are not object-oriented, however, the following capabilities, at a minimum, must be supported:

- common authorization type definitions
- a common user mapping paradigm
- a common approach for applying authorizations to managed elements

In addition to the problem of consistent handling and interchange of authorization information, the problem of user authentications must also be addressed. Common authentication problems include password synchronization, multiple logins across tools, maintenance of multiple authentication databases, and so on. In order to address the user authentication issue for the RASSP program, the following capabilities are desirable:

- a single point of entry / maintenance point
- a single, initial authentication
- password and general authentication synchronization

- a common user description
- a project orientation capability in support of tools such as the product data manager

The approach defined in the paragraphs to follow for handling authorizations and user authentications in the RRDM are consistent with the Lockheed Martin ATL proposal, meet the design goals listed above for authorizations, and suggest a mechanism for a third-party integration tool to manage RRDM user authentications for RASSP.

3.7.1. Authorizations

As per the authorization model proposed by Lockheed Martin ATL [LM-ATL, 1994a], authorization information for the RRDM may be defined as ordered triplets $\{o_i, r_j, t_k\}$, where o_i is an authorization object hierarchy, r_j is an authorization role in an authorization role hierarchy, and t_k is an authorization type in an authorization type hierarchy. Authorization objects are organized as per the RASSP Reuse Design Object Classification Hierarchy (RDOCH) for the RRDM. An authorization role is a collection of users that have the same set of authorizations on the same set of objects. Authorization roles may by organized as a directed, acyclic graph, as described in the ATL model. In other words, one authorization role must be capable of inheriting privileges from another (*i.e.,* can manage another). An authorization type defines the type of operation that may performed on a design object. Figure 3-1 defines the authorization type hierarchy that has been adopted for the RRDM.

"Grant" authorizations, as shown in the figure, are privileges to grant an authorization to another role in the role hierarchy. The directed links between any two nodes (*e.g.,* Grant Destroy/Delete, Destroy/Delete) represent an implicit relationship between the nodes. For example, authorization for a librarian to update Assembly information implies the authorization to update Module and Substrate data, and so on, as lower levels in the hierarchy are traversed. This authorization implies the same authorization for the RRDM DBA, who is a member of the authorization role parent class. Also, update authorization implies that the librarian has read and execute privileges, as per Figure 3-1.

The RRDM will support both positive and negative authorizations. An implicitly or explicitly positive authorization $\{o_i, r_j, t_k\}$ must exist for an operation of type t_k to be performed by a user belonging to role r_j on a data object belonging to the authorization object o_i. A positive or negative authorization specified on a node n_i in an authorization hierarchy may be overridden by an authorization on node n_j, occurring at a lower level in the hierarchy. Authorization objects may be defined at the class level, object level, or on a per class attribute basis in the RRDM.

The authorization object hierarchy and authorization role hierarchy may be customized by the RASSP system administrator or DBA. The authorization type hierarchy is static, however.

Authorization data exchanged between the RRDM and other tools will be modeled using the Configuration Management conformance class of the STEP standard AP203 [ISO, 1993], to the extent possible.

3.7.2. User Authentication

Two approaches to user authentication must be supported by the RRDM in order to meet the goals stated above: (1) to permit third-party tools such as the enterprise tools or domain-specific tools to launch the RRDM, uniquely identify the user and user role, and bring up a default configuration for that user without requiring a separate login, and (2) to support the use of the RRDM in a stand-alone mode, where an external login, including username and password, is explicitly required. Special applications that may run outside the standard RRDM GUI, (*e.g.,* API-based programs, batch data synchronization processes that are part of the product data manager to RRDM integration), must also use a well-defined mechanism for communicating authentication information.

For all users and user roles, authentication information for the RRDM will be maintained in the descriptive data repository. This guarantees that regardless of the authorizations granted to a particular user, role, or application globally, and regardless of the node in the WAN at which this user or application is running, access to the descriptive data and REs for a particular RRDM is controlled by the RRDM server. For a tightly-coupled federation of RRDMs, as described in section 3.5, authorization information must be shared across the federation.

Ideally, authentication information should be maintained centrally by an authentication server. This server should check a user’s password once, on initial login to the RASSP enterprise desktop environment, and provide an encrypted "ticket" to the desktop environment that can then be used to authenticate the user to any other RASSP tool. Specification of the requirements for this authentication server is beyond the scope of this document.

A command line launch facility, allowing an external process to automatically invoke the RRDM with the required user authentication and role information shall be provided by the RRDM. By default, user authentication will be taken from the user’s environment for command line launch purposes. If the username matches user authentication data contained in the RRDM, an automatic login for that user and user role shall be performed. User identification provided on the command line will take precedence over information available in the environment. If the username given differs from that available in the environment, however, explicit password verification will be required. This feature will allow the DMM to launch the RRDM so that the user interface will be automatically configured for that user and user role combination, giving the appearance of a "single point of entry" (*i.e.,* eliminating multiple logins from the users' perspective). It will also appear that password verification and general authentication was performed by DMM (or the desktop environment). It does not support a single entry/maintenance point for authentications data, however, which requires integration with an authentication server as proposed above.

3.7.3. Access Control Lists

All physical design objects maintained in the RRDM shall have access control lists (ACLs) associated with them. These lists shall specify what actions can be performed on the design objects themselves by which users. Authorization information shall be consistent with the model described in section 3.7.1, above.

3.8. Tool Interoperability and External Process Integration

Two kinds of problems must be addressed by the RRDM in order to provide a complete solution for integration between the RRDM and other external tools: *data integration* and *control integration*. Each of these areas will be discussed in the paragraphs that follow.

3.8.1. Data Integration

Data integration is the most obvious problem to solve for the RRDM, and includes batch data import and export capabilities, programmatic access (read/write) through application programming interfaces (APIs), and on-line data import and export through the graphical user interface. Batch data management facilities are required primarily for moving large numbers of objects between applications, loading new libraries, or supporting incremental updates of existing design data. These facilities must also support data model modifications, as defined in section 3.1. API access supports the same (or similar) functionality available through the user interface, programmatically, enabling integrators to incorporate additional capabilities for data translation and manipulation, as needed. API programs may run concurrently with other programs that perform additional operations on the data, but require the use of a license, and performance issues and/or resource limitations must be considered for large transactions. On-line import and export functions facilitate "copy and paste" and "drag and drop" operations from the RRDM GUI environment to windows in which another tool is running, but the volume of data that can be exchanged in this manner is limited. These three mechanisms, provided together, can support all of the RRDM data integration requirements identified to date.

