U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 4755

National PDES Testbed
Report Series

Issues and Recommendations for a STEP Application Protocol Framework

NATIONAL

TESTBED

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 4755

National PDES Testbed

NATIONAL TESTBED

Issues and Recommendations for a STEP Application Protocol Framework

Thomas R. Kramer
Mark E. Palmer
Allison Barnard Feeney

U.S. DEPARTMENT OF COMMERCE
Robert A. Mosbacher,
Secretary of Commerce

National Institute of Standards and Technology
John W. Lyons, Director

January 17, 1992

DEPARTMENT OF COMMERCE
UNITED STATES OF AMERICA

Disclaimer

No approval or endorsement of any commercial product by the National Institute of Standards and Technology is intended or implied.

Acknowledgments

The National PDES Testbed is funded by the Computer-aided Acquisition and Logistic Support (CALS) Office of the Department of Defense.

Partial funding for the work described in this paper was provided to Catholic University by the National Institute of Standards and Technology under cooperative agreement Number 70NANB9H0923.

Executive Summary

National and international groups of technical experts are developing the next generation product information exchange standards, called the Standard for the Exchange of Product Model Data (STEP). STEP is being developed within Sub-Committee 4 (SC4) of the Technical Committee 184 of the International Organization for Standardization. The STEP project is using the application protocol (AP) methodology for defining the information requirements of industry and for specifying standards for representing these requirements.

Chief among the many elements of an AP are the Application Reference Model (ARM) and the Application Interpreted Model (AIM). The ARM is a model of the information needed by the application written in the generally accepted terminology of the application. The AIM is an information model written in the EXPRESS information modeling language which uses entities from STEP resource models to represent the same information. The AIM may be implemented in software systems to do the tasks of the application. Thus, APs are the means by which the STEP standards are put into use.

In May 1990, SC4 requested member countries to select their top priorities for STEP AP projects from eighteen proposals. The recommended criteria for this selection were: is it feasible to realize the AP within one year, does the AP meet an existing industrial need, and is it feasible to implement the AP in commercial software systems. SC4 used the results of this survey to establish the first five AP projects. These five projects have produced draft APs, which exist as documents called STEP “Parts” and cover the applications: explicit draughting, associative draughting, configuration controlled design, boundary representation, and surface representation.

The Problem

The initial scopes and requirements of these first AP projects were defined in isolation of each other, without a detailed analysis of how these APs fit together within STEP, and without an assessment of the cumulative functionality and coverage provided by these APs. This process was acceptable for initiating the first prototype projects, but is not appropriate for defining subsequent AP projects. Since April 1991, there have been some discussions between AP projects on common requirements and terminology, but these discussions have not involved all AP projects simultaneously and have not examined long range planning issues.

There is ample evidence that the current situation needs improvement. There is overlap among the first five APs, but the APs do not form a coordinated set. AP developers have had difficulty defining the components of their APs.

Before the AP methodology is expanded to address more complex problems, and before any more AP projects are initiated by SC4, the fundamental methods for developing and testing APs must be completed and proven. Only after these techniques are stable and an appropriate framework for planning and implementing the development of APs has been established, will the STEP community be properly equipped to standardize APs for complex information domains.

The Report

This report provides a description of application protocols, a summary of AP development issues (supplemented by discussions and case studies in the appendices), a listing of the relevant documentation and issues papers, and recommendations for resolving the identified issues. This

includes recommendations on functional requirements for an AP framework, elements of AP structure, coordination of AP development, and AP classification, plus others. This report provides a baseline study for subsequent work on an AP Framework by members of the STEP community.

Principal Recommendations

This report recommends a broader definition of AP integration, i.e., that AP integration must include the evaluation of APs' scope, requirements, and ARM, possible refinements to the rules for developing AIMs, and the allocation of additional resources for the coordination and long term planning of AP projects.

AP Framework

To assure the rational development of APs, the STEP project requires an AP Framework for defining, planning, and managing AP projects. This Framework must provide:

1. criteria for defining overlaps and boundaries among APs
2. criteria for defining AP integration requirements
3. classification methods for APs
4. criteria for prioritizing AP projects
5. viable implementation strategies
6. methods for management of AP development
7. guidance for development of individual APs
8. a structure for building APs

AP Structure

An application interpreted construct (AIC) is a logical grouping of concepts that is shared by two or more AIMS. AICs did not exist during the development of the five existing draft STEP APs. AICs should be used in building APs. AICs provide modular building blocks for APs and will help make APs interoperable.

Coordination of AP Development

The coordination of AP development should be improved by establishing an AP Information Base and having some coordinating agency provide guidance on AP development, ensure that STEP policies and procedures are followed, study the effectiveness of the policies and procedures, and suggest changes, where needed. The Information Base should include, for example:

1. configuration managed copies of AP documents
2. registries of AP and AIC project memberships
3. various indexes to the contents of APs
4. classification information regarding APs
5. a registry of software certified as conforming

AP Classification

APs may be classified using an N-space. A 3-space in which the coordinate axes are type of product data, product class, and tasks supported may be suitable. A second classification method, dependency graph classification, deserves further investigation.

Contents

Figures

1 Introduction

National and international groups of technical experts are engaged in developing the next generation product information exchange standards, called the Standard for the Exchange of Product Model Data (STEP). The objective of STEP is to enable product data sharing and transfer between different computer systems and environments. STEP is being developed within Subcommittee 4 (SC4) of the ISO Technical Committee 184 (TC184). The STEP project is using the application protocol (AP) methodology for defining application information requirements and for specifying the STEP constructs for representing these requirements.

The application protocol methodology is fundamental to the STEP project for the following reasons. The AP methodology:

1. Provides the means to define industry requirements and to ensure that these requirements are fulfilled by STEP standards.

2. Provides the means of extending STEP to address new application requirements in a consistent manner.

3. Provides the means to validate the application protocol and to ensure that implementations are testable.

4. Provides the STEP vendor a specification that can be used in developing useful and reliable software products.

5. Provides the basis for conformance testing of STEP implementations.

6. Provides the end-user with the means to ensure procurement of conforming implementations of STEP.

To assure the rational development of APs, the STEP project requires a framework for defining, planning, and managing AP projects. This framework must provide a structure with which to classify APs, define AP scopes, identify integration requirements, and accommodate overlaps and interfaces between application protocols. In this paper, the authors analyze the issues pertaining to an AP framework and give recommendations for such framework.

1.1 Background

The STEP project was initiated in 1985 with only a general description of its intended scope. The scope included all product data necessary to completely define any product for all applications over the product's entire life cycle. This product data included geometry, topology, tolerances, features and even production process information [Brauner, pages 11 - 20]. After delivering the first draft of STEP, SC4 recognized that the practical implementation for a particular application of STEP would not require the use of every STEP entity and that the application context will constrain the STEP entities selected. Based on this understanding, the STEP project adopted the application protocol methodology in June 1989. Guidelines for developing APs have been drafted and are being used by AP developers [Palmer1].

In May 1990, SC4 requested member countries to select their top three priorities for STEP AP projects from eighteen proposals. The recommended criteria for this selection were: is it feasible to realize the AP within one year, does the AP meet an existing industrial need, is it feasible that the resources needed by the AP can be provided in the first edition of STEP, and is it feasible to implement the AP. SC4 used the results of this survey to establish the first five AP projects.

The first edition of STEP, divided into sections called “Parts”, will document the resources and mechanisms for defining and testing a physical file implementation of at least one STEP application protocol. STEP will include Parts documenting the EXPRESS language, the physical file structure, conformance testing, a set of application-independent information models called the “integrated resources models”, and the application protocols for “Explicit Draughting” and “Configuration Controlled Design”. Additional APs may be included if their inclusion does not affect the delivery of the minimum set of Parts for the first edition of STEP and if the “integrated resources” support their needs.

1.2 Current Situation

The five AP projects established by SC4 are STEP Parts 201 through 205 and are at various stages of completion. The initial scopes and requirements of these first AP projects were defined in isolation of each other and without a detailed analysis of how these APs fit together within STEP. This process was acceptable for initiating the first prototype projects, but should not be continued for defining any subsequent AP projects. Since April 1991, there have been some discussions between AP projects on common requirements and terminology, but these discussions have not examined long range planning issues.

There is ample evidence that the current situation needs improvement. There is excessive overlap in the five APs which are STEP Parts. Initial AP developers have had difficulty defining scopes for APs.

The set of Parts designated for the first edition of STEP have been and should be used as a baseline for testing and refining the underlying methodologies. Before the AP methodology is expanded to address more complex problems, and before any more AP projects are initiated by SC4, the fundamental methods for developing and testing APs must be completed and proven. Only after these techniques are stable and an appropriate framework for planning and implementing the development of APs has been established, will the STEP community be properly equipped to standardize APs for complex information domains.

1.3 Scope, Methodology, and Framework

The scope of an AP gives its domain of discourse: the type of data being handled, and the purposes served by the data. The scope of each AP must be rational, and the scopes of STEP APs taken as a group must cover industrial needs without excessive duplication. Providing this coverage will require answers to questions such as:

1. How many application protocols will be required?
2. What types of application protocols will be required - how can they be classified?
3. How will these application protocols relate to each other?

A methodology is a body of methods, rules, and supporting techniques which may be applied to some domain of discourse. A technically sound methodology for STEP APs is required to set the structure of APs, define boundaries of APs, and to assure the cost-effective construction of APs. This methodology already exists in part [Palmer1], but does not provide sufficient rules for defining boundaries of AP scopes or instructions for long range planning for suites of APs.

A framework is a set of policies, procedures, and organizational arrangements. The ISO STEP project requires a framework to support the planning, integration, and possible optimization of AP projects. Some policies and procedures are already in place, but a fully-developed and comprehensive AP framework is not.

1.4 About This Report

The target audience of this report is individuals already familiar with STEP and the basic concepts of application protocols. The report is intended for use in the STEP community and in the National PDES Testbed.

The central questions addressed in this report are:

1. What methods should be used for defining the scope and boundaries of APs and for dividing a broad area of activity into APs?
2. How should overlaps and interfaces among APs be defined and accommodated in STEP?
3. What are the requirements for a framework for planning and implementing STEP APs?

In preparing this report, the authors:

1. contacted key players for input,
2. reviewed all available papers in the domain of STEP on the subject,
3. conducted a set of case studies on AP scope definition, and
4. conducted a case study of common geometry requirements in the five STEP AP projects which are draft STEP Parts.

The results of these investigations were used to develop the recommendations in section 3. That section proposes recommendations on AP scoping, AP classification, elements of AP structure, and elements of an AP framework.

This report was written in the summer of 1991 and reflects the situation at that time. The issues with which this report deals continued to evolve, and the development of an AP framework proceeded during the fall and into the winter. Thus, some aspects of the report may not be up-to-date as of January 1992, the month when the report was cleared for publication.

2 Application Protocols

2.1 Function of Application Protocols

A fundamental concept of STEP is the definition of application protocols as the mechanism for specifying implementation requirements and for ensuring reliable information communication. An application protocol is a standard that defines the context and scope for the use of product data and specifies the interpretation of the standardized integrated resources in that context to satisfy an industrial need. Additionally, an AP enumerates the conformance requirements and test purposes from which its abstract test suite is derived.

The scope of an AP is defined by the type of product, the supported stages in the life cycle of the product, the required types of product data, the supported uses of the product data, and the disciplines that use the product data. The design of application protocols permits nesting of protocols and the reuse of application interpreted constructs to ensure consistent implementations and sharing of relevant data among applications.

2.2 Components of an Application Protocol

The four major components of a STEP AP are:

1. scope and application activity model;
2. information requirements and the corresponding application reference model;
3. application interpreted model that specifies the use of the integrated resource constructs to represent the application information, and
4. conformance requirements and test purposes.

The table of contents for a STEP AP is listed in Figure 1. Detailed descriptions of the components of an AP and the process for developing an AP are available in referenced documents. This section summarizes information from [Palmer1].

2.2.1 Scope and Application Activity Model (AAM)

The scope clause defines the domain of the AP and summarizes the functionality and data that are accommodated by the AP. A description of the functionality and data that are specifically outside the scope of the application may also be defined to clarify the domain. This clause may also describe any defined associations with other APs.

The scope of an AP is defined by the type of product, the supported stages in the life cycle of the product, the required types of product data, the supported uses of the product data, and the disciplines that use the product data. The scope clause is based on an application activity model (AAM). The AAM is used to establish understanding of the application activities, processes, and data flows and to get agreement on them. The AAM also describes the relevance and roles of the required concepts or data.

2.2.2 Information Requirements and Application Reference Model

The application reference model (ARM) formally describes the information content, structure, and constraints of the application domain. The ARM provides the basis for specifying and verifying the information requirements of the AP. The information requirements are organized by the different product types, life cycle phases, or application views supported by the AP.

Foreword
Introduction
1 Scope
2 Normative References
3 Definitions
4 Information Requirements
    4.1 Construct Definitions and Assertions
    4.2 Global Assertions
5 Application Interpreted Model
    5.1 Mapping Table
    5.2 AIM EXPRESS Short Form
6 Conformance Requirements and Test Purposes
    6.1 Conformance Requirements
    6.2 Conformance Test Group Structure
    6.3 Conformance Test Purposes

Annexes
A AIM EXPRESS Long Form (required and normative)
B AIM Short Names (required and normative)
C PICS (Protocol Implementation Conformance Statement) proforma (required and normative)
D Implementation Specific Requirements (required and normative)
E Application Activity Model (required and informative)
    E.1 AAM Definitions
    E.2 AAM Diagrams
F Application Reference Model (required and informative)
    F.1 Units of Functionality
        F.1.1 UOF Definitions
        F.1.2 UOF and ARM Correspondence
    F.2 ARM Diagrams
G AIM EXPRESS-G (required and informative)
H Application Protocol Usage Guide (optional and informative)
J Technical Discussions (optional and informative)
K Bibliography (optional and informative)
L Resource Entity Definitional References
Index

Figure 1. Table of Contents of a STEP Application Protocol

2.2.3 Application Interpreted Model (AIM)

The application interpreted model (AIM) specifies how the integrated resources are used to satisfy the requirements documented in the ARM. The AIM is defined in the EXPRESS language and is constructed from the resource constructs using EXPRESS mechanisms defined in ISO 10303-11. These mechanisms allow for the direct use of integrated resource constructs and/or their refinement. Each refinement arises out of the scope and requirements supplied by the particular application.

2.2.4 Conformance Requirements and Test Purposes

This clause describes the conformance requirements, completeness requirements, and test purposes for conformance testing. Test purposes with related objectives are organized into test groups. Each AP has a corresponding abstract test suite which is derived from the conformance requirements and test purposes.

2.3 Development of an Application Protocol

Each component of an AP proceeds through three basic steps:

1. define the requirements and evaluation criteria for the component,
2. develop the component, and
3. exercise the criteria to evaluate the component.

Each step in the development process builds upon the precision and documentation of the previous step.

The first phase of developing an AP is the definition of its context, scope, and information requirements. Definition of the scope and information requirements begins with the formulation of a concise statement of the application context and functional requirements for the AP. This statement must include the product data application(s) targeted for the AP and the intended usage of the product data within the application(s). The detailed scoping and information requirements definition proceed from this statement. As these become more detailed, the scoping statement prepared at the beginning of the scoping phase is refined to correspond.

Scope definition is refined via the development of an Application Activity Model. The AAM describes the use of the product data within the application domain, usually with a process modeling technique such as IDEF0. During this analysis example parts and usage scenarios from the application domain are documented. These usage examples are extremely valuable in defining the scope and in subsequent validation testing of the ARM and the AIM.

When the detailed scope and general information requirements have been defined, the information domain of the AP is described by the use of the application reference model. The ARM is developed using a standard information modeling technique, e.g., IDEF1X, NIAM, or EXPRESS. Each application information requirement deemed in scope must be expressed in the ARM. Conversely, each element of the ARM must satisfy a documented need of the application. The ARM should be sufficiently detailed to describe fully the data requirements of the application domain.

A basic mechanism for modularizing the scope of an AP into manageable components is to define units of functionality (UOFs) within the context of the ARM. A UOF is a collection of entities, attributes, and relationships that conveys one or more well-defined concepts within the context of the ARM. An ARM is not required to be subdivided into UOFs or even to contain any UOFs, but it is anticipated that most ARMs will contain conceptual UOFs and ARM developers will identify them.

The AIM is developed by selecting and constraining constructs from the integrated resources to convey the concepts and information requirements of the ARM. The process of developing the AIM includes ensuring consistency of STEP data representations across APs. APs that have common functional requirements should specify the same constructs from integrated resources for representing these requirements in their AIMs. When two or more APs have equivalent UOFs in their ARMs it is expected that an Application Interpreted Construct (AIC), or possibly a number of AICs, will be included in the AIM of each AP to provide the information defined by the UOF in the ARM. It may be necessary for a new AIC to be created. The developers of an AP are not expected to develop AICs or even necessarily to identify the need for an AIC if one does not exist, as this requires examining other APs. The responsibility for identifying the need for a new AIC lies with the people who handle integration of the AP into STEP. Responsibility for developing AICs is still an open issue.

An integrated resources interpretation report is written to summarize the rationale with which the AIM was derived from the ARM and to document any specializations of integrated resource constructs and any candidates for the AIC Schema Library. During the development of an AIM, constructs may not be available in the integrated resources to satisfy some portion of the ARM data requirements. In this event, constructs may be filtered into the integrated resources after thorough analysis by WG4.

The conformance requirements and test purposes are derived from three areas: the Scope clause, the ARM, and the AIM. A test group for each independent concept or entity of the ARM (an entity which may exist without a parent entity) is developed. Each test purpose identifies the required AIM constructs for representing the specified ARM concept. All test purposes are traceable to a documented requirement.

The development and validation of a STEP AP is an iterative process of progressive detailing and validation testing. Each step in this process provides feedback for the next version of the AP.

3 Recommendations on Methodology and Framework

3.1 Current AP Methodology and Framework

The concept of APs was introduced to the STEP project in January 1990 and formally adopted in July 1990. The AP methodology and the procedures for developing APs are still evolving. There are a number of AP development issues to be resolved. During the past year, the required contents of an AP have been modified to conform to the requirements for ISO standards and to resolve initial problems identified in the first AP projects.

The first APs designated as STEP Parts (201[Haas1], 202[McKee1], 203[Gilbert1], 204[Weick2], 205[Evensen1]) were developed as individual projects to meet separate sets of industrial requirements. The AP projects teams appear to have worked independently of each other, except for the coordination between 201 and 202 and between 204 and 205 described in section C.3. All subsequent AP projects should be developed with broader interaction and coordination.

The integration of a set of APs must begin with the review of their scopes and requirements and include possible integration of their ARMs. The process of integrating APs would be improved by requiring all ARMs to be defined using the same information modeling language, preferably EXPRESS. This would not preclude the initial development of an ARM with the modeling language best suited to the application domain or application experts. Rather, these initial ARMs would then be translated into EXPRESS prior to submission to the integration process.

Although the AP methodology has been incorporated into the STEP project, there is currently no framework for planning the rational development of APs. Several documents use the term “framework” or discuss application protocols ([Danner1],[Kirkley1], [Shaw2]) but none of these meets the functional requirements for an AP Framework given in the next section. The “AP Guidelines” [Palmer1] meet some of the functional requirements of an AP Framework. They are considered an integral part of the Framework given in section 3.3. The STEP project requires a framework for defining, planning, and managing AP projects. This framework must provide a structure with which to classify APs, define AP scopes and boundaries, identify integration requirements, specify required relationships between APs, and accommodate overlaps and interfaces between APs. Additionally, this requires a strategy for managing the evolution of the STEP parts and for minimizing the impact of changes that will be required.

3.2 Functional Requirements for an AP Framework

This section presents the functional requirements for an AP Framework.

1. Provide Criteria for Defining Overlaps and Boundaries Among APs - The Framework should provide a common mechanism for clearly defining the boundaries of AP scopes, how APs should overlap, and how to nest APs.

2. Provide Criteria for Defining AP Integration Requirements - The Framework should provide criteria for identifying AP integration requirements.

3. Provide Classification Methods for APs - The Framework should provide classification methods useful for defining and assessing the existing, planned and proposed APs and for managing AP development.

4. Provide Criteria for Prioritizing AP Projects - The Framework should provide criteria for prioritizing AP projects.

5. Support Viable Implementation Strategies - The Framework should include methods that support effective implementation strategies for APs.