3.8.2. Control Integration

The problem of sophisticated control integration, where an external process can "drive" the RRDM user interface, or through which the RRDM can command another application to take action, is much more difficult to address. Among other things, as suggested by Julienne and Holtz [Julienne and Holtz, 1994], facilities for control integration must:

- eliminate the necessity for NxN bilateral integrations between the RRDM and external tools
- allow other applications to be coded, compiled, linked, and installed independently
- allow RRDM client applications to be replaced or rewritten without requiring modification of the other applications with which they interact
- operate in the enterprise network environment or in single user mode transparently to the user
- be standardized to the extent that other third-party developers can use them without requiring in-depth knowledge of the underlying RRDM source code

Clearly, given the integration goals for the library management system in the RASSP environment, inherent support for tool interoperability must be provided. Given the requirements listed above, a SunSoft, ToolTalk, Object Management Group (OMG) Common Object Request Broker Architecture (CORBA), and/or SunSoft, Distributed Object Management Facility (DOMF) based interprocess communications protocol (IPC) for UNIX applications, including a standardized interface definition language (IDL) compiler for language-binding elements is required. An additional OLE -compatible IPC for Windows applications is also desirable, though not required to support the current enterprise environment.

4. External Interface Requirements

4.1. User Interface Design

As stated in section 2.3, due to the diversity in roles and level of computer literacy that RRDM users may have in a given corporate or enterprise environment, the graphical user interface (GUI) of the RRDM must be highly configurable on a per user and user role basis. Each individual user must be able to use the library management system at their own level of understanding, with minimal additional training. This implies not only ease of use from a GUI standpoint, including ease in transitioning from one client platform to another (*e.g.,* UNIX/X11/Motif to Microsoft, Windows) from a user perspective, but that the implementation of the RASSP Reuse Design Object Classification Hierarchy (RDOCH) must be intuitive and limit the amount of "thrashing" a user is required to do in order to find a reusable design object.

The following set of requirements addresses the GUI style and development methodology requirements for this task. An implementation approach for the RDOCH is given in section 5.1. Note that one of the LM-ATL goals is for all enterprise and domain-specific tools developed on the program to be Common Open Software Environment (COSE) compliant. For the RRDM, meeting the OSF/Motif, Level One Certification criteria is equivalent to compliance with COSE guidelines. Development of custom widgets, such as spreadsheet widgets for comparison of class attributes or instances of specific RDOCH classes is discouraged, and use of standardized, commercial widgets is preferred when feasible.

- From a behavioral standpoint, all RRDM UNIX-based client GUI applications shall comply with the *OSF/Motif, Style Guide* , and in particular, shall meet OSF/Motif, Level One Certification criteria.
- Client applications' GUI behavior shall be consistent across UNIX platforms, including SunOS 4.1.4, Solaris 2.4, and HP-UX 9.1, at a minimum.
- Cross platform behavior (*e.g.,* Motifï to Microsoft, Windows) shall be consistent to the extent possible.
- Context-sensitive help shall be provided for all RRDM GUI applications, for each unique window, menu option, and dialog box, at a minimum.
- All primary search, results, browser, editor, and data model maintenance windows shall be configurable (and saved if desired) on a per user basis for all classes in the RDOCH.
- Precedence for automatic loading/display of class configurations shall be (1 by user, (2) by user role for the session, and (3) the default system-generated configurations.
- Default configurations shall be automatically generated for each class, and modified appropriately as class attributes, relationships, and general display-related characteristics are updated.
- Graphical representations of the RDOCH and folder hierarchies of reusable design objects, with visual cues indicating the current class, shall be provided whenever possible.

4.2. Levels of Data and Tool Integration

As described in section 2.2, one of the key requirements for library management in the RASSP environment is to integrate the multiple libraries and sources of reuse data to provide a single source for searching. Once a reusable design object is identified by searching the RRDM, that RE may be inserted into a particular CAD tool's design data space using either a manual or automated insertion process. A *manual insertion process* implies that the RRDM will provide information regarding the location and the format of the design object, and the design engineer will need to retrieve that RE, perform any necessary data translations, and insert it into the CAD tool's design data space. An *automated insertion process* implies that the RRDM will communicate with a particular CAD tool through an inter-process communication (IPC) mechanism to insert the RE into the CAD tool's design object space. Any necessary data translation will also be automatically performed.

Data translations may be required when reuse data created in a CAD tool CT-A need to be used in another CAD tool CT-B, and the two tools use incompatible data representations. Data translations are performed using standard data representations as intermediate representations. For example, a data flow from CT-A to CT-B is supported by translating the design data produced by CT-A, from the native data representation of CT-A to a representation consistent with a Standard-X, and CT-B reading data represented in Standard-X, and translating it to the native representation of CT-B. A data flow from CT-B to CT-A is made possible using the reverse process. Examples of standards that may be used for data translation include the Electronic Data Interchange Format (EDIF) [EIA, 1993], the Initial Graphics Exchange Specification (IGES) [US PRO, 1993], and the CAD Framework Initiative Design Representation (CFI-DR) [CFI, 1994].

The individual CAD tools and their associated libraries may be integrated with the RRDM in one of the three levels described below:

- *Level 1 Integration* - maintenance of descriptive data only in the library itself, with references to the physical design objects within the native tool environment
- *Level 2 Integration* - maintenance of both the descriptive data and physical design objects in the library, with the capability to view the design objects in native form (*Level 1 Integration* plus design data integration)
- *Level 3 Integration* - automated design data exchange and meta-data synchronization between the library and enterprise/design tools (*Level 2 Integration* plus tool integration)

A CAD Tool CT-A is said to have a *level 1 integration* with the RRDM if the descriptive data repository of the RRDM is populated with the descriptive data associated with the reusable design objects in the CT-A library, with references to the location and composition of the reusable design objects. This level of integration enables searching of multiple vendor libraries from the RRDM. Once the appropriate design object is identified, the design object is inserted into CT-A's environment using a manual process. Given the requisite tool encapsulation information, some design objects may also be viewed using CAD-tool-specific viewers or viewers for graphical REs maintained in standard representations, such as Postscript and Graphical Interchange Format (GIF).

Level 2 integration of a CAD tool CT-A to the RRDM subsumes level 1 integration; additionally, the reuse data repository of the RRDM is populated with the reuse data objects in CT-A's library. This enables viewing as well as semi-automated transfer of the design objects to the CAD tool environment.

A *level 3 integration* of a CAD Tool CT-A to the RRDM subsumes level 2 integration, and provides for automated insertion of reusable design objects from the RRDM to the CAD tool, bi-directional meta-data synchronization, and customized data exchange through the user interfaces of the CAD tool and RRDM, as required.

4.3. Standalone Libraries

Standalone libraries consist of reusable design objects sourced from outside vendors, government and commercial libraries available in the public domain, libraries available through various research organizations, design data available within the virtual corporation, and program-specific libraries. They may be tool-specific proprietary data formats, or they may consist of standards-based formats that can be accessed by a number of different tools, such as VHDL simulation model and C++ software libraries. The RRDM must provide the facilities for classifying, cataloging, and maintaining the rich set of reusable libraries and design data identified to date, as well as other libraries that may be identified over the course of the program.

Examples of standalone libraries of reusable design data purchased for RASSP benchmark team use from an outside vendor include the Logic Modeling SmartModel Library of full-functional VHDL simulation models and the MentorGraphics Board Process Library (licensed separately from Falcon Frameworkï). Public domain RE libraries may include web-based software libraries, such as the ARPA ASSET (Asset Source for Software Engineering Technology) libraries, USAF ESD CARDS (Central Archive for Reusable Defense Software) libraries, and NASA COSMIC (Software Technology Transfer Center) libraries, for example. Libraries sourced from the research community may include RASSP-funded VHDL simulation model libraries from the University of Cincinnati and the University of Virginia, MIT design libraries for the Benchmark 2 SAR Processor, and so on. Additional Lockheed Martin corporate libraries currently in use on the RASSP program include EPI-team MentorGraphics LMS libraries of released component data (logic symbol, geometry, and catalog entry information), supplier management information from the existing GE Consolidated Purchasing System (CPS), software libraries available from other Lockheed Martin companies and divisions, etc. Program-specific libraries may include architecture models and LMS component libraries currently under development for Benchmark 3.

Several questions must be answered prior to integration of any standalone library. Some of these are implied in the workflow given in Figure 2-2. Reliability of the reusable design objects must be verified. They must be classified in the RDOCH, and new classes and/or attributes must be created, as needed. Sizing, network location, composition, and RDOCH class-specific meta-data must also be determined. Although most of the example libraries consist of data in standard formats, there are a few, including the MentorGraphics LMS data (Board Process and EPI-team libraries) that are CAD tool specific. These must be loaded into the Mentor Graphics Falcon Framework environment and maintained as *level 1* data, rather than being maintained in the Enterprise Reuse Data Repository as *level 2* data. Standalone libraries do not qualify for *level 3* integration, by definition. A complete description for classification of REs for incorporation in the RRDM is given in section 5.1.

4.4. Enterprise System Integration

As indicated in section 2.1, the enterprise framework provides the facilities required for workflow management, design data management, library management, computer-supported collaborative work, and remote tool access for RASSP. In support of these tasks, several interfaces for batch data synchronization and real-time data exchange between the library management system and other enterprise tools are required. The following sections describe the minimal set of functional requirements for these interfaces.

4.4.1. Design Methodology Manager Integration

The Design Methodology Manager (DMM) and related project maintenance tools form the basis for the enterprise framework from an RRDM perspective. RASSP users may launch any tool in the environment from DMM (given that they have proper authorizations), or a tool may be launched by DMM during execution of a specific workflow step. This interface may be described as a relatively straightforward tool encapsulation. Users who are logged in to the DMM and require access to the RRDM may select the appropriate icon from the available choices, automatically invoking the RRDM with the requisite authentication and user role information. Appropriate user identification and role information may be supplied to the RRDM on the command line for this purpose.

Underlying network and security features required by the RRDM for this interface are provided by the RASSP enterprise environment. These features will interact at an implicit level, through the standard operating system environment, and will be essentially transparent to the operations of the RRDM and related interfaces. The primary areas of interaction will be in the network TCP/IP named services area, required by the RRDM daemon servers and clients to connect to one another in either the LAN or WAN environment, and license management.

In addition to the command line launch feature set, the interprocess communications (IPC) mechanism described in section 3.7 will support:

- alerting the RRDM that a current task was initiated as a part of a workflow step (*e.g.,* creation of a new RE)
- notifying DMM that a specific task has been completed

The RRDM IPC shall include generic message types that provide a mechanism for exchange of alert/notification information between the RRDM and DMM. A mechanism shall also be provided to manage the status of current tasks initiated through the DMM on a per user basis, to facilitate task management across multiple sessions/logins.

4.4.2. Product Data Manager Integration

The Enterprise Product Data Manager (EPDM) provides the facilities necessary to support configuration management and control of design data that may reside at various sites in the enterprise network while these designs are considered active. For the purposes of this document, as stated in section 2.4, the EPDM is implemented in the RASSP enterprise framework in the Intergraph Data Manager (DM 2.0) tool. The information managed by the EPDM may include all product (or project) related descriptive and physical design data -- "data that is part of, or directly tied to, the development of a signal processor" [Intergraph, 1994]. The EPDM is responsible for controlling access to, maintaining version control of, vaulting, and archiving the product-related descriptive and physical design data developed in the RASSP enterprise environment.

Data managed by the EPDM may include any and all design objects that are required for execution of a given process step at any point in the life cycle of the product. Although product data management (PDM) systems have traditionally limited data management to the information required to reproduce/rebuild a particular product, this definition reflects a more comprehensive approach. As such, there has been some confusion on the program to date regarding where the line is drawn between library management and design data management. Distinguishing characteristics may include:

- released design data vs. work in progress
- design objects that are complete entities but would normally be considered a subset of the overall product, and would not typically be "sold separately" (*e.g.,* a released requirements specification, daughter card, processor module, algorithm library, network architecture model, etc.)
- supporting data vs. design objects required for use in a process step (*e.g.,* supporting libraries for the MentorGraphics Falcon Frameworkï tools may include logic symbols, geometries, simulation models, and so on, that are not called out in the workflow but are required for successful operation of the tools)

Design data maintained by DM 2.0 is characterized by the project/product in which it is used. Reusable design objects are described in the RRDM by their characteristics. For example, attributes for a VHDL model managed by the RRDM include external and internal timing and value resolution, model type, modeling language, etc., as per the RASSP taxonomy for VHDL models [LM-ATL, 1995c]. In the product data manager, it may be described at a very high level as a performance model for a particular architecture on a specific project, with attributes and references that may or may not be migrated to the RRDM.

In addition to the characteristics listed above, additional information that may be described by the RRDM and viewable on-line (but would not be considered product information) includes:

- product information (vendor data sheets, application notes, engineering alert notices, etc.)
- specifications, standards, and other traditional references
- standalone reference data (*e.g.,* the Logic Modeling SmartModel library, government software libraries, etc.)
- supplier information

Given this distinction, the exchange of data between the RRDM and DM 2.0 is limited to descriptive data and related design objects <u>at the level at which they are managed by DM 2.0</u> unless there is manual intervention by the RRDM librarian. If , for instance, the schematic for a particular design is identified as reusable, and it is managed in DM 2.0 as a single compressed binary large object (BLOB) rather than as a distinct set of "intelligent" design objects (*i.e.,* net list, native Design Architectï schematic, user documentation files, specification, etc.), it will be managed in the RRDM as a BLOB unless the RRDM librarian intervenes and executes a set of well-defined steps to convert it to an "intelligent" reusable design object.

Three usage scenarios have been defined by the enterprise team to describe the interaction between the RRDM and DM 2.0. The first of these is fairly straightforward, and reflects the requirement for designers to be able to reuse existing design data in a new design. At any point during the design process, when a designer sees the need to incorporate a new element (*e.g.,* function, module, algorithm, etc.), they will be provided with the capability to search for similar REs in the RRDM, find one or more that fit their needs (if they exist), and copy the descriptive data and physical design objects into the DM 2.0 (or CAD tool) environment for incorporation in their design.