6. Facilitate Management of AP Development - The Framework should facilitate the planning and management of STEP AP projects and possible coordination with companion standards or specifications. Among the activities here should be providing a central source of information about APs and providing guidance for the composition, working methods, and staffing of AP teams.

7. Provide Guidance for Development of Individual APs - The Framework should provide guidance for the development of APs. This is already being done under the leadership of WG4 and should be coordinated with other activities in the Framework.

8. Provide a Structure for Building APs - The Framework should provide a structure for building APs which defines the elementary units from which an AP is built and specifies how they can be combined. Some elements of structure are defined in the AP Guidelines [Palmer1], but more detail is required. The structure itself will limit the range of options for dealing with overlaps and boundaries. The structure should establish a building block approach under which new APs may be added in an efficient manner.

In addition to being complete and technically sound, a proposed AP Framework must be acceptable to the STEP community and software vendors or it will be of little value.

3.3 Proposed Methodology and Framework

This section provides specific recommendations for the AP methodology and Framework, based on the authors' preferred resolution of the issues. In order to have the proposal be brief, full discussions of the issues are not included here, but are deferred to section 4. Some of the rationale for this proposal stems from the case studies reported in appendixes C and D, particularly sections C.1.2.3, C.1.3.3, and D.6.

3.3.1 AIM Structure

The basic issue of AIM structure is: what are the building blocks of APs and how can they be combined? AIM structure issues are discussed in more detail in section 4.2.3. AIMs are important because they are what is implemented in software systems. AIM structure is important because AIM structure drives the structure of implementation software. Moreover, AIM structure largely determines how APs can be made to work together (the interoperability issue) and how new APs can be built using old ones (the extensibility issue). The structure proposed here is intended to provided for interoperability and extensibility of APs.

3.3.1.1 Atoms

Atoms are the smallest statement-like unit of AIM structure. Specifically, an atom is one of the following from the EXPRESS Reference Manual [Spiby]: constant-decl, entity-block, function-block, procedure-block, rule-block, type-decl.

Over the suite of all APs, no two atoms should have the same name, no two atoms should have identical function, and atoms with redundant function should be minimized.

Atoms not found in integrated resources may be created in the process of developing an AP. Under current policy, atoms created this way which are not subtypes or specializations of atoms in integrated resources must be migrated into integrated resources. This issue is discussed in section 4.2.5.3. The policy requires further thought.

3.3.1.2 Application Interpreted Constructs

An application interpreted construct (AIC) is a logical grouping of concepts that is shared by two or more AIMs. Atoms may be combined to form AICs. AICs take the form of EXPRESS schemas. AICs are discussed in more detail in section 4.2.3.2.

Over all of STEP, if an atom is defined in two or more AICs, the definitions must be identical. A stronger requirement that no two AICs may contain the same atom may be desirable. If the stronger version is adopted, any atom used in two or more AICs would be placed in an AIC itself, and the AICs that use it would reference the new AIC.

3.3.1.3 AIMs

AIMs may be defined using atoms, AICs, and other AIMs. At the extremes, an AIM might be composed of all atoms or all AIMs. Section 4.2.3.3 gives further discussion.

The AIM EXPRESS schemas, usage guides, and test suites of an AP should not provide any normative information about externally defined components (AIMs, AICs, or atoms in them); that information will already exist separately (or be prepared separately, concurrently). Complete normative information should, however, be provided for dealing with atoms defined in the AIM of the AP.

AIMs should not be built to mimic a data structure specific to some software system, unless that approach is already used in a majority of systems handling data of the type concerned. The case studies indicate that some AIMs may be tailored to a small group of existing systems. Physical file scoping requirements in Part 204 (boundary representation) stand out in this regard.

3.3.1.4 Functional Levels

Functional levels, as discussed in section 4.2.3.6, should be accommodated in the AP Framework. An example of functional levels is provided by Part 204, the boundary representation AP. This is discussed in the case studies, sections C.3.4 and C.3.5. The idea of a functional level is that a core group of related constructs would permit some minimum functionality in an AP at the lowest level. By having more constructs in higher levels of the AP, more functionality would be achieved.

The current version of EXPRESS appears to be adequate for implementing functional levels. The specific EXPRESS capability most needed is the means to use one schema in another, either by reference or by inclusion. If EXPRESS is changed, the need to implement functional levels should be kept in mind.

A section should be added to the AP guidelines specifying how functional levels are to be handled. It is desirable that higher levels should be compatible with lower levels.

3.3.2 AP Implementation

3.3.2.1 Conforming Implementations

A software system which is able to process product data (in physical files or databases) in the manner prescribed in an AP will be called a conforming implementation of the AP. The official ISO definition from [ISO] is given in section B.1. A conforming implementation may require a schema (or schemas) as input or it may embody a specific schema (or schemas) - i.e. the schema may be hard-coded in the software.

It may be desirable to develop and apply conformance criteria for AICs, as well. This suggestion has not been fully examined but should be given further consideration.

3.3.2.2 Implementation Requirements

Conforming implementations must implement all of the substructure (atoms, AICs, AIMs) as specified in the AP. The semantics implemented will be those of the AIM EXPRESS schema, including the text accompanying the schema.

Output from a conforming implementation must satisfy all constraints which are part of the AIM being implemented.

Conforming implementations are encouraged but not required to implement constraint checking for input. It is difficult to make input checking a requirement because it is not always preferred to specify what a system should do in case of an error in the input.

3.3.2.3 Conformance Certification

The STEP community needs to put some type of conformance certification system in place for certifying that software systems implement APs properly. The type of arrangements currently used in other standards arenas may be suitable: have conformance testing services (for-profit or non-profit) and a central agency (or several such agencies, worldwide) which certifies the testing services.

3.3.3 Coordination of AP Development

Additional resources from the STEP community should be assigned to the coordinated development of APs. The functions listed below are required. Some are already being carried out; others are not.

3.3.3.1 Maintain an Information Base

An AP Information Base should be maintained. It should contain copies of APs and basic information about AP development. It should also be able to provide answers to questions like: What are all the APs that use cartesian_point? What are all the APs related to mechanical products? What are the closest matches among existing entities in all APs to a proposed new entity? To avoid duplication of effort, it would be desirable for the AP Information Base to be integrated with a broader STEP information base that covered the integrated resources, as well, and possibly other STEP information. The AP Information Base should be easily usable by and accessible to AP developers.

The AP Information Base should include, for example:

1. configuration managed copies of all APs, AICs, and AP project documents. AP projects should include those proposed to SC4, SC4 approved, SC4 not approved, and related but not proposed to SC4 - e.g., Dutch Road Building AP.

2. registries of AP and AIC project memberships.

3. various indexes - e.g., an alphabetic index to all atoms used in any AP or resource, a subject index to APs, a title index to AICs

4. classification information regarding APs

5. a registry of software which has been certified as being conforming

3.3.3.2 Guiding AP Development

Some coordinating agency should:

1. Provide guidance to ensure that the overall pattern of AP development is rational and meets the needs of the STEP community.

2. Ensure that STEP AP projects adhere to the SC4-approved policies and procedures for AP and AIC development.

3. Study the effectiveness of policies and procedures being used for AP and AIC development and implement or make recommendations for changes, where needed.

3.3.4 Guidance for Overall Development of APs and AICs

AP and AIC development should combine both bottom-up and top-down methods. Most of the push behind AP development should be bottom-up, normally from a group of product producers or consumers who need standards for a specific range of products. In the case of AICs, the AP Integration Project should take the lead in determining needs and in development. Procedures for developing, approving and changing AICs should differ from those used for APs, by requiring broader input from the STEP community, for example.

Each AP should be a separate STEP Part, provided that all functional levels for a given type of data should be in the same Part, whether the functional levels are given in one AP or several. This is current policy and seems appropriate.

How AICs are packaged in STEP Parts will require further study. In some cases a set of small interrelated AICs might form a suitable Part.

3.3.5 AP Classification

3.3.5.1 N-Space Classification

It is feasible to use a 3-space (an N-space with N=3) for AP classification. A prototype set of axes for classifying APs is: type of product data, product structure, and tasks supported. These are presented in detail below. Different levels of abstraction should be applied when the classification is used for STEP AP project planning, industry planning of APs for types of products, or intra/inter-organizational information planning. The notion of using N-space classification and a description of various axes which might be used are discussed in more detail in section 4.2.2 and appendix F.

The identification of an AP within this classification may be supplemented with a listing of the engineering stages supported and the discipline views accommodated. Engineering stages and product life cycle, as described in section 4.2.2, will usually be accounted for adequately by tasks supported. Data is required to do tasks. Similarly, the type of data will be more relevant than the discipline view.

Axis 1: Type of Product Data

Shape: 2D, 3D (wireframe, surface, solid (levels of CSG and B-Rep), form features)
Shape tolerances
Physical properties: material, surface roughness, surface treatment
Functional, programmatic, or performance characteristics (e.g., kinematics, loading)
Product structure: assemblage, connectivity
Product configuration
Production process
Product life cycle support
Representation specialization (e.g., draughting data)
Visual presentation
Specialized representations for analysis (e.g., finite element analysis data)

Axis 2: Product Structure

Part
Component: an identified member of an assembly or system
Assembly: an aggregation of parts and/or components for specific purpose(s). Assembly definition includes location and orientation. Assemblies may be permanent assemblies (e.g., welded), temporary assemblies (e.g., bolted), or articulated assemblies (e.g., linkages)
System: an aggregation of parts, components, and assemblies to perform as a defined system classed by function (e.g., piping, structural, HVAC, electrical, fire suppression)
Inter-related systems: an aggregation of systems, with defined interactions/interfaces

Axis 3: Tasks Supported

- Product requirements documentation
- Functional design
- Drawing Generation and Maintenance
- Performance analysis and testing
  - Structural analysis
  - Thermal analysis
  - Energy consumption analysis
  - Manufacturability/Constructability analysis
  - Other simulations
- Detailed design
- Manufacturing process planning
- Assembly and installation planning
- Procurement
  - Bill of materials
  - Product catalogs
- Quality assurance
- Operations and maintenance
- Disposal

The five existing APs which are STEP Parts are classified using these axes in Figure 2 (APs 201 [Explicit Draughting] and 202 [Associative Draughting]), Figure 3 (AP 203 [Configuration Controlled 3D Product Design Data]), and Figure 4 (APs 204 [Boundary Representation] and 205 [Surface Models]). The classifications given in the figures are certainly open to debate. Some of the rationale for the figures follows.

The product of interest in APs 201 and 202 is the drawing itself, not the objects depicted in the drawing. Since these APs provide for having several coordinated views of the objects, these APs cover part, component, and assembly on the Product Structure axis. Although drawings made using these APs might have many uses, the STEP data for the drawings is intended only for the drawing generation and maintenance task in the case of 201, and for that task plus detailed design in the case of 202.

AP 203 provides for parts, components, and assemblies on the Product Structure axis. It is intended to be used in “conceptual, preliminary, or detail” design but not for any “disciplines outside of the design phase” [Gilbert, page 6]. The breakdown of design on the Tasks Supported axis has only two pieces (functional design and detailed design), so these have been used. This AP also has “design change” data, which does not occur anywhere on the Type of Product Data axis, and is thus missing from Figure 3.

The interpretation used for Figure 4 is that APs 204 and 205 contain shape information that may be used for a wide variety of tasks. Information of other sorts may well be required in carrying out the tasks, in addition to the shape information of APs 204 and 205. APs 204 and 205 provide for only single parts.

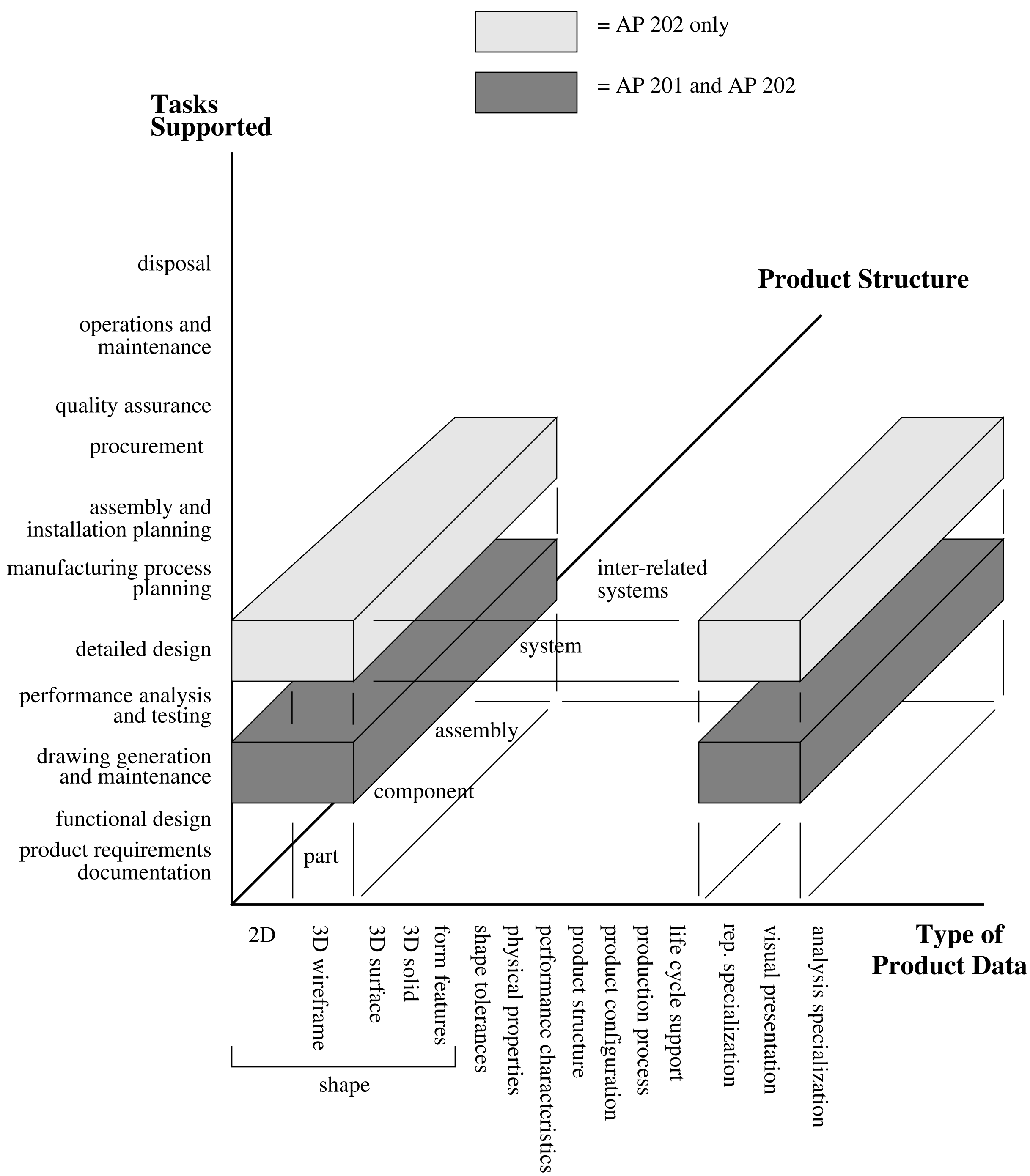

Figure 2. Classification of APs 201 and 202

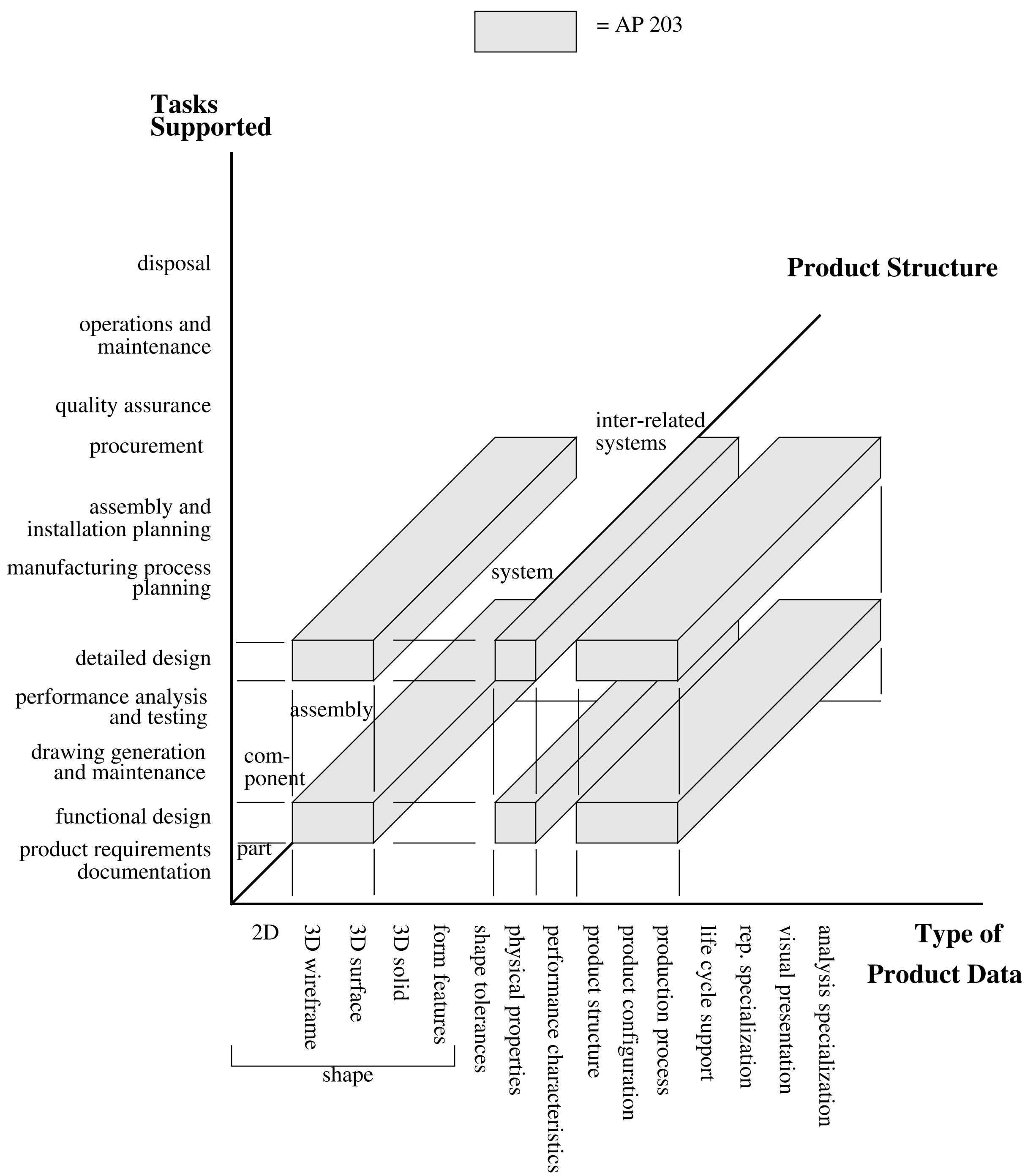

Figure 3. Classification of AP 203

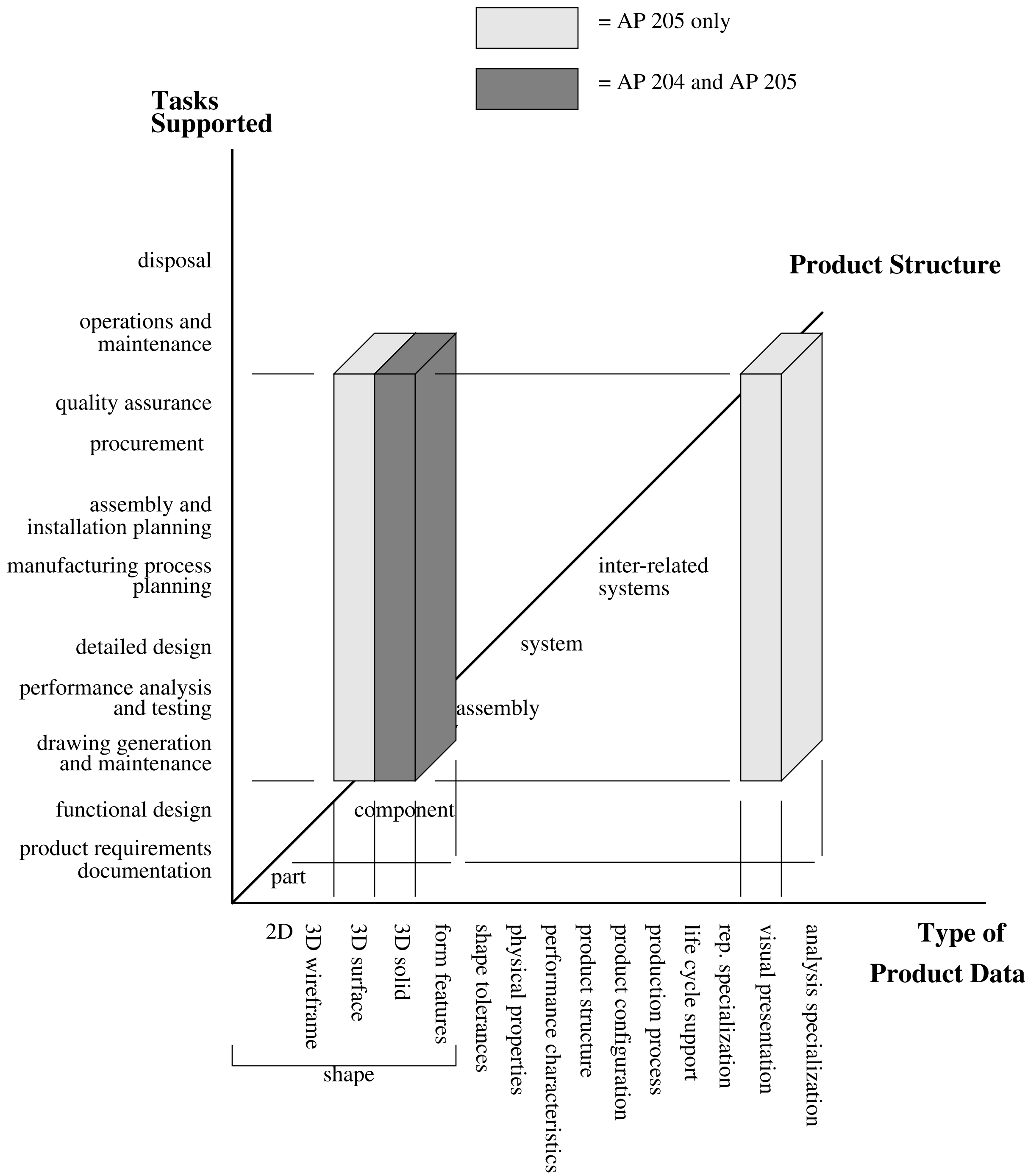

Figure 4. Classification of APs 204 and 205

3.3.5.2 Dependency Graph Classification

A second classification method, easily visualized, uses a directed graph (Figure 5 is an example) showing the dependency of APs and AICs on one another. In this method, APs and AICs are connected by dependency relationships. If one AP references another, the first is dependent on the second. Using the dependency relationship as an arc, the APs and AICs form a directed graph. It is not a hierarchy because, using normal graph terminology, a child node may have more than one parent. It is not a network because cycles are not permitted.