The other two scenarios are related to creation of REs from design information that is currently active in the DM 2.0 environment. On completion of a design, if the designer has identified one or more elements of that design as candidates for reuse, once the proper approvals have been obtained and data migration processes (if any) have been completed, the capability to copy the descriptive data and physical design objects from DM 2.0 to the RRDM is required. Secondly, at any time during the design process, prior to release of the entire project, the designer may identify an element as reusable. In this case, the descriptive data may be copied to the RRDM, but management of the physical design objects is retained by DM 2.0. As a result, to make this information available to other designers in the enterprise network, references to the physical design data must be created in the RRDM, with the notation that this RE is considered to be work in progress rather than a fully released element (unless this portion of the design has actually been released).

On completion of the design (*i.e.*, release to production), the actual physical data will be copied from DM 2.0 to the RRDM, and DM 2.0 references will be removed.

In summary, the following functions are required for integration between the EPDM (DM 2.0) and RRDM:

- enable creation of new/updated REs in the RRDM from descriptive data maintained in the EPDM at any phase of the design process

    1. For an active design, descriptive data for a new reuse element will be inserted in the RRDM, including references to the various design objects, documentation, test data, and so on, but the physical design objects will remain under DM 2.0 control.
    2. On completion of a design, both the descriptive data and physical design objects will be copied to the RRDM.

- enable transfer of existing design information from the RRDM to DM 2.0 for use in new and/or modified designs, including descriptive data and related design files
- provide for the maintenance of consistent access controls and version information between the two tools

Due to the fact that a high degree of user interaction will be required in order to complete the process of creating an individual reuse element and of incorporating reuse elements into current designs, the following minimal capabilities must be provided by the integration (both through the GUI and via API calls):

- the capability to launch the RRDM from within DM 2.0 and search for REs based on available descriptive data (RDOCH class and configuration selection, and search window population shall be automatic wherever possible)
- the capability to select one or more descriptive data objects from within the RRDM and drag and drop (export) them into the DM 2.0 environment, causing both the descriptive data and related physical design objects to be made available to DM 2.0 for incorporation in the user's workspace
- the capability within DM 2.0 to import descriptive data and physical design objects from the RRDM, resolve attribute discrepancies and complete required descriptive information through interaction with the user, and incorporate the RE into the current workspace
- the capability to identify selected physical design information in DM 2.0, determine (or solicit from the user) the RDOCH target class, create baseline RRDM descriptive data for that RE based on information available in DM 2.0, and drag and drop both the template descriptive data and physical design object(s) into the RRDM environment
- the capability to automatically invoke the RRDM editor from DM 2.0 (if it is not currently displayed), and display the partial instance of the RDOCH class to the user for completion and insertion, copying all reference information for the physical design data and/or the actual data objects to the enterprise reuse design data

repository, as required
- the capability to maintain mapping information within the RRDM to indicate which RRDM attributes correspond to which DM 2.0 attributes on a per class basis (including an indication of "ownership" and mechanisms for conflict resolution encountered during automated data synchronization)

The interprocess communications mechanisms necessary to fulfill these requirements (for the RRDM) are described in section 3.7.

4.5. Domain-Specific Tool Integration

4.5.1 Candidate Tools

Of the tools that have been encapsulated in the enterprise framework for Build 3, several have been identified as potential candidates for domain-specific (level 3) integration. These include:

- MentorGraphics Library Management System (LMS) and Design Architectï
- JRS Research’s NetSyn
- Alta Groupï of Cadence Design Systems’ Signal Processing WorkSystem (SPWï)

Additional Mentor Graphics Corporation (MGC) tools, MATLAB from The Math Works, Inc., MCCI’s autocode generation tools, Savantage’s SavanSysï , the LogicVision DFT tools, and others are also under consideration. Additional tool integrations may be identified over the course of the program.

To date, the requirements for these tool integrations, with the exception of MentorGraphics LMS and Design Architectï , remain to be defined.

4.5.2 Mentor Graphics Library Management System and Design Architectï

There are a number of overriding design goals for this integration, specific to the job function being performed (namely, the CAE/CAD/CAM librarian and hardware designer). From the librarian’s perspective, the integration should provide the facilities needed to locate and maximize reuse of existing component and library data (logic symbols, geometries, simulation models, catalog entries, etc.), sourcing the information from anywhere in the enterprise network. It should also minimize the necessity for manual intervention, which can introduce error when generating library data. The capability to create and modify REs identified by the librarian must be supported. Finally, this integration should provide as much automation as possible in synchronizing the RRDM descriptive data repository with the LMS catalog and libraries across the enterprise. For the hardware designer, the ability to search for and easily incorporate REs in a new design is the primary objective. The designer should also be provided with the facilities necessary to create new REs and maintain existing library data, as needed.

The primary functions required for this integration include:

- bi-directional data exchange, enabling (1) creation of new and/or updated LMS catalog files during development of a part within *lms_libr* , either on-line through the RRDM graphical user interface or nightly, in a batch mode if the descriptive data repository has been updated, based on the descriptive data contents, and (2) maintenance of hardware design view descriptive data and related physical design objects (or references to them) in the RRDM based on LMS catalog files on release of a part from *lms_libr*
- the capability to launch the RRDM from within *da_lms* and search for REs based on (1) the part instance (logic symbol selected in the schematic) and (2) the *lms_user* Part Selector menu item
- the capability to place part instances in *da_lms* based on descriptive data and REs maintained in the RRDM
- the capability to launch the RRDM from within *lms_libr* and search for applicable REs using the LMS library selector mechanism
- the capability to create/modify LMS catalog files based on data contained in the RRDM, and automatically open the appropriate catalog file within *lms_libr* for edit
- the capability within the RRDM to maintain mapping information indicating which RRDM attributes correspond to which LMS catalog file properties on a per class basis, including identification of "ownership" to resolve conflicts in data synchronization
- the capability to maintain additional mapping information indicating which *da_lms* properties correspond to which RRDM attributes for the purposes of launching a search in the RRDM based on an instance in a *da_lms* sheet
- the capability to create and modify a shopping list of parts in the RRDM and transfer the catalog references to *lms_user* (creating the shopping list in *da_lms*) from which parts may be selected during the design process

The use of standard OSF/Motifï features, such as drag and drop, for ease of use is recommended whenever feasible. Additional requirements may be defined based on user feedback.

5. Reuse Design Object Classification Hierarchy Development

5.1. Methodology

Program goals related to development of the RASSP Reuse Design Object Classification Hierarchy (RDOCH) include:

- creation of a model that is general enough such that it fits most corporate environments or can be easily adapted to do so
- ensuring that the data model, data dictionaries, and related specifications for reusable design object sets provide complete, consistent, and correct classification of data regardless of the source or format
- creation of a model based on a methodology and data dictionaries for individual classes that can become the basis for an industry standard

An important criterion for the RDOCH is that it should be general enough to be adopted as an industry standard. As a standard hierarchy, it should lend itself to extensions without requiring destructive changes. Destructive changes include deletion of classes in the hierarchy, deletion of attributes in the hierarchy, and moving classes within the hierarchy. Adding new classes to the hierarchy and adding attributes to the existing classes will be allowed. These restrictions are necessary so that future releases of the RDOCH will be upwardly compatible with previous releases (i.e., previous integrations of CAD tool libraries with the RDOCH will continue to be valid). The model must be generic, or integration of additional classes over time will require numerous modifications, and it will not be acceptable to the corporate community that may benefit from its development. Consistent, standardized classification of data is a necessity, or search results may be unpredictable.