Figure 5 shows a portion of a possible dependency graph. In the figure, AICs and APs are represented by boxes. The boxes are connected by arrows. An arrow between boxes means the AP or AIC at the head of the arrow is referenced by the AP or AIC at the tail. The EXPRESS schema for the AP or AIC at the tail of the arrow would have a USE or REFERENCE statement citing the schema for the AP or AIC at the head.

As an alternative to the bottom-up method of determining what APs and AICs are needed, the STEP community might lay out a complete proposed dependency graph and charter development teams to develop individual APs or AICs or sets of closely linked APs or AICs. This would have the advantage of making clear the role of each AP or AIC. The scope and general functionality of each AP or AIC would be determined by the STEP community, not the development team. Developing APs and AICs in this mode would be iterative, with refinements in the scope and functionality of each AP or AIC being made as each is developed. The location of each AP or AIC in the dependency graph might also change as the planned graph becomes the realized graph.

We recommended that this alternative be given further consideration.

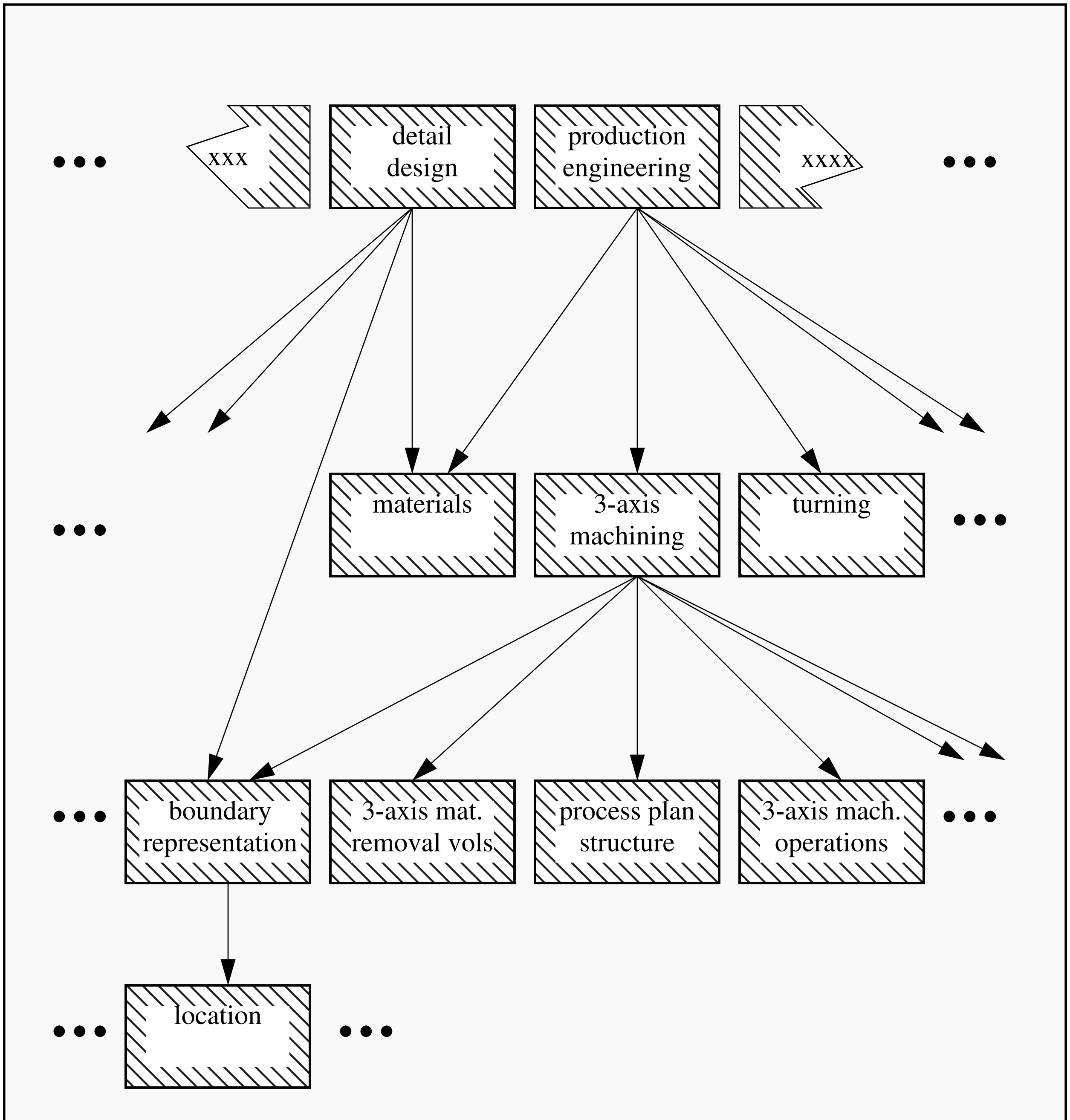

Figure 5. Dependency Graph of APs and AICs

3.3.6 Guidance for Development of Individual APs and AICs

3.3.6.1 General

Much guidance for developing individual APs is provided in *Guidelines for the Development and Approval of STEP Application Protocols* [Palmer1] and is not repeated here. The remainder of this section is intended to be complementary to the *Guidelines*. Some of this material may be appropriate to add to the *Guidelines*.

3.3.6.2 Deciding Between APs and AICs

If data under consideration for an AP is inherently widespread (such as basic shape), or is used in a very large number of applications for some other reason, (such as a basic bill of materials), the data should be handled in an AIC rather than an AP.

3.3.6.3 Scoping APs and AICs

Consideration should be given to aligning AP and AIC scope boundaries close to the boundaries of existing software systems dealing with a general area, if the boundaries of such systems are similar to one another. This is the case in computer-aided design, for example.

If an AP or AIC is largely an implementation version of a STEP integrated resource, the boundaries of its scope should generally coincide with the boundaries of the resource.

AP scope boundaries should generally coincide with the boundaries of discipline view(s), engineering stage(s), product class, and product structure. AIC scopes should generally ignore these boundaries. Atoms in AICs are expected to be used on both sides of boundaries of one or more of these types.

In conjunction with the development of an AIM from an ARM, the scope of the AP should be analyzed for possible alignment with the boundaries of any APs used to construct the AIM.

3.3.6.4 Level of Detail in APs

Data which is expected to pass between software systems from different sources should be modeled in APs, regardless of whether the systems perform the same functions or different functions. It will usually not be necessary to model data which will never be exchanged. For example, a complete description of the shape of a designed object should be covered by an AP applicable to a CAD system, but it is not necessary to have an AP for the construction geometry used during the design of the part, or for journal files, rollback files, or user preferences.

3.3.7 Other Issues

3.3.7.1 Gaps in Resources

Measures should be taken in STEP to identify and fill gaps in resources which will impede the building of APs. This issue is discussed in more detail in section 4.2.5.3.

3.3.7.2 Constraints

As a rule (to which there will be exceptions), constraints should be defined liberally but applied sparingly in resource models. If this is done, it should not be necessary to provide the capability to delete constraints from resources in using them for APs. This issue is discussed in more detail in section 4.2.5.1.

3.3.7.3 Efficiency

Efficiency of representation should be sought in constructing atoms and sets of interacting atoms. A good deal of effort to get efficient representations is justified while building APs. Where a trade-off between efficiency and redundancy is contemplated, a large gain in efficiency should be required to justify a small amount of redundancy. This issue is discussed in more detail in section 4.2.4.4.

3.3.7.4 Ambiguity

Each instance of ambiguity should be judged on its own merits. It is likely that most instances of ambiguity within the context of an application will be unacceptable. This issue is discussed in more detail in section 4.2.4.2.

3.3.7.5 Redundancy

Redundancy of function in atoms or sets of atoms should be minimized, but permitted where significant gains in efficiency may be obtained. Also see the next section. This issue is discussed in more detail in section 4.2.4.3.

3.3.7.6 APs Outside STEP and Pre-STEP APs

AP coordination efforts should include a method of registering APs developed outside STEP and submitted to STEP for registry. Such APs should be reviewed to see if it would be useful to have them become STEP APs. If so, efforts to bring them into STEP should be made by the STEP community.

Industry consortiums should establish mechanisms for identifying requirements for APs and for testing the utility and viability of developing specific APs. Initial ideas and scope statements for possible APs should be carefully analyzed and documented by interested organizations. Once these organizations have established confidence in the scope statement (and its relationship with existing AP projects), the application activity model, and possibly the application reference model, a Candidate AP Summary should be submitted for review and approval by the SC4 PMAG as a new STEP AP project. Some Candidate APs' scopes may be outside the defined scope of STEP, and others will define additional requirements that the STEP integrated resources must eventually fulfill.

3.3.8 Recommendations for Planning APs for a Product

The following steps are suggested for planning the suite of APs needed to provide for the data required in a given product.

1. Identify the product structure, the type of product data, and the applications/tasks that will use the AP. Locate on the AP classification axes given in section 3.3.5.1.

2. Identify the set of components comprising the product. Select a representative set of test pieces for the target application domain.

3. Define preliminary AAM and identify pertinent product life cycle stages.

4. Define relevant product life cycle activities and information flows.

5. Define relevant product characteristics (product structure, material, shape, tolerances, functional requirements, performance requirements, …).

6. Define detailed AAM (including multiple levels of decomposition) for the product life cycle. This may be supplemented with AAMs from different discipline views.

7. Identify major information transfers between participants in the product life cycle (e.g., from detail design to fabrication and assembly) and shared information requirements.

8. Define product information requirements for distinct life cycle stages.

9. Define the product data required for each targeted application.

10. Identify data requirements and functionality common to two or more applications or tasks. Identify data requirements and units of functionality common with other APs.

11. Define initial AP suite.

12. Identify relevant APs or AP projects and assess the utility of inter-project collaboration.

13. Perform a cost, benefit, and risk analysis on developing the AP or AP suite.

14. Prioritize APs and define AP development plan. This plan should include interaction between overlapping/interfacing APs during development.

4 Issues

4.1 Main Issues

4.1.1 Issues

What methodology should be used for defining the scope of APs and dividing a broad area of activity into APs? What is a suitable Framework for planning and implementing the development of APs?

4.1.2 Context of Issues

The context of this issues was set in section 1. Additional details follow.

The NIST National PDES Testbed first recognized this issue in planning for a series of interrelated APs for a STEP Production Cell ([Fowler], [Stark]). A working group formed to plan APs could not achieve an agreement, in part because no method exists for dividing the planned activities into separate APs.

The IPO Mechanical Products Committee also recognized this issue and solicited ideas from committee members for how to deal with it [Cain]. The responses [Engelberth], [Kramer1], and [McKay] did not provide conclusive answers. Committee efforts to build APs for mechanical products have been hampered by lack of a method for dividing APs.

The Air Force has issued formal requests for proposals ([Drago1], [Drago2]), one for developing an AP suite for composites, the other for developing APs for electronics. Successful performance of contracts could depend upon resolution of this issue.

4.2 Related Issues and Sub-Issues

The main issue has several sub-issues. There are also several closely related issues whose resolution will be affected by the way in which APs are built. This section deals with these two types of issues. The subsections cover the following issues:

- 4.2.1 Coverage of one AP
- 4.2.2 Classification of APs
- 4.2.3 AP Substructure
- 4.2.4 Semantics, Ambiguity, Redundancy, and Efficiency
- 4.2.5 Using Integrated Resources
- 4.2.6 Tailoring APs to Specific Disciplines
- 4.2.7 Extensibility and Upward Compatibility
- 4.2.8 Interoperability
- 4.2.9 Pre-STEP AP Development and APs Outside STEP
- 4.2.10 Acceptability of APs and AP Framework
- 4.2.11 Testing a Methodology
- 4.2.12 Implementing a Framework
- 4.2.13 Design of APs
- 4.2.14 Interaction Between AP Development, Scope and Structure
- 4.2.15 AP Teams

4.2.1 Coverage of One AP

The scope of an AP defines an area of discourse covered by the AP. What guidance should be provided on coverage? If the focus is on a specific type of data, without careful examination of the uses of the data for industrial purposes, it is likely that the data will be inadequate for some important purposes. If the focus is on an industrial purpose alone, it is likely that essentially identical data types will be defined differently in different APs. Appendix D provides examples of this pitfall.

APs are to be developed to support data types for specific purposes, not to support data types alone or purposes alone. This premise allows, for example, the development of a B-Rep AP for transfer of designs among design systems, finite element analysis modeling systems, process planning systems, and NC-programming systems.

The design of APs must support product data exchange for similar systems (e.g., between two different design systems or from one design system to itself at different times) and dissimilar systems (e.g., a design system and a process planning system). Is there any difference in the APs required for these kinds of transfer?

At the finest level of detail, it would seem reasonable to have one AP cover the transfer of one type of information between two or more systems, all of which produce such information as output or require it as input.

At broader level, it may be useful to have an AP for a large system (e.g., a machine shop) which consists of a list of all the detailed APs required for the business of the larger system. One can imagine, for example, an AP for a “class 2 machine shop” which contains an AP for B-Rep designs, several process planning APs, a bill of materials AP, a cutter AP, etc.

4.2.2 Classification of APs

Some method of classifying APs is required. The classification of APs should serve as an aid in managing AP development by providing understanding of what areas of product data are covered by existing APs and what areas remain to be covered.

4.2.2.1 Selecting Axes for 3-Space Classification

One classification method useful for visualizing AP coverage is to view the scope of an AP as covering some portion of an N-dimensional (usually 3-dimensional) space. Appendix F gives technical details of N-spaces. Using a 3-dimensional space requires assigning AP properties to each axis.

The criterion by which a set of axes should be judged is: Does classifying APs using this set of axes provide information which is useful to STEP?

Many different axes have been proposed which might be used for classifying APs. These include:

1. ***Shape Representation*** - primitive instancing, wire frame, surfaced wire frame, B-Rep, exact constructive solid geometry, approximate constructive solid geometry (such as octree), form features.

2. ***Discipline View*** (partial list) - draughting, analysis (e.g., finite element analysis), process planning, configuration management.

3. ***STEP Version*** - The STEP version number of the AP.

4. ***AP Level of Functionality*** - levels of functionality (not to be confused with units of functionality) may be defined within an AP (see section 4.2.3.6).

5. ***Engineering Stage or Product Life Cycle*** - design (conceptual design, engineering design), manufacturing (manufacturing engineering, scheduling, production, inspection), use (testing, operation, maintenance, logistics support), decommissioning, disposal.

6. ***Product Class*** - a standard method for describing types of products, such as the Office of Management and Budget's Standard Industrial Classification list [OMB].

7. ***Type of Product Data*** - a classification based on type of data - configuration management, geometry, materials, etc.

8. ***Tasks Supported*** - design, process planning, scheduling, analysis, etc.

9. ***Product Structure*** - part, component, assembly, system, multiple interacting systems.

10. ***Product Material*** - the material from which the product is made (e.g., steel, glass, plastic, etc.).

11. ***Product Shape*** - a classification by physical shape characteristics (rotational parts, prismatic parts, etc.) for products which have a fixed physical shape.

12. ***Product Properties*** - a classification that combines product structure, product material, and product shape on a single axis.

13. ***Category*** - [Kirkley, page 24] a combination of discipline view and data type.

14. "***Applicable to All Subjects***" - [Schuldt, page 4] physical, functional, process.

15. "***Applicable in the Context of a Product***" - [Schuldt, pages 4 and 5] product requirements data, product description data, product production data, and eight other types of product data.