Specific goals for the RDOCH itself are:

- to provide a framework for classifying design objects in the RASSP domain
- to capture all descriptive data relevant for design objects in the RASSP domain using an object-oriented paradigm
- to provide a framework for searching for reusable design objects using an object-oriented paradigm

The user may execute a search based on the descriptive data, locate the appropriate design object, and then access that design object for incorporation in a new or revised design. The design data associated with a design object may include a data sheet, a simulation model, a schematic, etc.

Additionally, the descriptive data repository developed under the RASSP program must be populated sufficiently to demonstrate its utility. Ideally, the resultant library demonstration should also show how any implementation of the classification hierarchy can be customized to fit a variety of corporate environments that may use a different mix of tools and COTS source data, or that may produce a wider variety of products.

To achieve these goals we have extended the methodology described in International Electrotechnical Commission Draft International Standard 1360-1 for classifying electric components [IEC, 1994]. The salient features of this methodology can be summarized, as follows:

- select the type of reuse data to model
- source data from applicable tool vendors, government and commercially available libraries, RASSP enterprise and demonstration/beta site team members, and so on
- determine the set of potential descriptive data attributes to be managed (based on RASSP requirements, user input, analysis of products output by the tools, analysis of the library data required to feed the workflows and tools, etc.)
- for each potential attribute, specify the semantics and valid values
- determine where this descriptive data fits in the RDOCH, creating new classes only in cases where there are either a high number of unique attributes or several significant, unique attributes
- create a complete specification for both the descriptive data dictionary and reuse design objects

Specifications for each class of reusable design data consist of:

- preliminary and final classification trees
- a list of standards and/or references used to design the class attributes
- source data dictionaries (attribute lists and definitions available from vendors, government organizations, benchmark team members, etc.)
- spreadsheet comparison of potential attributes and valid values
- final, standardized attribute list and definitions, with rationale
- legal/valid enumerated values or valid range for each attribute, appropriate unit of measure, and sort order
- source data object list across suppliers/sources, and set of target vendors and libraries
- classification rules, vendor-specific data normalization rules
- definition of a complete design object (e.g., VHDL models may have related documentation, test vectors, etc.)
- location and management methodology (including level of integration) for the physical design objects
- the set of tools to be encapsulated in order to view the design objects, and encapsulation information (location, invocation syntax, and so on)
- sizing projections for both the physical design objects and descriptive data
- modification frequency, methodology, and access limitations and/or restrictions

A complete RASSP Reuse Design Object Classification Hierarchy (RDOCH), data dictionaries defining the descriptive data for each class, and specifications for the reusable design objects described by each class will be developed over the course of the program and published for review. A preliminary high-level version of the RDOCH is given in Figure 5-1.

5.2. Data Views and Relationships

The implementation model for the RDOCH will support multiple views of the data managed by the RRDM. These may include (but not be limited to):

- Descriptive Data View
- Design Data Views (i.e., views that are applicable to multiple high-level RDOCH classes, such as Simulation Model, Design For Test, and Documentation views)
- Hardware Design Data Views (e.g., BSDL File, Geometry, LMS Catalog, Logic Symbol, Package, Pin Property, and Schematic)
- Supplier/Manufacturer Views (e.g., corporate aliases, addresses, certification information, commodity tracking, and so on)

The Descriptive Data View of the RRDM is provided by the RDOCH . It models the descriptive data about the design objects that may be created/used within a RASSP design environment. The interior nodes of the hierarchy are abstract classes, and the leaf nodes are concrete classes that may be populated with real design objects. Examples of concrete classes include "Microprocessor", "Demodulation Primitive", "FIR Filter",

Figure 5-1. The RASSP Reuse Design Object Classification Hierarchy (Preliminary)

"Instrumentation Interface Module", etc. Relationships between classes may include references from the Simulation Model class to the Author (Contact/Employee) of the model, Component Supplier, and Design Tool and configuration used to generate the model, for example, or from the Component to the Supplier/Manufacturer, Package, and Product Information available. The hierarchy and specification of these relationships will be further refined and developed by analyzing system use and user feedback over the course of the program.

Views provide an alternate way of accessing design data in the RRDM. For example, a user may go to an on-line data book and use the table of contents to locate an appropriate data sheet. The RRDM will also maintain limited descriptive data to locate the design objects within each individual view.

5.3. Domain-Specific Libraries

Numerous standalone and tool-specific libraries of reusable design data have been identified for use on the program. These are grouped by high-level classification area, below. Note that many of the libraries include data that is considered to be design view specific, such as the Logic Symbol information available in the Mentor Graphics Board Process Library, providing supporting information required for the design process.

Reusable *Hardware Design* elements may be identified at many levels in the design process, from low level component package dimension information to the schematic for an entire rack of equipment designed to fulfill program requirements. At the top level, the Hardware Design class contains the following subclasses:

- Assembly - for Module and Substrate information
- Component - containing descriptive data for Electrical Component, Electro-Mechanical Component, and Mechanical Component
- System - for higher-level design information, such as card cage wiring

Additionally, there are Design View (Simulation Model, DFT, and Documentation) and Hardware Design View (BSDL File, Geometry, LMS Catalog, Logic Symbol, Package, Pin Property, and Schematic) classes related to the Hardware Design class (see Figure 5-1). To date, the descriptive data for the Component classes has been sourced from Aspect Development, Inc. This information includes high-level descriptions of the components as well as low-level timing information and package dimensions for a variety of manufacturers. Manufacturer information, package information (including HPGL drawings), and product documentation (TIFF images of data book and data sheet information, by manufacturer), are provided as a part of this library. Simulation model libraries related to the Hardware Design classes are available through a number of sources. These include the following standalone libraries:

- Logic Modeling SmartModel Library
- University of Cincinnati
- University of Virginia
- LM-ATL Benchmark Team library

The Simulation Model class of the RDOCH is designed based on the *RASSP VHDL Modeling Taxonomy and Terminology* [RASSP, 1995] paper given at the Second Annual RASSP Conference. Additional examples of libraries of reusable hardware design view and reusable design object data identified for use on the program include:

- Mentor Graphics Board Process Library - logic symbol, geometry, LMS catalog, and simulation model data for use with the Mentor Graphics tools
- Mentor Graphics Falcon Framework internal libraries
- LM-ATL EPI team library of released parts (logic symbol and geometry classes, primarily)
- LM-ATL RASSP team library of released designs (from prior benchmark efforts)

Architecture Design classes are modeled based on the RASSP Model Year Architecture (MYA) methodology developed by the LM-ATL team [LM-ATL, 1994d]. The primary external library identified to date for integration in this class is the JRS Research Laboratories, Inc. architecture selection library.