4.2.2.2 An Example of 3-Space Classification

Figure 6 is an example showing a 3-space using the axes proposed in section 3, namely: type of product data, product structure, and tasks supported. A shaded block shows the subspace where:

1. Tasks Supported are drawing generation and maintenance, performance analysis and testing, and detailed design,

2. Type of Product Data is 3D surface, and

3. Product Structure is part.

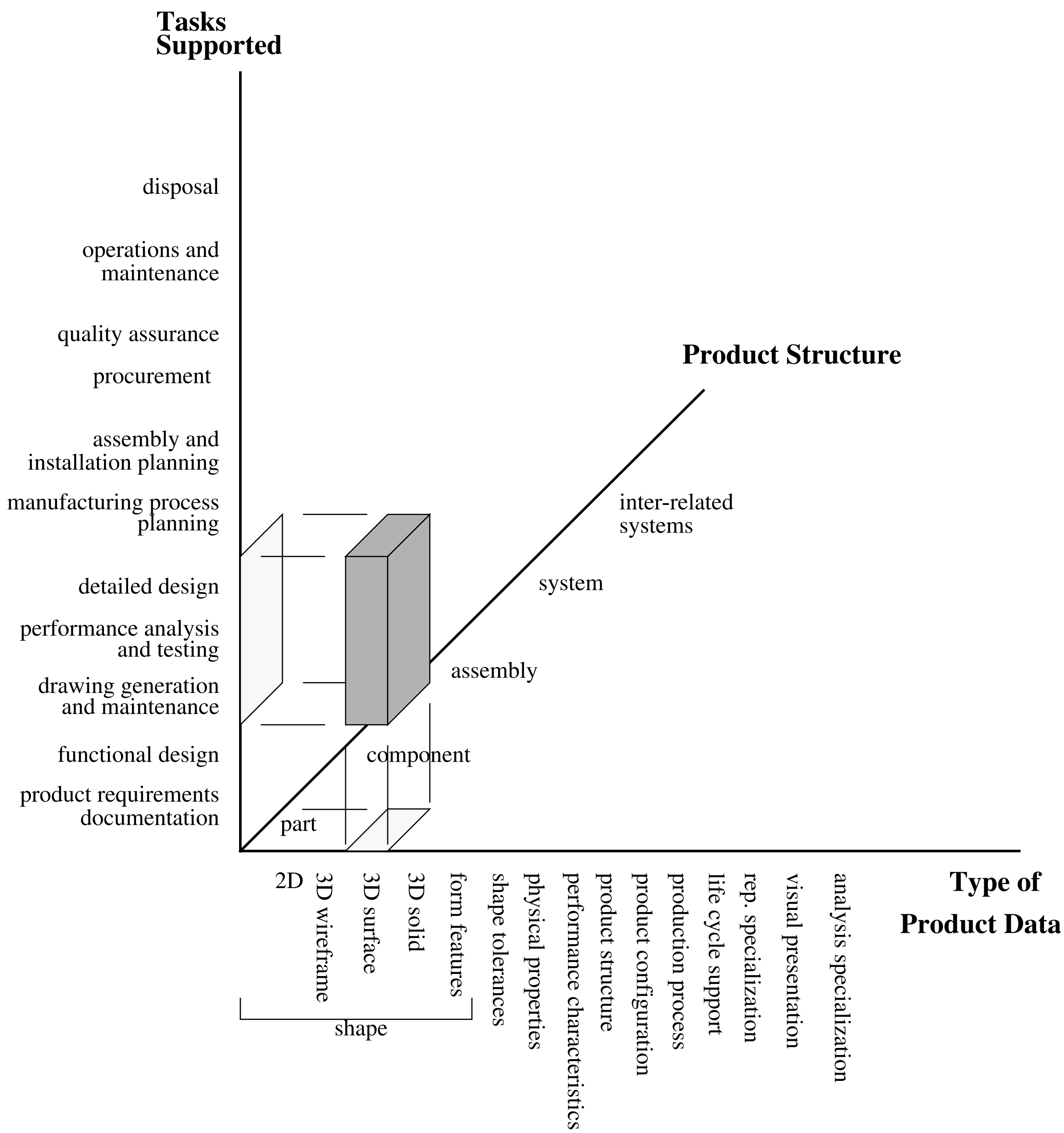

Figure 6. 3-Space Example

4.2.2.3 Other Classification Methods

One alternative method of classifying APs is a dependency graph. This was discussed in section 3.3.5.2.

The use of classification methods for APs has not been thoroughly investigated in the STEP project. Some initial papers have been considered without any resolution ([Schuldt], [Palmer4], [Kirkley]). The STEP project requires a comprehensive analysis of this topic that should provide input to the planning of future STEP Parts. This paper provides only a partial analysis. A thorough study is needed.

4.2.3 AIM Substructure and Other AIM Issues

What type of substructure should the AIM of an AP have and how does AIM substructure interact with other issues. Methods of dealing with overlaps in APs, interoperability of APs, extensibility of APs, and functional levels in APs are largely embodied in AIM substructure. The fact that AIMs are to be implemented has implications for setting AP boundaries.

4.2.3.1 Substructure Forms

An “atom” is the smallest statement-like unit of AIM structure, specifically, one of the following from the EXPRESS Reference Manual [Spiby]: constant-decl, entity-block, function-block, procedure-block, rule-block, type-decl. The only EXPRESS grouping construct is the schema, so both application interpreted constructs and AIMs translate into EXPRESS as schemas. EXPRESS builds schemas from a mixture of atoms and other schemas (via an interface that currently consists of “reference” and “use” clauses).

Functions and procedures only affect entities by use in “where” clauses or rules. More is said about “where” clauses and rules in section 4.2.5.1.

The second AIM substructure form is the application interpreted construct (AIC) - defined as a logical grouping of concepts shared by two or more AIMs. AICs are composed of atoms. It may be desirable to allow AICs as components of other AICs; the idea needs further examination.

AIMs are constructed using atoms and AICs. It may be desirable to also allow other APs to be used as components of APs; this idea also needs further examination.

4.2.3.2 Application Interpreted Constructs

AICs are intended to be usable in several different APs. Thus, methods of defining AICs and revising the definitions will be required which ensure that AIC development is coordinated with AP development and that AICs span APs. What process could be used to make minor modifications to an AIC to make it useful to an AP under construction without causing problems for existing standardized APs already using the AIC?

A formal STEP method of dealing with AICs has not yet been developed, but will be required. AICs have been discussed in [Danner2] and [Palmer1].

The background study for an AIC will be different from the background study of an AP. An AP focuses on the activities of a specific application via the AAM in order to discover what data is required. In determining what should be included in an AIC, it will be necessary to focus first on the data known to be common to some set of applications and then try to discover other applications of the data, in order to be sure the data will serve the needs of as broad a group as possible.

Should application interpreted constructs be nested? EXPRESS provides the tools for conceptual nesting of schemas, but should those tools be used on AICs? To keep software simple, it would be desirable to avoid nesting. On the other hand, AICs may be naturally hierarchical in some areas (geometry seems a likely case).

Can an atom be included in more than one AIC? If nesting of AICs is not allowed, it is almost certain that some atoms will be required in more than one AIC. Because AICs are to be used as building blocks, it is clear that if an atom is used in any AIC, it must have the same definition in all AICs in which it is included, and in any AP in which it is used outside of an AIC.

Are limitations on the number of atoms and subordinate AICs in an AIC desirable? It seems desirable to have at least a handful of atoms in an AIC in order that the number of AICs required to make up an AIM is not excessive. It also seems desirable not to have AICs with very large numbers of atoms, since such AICs would be hard to implement and would be unlikely to serve well as building blocks.

It is desirable that most or all of the atoms in a given AIC should be useful to each of the AIMs that employs that AIC. It may be, however, that a given application would need some but not all the atoms from an AIC. Thus, if the requirement that an application must implement an entire AIC if any of the AIC is implemented is imposed, this may require application systems to be able to deal with a lot of atoms not required by the application. The requirement seems desirable, but it remains to be seen whether AICs can be devised cleverly enough to avoid this problem.

4.2.3.3 Mapping ARM Substructure to AIM Substructure

It is expected [Palmer1, page 4-9] that any team building an APs will identify units of functionality in the ARM. The AIM counterpart of a UOF identified in an ARM may fit within some existing AIC. In other cases, a UOF might be translated into a new AIC when the AIM is developed.

4.2.3.4 Implications of AIMs for AP Boundaries

The requirement that an AIM be implementable must be a taken into account in setting the boundaries of an AP. Since AIMs are what gets implemented in STEP-compliant software systems, AIMs will tend to delineate the boundaries of software systems. It may be desirable to place AP boundaries roughly in the same place as the boundaries of existing software systems. In the case of large multipurpose systems (such as solid modeling combined with rendering and NC-programming), the AP boundaries would correspond to the submodules of the system. Software developers should participate in setting AP boundaries.

The information needs for controlling specific types of equipment should also be considered in setting AP boundaries.

Consideration should also be given, while setting boundaries, to which atoms have instances in sets of data transmitted from one system to another or stored and retrieved later. Having the boundaries of AIMs include all atoms with instances in such sets would be useful, since the transmission or storage and retrieval can be governed by an AIM.

4.2.3.5 Overlaps

Should the scope of APs be disjoint or overlapping? Certain types of data are required in many different applications. For example, a need to know the shape of a product is extremely common, and bills of materials are required for almost all types of manufactured products. Thus, the scope of APs almost certainly will overlap.

It is desirable that where different applications use the same type of data, they use the same representation for that data. The shape representation of a computer chip used to analyze heat flow inside a computer, for example, should be usable by a robot for placing the chip on a circuit board for the computer. It is not desirable, however that this be accomplished by duplicating entities in many different APs, because of the difficulty this makes in keeping entities the same when versions change. Rather, such overlaps should be dealt with by having an information model (an AIC or AP) defined once and then referenced by all the APs using that type of data. In this way the overlaps among APs will be modular, and managing the configuration of overlaps will be more tractable.

4.2.3.6 Functional Levels

The concept of functional levels was described in section 3.3.1.3. To what extent should APs have functional levels, such as the three proposed levels of the B-Rep AP [Weick2]? It seems desirable to allow functional levels, since enterprises and systems may often be grouped in functional levels. An inexpensive software package might be adequate to implement a low functional level of an AP and be suitable for some enterprises. A more expensive software package might be required to implement a higher functional level and be suitable for an enterprise making more complex products.

If functional levels are used, some method of describing them will be required. The definition of a functional level may be quite simple to state. In the case of the B-Rep AP, for example, if a software package can handle all the atoms required for a “facetted_brep_occurrence” [Weick2, page 37], it implements the lowest level of that AP. It may be sufficient to allow descriptions of this sort in natural language, or it may desirable to add the notion of levels using EXPRESS in a formal manner.

Should functional levels be required to be nested? That is, must a lower level be a subset of some higher level? This is desirable so that software for handling STEP can be built hierarchically and so that physical files prepared according to the rules for a lower level can be used for a higher level. The semantics of any entity should be the same at all levels. Semantics is discussed in section 4.2.4.

In addition to fixed levels, it may be desirable to have extensions and restrictions to fixed levels to allow modularity. For example, there might be a “text” extension to the B-Rep AP at level 2 of that AP, or there might be a combined restriction and extension of level 3 of the B-Rep AP which disallowed b-spline surfaces (a restriction) and substituted some other surface model in their place (an extension).

4.2.4 Semantics, Ambiguity, Redundancy, and Efficiency

The notions of semantics, ambiguity and redundancy are essential to an understanding of APs, yet misuses of these terms are common in discussions of STEP issues. Misunderstanding of the situation regarding semantics and redundancy in STEP is also common. Further, how to deal with these matters in APs is an important issue.

4.2.4.1 Semantics

The "semantics" of an EXPRESS schema is simply the meaning of the schema, the information the schema is intended to convey.

In general, the transfer of information via data requires that the sender and the receiver have a common body of knowledge which provides the context in which the data can be interpreted. The definition of EXPRESS and the definition of the rules for constructing physical files from EXPRESS schemas provide some of that common body of knowledge. Much of the meaning conveyed in EXPRESS schemas, however, is based on common knowledge of natural language or common knowledge of specific disciplines such as mathematics. Much other meaning required for understanding the data in physical files cannot be stated in EXPRESS, but is given in natural language text accompanying EXPRESS statements. This is stated in [Kirkley, page 15] as follows.

> *Conveyance of semantic intent is effected by careful selection of names for model elements, selection of appropriate attributes and relationships, and careful wording that accompanies and describes the guidance for use of each element.*

The definition of a circle in the STEP geometry schema [Goult, page 44] provides a good example. The EXPRESS definition of a circle is as follows:

```
ENTITY circle
SUBTYPE OF (conic);
 radius       : length_measure;
 position     : axis2_placement;
DERIVE
 dim          : INTEGER := coordinate_space(position);
WHERE
 WR1 : radius > 0.0;
END_ENTITY
```

The EXPRESS statement does not tell what shape is intended. Only because the name of the entity is "circle", does the reader understand what is intended. If the reader does not know English, the meaning is lost. Exactly the same EXPRESS statement could be used to define a square if "c i r c l e" happened to mean square in English. The corners could be at (R, 0, 0), (0, R, 0), (-R, 0, 0), and (0, -R, 0), where R is the value of "radius", for example. In fact, that EXPRESS statement could define any curve of known shape whose definition requires only one real number to specify size - a straight line segment, an equilateral triangle, whatever. The orientation of the figure with respect to the axis2_placement has to be specified in the text accompanying the EXPRESS statement in any event. Much of the information about circles required to use them is contained in the text [Goult, page 42]. This includes where the circle is located with respect to the axis2_placement, and which direction around the circle is the positive sense of the circle. A reader who does not understand the vector notation and parametric equations used in the text will not be able to determine these facts about an instance of a circle in a STEP physical file.

Since a given EXPRESS statement may have different semantics in different schemas, the issue arises: should this be permitted in STEP? Allowing different semantics can be handled automatically if the EXPRESS schema under which a file was prepared is known, and the semantics of the schema have been embodied in the software handling the file. However a given file may be prepared using several schemas. If the same EXPRESS statement has differing semantics in those schemas, the meaning of the data in the file will be unclear. It seems desirable not to allow different semantics.

4.2.4.2 Ambiguity

The general meaning of ambiguity is the quality of being open to more than one interpretation. Ambiguity is a problem when it is important that only one interpretation be possible. If it does not matter which of several possible interpretations is made, then ambiguity is not a problem, and there is no point in getting rid of it. Thus, ambiguity is acceptable in some situations, unacceptable in others.

One of the best known examples of acceptable ambiguity regards threads. Imagine a threaded bolt hole in a flat surface. The opening of the hole onto the surface is roughly a circle. The edge of the thread is a helix which emerges from the hole at some one point on that circle. The position of that point on the circle is usually left ambiguous because the usual use of a bolt hole is for fastening with a bolt, and for this purpose it does not matter where that point is. When threads are specified, no information that would determine that point is given.

One of the best known examples of unacceptable ambiguity regards interpreting a wire frame model as a solid model. The example is a block with a passage through it, as shown in Figure 7. The wire frame on the left can be surfaced with planes to form a solid as shown either in the middle figure (where the passage is from top to bottom) or in the figure on the right (where the passage is from left to right).

The Framework should provide that each instance of ambiguity be judged on its own merits, in the context of the application. It is likely that the vast majority of instances of ambiguity will be unacceptable, since ambiguity is usually accidental, not intentional.

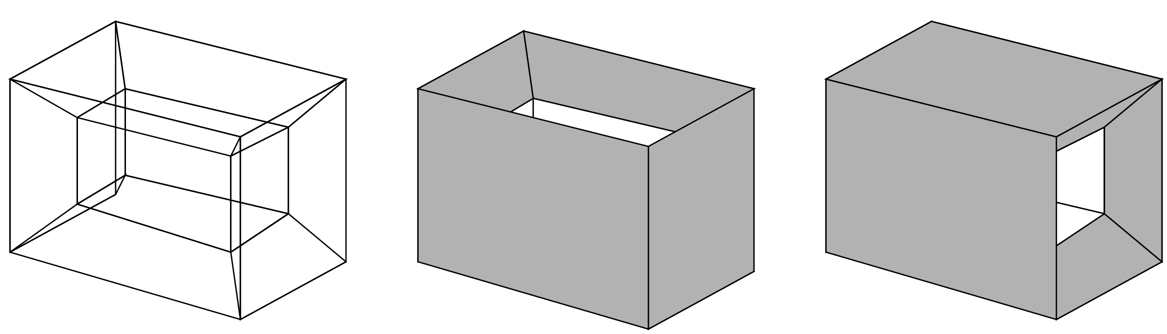

Figure 7. Ambiguous Wire Frame

4.2.4.3 Redundancy

The concern here is with the number of ways in which a single situation can be represented. If there is more than one way to represent a situation, it is difficult to determine if two representations are representing the same situation. In STEP the representations are constructed of instances of entities defined in an EXPRESS schema.

Redundancy is defined in [Schenck, page 2] as follows:

> *A redundant condition exists when it is possible to transform one instance into another and vice versa without loss of content, intent or functionality.*

In STEP discussions, redundancy is often presented as a problem of having two different entity types which can represent the same thing in some or all situations (an ellipse with equal major and minor radii can represent a circle, for example). Less frequently, it is presented as a problem of different sets of attributes being used to define an entity (a circle may be defined by a center and a radius or three points on the circumference, for example). But perhaps the most serious problem has gotten almost no ink: The same entity instance may often be represented in more than one way even if all the representations use the same entity type with the same set of attributes by using different values for the attributes. A line, for example, is represented in STEP geometry by a point on the line and a direction. Since there are an infinite number of points on a line, the choices of value for the point attribute are infinite. Since a line goes in two directions, there is a choice of two values for the direction attribute. The chances that two different CAD systems (or two different people) would represent the same line the same way are very small.

It is often stated that one of the goals of APs is to eliminate the possibility of redundant representation ([Schenck, page 1], [Anderson1, page 3], [Assiff2, several mentions], [Danner2, item 5 of Introduction], etc.), but the last kind of redundancy is not properly recognized.

In some cases, redundancy may be desirable. In the draft AP for boundary representation [Weick2], for example, a loop around a face may be either an edge_loop or a poly_loop. These two entities are functionally equivalent if the face is a polygon. The edge_loop is more general, and the poly_loop is more concise. Poly_loops are used with facetted B-Reps (the lowest functional level of that AP) and edge_loops are used with elementary (intermediate functional level) and manifold_solid (highest functional level) B-Reps. The poly_loop is known to be redundant and is intentionally efficient: *The poly loop was created specifically for the efficient communication of facetted B-rep models* [Goult, page 16].

When an AIM is developed, the developers are supposed to select the representations most useful to satisfy requirements in the ARM. Even if this is done, it may be that the ARM entities are already redundant or that a selected entity allows redundant instances, as discussed earlier.

The Framework should take a position on whether redundancy should be allowed, and, if so, guidelines for what is acceptable should be provided.

4.2.4.4 Efficiency

It is generally agreed that STEP should strive for efficiency (minimizing the amount of data required to represent a given situation) as long as there is no cost for efficiency. The issue of efficiency is: if efficiency has a cost, how much should we be willing to pay. This issue has been discussed in [Christensen1].

The usual cost of efficiency is loss of generality. Typically, a more efficient representation will be redundant with a more general representation. It is not unusual for efficient representations to differ from general representations by a factor of 10 to 100 in the amount of data needed. For example, three real numbers (representing length, width, and height) suffice to define a box, but a facetted_brep (the most efficient representation of the draft B-Rep AP [Weick2]) would require 43 entities.

STEP action regarding efficiency has not been consistent. As the quote in the preceding section indicates, efficiency was a factor in Part 42. On the other hand, a current STEP document regarding integration [Danner2, page 3] states: *Constructs shall not include ideas or mechanisms that are motivated by convenience in practices, computer technologies, or efficiency requirements for implementation.*

The AP Framework should suggest a position on efficiency. The position should be made consistent with the larger STEP framework.

4.2.5 Using Integrated Resources

4.2.5.1 Constraints on Entities

Constraints on entities are implemented in EXPRESS by the use of "where" clauses, "unique" clauses, and rules. The "where" clauses and rules may refer to functions and procedures.

STEP integrated resources are intended to be broadly useful. To make them so, it may be useful for integrated resources to minimize applying constraints, as a general rule, on the expectation that appropriate constraints will be applied in APs. To prevent proliferation of duplicate functions and procedures in APs, the functions and procedures required should be defined in resource models, even if not applied there. STEP provides that constraints may be added in APs but does not provide for deleting them. If the application of constraints is not minimized in resource models, it may be desirable to allow deletion of constraints in APs.

Some resource models (geometry and topology, for example) currently define and apply large numbers of constraints. Many geometry and topology constraints apply to any use of geometry and topology entities, but others do not. The restriction that the loops on a face shall not be a mixture of poly_loops and other loop types [Goult, page 147], for example, does not seem necessary in any situation, and the constraint that the orientation of a face agree with the flagged orientation of the underlying surface seems unnecessary in some applications.

4.2.5.2 EXPRESS

Does EXPRESS require modification to facilitate development of APs? EXPRESS facilities (formerly "map", currently "reference" and "use") designed to help use integrated resources in applications have been changing. It seems reasonable to expect that further changes will be required until AP definition methods have been proven.

Several sections of this report discuss the use of EXPRESS in connection with specific issues: the deletion of constraints in section 4.2.5, functional levels of APs in sections 3.3.1.4 and 4.2.3.6, tailoring APs to specific application domains in section 4.2.6, user-defined attributes in section 4.2.13.4, and prescribing scoping in section C.3.4.3.

As discussed in section 4.2.4.1, the semantics of EXPRESS statements are carried in part in the text accompanying the EXPRESS statements. Currently, there is no method in EXPRESS of associating a comment with an EXPRESS statement. Proximity of a comment to an EXPRESS statement in a file has no meaning in EXPRESS. As a result, there is no way for a parser to preserve semantics given in text when the parser reads an EXPRESS schema. It may be desirable to add such a method to EXPRESS. It is unfortunate that in formal languages this capability is often taken as an invitation to avoid using the facilities of the language, because in EXPRESS, semantics are lost when text is lost.