Algorithm Design classes are modeled based on a convergence of the MCCI specification [MCCI, 1995], Q003 Specification [AT&T, 1993], discussions with members of the Alta Group RASSP team and their library reference documentation, discussions with members of the JRS RASSP team, and internal LM-ATL RASSP team analysis. Primary sources for algorithm design data include:

- Alta Group of Cadence Design Systems, Inc. Signal Processing WorkSystem libraries of reusable function blocks for DSP design
- JRS NetSyn libraries of signal processing primitives
- MCCI libraries of application graphs, partition graphs, and primitives
- The Math Works, Inc. (MATLAB) libraries of reusable algorithms
- LM-ATL internal algorithm libraries

Software Design classes are modeled after information published by the Reuse Library Interoperability Group (RIG), Lockheed Martin internal software development libraries, and information provided by various government organizations. This area, targeted for development in early 1996, is the least stable at this time, and will be explored in depth as the program continues.

Other areas that are currently under development include the Design For Test classes (which may include libraries from LogicVision, Savantage, and others), supplier management classes (currently based on the GE Consolidated Purchasing System and EPI and ATL practices), and Specification and Standard classes.

Appendix A

Definitions, Acronyms, and Abbreviations

A/D Analog to Digital

ABBET A Broad Based Environment for Test

ACL Access Control List

ADP Automatic Document Processing

AFAL Air Force Avionics Laboratory

AIM Application Interpreted Model

AIRST Advanced Infrared Search and Track

ALC Ascent Logic Corporation

AMPLE Advanced Multi-Purpose Language

ANSI American National Standards Institute

AP Application Protocol

API Application Programming Interface

ARL Army Research Laboratory

ARM Application Reference Model

ARPA Advanced Research Projects Agency

ASCII American Standard Code for Information Interchange

ASEM Application-Specific Electronic Module

ASIC Application-Specific Integrated Circuit

ASSET Asset Source for Software Engineering Technology

AST Architecture Selection Toolset

ASW Anti-Submarine Warfare

AT&T American Telephone & Telegraph

ATAG ABBET Technical Advisory Group

ATD Advanced Technology Demonstration

ATE Automatic Test Equipment

ATL Advanced Technology Laboratories

ATM Automatic Test Methods

ATPG Automatic Test Pattern Generation

ATR Automatic Target Recognition

ATS	Automatic Test Sets
BAA	Broad Agency Announcement
BDE	Block Diagram Editor
BDT	Berkeley Design Technology, Inc.
BFM	Bus Functional Model
BIST	Built-In Self Test
BIT	Built-In Test
BITE	Built-In Test Equipment
BLOB	Binary Large Object
BONeS	Block-Oriented Network Simulator
BSDL	Boundary Scan Definition Language
BTD	Benchmark Technical Description
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CALS	Computer-Aided Logistics Support
CAM	Computer-Aided Manufacturing
CAPE	Computer-Aided Parametric Estimating
CARDS	USAF ESD Central Archive for Reusable Defense Software
CASE	Computer-Aided Software Engineering
CAT	Computer-Aided Test

CDEM Customizable Debugging Environment for Multiprocessors

CDMS Computer-Aided Design Data Management System

CDR Critical Design Review

CE Concurrent Engineering

CECOM Communication-Electronic Command

CEENSS Continuous Electronics Enhancement Using Simulatable Specifications

CFAR Constant False Alarm Rate

CFG Control Flow Graph

CFI CAD Framework Initiative

CFPI Catalog File Procedural Interface

CGS Code Generation Tool

CHPC Center for High-Performance Computing

CIS Component Information System

CLMS Component and Library Management System

CM Configuration Management

CMU Carnegie Mellon University

CND Cannot Duplicate

CNI Communication, Navigation, Identification

CORBA Common Object Request Broker Architecture

COSE Common Open Software Environment

COSMIC NASA’s Software Technology Transfer Center

COTS Commercial Off-The-Shelf

CP Command Program

CPS Consolidated Purchasing System

CPU Central Processing Unit

CSCI Computer Software Configuration Items

CT CAD Tool

DB Database

DBA Database Administrator

DCE OSFï Distributed Computing Environment

DDMS Design Data Management System

DEMVAL Demonstration/Validation

DFD Data Flow Diagram

DFG Data Flow Graph

DFT Design For Testability

DICE DARPA Initiative in Concurrent Engineering

DM Data Management

DMA Defense Mapping Agency

DMA Direct Memory Access

DMM Design Methodology Manager

DoD Department of Defense

DOE Distributed Object Environment

DOM CFI Standard for Design Object Management

DOMF Distributed Object Management Facility

DR CFI Standard for Design Representation

DRAM Dynamic Random Access Memory

DSP Digital Signal Processor

DT&E Development Test and Evaluation

ECM Electronic Countermeasures

EDA Electronic Design Automation

EDB Electronic Data Book

EDIF Electronic Data Interchange Format

EDM Enterprise Desktop Manager

EIA Electronic Industry Association

EIF Enterprise Integration Framework

EINet Enterprise Integration Network

EPDM Enterprise Product Data Manager

EPI Engineering Process Improvement

ER Entity-Relationship

ESS Environmental Stress Screening

EW Electronic Warfare

Express-G Information Modeling Language - Graphical Representation

FDDI Fiber Distributed Data Interface

FLIR Forward-Looking Infrared

FMEA Failure Modes and Effects Analysis

FMECA Failure Modes and Effects Criticality Analysis

FPGA Field Programmable Gate Array

FSED Full-Scale Engineering Development

GBR Ground Based Radar

GDB GNU Debugger

GE General Electric

GEDAE Graphical Entry Distributed Application Environment

GES Government Electronic Systems

GFE Government-Furnished Equipment

GFI Government-Furnished Information

GFLOPS Billion Floating-Point Operations per Second

GIF Graphical Interchange Format

GIP Graph Instantiation Parameter

GOTS Government Off-The-Shelf

GRAIL Graph Translator

GRED Graphical Editor

GrTT Graph Translation Tool

GUI Graphical User Interface

GW Gateway

HCI	Human-Computer Interface
HDI	High-Density Interconnect
HDL	Hardware Description Language
HI-TEA	High-Level Test Strategy and Economics Advisor
HOL	High-Order Language
HSIM	Heterogeneous Simulation Interoperability
HTML	Hyper Text Markup Language
HW	Hardware
I/O	Inputs/Outputs
IC	Integrated Circuit
ICAM	Integrated Computer-Aided Manufacturing
ICD	Interface Control Document
ICNI	Integrated Communication, Navigation, Identification
IDAS	Integrated Design and Assessment
IDDq	Quiescent Source to Drain Current Test
IDEF0	ICAM Definition Methodology for Functional Modeling
IDEF1x	ICAM Definition Methodology for Data Modeling
IDEF3	ICAM Definition Methodology for Process Description Modeling (standard workflow graphical representation format)
IDL	Interface Definition Language
IEEE	Institute of Electrical and Electronics Engineers
IGES	Initial Graphics Exchange Specification
ILS	Integrated Logistics Support

IPC Inter-Process Communications

IPDT Integrated Product Development Team

IPPD Integrated Product/Process Development

IR Infrared

IRST Infrared Search and Track

ISA Instruction Set Architecture

ISO International Standards Organization

ITC CFI Standard for Intertool Communication

JCALS Joint Computer-Aided Logistics Support

JIAWD Joint Integrated Avionics Working Group

JRS JRS Research Laboratories, Inc.