On the other side of the coin, it is generally feasible to write EXPRESS statements to implement some or all of the semantics of entities, the constraints on entities and relationships between entities. AIMs should be written so that all the semantics that can be put into EXPRESS are.

4.2.5.3 Gaps in Integrated Resources

Current policy in STEP is that entities which are not defined in some resource model should not be used in AIMs, except rarely ([Mason]). Under this policy, it will be difficult to develop APs for large areas of product data, because the holes in the coverage of current resource models are gigantic. Process planning and dynamics (velocity, energy, etc.), for example, are completely absent.

A process planning model [Paul] is under development in the Manufacturing Technology Committee of the IPO. This process planning model could be thought of as a candidate resource model but is not yet a STEP Part. This model is designed to deal with discrete processes, but has no entities specific to any manufacturing domain, such as machining.

A prototype STEP-based system for generating NC-code for a machining center built by one of the authors [Kramer4] illustrates the gaps in integrated resources. It was desired to put all input in STEP physical files. To do this, it was necessary first to adopt an EXPRESS schema for a general-purpose process planning language, and then to add a suite of machining operation entities to that schema. It was also necessary to define an EXPRESS schema for material removal volumes for that system. It may be expected that commercial systems for NC-programming will have similar data requirements in the future, and APs will be needed to describe that data.

Much of the data required for specifying machining operations is specific to machining. It may be expected that other machine types (turning centers, coordinate measuring machines, etc.) will have operations that are specific to the machine type but are universal in the sense that such machines are used worldwide. If APs for the data required to control such machines are not developed, STEP will not achieve its objective of providing standards for product data. On the other hand, if all entities required for all machine types are made part of integrated resources, it may be difficult to find the broadly applicable general-purpose entities among all the special-purpose entities.

It is clearly desirable to fill the gaps in integrated resources. Even when these gaps are filled, however, it may be desirable to allow APs to include large numbers of entities which are not part of integrated resources. Application interpreted constructs containing such entities could be defined and moved into integrated resources if the entities are discovered to have multiple uses.

Allowing large numbers of application-specific entities will complicate the problem of building AICs and the problem of identifying redundant entities.

This issue goes beyond APs into the overall framework of STEP. The AP Framework should address the issue, but the provisions of the AP Framework will have to be coordinated with provisions of a STEP framework.

4.2.6 Tailoring STEP APs to Specific Disciplines

Currently in STEP there are several techniques for tailoring APs to specific disciplines. All work through EXPRESS. This is described in the AP Guidelines [Palmer1, page 4-13 through 4-16] and the EXPRESS Usage Guide.

1. Subtypes of an entity defined in a resource model may be created in the AIM of an AP. This allows the addition of attributes, rules, and constraints. Rules may require values for optional attributes to be present or to be absent in instances of entities.

2. Aliasing entity names is provided in the EXPRESS “use” clause [Spiby, page 66]. Aliasing names of any atoms is provided in the EXPRESS “reference” clause [Spiby, page 67].

Other types of tailoring are possible in EXPRESS but not currently allowed. As was discussed in section 4.2.5, defining new types of data not provided in integrated resources or other APs could be permitted.

Another idea is to permit use of the usual terminology of the discipline by providing for aliasing attributes of entities. Aliasing attribute names is not provided in EXPRESS.

If the motivation for allowing aliasing is to permit standard usage of terms by various specialties, then it will be necessary to have STEP data processing software able to deal with multiple meanings for the same term, since there will certainly be cases of different specialties using the same term different ways. Moreover, a method of dealing with the same term meaning two different things within the same file will have to be considered. One such method is mentioned in section 4.2.8.

4.2.7 Extensibility and Upward Compatibility

It may be expected that new products and new methods of manufacturing will come into being. Both may require the use of new types of data. STEP must provide smooth methods for extending old APs and introducing new ones to handle this data. By extensibility of STEP, we mean the concept of adding such extensions to STEP without changing the existing contents.

By upward compatibility, we mean the compatibility of data produced under version n of some STEP Part with data produced under version n+1 (or version n+m). Upward compatibility has been discussed in several papers ([Bloom2], [Eirich], [Shaw]). Upward compatibility subsumes extensibility, since any extension will be made in a new version, but upward compatibility deals with changes in existing data models, as well as new ones. APs have their own specific problems with changes in version. What happens to an AIM or AIC when a resource it uses changes? What happens when an atom migrates from an AIM into a resource?

4.2.8 Interoperability

The general issue of interoperability is: how can two or more APs be made to work together? This has been discussed in [Bloom1] and [Burkett].

There is some sentiment that it should not be necessary to know what AP was used in preparing a set of data. This does not appear to be feasible. Rather, as stated in [Burkett, page 3]: *It is … clear that the receiving system must know the AP under which the data was created.*

Assuming the identity of the AP (or identities of the APs) under which a data set was prepared is known, there would not be a problem unless the data set was prepared using more than one AP, the APs contain entities having the same name but different definitions or constraints, and an instance of such an entity occurs in the data set.

The draft AP for B-Reps [Weick2, pages 50 - 51] proposes that physical files be divided into sections, with each section containing entities prepared according to a single AP. This solution appears workable for physical files. A solution for databases is not so easy. Labelling every entity instance with its AP would consume a lot of memory, and dividing the database by AP might be difficult or impossible.

A simpler solution is not to allow atoms with the same name to be used in two APs with different definitions. A master index to names used in all AICs and APs would be needed to implement this restriction. If the restriction were implemented and the master index were made available, APs would not necessarily have to be linked to entity instances in physical files or databases, since the link could be formed using the index.

4.2.9 Pre-STEP APs and APs Outside STEP

APs may be constructed outside of STEP. These may be complete formal APs, or they may be partial APs, most likely including an AIM but missing one or more of the other required AP components. We will call the latter "ad hoc AIMs." The formal APs may be developed with the intention of becoming STEP APs; these we will call "pre-STEP APs." The concept of "vendor protocol" has been developed as a type of non-STEP protocol that is not intended to become part of the STEP standard.

SC4 made the following statement in March 1991 about APs defined outside STEP in resolution 64 [Smith]:

> *As well as the need for normative APs, SC4 recognizes that there exists an industrial requirement for additional APs to be defined outside the SC4 organization at the industry, national, or regional level. SC4 requests the PMAG to define requirements for guidelines to permit the development of APs both inside and outside SC4.*
>
> *Provision should also be made for the registration of APs with SC4 to ensure the consistency of related standards.*

4.2.9.1 Complete Pre-STEP APs

Several APs are being developed and tested prior to making a formal application to SC4 for approval as a sanctioned AP project. These projects are focused on defining the information requirements for their application domains. In conjunction with the STEP community, these programs are analyzing the current set of STEP models to identify voids and are defining strategies to ensure that STEP will eventually provide the necessary integrated resource constructs to meet their requirements. All of these are being coordinated with STEP but are not yet formally part of STEP. These projects plan to submit their results for inclusion in the STEP standard.

The Navy-Industry Digital Data Exchange Standards Committee (NIDDESC) is developing a "NIDDESC Piping Application Protocol" [NIDDESC]. The ARM for it has been written, but an AIM does not yet exist. This AP has been developed in conjunction with the STEP committees.

As noted earlier, the U.S. Air Force has released requests for proposals ([Drago1], [Drago2]), one for developing an AP suite for composites, the other for developing APs for electronics. The specifications for these APs require close coordination with STEP, but do not require formal STEP approval of the APs developed.

4.2.9.2 Ad Hoc AIMs

An ad hoc AIM is defined here as an implemented EXPRESS schema having no formal standing in the ISO or other organizations developing STEP. An exchange of STEP format physical files can be accomplished as long as it is governed by an EXPRESS schema agreed between the sending and receiving parties. The situation in which an ad hoc AIM would be likely to be formulated is when two parties find that they wish to exchange specialized data which does not appear in existing APs or resource models.

It seems unlikely that the parties to an ad hoc AIM would prepare full APs for their own use, since having the AIM and a common understanding of its semantics would suffice for exchanging information in physical files. Vendors of special-purpose software might develop ad hoc AIMs as a method of loosely integrating their software with other STEP-based software. Vendors would be even be able to use ad hoc AIMs for dealing with data not intended for transmission outside the system, such as tables of user preferences, journals, or command files. The advantage to the vendor is to be able to use schema independent modules to handle such data.

On the one hand, this seems desirable as a method of initiating APs and as a method of providing extensions to existing STEP-approved APs. The Framework might provide incentives for notifying STEP of the existence of ad hoc AIMs, for documenting the context, and for getting them submitted to STEP for expansion into complete APs and formal adoption.

On the other hand, encouraging ad hoc AIMs opens the door to duplication of efforts and uncontrolled growth.

4.2.10 Acceptability

Both the acceptability of the Framework and the acceptability of individual APs are at issue. The Framework itself must be acceptable to the STEP community, and it must be constructed so as to promote the acceptability of individual APs developed under it.

4.2.10.1 STEP Community

One of the purposes of STEP is to enhance the direct exchange of information between computers. APs must be constructed so as to facilitate such exchange. Since current business practices are rarely optimized in this direction, it is likely that APs designed to facilitate computer sensibility will be a motivating force for changes in business practices. This may detract from the acceptability of STEP APs in the STEP Community. Conversely, it seems likely that designing APs for acceptability will tend to promote the status quo, not computer sensibility.

4.2.10.2 Vendors

The existence of APs is likely to be a major factor setting the boundaries of commercial software systems. If APs are very broad, they are likely to go beyond what vendors would prefer to provide. The Framework should provide for obtaining vendor input as part of setting the scope of an AP. How far should STEP go to accommodate vendors? How much will vendors have to change to adapt to STEP?

It is very helpful in the development of consensus standards if all vendors can be treated equally. STEP is in a good position in this regard by virtue of being new. Often in the development of consensus standards a standard is being sought for something already being handled by competing commercial systems, and it is a challenge to develop a standard which is sensible but does not give preferential treatment to some subset of existing systems. There are no STEP-based commercial systems, so the problem doesn't arise.

4.2.11 Testing a Methodology

It would be very desirable to test any proposed methodology for dividing a broad area of activity into application protocols before making a commitment to the methodology or (especially) to any organizational arrangements for STEP-wide implementation of the methodology. Research and prototype facilities such as the STEP Production Cell proposed for the National PDES Testbed might serve as testbeds for a proposed methodology.

It is not clear that the STEP community can afford to wait as long as a test is likely to take. This period is a conceivable minimum of 5 months in a crash program using mostly people knowledgeable about STEP, and is estimated to be about two years for a steady but unforced implementation using many people originally unfamiliar with STEP.

4.2.12 Implementing a Framework

Any Framework will require that complete knowledge of the contents of STEP be available in one place.

Many possible Frameworks will require some agency (person, organization, etc.) with a broad knowledge of STEP to review proposed AICs.

Certainly a STEP-wide AP development policy should be adopted.

Should a STEP-wide AP development plan be formulated? If there is such a thing, an agency for getting it carried out will be required.

4.2.13 Alternatives to APs and Variations in the Design of APs

Is there an alternative to APs that would serve the same purposes better? Are improvements required in the contents, the development process, or the approval process for APs? The ideas in this section have been put forth and references are given. We are not aware of any others.

4.2.13.1 Kernel or Core

The notion of a STEP "Kernel" or "Universal Core" has been suggested as an adjunct or alternative to APs ([Anderson1], [Anderson2], [Anderson4], [Assiff2], [Christensen2], [Schenck]). We will use "Core" here to denote the idea. Although the approaches of these authors differ somewhat, the Core may be defined as the set of all atoms from integrated resources being used by some application protocol. As the number and content of APs change, the Core changes. If it is required that any atom used in an AP also be in integrated resources, then the Core obviously contains all the atoms used in any AP. As defined, the Core is a subset of all the atoms in integrated resources.

Other than as an aid in understanding STEP, it is not clear what use might be made of the Core that would not be served by AICs and APs.

4.2.13.2 Resource Protocol

The notion of a "resource protocol" is discussed in [Assiff1] and [Assiff2]. *A "Resource Protocol" is a particular subset of entities, properly constrained, from a resource model* [Assiff2, page 6]. The term "resource protocol" appears synonymous with what we are calling "application interpreted construct," with the limitation that only unmodified atoms already found in integrated resources can be put into a resource protocol. As such, it may be a good idea, but the name is not needed.

4.2.13.3 Direct Implementation of Resources

It has been suggested that an alternative to APs might be to allow the direct implementation of atoms from integrated resources. Those atoms to be implemented would be selected on the basis of need. Thus, the Core (or resource protocols) would be constructed directly, and APs would not be needed. It must be admitted that APs have not existed, yet people (including the authors) have been dealing with STEP physical files for years. Files have not, however, been exchanged between independently developed systems. A cursory look at a few such files will convince a doubter of the need for additional standards.

A mature monolithic Core is unworkable because the atoms in it will be very heterogeneous and only a small fraction of them will be of interest to any one application. Commercial software vendors are not going to build software to deal with a large monolithic Core. So, in the rest of this section we will assume the Core has been divided into non-intersecting subsets of atoms.

The proposal to implement this modular Core has serious deficiencies. In a nutshell, direct implementation does not deal with the issues ancillary to an AIM that are considered in the AP contents and development process specified in the AP Guidelines [Palmer1]. First, direct implementation does not provide for test purposes or test suites. Second, it does not provide for certification. Third, it does not provide for a user guide. Without these three items, uniform implementation of a Core will not happen. In addition, some process for deciding what should go into the Core will have to be developed. This has been anticipated by the requirements for an AAM and ARM in the AP Guidelines.

4.2.13.4 User-defined Attributes

It is a stated requirement of the Piping Application Protocol [NIDDESC, page 4], which is a pre-STEP AP in the process of being built, that user-defined attributes be supported. Since an AIM has not yet been written, it is not clear what effect the requirement will have on systems for handling data prepared according to this AP. It sounds as though the intent is to define some entities in the EXPRESS schema of the AIM only partially, and let the user change the attributes of those entities, perhaps by putting EXPRESS statements as well as data in physical files.

User-defined attributes in physical files appear to be a bad idea, since, as discussed in section 4.2.4.1, the semantics of the user-defined attributes would not be in the EXPRESS. Schema-independent software could be constructed to read the definitions of user-defined attributes and then read in data and check its type, but the receiver would not be able to use the data since its meaning would be unknown.

As discussed in section 4.2.9, if there is agreement between the sender and receiver, an ad hoc AIM could be used to fill the apparent need for user-defined attributes. This must be a joint decision between the sender and the receiver, not a unilateral decision by the sender. The semantics of the non-standard atoms must be built into the software handling the data.

Because EXPRESS is a rich language, it may be feasible in building an AIM to define EXPRESS atoms that meet the actual needs which are described in the Piping Application Protocol as a need for user-defined attributes. Details of the required capability are not given in that document.

4.2.14 Interactions Between AP Development, Scope, and Structure

4.2.14.1 Delaying Effect of Broad Scope

If APs have broad scope, it will be necessary to bring experts from all the subject areas in that scope together to consider the AP. A larger group necessarily works more slowly than a smaller one. Thus, having broad scopes may delay AP development.

4.2.14.2 Organizational Implications of AP Structure and Scope

If the structure of APs is complex, as seems likely, it may not be reasonable to expect committees of volunteers to be able to produce well-structured APs without the assistance of professionals in AP development. One possibility is to have a professional group of AP developers assigned as staff to work with committees of volunteer subject matter experts. The staff would move from project to project as APs were completed.

4.2.14.3 Total Coverage of STEP

What portion of the world of product data is within the scope of STEP?

The scope of STEP may be viewed as a tree, with the branchings representing specialization. For example, construction might be one main branch. Construction might subdivide by specialty into the branches: wood frame house construction, shipbuilding, road building, steel girder frame construction, etc. The question is, at what level of specialization does STEP stop? Since each level of specialization tends to increase the number of entities by a multiplicative factor, this is an important issue. It would seem practical to decide each first instance of specialization on its own merits and its implication for the proliferation of APs.

One area in which there is clearly the potential for a large number of APs is process planning for discrete processes. That main branch splits naturally by machine type: machining, turning, coordinate measuring, grinding, wire edm, etc., and most of those split several ways. Machining, for example, splits by the number of simultaneously controlled axes.

4.2.14.4 How Many APs

The number of APs required in STEP is important because if the number is large, AP development procedures will have to promote either rapid development or parallel development of APs (or both) in order to provide the required APs in a reasonable time period.

Obviously the number of STEP APs will depend directly on the size of the scope of STEP. As discussed in the previous section, this does not seem currently to be well-defined.

The Framework should provide for monitoring the growth of the universe of APs in order to be able to take timely action in case growth problems arise. The growth of AICs should be monitored for the same reasons.

Assuming that process planning is split as described above, several dozen APs might be required for process planning. APs to cover the areas in which resource models have been built or in which APs have been proposed would number a few dozens more. No clear upper limit is in view.

4.2.14.5 Management of AP Coverage

The portion of the world of product data within the scope of STEP needs to be covered by APs. What is the best way to generate a good covering? It is desirable both to let the STEP community focus its energy on those areas it most wants to cover and to ensure that the coverage is complete and rational at the end of the process.

Any method of generating a covering will require registration by a central agency of the STEP community of the scope of individual APs. The most extreme laissiez-faire position for generating the covering would be to allow assignment of scopes on a first-come, first-served basis, with no limit on the scope of a single AP and no limit on the number of APs assigned to an individual developer. The most extreme centralized position would be to have a central agency specify all the AP scopes it deems necessary, and then have the APs developed by developers selected by the agency according to a schedule determined by the agency.

These extremes are unworkable. The laissiez-faire position encourages a land-rush with fights among concerned parties, since any group with an interest in a type of data has to lay claim to it to protect its interest. This position is also not going to result in a rational division of the AP map, because there is no provision for rational division in its procedures.

The extreme centralized position has the usual drawbacks of centralization and makes no provision for protecting the interests or harnessing the energies of the volunteer portion of the STEP community.

4.2.15 AP and AIC Teams

How should the teams formed to develop APs and AICs be constituted, and how should they operate? How much training should team members be given?

As has already been discussed, the teams that deal with AICs will certainly have to be more broadly constituted than AP teams and may have different staffing needs. It may also be desirable to have some teams devoted solely to defining scopes for APs and other teams devoted to developing APs whose scope has already been defined.

It is clear that each team will need experts in the subject matter with which the team is dealing. It is also clear that at least some members of the team will need to be familiar with the details of constructing APs.

5 Summary and Conclusion

This report has described the main issues and many sub-issues of STEP application protocols and made recommendations regarding many of them. There is a clear need for a STEP AP Framework. This report has proposed functional requirements for the Framework and suggested what many elements of the Framework might be. The methodology of APs is still evolving. This report has made many recommendations regarding AP methodology. The chief recommendation regarding methodology is to make much more extensive use of application interpreted constructs (AICs) in building APs. Methods of developing and using AICs must be devised and formally incorporated in STEP practices.

This report provides many suggestions. Few of them are definitive. The STEP community will need to give further consideration to what the AP Framework should be and will need to experiment with AP methodology to determine what works. There is a great deal to be done.

A References

This list includes documents available to the authors in the summer of 1991. In some cases, later versions have appeared but are not referenced, since they were not available at the time this report was being developed. Where a referenced document had appeared in several versions by the summer of 1991, only the latest version available at that time is included here (with one exception, where two versions are cited). Several references here are to working papers which were not intended to be polished documents and may not be readily available.

[Anderson1] Anderson, William; working paper *Toward a Core Subset for PDES*; April 20, 1990; 6 pages

[Anderson2] Anderson, William; working paper *A Core Subset for PDES Revisited*; December 3, 1990; 6 pages

[Anderson3] Anderson, William; memo minutes of a PDES, Inc. *TM/L Team meeting on the Inter-operability of APs*; March 27, 1991; 2 pages

[Anderson4] Anderson, William; working paper *Perspective: The Kernel as a STEP Part*; March 28, 1991; 4 pages

[Assiff1] Assiff, Thomas C.; working paper *Rules Governing Geometry Resource Protocols*; January 7, 1991; 8 pages

[Assiff2] Assiff, Thomas C.; working paper *On Application Protocols and Universal Cores*; April 5, 1991; 8 pages

[Bloom1] Bloom, Buzz; and Silvestri, Mark; working paper *Application Protocol Interoperability Issues*; April 4, 1991; 6 pages

[Bloom2] Bloom, Buzz; working paper *Levels of Upward Compatibility*; February 27, 1991; 5 pages

[Bloom3] Bloom, Buzz; working paper *Two View of the Resource Model Development Process*; April 2, 1991; 1 page

[Brauner] Brauner, Kalman; et al; *PDES Initiation Activities*; IPO, May 1986; 240 pages

[Burkett] Burkett, William C.; Anderson, William; and Gilbert, Mitch; working paper *Application Protocols and Implementations: A Discussion of AP "Interoperability*;" April 5, 1991; 6 pages

[Cain] Cain, William D.; memo to 15 addressees *Concerns from AP Developers*; March 12, 1991; 2 pages

[Christensen1] Christensen, Noel C.; *The Need for Balance between Efficiency and Entity Count*; May 16, 1989; 6 pages

[Christensen2] Christensen, Noel C.; *STEP Kernel 3D Shape for Maximizing Information Sharing*; May 21, 1990; 11 pages

[Danner1] Danner, William F.; *A Proposed Integration Framework for STEP (Standard for the Exchange of Product Model Data)*, National Institute of Standards and Technology; Interagency Report 90-4295; April 1990; 22 pages

[Danner2] Danner, William F.; Sanford, David T.; and Yang, Yuhwei; *STEP (Standard for the Exchange of Product Model Data) Resource Integration: Semantic and Syntactic Rules*, National Institute of Standards and Technology; Interagency Report 91-4528; March 1991; 20 pages

[Danner3] Danner, William F.; and Yang, Yuhwei; STEP working paper *Application Protocol Integration Strategy*; May 1991; 3 pages

[Drago1] Drago, Sharon L.; U. S. Air Force request for proposal *Product Data Exchange Using STEP (PDES) Application Protocol Suite for Composites*; RFP F33615-91-R-5713; March 12, 1991

[Drago2] Drago, Sharon L.; U. S. Air Force request for proposal *Product Data Exchange Using STEP (PDES) Application Protocols for Electronics*; RFP F33615-91-R-5718; April 18, 1991

[Eirich] Eirich, Peter; *White Paper: Principles of Upward Compatibility*; April 11, 1991; 5 pages

[Engelberth] Engelberth, Paul A.; memo responding to William Cain; March 12, 1991;1 page

[Evensen] Evensen, Per; and Haenisch, Jochen; draft STEP Part 205 *Application Protocol for the Representation of Mechanical Design Surface Models by means of ISO 10303*; April 3, 1991

[Fowler] Fowler, James E.; *Development Plan STEP Production Cell*; National Institute of Standards and Technology; Interagency Report 4421; September 1990; 29 pages

[Gilbert] Gilbert, Mitchell; draft STEP Part 203 *Application Protocol for the Exchange of Configuration Controlled 3D Product Design Data*; Version 0.4; April 8, 1991; 167 pages

[Goosen] Goosen, Ted; draft *Exchange of Product Design Data to Process Planning for NC Systems*; March 12, 1991

[Goult] Goult, Ray; draft STEP Part 42 *General Resources: Shape Representation*; April 1991; 218 pages

[Haas] Haas, Wolfgang; draft STEP Part 201 *Explicit Draughting*; May 11, 1991; 120 pages

[Harrison] Harrison, Randy J.; and Palmer, Mark E.; *Guidelines for the Specification and Validation of IGES Application Protocols*; National Institute of Standards and Technology; Interagency Report 88-3846; January 1989

[ISO] ISO/IEC JTC 1 / SC 21; *Information Technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General Concepts*; March 14, 1991

[Jurrens] Jurrens, Kevin; *Test Plan for Validating A Context Driven Integrated Model (CDIM) For Sheet Metal Die Desig*n; August, 1991; 48 pages

[Kennicott1] Kennicott, Philip R.; memo to Mark Palmer *AP Integration Remarks*; May 31, 1991; 3 pages

[Kennicott2] Kennicott, Philip R.; memo to Mark Palmer regarding the notion of Vendor Protocol; June 7, 1991; 1 page

[Kirkley] Kirkley, James R.; and Seitz, Brian K.; draft STEP Part *STEP Framework Concepts and Principles*; version 3.000; July 17, 1991; 53 pages

[Kloetzli] Kloetzli, John W. and Billingsley, Daniel W.; *NIDDESC: Meeting the Data Exchange Challenge Through a Cooperative Effort*; July 4, 1989; 16 pages

[Kramer1] Kramer, Thomas R.; memo responding to William Cain; March 12, 1991; 3 pages

[Kramer2] Kramer, Thomas R.; working paper *Comments on Application Protocol Guidelines, Version 0.5*; January 2, 1991; 4 pages

[Kramer3] Kramer, Thomas R.; working paper *Comments on Application Protocol Guidelines, Version 0.*7; February 27, 1991; 3 pages

[Kramer4] Kramer, Thomas R.; <u>The Off-Line Programming System (OLPS): A Prototype STEP-Based NC-Program Generator</u>, proceedings of a seminar *Product Data Exchange for the 1990's*; New Orleans, Louisiana; NCGA; February 1991; Vol. 2

[Mason] Mason, Howard; draft *STEP Part 1: Overview and Fundamental Principles*; version 6; March 25, 1991; 32 pages

[McKay] McKay, Alison; memo responding to William Cain; March 14, 1991; 3 pages

[McKee] McKee, Larry; draft STEP Part 202 *Application Protocol: Associative Draughting*; Version 0.1, April 5, 1991; 62 pages

[McLean] McLean, Charles; memorandum on the subject *Application Protocol Framework Study*; April 23, 1991; 3 pages

[NCGA] National Computer Graphics Association; *IGES/PDES Organization Reference Manual*; April 1991; 61 pages

[NIDDESC] NIDDESC Outfit Working Group; pre-STEP AP *Piping Application Protocol*; Version 0.4; May 1991; 88 pages

[OMB] Office of Management and Budget; *Standard Industrial Classification Manual*; 1987

[Palmer1] Palmer, Mark E.; Gilbert, Mitch; and Anderson, Jody; STEP working paper *Guidelines for the Development and Approval of STEP Application Protocols*; Version 0.9; September 17, 1991

[Palmer2] Palmer, Mark E.; working paper *Current Issues on STEP APs*; April 5, 1991; 1 page

[Palmer3] Palmer, Mark E.; and Reed, Kent A.; *3D Piping IGES Application Protocol,*; version 1.0; National Institute of Standards and Technology; Interagency Report 4420; September 1990; 257 pages

[Palmer4] Palmer, Mark E.; memorandum to IGES/AEC Committee Members *Development of IGES/AEC Application Subsets*; June 3, 1987; 3 pages

[Parks] Parks, Robert E.; Draft STEP Part 101 *Application Resources: Draughting*; Version 2.4.0; April 3, 1991; 85 pages

[Paul] Paul, Greg A.; STEP working paper *Manufacturing - Process Plan*; Version 3.1; January 2, 1991; 95 pages

[PDES, Inc.1] PDES, Inc.; *TC-RAP 1 Final Report*; June 28,1989; 36 pages

[PDES, Inc.2] PDES, Inc.; *Data Model and Test Criteria Development for CDIM A1 Final Report*; October 26,1989; 31 pages

[PDES, Inc.3] PDES, Inc.; *CDIM A2 Project Plan for PDES, Inc.*; August 9, 1990; 24 pages

[PDES, Inc.4] PDES, Inc.; *Program Technical Development Plan (Phase II and Updated Phase I)*; August 14, 1990; 104 pages

[PDES, Inc.5] PDES, Inc.; *CDIM A2 Context Document for PDES, Inc.*; November 2, 1990; 47 pages

[PDES, Inc.6] PDES, Inc.; *CDIM B4 Project Plan for PDES, Inc.*; November 2, 1990; 36 pages

[PDES, Inc.7] PDES, Inc.; Draft *CDIM B4 Context Document*; November 2, 1990

[PDES, Inc.8] PDES, Inc.; *Test Report for Context Driven Integrated Model Application B3 Exchange of 3-D Product Design Data to NC Programming Systems*; February 1, 1991; 23 pages

[PDES, Inc.9] PDES, Inc.; *CDIM SM1 Project Plan for PDES, Inc.*; March 13, 1991; 29 pages

[PDES, Inc.10] PDES, Inc.; *Minutes from a Sheet Metal Planning Meeting*; May 23, 1991; 14 pages

[Schenck] Schenck, Doug; working paper *A View of the Universal Core*; November 16, 1990; 4 pages

[Schuldt] Schuldt, Ronald L; and Gielingh, Wim F.; *Universal Data Classification a Necessary Tool in the Evolution of STEP*; unpublished; 12 pages

[Shaw] Shaw, Nigel; working paper *Upwards compatibility A personal view*; February 27, 1991; 4 pages

[Smith] Smith, Bradford; draft *Policy and Procedures Manual for the ISO TC 184 / SC 4 Subcommittee*; March 22, 1991

[Spiby] Spiby, Philip; draft STEP Part 11 *EXPRESS Language Reference Manual*; April 29, 1991; 160 pages

[Stark] Stark, Chuck; and Mitchell, Mary; *Development Plan: Application Protocols for Mechanical Parts Production*; National Institute of Standards and Technology; Interagency Report 4628; July 2, 1991; 33 pages

[Van Maanen] Van Maanen, Jan; draft STEP Part 21 *Clear Text Encoding of the Exchange Structure*; March 25, 1991; 61 pages

[Weick1] Weick, W.; draft STEP Part 204 *Application Protocol for the Data-Transfer of STEP B_Rep Models Via a Physical File*; Version 2.0; September 11, 1990; 81 pages

[Weick2] Weick, W.; draft STEP Part 204 *Application Protocol for the Data-Transfer and Use of STEP B_Rep Models in a Mechanical Design Context Via a Physical File*; Version 3.0; March 25, 1991; 64 pages

[Woodall] Woodall, Alison; Pearson, Mark; Owen, Jon; Shaw, Nigel; Palmer, Mark; STEP working paper *Critical Issues for Application Protocols*; January 3, 1990; 11 pages

B Definitions and Acronyms

B.1 Definitions

AP Framework: A methodology for dividing a broad area of activity into APs and for planning and implementing the rational development of APs.

Application Activity Model (AAM): A representation of one or more activities which use product data in specific application context. An AAM is used to establish an understanding and agreement of the application activities and processes. [Palmer1, page 2-1]

Application Interpreted Construct (AIC): A logical grouping of concepts that is shared by two or more AIMs. An AIC may represent a unit of functionality defined in an ARM. [Palmer1, page 2-1]

Application Interpreted Model (AIM): A model that describes the interpretation of the integrated resource constructs that provide functional equivalence to the AP's information requirements as specified in the application reference model.[Palmer1, page 2-1]. The form of an AIM is an EXPRESS schema.

Application Protocol (AP): A standard which specifies implementable STEP constructs. An AP defines the context, scope and information requirements for the designated application(s) and specifies the STEP constructs for these requirements. Additionally, an AP enumerates the conformance requirements and test purposes for use when performing conformance testing on an implementation of the AP. [Palmer1, page 2-1]

Application Reference Model (ARM): A model that formally describes the information requirements and constraints for an application domain. The information model uses application-specific terminology and rules familiar to an expert from the application domain. The model is independent of any physical implementation. [Palmer1, page 2-1]

Atom: The smallest statement-like unit of AIM structure, specifically, one of the following from the EXPRESS Reference Manual [Spiby]: constant-decl, entity-block, function-block, procedure-block, rule-block, type-decl.

Conforming Implementation: An Implementation Under Test which satisfies both static and dynamic conformance requirements, consistent with the capabilities stated in the Protocol Implementation Conformance Statement [ISO, subclause 3.4.10]. Roughly speaking, this means a software system which is able to process EXPRESS schemas and product data (in at least one implementation form - physical files or databases) in the manner prescribed in an AP and its normative references.

EXPRESS Schema: A formal construct in the EXPRESS information modeling language which is most often used in STEP to define a group of related EXPRESS "Entities", constraints on the entities, and associated information.

Framework: A shortened form of "AP Framework".

Integrated Resources: A set of STEP Parts which provide application-independent information models for widely-used types of information. Integrated resources support communication between diverse applications by providing an agreed upon set of definitions and meanings for data that are independent of specific application requirements.

Part: A separate document in the STEP (ISO 10303) series.

Scope of an Application Protocol: A statement defining the domain of discourse of the AP. A statement of scope is a required normative clause of an AP.

STEP Production Cell (SPC): A proposed prototype STEP-based manufacturing cell envisioned to be built as part of the NIST National PDES Testbed. The SPC would demonstrate how parts are produced in a STEP environment. [see Fowler]

Unit of Functionality (UOF): A collection of entities, attributes, and relationships that conveys one or more well-defined concepts within the context of an AP's ARM such that removal of any component would render the concept(s) incomplete or ambiguous. [Palmer1, page 2-1]

B.2 Acronyms

AAM	Application Activity Model
AIC	Application Interpreted Construct
AIM	Application Interpreted Model
ANSI	American National Standards Institute
AP	Application Protocol
ARM	Application Reference Model
CDIM	Context Dependent Integrated Model
IGES	Initial Graphics Exchange Specification
IPO	IGES/PDES Organization
ISO	International Organization for Standardization
NIDDESC	Navy-Industry Digital Data Exchange Standards Committee
NIST	National Institute of Standards and Technology
PDES	Product Data Exchange Using STEP
PMAG	Project Management Advisory Group (of SC4)
SC4	Subcommittee 4 (of ISO Technical Committee 184), named Industrial Data and Global Manufacturing Programming Languages
SPC	STEP Production Cell
STEP	Standard for the Exchange of Product Model Data
TC-RAP	Testing Criteria Requirements Analysis Planning
TC184	ISO Technical Committee 184, named Industrial Automation Systems
UOF	Unit of Functionality
WG3	Working Group 3 (Product Modeling) of SC4
WG4	Working Group 4 (Qualification and Integration) of SC4
WG5	Working Group 5 (STEP Development Methods) of SC4

C AP Case Studies

This appendix provides a set of case studies of recent or ongoing AP projects. These case studies were developed to identify problems with the current scope definition methods and with the current framework for AP development. A common case study template specifying what to look for was used by the authors for evaluating the AP projects. The template focused particular attention on AP scoping. The reports of the case studies which follow do not have identical formats, although most consist of background, description and summary.

Some conclusions drawn from these case studies are included in this appendix. Others are incorporated in the recommendations of section 3.

C.1 NIDDESC AP Projects

C.1.1 Background

With the increasing use of CAD/CAE/CAM systems in the design, construction, and maintenance of ships and submarines, the U.S. Navy and the marine industry are faced with challenging information exchange requirements. These organizations must have the capability to exchange information between multiple applications during the life cycle of ships and among a constantly changing set of project participants.

> *Naval ships are among the most complex devices known to man. Their design and construction requires from 7 to 12 years. They roam the oceans for 30 years following their construction. … Only a few of each type are built, with each hull differing to some extent from her sisters. By the standards of most industries, these collections of 8,000,000 or so parts are all engineering prototypes* [Kloetzli, page 1].

The Navy/Industry Digital Data Exchange Standards Committee (NIDDESC) was formed in 1986 as a cost-sharing venture between private firms and government organizations, for the purpose of cooperatively developing techniques for enabling the exchange of digital information. NIDDESC's basic goals are to define the information requirements for the life cycle of Naval ships and to establish industry-wide agreements regarding the transfer of that information.

C.1.2 3D Piping IGES Application Protocol

C.1.2.1 Background

The 3D Piping IGES Application Protocol was based on IGES representations developed under the U.S. Navy's SEAWOLF submarine program. That material was enhanced and developed into the first IGES AP with the support of NIDDESC and the participation of representatives of the U.S. and U.K. process plant industry. Version 1.0 was released in September 1990.

C.1.2.2 Description

For Naval ship projects, a primary information transfer occurs when the detailed design is complete and the project information is transferred to the organization responsible for fabricating and assembling the system, module, or ship. Similar transfers of information and responsibility occur in the design and construction of process plants. Additionally, both of these industries recognize the three-dimensional (3D) computer model of the piping system design as highly valuable information for the successful fabrication and construction of piping systems. This common information exchange requirement was the basis for selecting the detailed design of the piping system model as the initial scope for the 3D Piping IGES AP.

Establishing multi-industry and international consensus on an AAM for the “as is” or the “to be” piping design environment became increasingly challenging with each level of decomposition. Due to these factors, the AAM was only developed to the level of identifying the “piping system model” as an output. The piping system model was not further decomposed in the AAM.

The most productive detailed definition of the AP’s scope was accomplished by establishing the minimum set of task or applications which could use the AP, and then negotiating agreements on the minimum set of product data required for each of these applications. This mechanism proved valuable for defining viable objectives for the first version of the AP and for planning subsequent expansions of scope for future versions.

C.1.2.3 Summary

This was the first IGES AP to be delivered to industry and was an important prototype example for both IGES and STEP APs. This AP project provided several insights into the AP development process. These include:

1. The precise definition of scope and terminology becomes increasingly difficult as more industry and discipline views are included.
2. The process of defining scope must employ the optimum level of abstraction at each stage of developing the AP. Rules for determining these levels of abstraction are not available.
3. The definition of the minimum set of tasks which the AP must accommodate, and then defining the minimum set of product data required for these tasks is a successful mechanism for defining the detailed scope of an AP.
4. Representative test pieces from the target application domain (e.g., sample piping models with application queries and reports) must be used in defining the scope and information requirements. These “usage tests” are extremely valuable in the validation of the ARM and the AIM and for interoperability testing.
5. Many users of APs will need to extend the minimum set of required data to provide additional functionality. AP developers should consider including a mechanism for user defined extensions.

C.1.3 NIDDESC Piping Application Protocol

C.1.3.1 Background

NIDDESC is currently developing an AP for piping systems. This AP is based on the NIDDESC Distribution Systems Model and the 3D Piping IGES AP. To date, the scope, information requirements, ARM, and some conformance requirements and usage tests have been defined. The initial scope and requirements were defined in February 1991, and Version 0.4 of the AP was distributed in May 1991 [NIDDESC]. A candidate AP summary will be submitted to SC4 in 1991.

C.1.3.2 Description

The purpose of this AP is to specify the content and format of data describing piping systems for use in a defined set of applications [NIDDESC]. The selection of the supported applications is based on the minimum acceptable functionality for the life cycle of Naval ship product data. The supported applications include: contract/functional design, detail design, production engineering, and support engineering. These activities are defined within the context of the AAM.

The scope of the AP is further defined with a listing of the data requirements for each of the supported activities and a summary of required functionality. This functionality includes: versioning and configuration management, product structure, use of external component catalogs, external references and connectivity, and customization with user-defined attributes. Units of functionality have not been defined for this AP, although the definition of required functionalities provides the basis for defining units of functionality.

The basic building block of the ARM is the abstract supertype Ship_Product_Item, from which all other entities are built. The ARM makes extensive use of subtyping for structuring information from a generalized entity to a very specialized entity. The hierarchy of conceptual objects used in this ARM appears extensible to other subtypes of distribution systems, e.g., HVAC (Heating, Ventilation and Air- Conditioning), and Electrical and Cableway.

This AP project has incorporated the requirements for validation testing and conformance testing into the development of the ARM. The AP conformance requirements and usage tests are being defined in parallel with defining functional requirements and information requirements. The usage tests are used to validate the ARM and to establish confidence that the ARM meets the stated requirements.

C.1.3.3 Summary

The lessons learned to date in this AP project include:

1. When modeling a broad domain, such as ship product data, a high level planning model and a detailed AAM are useful. The AAM should identify all data flows and define the possible “in scope” applications or tasks.

2. Although the planning model and the AAM provide a basis for scope definition and information modeling, these must be supplemented with a defined set of application views of the domain, e.g., designer’s view, fabricator’s view. Once this set of views of the information has been modeled and tested, additional views that must be accommodated may be used to refine the content and structure of the model.

3. During the first stages of AP development, project teams should focus their efforts on defining and validating the application reference model(s) before diverting their attention to analyzing or using the STEP integrated resources. The integrated resources must be stable and available in electronic format for the AP development team to produce an application interpreted model efficiently.

Having defined basic information categories and submodels for ship product models, NIDDESC is developing and testing ARMs for high priority domains. As these models are validated, they will be used to test the utility and coverage of the STEP integrated resources and for initiating STEP AP projects.