JTAG Joint Test Action Group (IEEE-1149)

JTIDS Joint Tactical Information Distribution System

LAN Local Area Network

LCC Life Cycle Cost

LM Lockheed Martin Corporation

LM-ATL Lockheed Martin Advanced Technology Laboratories

LMG Logic Modeling Group, Synopsis, Inc.

LMS Library Management System

LOCST LSSD On-Chip Self Test

LRM Line Replacement Module

LRU	Line Replacement Unit
LSA	Logistics Support Analysis
LV	LogicVision Software, Inc.
MANTEC	Manufacturing Technology
MATLAB	Matrix Laboratory (software tools for matrix computation)
MCC	Microelectronics and Computer Technology Corporation
MCCI	Management Communications and Control, Inc.
MCM	Multi-Chip Assembly
MCM	Multi-Chip Module
MFBIT	Multi-Feature Bayesian Intelligent Tracker
MFLOPS	Million Floating-Point Operations per Second
MGC	Mentor Graphics Corporation
MIMD	Multiple Instruction, Multiple Data
MIT	Massachusetts Institute of Technology
MMIC	Monolithic Microwave Integrated Circuit
MOE	Measure of Effectiveness
MOS	Messaging Object Service
MOU	Memorandum of Understanding
MPID	MIMD Primitive Interface Description
MPIDGen	MIMD Primitive Interface Description (MPID) Generator
MSDA	Multi-Chip Systems Design Advisor

MSI Management Services, Inc.

MTBF Mean Time Between Failures

MTI Moving Target Indicator

MYA Model Year Architecture

NASA National Aeronautics and Space Administration

NEP Node Execution Parameter

NFS Network File System

NGCR Next-Generation Computing Resources

NIIIP National Industrial Information Infrastructure Protocols

NIS Network Information Service

NIST National Institute of Standards and Technology

NRL Naval Research Laboratory

NSS Network Synthesis System

OCR Optical Character Recognition

ODBC Open Database Connectivity

OLE Object Linking and Embedding

OMG Object Management Group

ONR Office of Naval Research

OO Object-Oriented

OOA Object-Oriented Analysis

OR Object-Relational

OS Operating System

OSF Open Software Foundation, Inc.

OSI Open Systems Interconnect

PC Personal Computer

PCA Printed Circuit Assembly

PCB Printed Circuit Board

PDCM Product Data Control Module

PDES Product Data Exchange using STEP

PDM Product Data Manager

PDR Preliminary Design Review

PDP Product Data Package

PDT Product Development Team

PE Processing Element

PGM/DFL Parallel Graph Method / Data Flow Language

PGSE Programming Graph Simulation Environment

PI Procedural Interface

PIDGen Primitive Interface Description Generation

PLD Programmable Logic Device

PMB Performance Measurement Baseline

PML Process Modeling Language

PMO Program Management Office

POSIX Portable Operating System Interface

PreAMP Pre-Competitive Advanced Manufacturing Process

PRICE Parametric Review of Information for Costing and Evaluation

PRR Program Readiness Review

PWA Printed Wiring Assembly

PWB Printed Wiring Board

QA Quality Assurance

R&M Reliability and Maintainability

RAM Random Access Memory

RAM/ILS Reliability, Availability, Maintainability / Integrated Logistics Support

RASSP Rapid Prototyping of Application-Specific Signal Processors

RCS Revision Control System

RDBMS Relational Database Management System

RDD-100 System Design CAD Tool - Ascent Logic

RDOCH RASSP Reuse Design Object Classification Hierarchy

RE Reuse Element

RI Rockwell International

RIG Reuse Library Interoperability Group

RL Rome Laboratory

RMWG RASSP Methodology Working Group

RPC Remote Procedure Call

RRDM RASSP Reuse Data Manager

RSS Reusable Software System

RTL Register Transfer Level

RTM Requirements Traceability Matrix

RTOK Retest OK

SA Systems Administrator

SAL Signal Processing Algorithm Library

SCRA South Carolina Research Authority

SDR System Design Review

SE Switching Element

SEM Standard Electronic Module

SEMP Systems Engineering Management Plan

SEMS Systems Engineering Management Schedule

SES Systems Engineering Station

SGML Standard Generalized Markup Language (ISO 8879)

SIMD Single Instruction, Multiple Data

SMT Surface Mount Technology

SNR Signal to Noise Ratio

SOW Statement of Work

SP Signal Processor

SPGN Signal Processing Graph Notation

SPWï Signal Processing WorkSystem

SQL Software Query Language

SRAM Static Random Access Memory

SRR System Requirements Review

SRS System Requirements Specification

STA Science and Technology Associates

STEP Standard for The Exchange of Product Data

STS Self Test Services

STUMPS Self Test Using MISR and Parallel SRSG

SVI Standard Virtual Interface

SW Software

SWAP Size, Weight, and Power

T&E Test and Evaluation

T&M Bus Test and Maintenance Bus

T/R Transmit/Receive

TAP Test Access Port

TBD To Be Determined

TBR To Be Resolved

TCL/TK Tool Command Language / Tool Kit

TCP/IP Transmission Control Protocol / Internet Protocol

TES CFI Standard for Tool Encapsulation

TI Texas Instruments

TIGER Testability Insertion and Guidance Experts for RASSP

TK Toolkit for X11 Window System

TPM Target Primitive Map

TPS Test Program Set

TRD Test Requirements Document

TRP Technology Reinvestment Program

TRSL Test Requirements Specification Language

TSD Test Strategy Diagram

USAF United States Air Force

UUT Unit Under Test

VHDL VHSIC Hardware Description Language

VHF Very High Frequency

VHSIC Very High Speed Integrated Circuits

VLSI Very Large Scale Integration

VPS Virtual Prototyping System

VTEST Virtual Test

WAN Wide Area Network

WAVES IEEE Standard for Waveform & Vector Exchange

WBS Work Breakdown Structure

WEC Westinghouse Electric Corporation

WL Wright-Patterson Laboratory

Appendix B
References

[AL, 1992] Armstrong Laboratory, *IDEF3 Process Description Capture Method Report,* AL-TR-1992-0057, Wright Patterson Air Force Base, Ohio, 1992.

[Alta, 1995a] Alta Group of Cadence Design Systems, Inc. , *Signal Processing WorkSystem Communications Library Reference* , Foster City, California, 1995.

[Alta, 1995b] Alta Group of Cadence Design Systems, Inc. , *Signal Processing WorkSystem DSP Library Reference* , Foster City, California, 1995.

[Alta, 1995c] Alta Group of Cadence Design Systems, Inc. , *Signal Processing WorkSystem Radar Library Reference* , Foster City, California, 1995.

[Alta, 1995d] Alta Group of Cadence Design Systems, Inc. , *Signal Processing WorkSystem Interactive Simulation Library* ï *Reference* , Foster City, California, 1995.