This is a viable process for an industry consortium to document and submit requirements to the STEP project. The information requirements of Naval ship projects are not supported in the first edition of STEP. NIDDESC has planned its projects to ensure that a subsequent edition will support piping system models and other domains. This process will be successful only if close coordination and participation is maintained with the development of the STEP integrated resources and the other STEP Parts.

C.2 PDES Inc. CDIM and AP Projects

C.2.1 Background

PDES, Inc. is a joint industry and government effort initiated to accelerate the development, validation and implementation of STEP. The first plan to test and validate the standard within PDES, Inc. focused on testing each topical model in the December 1988 “Tokyo” Draft version. It soon became clear that testing without a clearly defined set of requirements to satisfy was not beneficial. A Testing Criteria Requirements Analysis Planning (TC-RAP) Task Team was formed to address the development of PDES testing criteria [PDES, Inc.4]. A significant result of the TC-RAP activity was the development of application Context-Driven Integrated Models (CDIMs). CDIMs function as a bridge between the application user’s requirements and the conceptual models in the testing environment. Results of CDIM tests are used to recommend changes to the draft conceptual models in the areas of validation, correction, improvements and voids.

Part of the PDES, Inc. charter is to ensure that the requirements of U.S. industry are incorporated into STEP. To this end, in April 1989, a Testing and Validation Candidate Applications Survey was given to the member companies of PDES, Inc. asking them to select three top priority applications. Their responses, including write-ins, were ranked by selection criterion determined by the TC-RAP methodology. The top three priorities became the first four CDIM projects [PDES, Inc1, pages K2-K7]. Additionally, due to member company interests, a Sheet Metal Project has commenced, and other projects are in planning stages.

These CDIMs are developed with the future development of implementable APs in mind. The CDIMs will provide the baseline for efforts that would be beneficial to both PDES, Inc. and the larger ISO community.

C.2.2 Methodology

The methodology applied to CDIMs was developed through the TC-RAP Project and has been modified slightly with use. Testing, modeling, and support teams are assembled with as broad a background as possible. The CDIM teams are composed of roughly ten or more full time equivalents representing almost as many companies and government associates. At a kickoff meeting, the core and support teams meet to reach consensus on objectives, critical success factors and the initial scope of the project. The next phase of the project consists of modeling the activities in the “to be” environment. The activity modeling done in this phase establishes the breadth and depth of the application to be modeled and tested. The IDEF0 application activity models (AAMs) are validated by selected member company experts. An IDEF1X planning model is developed from the data elements discovered through the AAM. At this point, the testing and modeling teams split up. The modelers search the STEP topical models for entities to satisfy the identified needs. The existing EXPRESS for these STEP entities forms the CDIM. The testers, meanwhile, interview Application Experts from member companies and modify the activity model to reflect specific company views. Based on these specific view activity models, test suites are created and executed. A test, fix, test cycle improves the CDIM. [PDES, Inc.3, p.8,9]

This methodology makes extensive use of industry experts in defining, reviewing and validating the scope with the intention of making a thorough test of the models based on application context. The AAMs are a key part of this process. They drive out the in-scope data elements and are also used as a tool to communicate scope to experts.

C.2.3 Summaries

C.2.3.1 CDIM A1

"Exchange of 3D product design data in a configuration controlled environment" was the number two priority according to the results of the survey of PDES, Inc. member companies mentioned in section C.2.1. This was CDIM A1's initial scoping statement. CDIM A1 was kicked off in June 1989. The CDIM A1 Final Report [PDES, Inc.2], the documentation reviewed for this paper, was issued in October 1989.

CDIM A1 was responsible for much of the CDIM process refinement that has taken place. Additionally, A1 recognized a number of issues. There is a lack of compatibility of semantic content between EXPRESS, IDEF, NIAM and the other data modeling techniques used by different groups within the STEP community. There are limitations to the CDIM work:

1. the data format is not unambiguous due to the heavy use of the existing STEP topical models,
2. exchange is only effective in the context of a recognized "exchange context," and
3. the scope of the effort was extremely small.

Finally, the question of how to determine the point where the CDIM has been adequately tested was raised.

The CDIM A1 work is now being built upon by PDES, Inc. and ISO representatives to create the Draft Part 203. See section C.3.3 for more information on Part 203.

C.2.3.2 CDIM A2

CDIM A2 involves the "exchange of 3D product design data to 2D drafting systems" which was the number one priority in the PDES, Inc. member company survey. The documentation reviewed for this paper was the CDIM A2 Project Plan [PDES, Inc.3] and the CDIM A2 Context Document [PDES, Inc.5].

CDIM A2 was initiated in June 1990 and scheduled to wrap up in February 1991. Instead, the decision was made to evolve CDIM A2 into Part 202 before testing. This decision was made for several reasons:

1. there was some hope that Part 202 could then be ready for acceptance into STEP Version 1.0;
2. the change would eliminate the unnecessary overlap in documentation between PDES, Inc. and ISO by providing only ISO required documentation;
3. continuing the CDIM would require the development and maintenance of two versions of EXPRESS models because the testbed tools were not based on the current version; and
4. resources from CDIM A2 could be reassigned to Part 203.

Further discussion of CDIM A2's development is deferred to section C.3.2.

C.2.3.3 CDIM B3

"Exchange of 3D product design data to numerical control programming systems" was the third priority application among PDES, Inc. member companies according to the April 1989 survey. This became the initial scope of CDIM B3. CDIM B3 was kicked off in July 1989. The Test Report [PDES, Inc.8], which was the documentation examined for this paper, was delivered in February of 1990.

The CDIM B3 model was significantly improved through the testing and issue resolution process. The general structure of the model was determined to be sound, but corrections and refinements are based on test results, and as of the test report writing, the model was 85% tested. At this point, the model could have been used for exchange in a prototyping or learning environment, but not in a production environment.

This CDIM made several recommendations for follow-on work. Much of that work was to be done in CDIM B4.

C.2.3.4 CDIM B4

The scope of CDIM B4 is an extension of the work done by B3 to include *the sharing of product data from engineering design to process planning for numerical control and NC functions, and the sharing of the data among process planning for NC and NC functions*. CDIM B4 was initiated in August 1990 and is scheduled to conclude in July 1991. The documentation examined for this paper includes the B4 Project Plan [PDES, Inc.6] and two draft AP Summary Sheets [Goosen].

CDIM B4 has done extensive activity modeling, data modeling, and testing in this area.

The CDIM B4 work has resulted in two draft AP proposals; one Process Plans for Machined Parts, the other Numerical Control (NC) Processes for Machined Parts. These AP summaries name several organizations doing work in this area. PDES, Inc., CAM-I, the ESPRIT project, U.S. Navy's RAMP project, NIST's STEP Production Cell, and the University of Leeds/Loughborough University of Technology's Information Support System for Design and Manufacturing are all mentioned in the proposal. There does not appear to be any framework given for additional APs in the same area, and it is unclear how this work will integrate with future work.

C.2.3.5 CDIM SM1

CDIM SM1 is the first major effort under PDES, Inc.'s sheet metal project. The context of the CDIM is the *standardization of the product data for a shared data base environment to support the design of dies/blocks for sheet metal parts. This includes the interrelationships between the dies/blocks and the parts they fabricate* [Jurrens, page 2]. The documentation examined for this paper is the CDIM SM1Project Plan [PDES, Inc.9] and the CDIM SM1 Test Plan [Jurrens]. CDIM SM1 was kicked-off in August 1990. The effort is expected to take eleven months.

The CDIM was first released in May 1991 and is currently being tested. Three separate versions of the CDIM will be tested, with issue resolutions incorporated into subsequent releases. So far, testing has consisted of manual data population.

A draft AP Summary for an AP in this area has been created by the developers of CDIM SM1. This proposal has been submitted to IPO/ISO for possible inclusion in the next release of STEP. No framework for further APs in this area is in place. AP 205 supports the surfacing capabilities needed by a sheet metal AP, but there are no documented plans for integration. In a long term planning meeting of the Sheet Metal Project [PDES, Inc.10] development of AP SM1 was ranked fourth priority and determining an approach to define the number of APs needed to support the sheet metal industry was ninth.

C.3 STEP AP Projects

In January 1990, SC4 established a group to identify topics on which to develop application protocols for the first edition of STEP. The criteria for this selection were:

1. it should be feasible to realize the AP within one year;
2. the AP must meet an existing industrial need;
3. resources needed by the AP should be included in the minimum intended Parts for the first edition of STEP, and
4. it should be feasible to implement the AP in commercial software systems.

SC4 member countries reviewed eighteen candidate AP summaries and in July 1990 selected five for STEP AP projects. The APs produced by these five projects are reviewed in the following sections.

C.3.1 Part 201: Explicit Draughting

C.3.1.1 Background

Part 201, originally titled “Exchange of 2D Geometrically Explicit CAD Drawings with Explicit Annotation”, was selected by SC4 as the top priority AP for STEP. Although this was not the position of all member countries, the majority of SC4 members believed that an AP with this limited functionality would meet an immediate, industrial need. Many companies, particularly in European countries where IGES is not widely used, are in need of a protocol for exchanging 2D drawings. Part 201 is being developed by an international team and reviewed by committee AP#3 under WG3. The documentation studied for this paper is Version 1.6 of the AP [Haas].

The ISO STEP Draughting Project has defined four levels of protocol based on useful functionality for the exchange of drafting data. In effect, these define a framework for the development of Application Protocols in drafting. Level 0 is defined as *the information about a drawing and the sheets of which the drawing is comprised; where the intent or requirement for use is presentation of the individual sheets of a particular drawing* [Parks, page 81]. Level 1, the functionality satisfied by Part 201, is *the drawing and sheet information, as in the previous level, plus geometric representations of shape, and explicitly defined drafting annotations, where the intent is 2-dimensional drawing maintenance and presentation* [Parks, page 81]. Level 2, supported by Part 202 is *the drawing and sheet information as in the first level, plus drawing view(s) which reference product data shape and explicitly defined drafting annotations with possible references to other product data such as dimension value and tolerance information. The intent or requirement for this level is product data related drawing maintenance and presentation* [Parks, page 81]. The drafting resource model, STEP Part 101, supports levels 0, 1, and 2 as they correspond to the levels of data generated on present systems. The intention behind this framework is that drafting APs mapped from this common drafting model should be upward compatible [Parks, page 82].

C.3.1.2 Description

Version 1.1 of Part 201 did not include an AAM, so activity modeling has played no apparent role in determining the scope or developing the ARM of this AP. The ARM included in this version of the AP is in IDEF1X and appears to be quite complete.This version does not contain defined UOFs or an ARM to AIM mapping. These are currently under development.

This AP is closely related to Part 202: Associative Draughting and Part 101: Draughting Resource. Part 101 has been developing and changing along with the two APs that reference it heavily. Efforts have been made to coordinate work on these three projects, but the wide geographic separation between teams is a source of difficulty. At the San Diego IPO Meeting in April 1991, the teams agreed on definitions of terms and defined an initial set of UOFs. It is hoped that there will be several common units of functionality between the two APs.

C.3.1.3 Summary

This AP is incomplete, untested and therefore subject to change. The framework that was defined by the PDES Drafting Committee has helped structure and scope the AP. However, it is not clear that the APs will be as integrated as was planned. In addition, the stability of the AP can be no better than the stability of the underlying models. Part 101 is still changing, as are several other resource models. The AIM mapping that is currently being developed will consist of some guesses, due to these changes.

C.3.2 Part 202: Associative Draughting

C.3.2.1 Background

As mentioned in section C.2.3.2, Part 202 has evolved from the work commenced in PDES, Inc.'s CDIM A2. Work on the AP documentation and deliverables began in February 1991. The documentation reviewed for this work is Version 0.1 of the Part [McKee] and e-mailed team progress updates. Further background information may be found in section C.2.

C.3.2.2 Description

The AAM and ARM are revised editions of those developed as part of CDIM A2, using the methods described in section C.2.2. Currently, the ARM is being improved and fully attributed, UOFs have been defined, and the AIM has not yet been developed.

Units of Functionality were initially defined at the San Diego IPO in the PDES Drafting Project meeting jointly with the 201 development team. A great deal of effort was made to use units of functionality defined in Part 203 and to coordinate UOFs with Part 201. These UOFs are being defined as narrowly as possible to encourage their use by APs in the same area that may require a slightly wider scope. The UOFs defined in Part 202 are: Product Association, Drawing Structure and Configuration Management, View Projection, Wireframe Geometry for 3D design of Mechanical Parts and Rigid Assemblies, Surface Geometry for 3D Design of Mechanical Parts and Rigid Assemblies, Annotation Element, Associative Drafting Resources, Drafting Annotation, and Grouping. The two geometry UOFs have been taken verbatim from Part 203.

C.3.2.3 Summary

Part 202 is still in early development. The background CDIM work has helped it progress quickly, but it is incomplete and untested. The CDIM methodology has created a robust model that should be widely useful. Resource model stability is presently the most troublesome issue, as it is with several APs currently being developed.

C.3.3 Part 203: Configuration Controlled Design

C.3.3.1 Background

Part 203 has evolved from the PDES, Inc. sponsored CDIM A1 effort as discussed in section C.2.3.1. The documentation reviewed for this work is Version 0.4 of the Part [Gilbert 1].

C.3.3.2 Description

The AAM and ARM provided to the Part 203 effort from CDIM A1 was developed using the methods described in section C.2.2. The AAM has been updated under Part 203, with the revised AAM being validated by industry experts. The ARM of this AP contains the entity set and attributes developed by A1. The testing of these models through CDIM A1 has generated a significant level of confidence that the entities and attributes of CDIM A1 which were populated during testing are required to support the activities within the scope of this AP. For this reason, the development of Part 203 has focused on the portions of CDIM A1 that were queried during testing.

AP 203 has defined seven UOFs: Effectivity, Assembly Component Structure, Design Change, Wireframe Geometry for 3D Design of Detailed Mechanical Parts and Rigid Assemblies, Surface Geometry for 3D Design of Detailed Mechanical Parts and Rigid Assemblies, Surface Bounding Topology, and Shape. In defining UOFs, a lesson learned was that UOFs should be defined with a narrow focus to promote reusability. At this point, the geometry UOFs defined by Part 203 are being used by Part 202.

C.3.3.3 Summary

Part 203 is nearly complete, but has not been tested in its AP form. The CDIM methodology provided a strong foundation on which to build the AP; the scope had been widely reviewed and the CDIM had been tested. Part 203 appears to be the most stable AP created to date. Part 203 is also affected by resource model instability and will likely go out for ballot without updating to the latest version of one of its resource models.

C.3.4 Part 204: Mechanical Design Using Boundary Representation

C.3.4.1 Background

The documentation available for this project is version 2.0 [Weick1] and version 3.0 [Weick2] of the AP for boundary representation. Additional documentation exists but was not obtained for this project. According to [Weick2, page1],

> *During the ISO/TC184/SC4 meeting in Frankfurt in June 1989, the concept of Application Protocols (AP) was introduced to STEP. In order to test the feasibility of this approach, the ESPRIT projects CAD*I and CADEX proposed to develop an AP for the transfer of boundary representation models based on analytical surfaces.*
>
> *… an essential element in the development of this AP was to insure that the experience gained with B-Rep transfer among five commercial systems … in the CAD*I project was fed into the AP.*

Work on this AP began in the summer of 1989. Version 1.1 appeared in December 1989, version 2.0 in September 1990, and version 3.0 in March 1991.

This AP has a close relationship with the Surface AP described in section C.3.5. Both were developed in connection with the CADEX project in Europe. Several of the AAM diagrams are identical between the two APs.

C.3.4.2 Description

This AP appears to have been built by converting an existing CAD*I protocol for exchanging B-Rep information among five commercial CAD systems (noted above) into the AIM of a STEP AP. Other elements required of a STEP AP were then added, or are being added.

The scope of this AP seems to have been determined largely by using the scope of the existing CAD*I protocol. Version 2.0 of the AP had no AAM, so there was no role for an AAM in defining scope for that version. Some AAM diagrams were added in version 3, but clearly as an afterthought. They obviously did not play any role in determining what entities were required to carry out specific activities because the only entities in the ARM that are mentioned in the AAM are the three top-level entities: polyhedron, elementary B-Rep, and advanced B-Rep.

The section of this AP called an ARM looked in Version 2.0 like what is described in the AP Guidelines as an ARM-to-AIM mapping, since it largely described how entities differed from STEP geometry and topology entities of the same name. The ARM of Version 3.0 does not refer to STEP geometry and topology.

A significant innovation, introduced between versions 2.0 and 3.0, is the use of three levels of functionality: polyhedron, elementary B-Rep, and advanced B-Rep. Conceptually, these represent units of functionality, as defined earlier.

The elementary level is compatible with the advanced level, in the sense that the shape of a part designed at one level could be modified slightly and become a design at the other level, without having to change any entity instances representing unmodified shape. The polyhedron level is not compatible with the elementary level in this sense. If a small change is made in one face of a polyhedron requiring a surface from the elementary level, the representation of every face of the polyhedron must change. Changing from elementary level to polyhedron level also requires changing every face.

The AP includes a sample physical file for level 2, which is very helpful for interpreting the rest of the AP.

C.3.4.3 Summary

This AP is a prime candidate for conversion to being an application interpreted construct (AIC). B-Reps are used to express shape for many different purposes. As noted in the activity models at the end of [Wieck2], this includes at least: NC-programming, NC-simulation, robot programming, robot simulation, tool description, manufacturing planning, and assembly planning. There is little point in requiring the authors to write a giant AAM documenting fully the activities needing B-Reps. We know there are lots.

The ARM for this AP was obviously constructed with an eye on the STEP geometry and topology resources. There are dozens of slightly different methods of constructing B-Reps. The chances of being able to build an ARM-to-AIM mapping using STEP geometry and topology effectively would be close to nil if the ARM entities were not intentionally built to be similar to STEP entities.

This AP does not appear stable. The biggest issue is whether it should be an AIC. There is also the issue mentioned above of incompatibility between polyhedron and elementary representations.

An additional issue is that the authors of the AP seem not to be willing to use the STEP rules for physical file construction [Van Maanen], stating (unconvincingly) that implementors need more guidance, STEP changes too fast, and the scoping capabilities of EXPRESS and the mapping rules are inadequate [Wieck2, page 30]. To provide rules for mapping to a physical file, a file construction scheme written in Wirth Syntax Notation is included. Files written using this scheme are very similar to how they would appear if they were written using the rules from [Van Maanen].

As discussed in the next section, the B-Rep AP and the Surface AP overlap excessively.

C.3.5 Part 205: Mechanical Design Using Surface Representation

C.3.5.1 Background

The documentation available for the Surface AP project is [Evensen1]. Additional documentation exists but was not obtained for this project. Details of how this AP was developed are not provided in [Evensen1], but several large companies are acknowledged in the foreword for having made “outstanding contributions”.

This AP has a close relationship with the B-Rep AP discussed in section C.3.4. Both were developed in connection with the CADEX project in Europe. Several of the AAM diagrams are identical between the two APs.

C.3.5.2 Description

This AP provides for building representations of surfaces. Three levels of representation are provided, according to the amount of topology to be used. Level 1 has no topology, while levels 2 and 3 have topology. Level 2 is described as being “non-manifold” and consists of a set of connected face sets, while level 3 is “manifold” and consists of a set of shells.

It is not at all clear why levels 1 and 2 are defined. The AP does not give any reasons for having three levels. There is probably a modest gain in efficiency of representation in levels 1 and 2, as compared with level 3, but even this is not cited as a reason. Given the lack of any explanation, one suspects that the AP levels are tailored so that some existing systems will not have to be changed much in order to comply with one of the three levels.

This AP at level 3 appears to be largely duplicative of the B-Rep AP at the advanced level. Both APs include B-spline surfaces, surfaces of extrusion, and surfaces of revolution, as well as identical sets of elementary surfaces. The only surface type in the Surface AP not in the B-Rep AP is offset surface. The Surface AP allows a shell to be open or closed, whereas a B-Rep shell must be closed. Any part which could be modelled under the B-Rep AP could be modelled using the Surface AP.

Although the surface description methods of the two APs are nearly identical, the Surface AP has a few capabilities missing from the B-Rep AP having to do with replication and presentation.

The AAM diagrams in the Surface AP probably did not play any role in determining what entities were required to carry out specific activities because only a few of the top-level entities in the ARM are mentioned in the AAM. It seems likely that, as with the B-Rep AP, the AAM was developed after the ARM.