[Aspect, 1994] Aspect Development, Inc., *CIS 2.0 Users Guide* , Mountain View, California, 1994.

[Aspect, 1995a] Aspect Development, Inc., *RASSP Reuse Data Manager (RRDM) and Reuse Strategy Requirements Specification (Draft*), Mountain View, California, 1995.

[Aspect, 1995b] Aspect Development, Inc., "RASSP Reuse Library Integration Strategy" (Presentation given 3/9/95 at LM-ATL), Mountain View, California, 1995.

[Aspect, 1995c] Aspect Development, Inc., *RASSP Reuse Data Manager (RRDM) and Reuse Strategy Implementation Plan (Draft*), Mountain View, California, 1995.

[AT&T, 1993] AT&T Federal Systems Advanced Technology, *AN/UYS-2 SEM E Graph Primitives Specification Library - Floating Point, Revision 3.0*, Greensboro, North Carolina, 1993.

[CFI, 1994] CAD Framework Initiative, Inc., *Design Representation Programming Interface*, Austin, Texas, 1994.

[Connell and Shafer, 1995] John Connell and Linda Shafer, *Object-Oriented Rapid Prototyping*, Prentice Hall PTR, Englewood Cliffs, New Jersey, 1995.

[EIA, 1993] Electronics Industries Association, *Electronic Design Interchange Format Version 3 0 0,* Washington D.C., 1993.

[IEC, 1994] International Electrotechnical Commission, *Standard Data Element Types with Associated Classification Scheme for Electric Components -- Part I: Definitions, Principles and Methods*, Draft International Standard 1360-1, Netherlands, September 1994.

[Intergraph, 1993] Intergraph Corporation, *Design Methodology Manager -- Users Guide,* Huntsville, Alabama, 1993.

[Intergraph, 1994] Intergraph Corporation, *System Requirements Specification for the Enterprise Product Data Manager of the Rapid Prototyping of Application-Specific Signal Processors Project*, SED-IO632/94, Huntsville, Alabama, 1994.

[Intergraph, 1995] Intergraph Corporation, *DM/Manager -- Users Guide*, Huntsville, Alabama, 1995.

[ISO, 1993] International Standards Organization, *Product Data Representation and Exchange: Overview and Fundamental Principles*, ISO 10303-1, Fairfax, Virginia: U.S. Product Data Association, 1993.

[Julienne and Holtz, 1994] Astrid M. Julienne and Brian Holtz, *ToolTalk & Open Protocols - Inter-Application Communication*, SunSoft Press, Prentice Hall PTR, Englewood Cliffs, New Jersey, 1994.

[LM-ATL, 1993] Martin Marietta Advanced Technology Laboratories, *Rapid Prototyping of Application-Specific Signal Processors (RASSP) CAD System Description - Version 1.0 (CDRL A007),* Moorestown, New Jersey, 1993.

[LM-ATL, 1994a] Martin Marietta Advanced Technology Laboratories, "The Authorization Model for the RASSP System", Moorestown, New Jersey, 1994.

[LM-ATL, 1994b] Martin Marietta Advanced Technology Laboratories, "The Configuration Management Model for the RASSP System", Moorestown, New Jersey, 1994.

[LM-ATL, 1994c] Martin Marietta Advanced Technology Laboratories, *Rapid Prototyping of Application-Specific Signal Processors (RASSP) Methodology Overview - Version 1.0,* Moorestown, New Jersey, 1994.

[LM-ATL, 1994d] Martin Marietta Advanced Technology Laboratories, *Rapid Prototyping of Application-Specific Signal Processors (RASSP) Model Year Architecture Working Document - Version 1.0,* Moorestown, New Jersey, 1994.

[LM-ATL, 1995a] Lockheed Martin Advanced Technology Laboratories, "The Library Management Model for the RASSP System, Version 2", Camden, New Jersey, 1995.

[LM-ATL, 1995b] Lockheed Martin Advanced Technology Laboratories, *RASSP Second Annual Interim Technical Report (CDRL A002),* Camden, New Jersey, 1995.

[MCCI, 1995] Management Communications and Control, Inc., *RASSP Domain Primitive Library Specification (Preliminary*), Arlington, Virginia, 1995.

[Martin and Odell, 1995] James Martin and James J. Odell, *Object-Oriented Methods: A Foundation* , PTR Prentice Hall, Englewood Cliffs, New Jersey, 1995.

[Mentor Graphics, 1992a] Mentor Graphics, *AMPLE Users Manual, Version 8.2,* Wilsonville, Oregon, 1992.

[Mentor Graphics, 1992b] Mentor Graphics, *AMPLE Reference Manual, Version 8.2,* Wilsonville, Oregon, 1992.

[Mentor Graphics, 1993a] Mentor Graphics, *Common User Interface Users Manual, Version 8.2,* Wilsonville, Oregon, 1993.

[Mentor Graphics, 1993a] Mentor Graphics, *Common User Interface Reference Manual, Version 8.2,* Wilsonville, Oregon, 1993.

[Mentor Graphics, 1994a] Mentor Graphics, *Design Architect Reference Manual,* Wilsonville, Oregon, 1994.

[Mentor Graphics, 1994b] Mentor Graphics, *Library Management System -- Users and Reference Manual for Designers,* Wilsonville, Oregon, 1994.

[Mentor Graphics, 1994c] Mentor Graphics, *Library Management System -- Users and Reference Manual for Librarians,* Wilsonville, Oregon, 1994.

[Mentor Graphics, 1994d] Mentor Graphics, *Catalog File Procedural Interface Manual, Version 8.4,* Wilsonville, Oregon, 1994.

[Mentor Graphics, 1995] Mentor Graphics, *Board Process Library Specification, Release A.3 (Final Review Draft),* Wilsonville, Oregon, 1995.

[OSFï , 1993] Open Software Foundation, *OSF/Motif ï Style Guide, Revision 1.2* , Prentice Hall PTR, Englewood Cliffs, New Jersey, 1993.

[RASSP, 1995] C. Hein, T. Carpenter, P. Kalutiewicz, and V. Madisetti, "RASSP VHDL Modeling Terminology and Taxonomy - Revision 1.0", in *Proceedings, 2nd Annual RASSP Conference* , Arlington, Virginia, 1995.

[Rosenberry et al, 1992] Ward Rosenberry, David Kenney, and Gerry Fisher, *Understanding DCE* , O'Reilly & Associates, Inc., Sebastopol, California, 1992.

[Rumbaugh et al, 1991] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, and William Lorensen, *Object-Oriented Modeling and Design* , Prentice Hall PTR, Englewood Cliffs, New Jersey, 1991.

[SCRA, 1994] South Carolina Research Authority, *RASSP PCA Manufacturing Interface Definition* , North Charleston, South Carolina, 1994.

[Synopsys, 1994] Logic Modeling Group, Synopsys, Inc., *IC Manufacturer to MCM Designer Die Information Exchange (DIE) Format Reference Manual* , Version 1.0, Milpitas, California, 1994.