The AP does not include an EXPRESS schema for the AIM.

The part of the AP entitled “usage guide”, section C.4, is three alphabetically arranged lists of entity names with no guidance for usage.

The AP does not include a sample file, so it is often difficult to understand what is intended.

As with the B-Rep AP, the authors have not used the STEP rules for mapping EXPRESS to a physical file, but have provided rules written in Wirth Syntax Notation.

C.3.5.3 Summary

Because the contents of the Surface AP overlap so much with those of the B-Rep AP, it does not seem reasonable to have both as STEP Parts in their present form. Either the two should be combined, or common elements should be extracted and put into a separate AP or AIC which would be referenced by both. Case Study of Geometry in Five STEP AP Projects

D Case Study of Geometry in Five STEP AP Projects

One of the objectives of STEP is to provide standard methods of data representation. The five draft APs with STEP Part numbers cover overlapping areas, all of which deal to some extent with geometry. Five of the most basic geometry entities were selected from the STEP geometry schema: cartesian_point, direction, axis2_placement, line, and circle. The five APs were examined for how these were treated. An attempt was made to determine whether a file reader programmed to read physical files prepared according to one of these APs could read instances of these five entities written by a file writer programmed to write files according to one of the other APs.

This exercise was very difficult for several reasons. None of the draft APs contains text for all the sections listed in its table of contents. Several of them are missing portions that are required in order to tell what is meant (AP 201, for example, has many stub definitions in EXPRESS). APs 204 and 205 have their own physical file mapping rules. In general, these rules are the same as the STEP rules. Exceptions are noted below. Where there are exceptions, entity instances will be represented differently.

Table 1 shows page numbers where these entities are referenced; it is probably incomplete. Identical or very similar concepts often have different names (ARM names seem to have been intentionally chosen to differ from AIM names in some cases - which is reasonable). The same name has been given different meanings between the ARM and the AIM in some instances (which is not so reasonable).

Some references given in the index to AP 203 which were to incidental uses of a term where the term itself was not under discussion are intentionally omitted in Table 1. There are no other intentional omissions.

D.1 Cartesian point

In AP 201, the thing called “2d_point_occurrence” seems to be close to a 2d STEP cartesian_point. This “2d_point_occurrence” is certainly 2-dimensional, and is therefore different from the others, which are all 3d. The EXPRESS definition does not provide attributes for locating the point in space.

AP 202 describes point in English roughly the same as a STEP 3-dimensional cartesian point, but does not tell how to represent one in any modeling language.

AP 203 uses “point” in its ARM to correspond to “cartesian_point” in the AIM. The AP also uses “point” (as defined in STEP geometry) to denote the parent type of which “cartesian_point” is a subtype. The text states *a cartesian_point must have a z coordinate value*, but this rule does not seem to be implemented in EXPRESS. The AIM short form says to use cartesian_point from the STEP geometry schema. The AIM long form does not define cartesian_point.

In AP 204, the “point” entity of the ARM is the “cartesian_point” of the AIM. The x, y, and z values of a cartesian_point are real numbers.

In AP 205, the “cartesian_point” entity of the ARM is the “cartesian_point_205” of the AIM. The x, y, and z values of a cartesian_point are real numbers. The entity name in a physical file is “crtpnt_205.”

AP → / *entity* ↓	**201 Explicit Draughting**	**202 Associative Draughting**	**203 3D Design Data**	**204 Boundary Represen-tation**	**205 Surface Represen-tation**
cartesian point	2d_point _occurrence 16, 54, 56 79, 104	point 14, 54	40, 42 109, 118, 162 point 9, 28, 40, 134	24, 48 point 17, 24	23, 30, 94, 102 cartesian _point_205 16, 23, 32, 34, 36, 42, 48, 54, 60
direction	16	direction_vector 14, 54	40, 42 118, 162 direction_vector 28, 40, 134	24, 29, 49 normal_direction 24	43 direction_205 16, 43, 49, 55, 64
axis2_ place-ment		axis_system 15, 54	41 118, 162 axis_system 9, 28, 41, 134	29, 49	21, 41, 46, 59
line	2d_straight _line 16, 52, 75	15, 54	9, 28 41, 134	22, 24 28, 46	11, 41, 46 69, 95, 103 113
circle	60, 86, 87	15, 54	9, 28, 41 118, 134, 162	22, 24 28, 47	10, 41, 46 61, 95, 103 113

Notes

1. If the AP has a different name for the same conceptual entity, the name is listed in the appropriate box, and the page numbers below the new name refer to it.

Table 1. References to Five Geometry Entities

D.2 Direction

In AP 201, the word "direction" is mentioned once and explained nowhere, so it is not possible to tell what is intended, but it is almost surely 2-dimensional, and therefore different from the others, which are all 3d.

AP 202 describes "direction_vector" in English as roughly the same thing as a STEP "direction", but does not tell how to represent one in any modeling language.

The ARM of AP 203 uses "direction_vector" to mean what is called "direction" in the AIM. The text states *a direction must have a z value*, but this rule does not seem to be implemented in EXPRESS. The AIM short form says to use direction from the STEP geometry schema. The AIM long form does not define direction.

The ARM of AP 204 uses "normal_direction" for the "direction" of the AIM, which is the same as a STEP geometry "direction", with the limitation that the z_coordinate must exist.

In AP 205, the "direction" entity from STEP geometry is used with the name changed to "direction_205" and the limitation that the z_coordinate must exist.

D.3 Axis2_placement

For AP 201, this is the only blank box in Table 1. This AP deals with translations and rotations using entities called "position", "scale" and "transformation" [Haas, page 98]. The axis2_placement could have been used, but wasn't. This AP deals with scaling, while the others do not, so axis2_placement is not quite sufficient.

AP 202 describes "axis_system" in English as something similar to a STEP "axis2_placement", but does not tell how to represent one in any modeling language.

The ARM of AP 203 uses "axis_system" to mean what is called "axis2_placement" in the AIM. The AIM short form says to use axis2_placement from the STEP geometry schema (although it is misspelled). The AIM long form does not define axis2_placement.

AP 204 uses "axis2_placement" from STEP geometry, except that the directions, "axis" and "ref_direction," are mandatory, not optional.

AP 205 uses "axis2_placement" from STEP geometry, except that the rules for writing a physical file use "axsplz" as the entity name.

D.4 Line

AP 201 includes a "2d_straight_line". This "2d_straight_line" is certainly 2-dimensional, and is therefore different from the others, which are all 3d. The EXPRESS definition is a stub.

AP 202 describes "line" in English as a line segment, but does not tell how to represent one in any modeling language.

AP 203 uses "line" in its ARM to refer to a line segment, but does not use the definition of "line" given in the STEP geometry schema by any name. The "polyline" entity must be used to represent any line segment, and unbounded lines cannot be represented. This is implicit in the EXPRESS, diagrams, and tables of the AP, but is not explained in text anywhere in the AP. The word "line" does not even appear as an entry in the index, which is otherwise quite detailed.

In AP 204, there appears to be an error in the rules for writing “line” entities to a physical file (&SCOPE and ENDSCOPE seem to have been inadvertently omitted). The rest of this appendix assumes this error has been corrected. The line is as described in STEP geometry.

The line in AP 205 is as described in STEP geometry.

D.5 Circle

AP 201 mentions “circle”, but gives not the least clue as to what is intended. It is not described in English or any modeling language. The AP includes an entity named “bounded_2d_conic”, [Haas, pages 52, 76] which probably subsumes circular arc, but it is never explicitly stated. The AP mentions entities called “diameter_dimension” and “radius_dimension” [Haas, pages 17, 19 58, 90], but makes no connection between those entities and either circles or bounded_2d_conics. Whatever is intended here, it is surely a 2-dimensional thing.

AP 202 describes “circle” in English, but does not tell how to represent one in any modeling language.

In AP 203, the short form of the AIM says to use the version of circle given in the geometry schema. The AIM long form does not define circle.

AP 204 uses a real number for the radius of a circle, while 203 and 205 use length_measure. The radius position in the file must contain a real number.

In AP 205, the short form of the AIM says to use the version of circle given in the geometry schema, but the AP gives its own mapping to a physical file, which differs from the STEP mapping. The radius position in the file must contain an entity name.

D.6 Conclusions

All five entities were found in all five APs, with the one exception that axis2_placement is not in AP 201.

The versions of these five entities in AP 201 are all 2-dimensional, and are thus necessarily conceptually different from the versions in the other 4 APs.

The other 4 could have used identical versions of each of the five entities.

AP 202 has no EXPRESS yet, and does not define any of these entities in any other modeling language, so it is impossible to tell what an instance of any of them in physical file prepared according to AP 202 might look like.

Thus it is only possible to consider physical files prepared according to three of the five APs: 203, 204, and 205

For cartesian_points, directions and circles, APs 203 and 204 can deal with instances in each other’s physical files, but those two are both incompatible with AP 205.

For axis2_placements, APs 203 and 204 can deal with most instances in each other’s physical files. The one incompatibility is that a file reader programmed to read files according to AP 204 should signal an error if a “ref_direction” were not provided in a 203 file. APs 203 and 204 are both incompatible with AP 205.

For lines, APs 204 and 205 can deal with instances in each other's physical files (ignoring name differences for cartesian_points and directions which might be in the scope sections), but those two are both incompatible with AP 203, which does not use lines.

This is a terrible mess. In two of the five APs, it is not possible to tell what is intended when these entities are used, and none of the entities is the same across the three APs which do provide understandable definitions.

There does not appear to be any reason why the same version of these five entities should not be used in the four APs which are 3-dimensional.

The APs should not be approved in current form. AICs should be developed to provide a single definition of these five entities (and many others) which can be used by reference in each of these APs (and any others developed in the future). The APs should be revised to use the AICs.

E APs for the STEP Production Cell

The STEP Production Cell intended here is the one described in [Fowler], which is planned to consist of seven major subsystems: design, process planning, equipment programming, machining workstation, inspection workstation, STEP Data Repository, and Network Communications. As stated in [Fowler, page 3], *the STEP Production Cell will help verify that the STEP standard is workable through production level testing.* One function of the SPC will be to test STEP APs.

E.1 AP Hierarchy

It is useful to think of the SPC AP suite as a hierarchy, with a single top-level AP for the SPC. The top level consists of middle-level APs corresponding to the SPC subsystems which require STEP data. That is, one each for at least design, process planning, and numerical control (equipment) programming, as discussed in [Stark]. Middle level APs for the machining and inspection workstations also seem desirable. The middle level APs would consist of a collection of lower level APs, as described below. In general, each AP from the lowest level would appear in several middle-level APs.

E.2 Strategy for SPC AP Development

E.2.1 Bootstrap

There is a chicken-and-egg problem with STEP APs. The nature of the AP development process and of APs themselves is still changing rapidly. It would be foolish to prepare highly polished APs following the current procedures, because the procedures are quite likely to change, making the APs obsolete. It will not be possible to have stable procedures, however, until it can be determined that the procedures work. This means using APs in software systems, but the software systems cannot operate without the APs.

This situation dictates a bootstrapping strategy, and provides the opportunity for the SPC to contribute substantially to refining the AP development process. The essence of the strategy is for NIST to start by developing a few unpolished APs for the SPC rapidly, develop software for some portions of the SPC to use these APs, run the software to test the APs, and feed back the results of the tests both for refining the APs and for refining the AP development process. After the first round of APs and software has been built and tested, another round would be undertaken, with AP development preceding software development by a short time.

E.2.2 Coordinate

One objective of developing APs for the SPC should be to develop APs that could be used by the rest of the world. At no stage of development should it seem impossible for an AP for the SPC to evolve to the stage of being generally useful. Some SPC APs might evolve into STEP APs, but probably most will be put aside when STEP APs covering the same area become available. The SPC APs should be designed so that that transition will be smooth.

If STEP APs are available in areas of interest to the SPC, even in draft form, the SPC should use them if possible.

The acid test of STEP is whether data generated at one site can be used at another site using different software. Two sites cannot exchange STEP data and have it be usable unless both are using the same AIM and usage guides for the data. In developing APs for the SPC, NIST should try to get enough low level APs in common with other sites so that data exchange tests can be conducted with other sites early in the life of the SPC. The RAMP project is a prime candidate for a distant site with which to coordinate.

E.2.3 Evolve

At the outset, NIST should lay out the entire suite of APs expected to be required. A first cut at this is made below. The contents of the suite may be expected to evolve as STEP APs and SPC APs are developed, implemented, and tested. Radical changes may be expected as a result of this evolution.

So that tests involving production of metal parts can be carried out early in the life of the SPC, the initial focus of SPC development should be on the software and data most essential for production. Data for workstation and shop levels of control can be deferred, as can scheduling and resources management data. Resource description data (fixtures, stock, workpieces, tooling) will be needed early.

E.3 What the SPC Should Be

The subsystems of the SPC are described in [Fowler], but the nature of the SPC is not. NIST should view the SPC broadly as a prototype of a small, general-purpose machining shop, (henceforth shortened to “shop”) and account for the data required by all activities of a shop in the APs developed for the SPC.

The current design of the SPC includes only one metal-cutting workstation. In general, a shop is expected to have several metal-cutting workstations. To avoid building a one-station bias into the SPC APs, and since RAMP focuses on turned parts, the SPC APs should be developed under the assumption that the SPC might include a turning workstation.

E.3.1 Model

A model of the physical resources, human resources, and activities of a shop should be formulated at the outset and serve as a testbed for thought experiments on application protocols. The shop model should be complete at a coarse level of granularity, but only those parts of the model which are to be implemented in the SPC will be completed at finer levels. The remainder of section E.3 gives an overview of the model.

E.3.2 Business

The business of a shop is to produce small batches of piece parts by machining, and to produce data which can be used by other shops so the other shops can produce parts or data. The shop has the capability to do cleaning, deburring and some surface or bulk treatments on parts which have been machined. The shop does not do any casting, sheet metal forming, forging, rolling, punching, powder metallurgy, bending, joining (welding, soldering, gluing, etc.), composite forming, or assembly.

E.3.3 Quality and Efficiency

The shop is able to make parts right on the first try. To accomplish this, it uses the results of inspection and in-process measurements to improve its operations, both by real-time control and by modifying its business practices.

The shop strives to be efficient. To do this, it takes time and cost into account in its decisionmaking, and measures time and cost to provide data for decisionmaking.

E.3.4 Input

The input a shop can handle includes:

1. design information in any form - from a STEP B-Rep to a customer with an idea in mind,
2. any or all of the intermediate data needed to produce a part from a design, such as process plans, NC-programs, setup descriptions, and descriptions of volumes of material to be removed,
3. orders for parts for which the design is already on hand (possibly with changes),
4. raw stock (bar, plate, tube, etc.) or near-net-shape workpieces (castings, in particular) and other supplies, and
5. fixtures and cutting tools.

It is expected that any design may include specification of materials, tolerances, surface finishes, and treatments, as well as nominal shape. It is not expected that a design will include any information regarding functionality.

E.3.5 Resources

The specialized resources of a shop include

1. hand tools,
2. manually controlled machine tools,
3. numerically controlled machine tools,
4. a staff of machinists with a moderate to high level of expertise in operating all the tools, and
5. computing and communications equipment and software.

The numerically controlled machine tools are 3-axis machining centers or turning centers without turrets or active tooling. The shop does not have numerically controlled tools of the following types: drilling, grinding, wire EDM, turning centers with turrets or active tooling, 4-axis or 5-axis machining centers, laser-cutting, flame cutting, broaching, shaping, slotting, hobbing, or gashing. For the purposes of making any one batch of parts, the resources of the shop are regarded as static.

The shop also has the resources for moving workpieces between workstations and within workstations.

E.3.6 Materials

The shop is prepared to machine commonly machined homogeneous materials which can be handled by conventional techniques on conventional machines. This includes many metals (common varieties of iron, steel, aluminum, copper, bronze, and brass, for example) and some plastics. It does not include non-homogeneous materials such as wood or composites such as those made from fiberglass or graphite fibers impregnated with resins. It does not include hard-to-machine materials such as titanium or carbides. It does not include non-machinable ceramics, glass, elastomers (such as rubber), hazardous materials, (such as sodium or plutonium), or materials whose value justifies special handling (such as gold or platinum).

E.3.7 Range of Part Size and Shape

The range of part size (including weight) and shape (including tolerances) should follow from the capabilities of the NC-machines in the shop. Since different machines can handle different ranges of shapes, a set of shape ranges should be constructed, one for each machine type. Tolerance ranges will depend upon the type of feature to which the tolerance applies. For example, on an NC-machining center, it is difficult to mill a complex pocket and keep the position of all walls within 0.002 inch of their nominal location, but it is relatively easy, using a ream, to make a hole whose diameter is within 0.0005 inch of its nominal value.

Size ranges are a more difficult problem. The only real constraints are: is the fixturing that holds the workpiece strong enough to bear the weight of the workpiece and machining forces, and will the workpiece fit into the space available. Determining whether these constraints can be met goes beyond an examination of the machine being used. For example, it is possible to drill a hole in an I-beam ten meters long on a machine whose work space is one meter long, provided that the machine is in a large enough room.

E.3.8 Shop Organization

The shop has:

1. a business office which handles all dealings with customers, shippers, utilities, banks, governments, etc., and tells the resource office and manufacturing office what to do,
2. a resource office (which covers what is usually divided among a tool crib, a stockroom, and a data systems office)
3. a manufacturing office that handles the production of parts and data, including scheduling and real-time shop control for all parts of the shop except the business office and resource manager,
4. data preparation stations which can perform design, process planning, and NC-programming,
5. two or more machining workstations,
6. an inspection workstation.

The shop may also have:

1. a cleaning and deburring workstation and
2. one or more treatment workstations (anodizing, shot peening, polishing, annealing, knurling, etc.).

The model does not require that the business, resource, and manufacturing offices be physically separate, but only that the shop carry out the functions of those offices. An actual shop should be able to have only one machining workstation and still be able to use data prepared according to the SPC APs, but the model should provide for two or more.

E.3.9 Process Plans and NC-Programs

Each control system at each hierarchical level in the shop (the shop itself, the workstations, and equipment within the workstations) operates by carrying out a process plan, provided that, at the equipment level, it is usually the case that the process plan must be converted into an NC-program, since most equipment controllers are built to carry out NC-programs, not process plans. For each control system in the shop, a library of operations which can be carried out on that system will be defined. Only the operations in the library can be included in a process plan for a system of that type. A single general-purpose process planning language, such as the one being developed in STEP [Paul], will be used in conjunction with the libraries. The general-purpose language will provide for expressing the order of operations, alternative operations, timing, physical resource requirements, etc.

In conjunction with process plans, it is expected that shape information will be prepared to express:

1. the shape of volumes of material to be removed by machining operations,
2. the shapes of workpieces,
3. the shapes of fixtures,
4. the shapes of workspace environments, and
5. the shapes of cutters and holders.

E.3.10 Work Flow

If a complete design is received as input, but no manufacturing data is received, the normal course of work flow for making a part from the design would be as follows (ignoring scheduling, data storage and retrieval, cleaning and deburring, and feedback). To simplify, the following example deals with a batch size of one for a part requiring machining in one fixturing in one workstation. The system must be able to deal with cases in which all of these are more than one, of course.

1. Process plans for manufacturing the part are produced by process planning. This includes at least one shop-level plan, one machining plan, and one inspection plan. Process planning also defines the initial workpiece, fixturing, setups, and material removal volumes. If a special fixture is required, it will be designed at the design station and be processed like another part.
2. An NC-program for a machining center is generated from the data prepared by process planning. An NC-program for inspection may also generated.
3. The initial workpiece, required fixture, and required cutters (with their holders) are obtained from the resource manager and moved to the workstation.
4. The workstation is set up according to the setup data.
5. The NC-program is run to machine the part.
6. The part is moved to inspection.
7. The part is inspected manually or using an NC-program
8. The part is moved out of the manufacturing area.

E.3.11 Feedback

Feedback from one subsystem to another is common in the shop. Feedback is of two broad types: feedback of status information from a subordinate controller to a superior controller, and feedback of information about the effectiveness of manufacturing. In the second category, the SPC APs will provide for data that provides a reverse trace from each feature of the finished part (or intermediate workpiece) to the manufacturing processe(s) which produced the feature. This will make it possible for the results of inspection to be used to change the rules or algorithms used for process planning.

Various types of change requests are used, such as requests from process planning or NC-programming for changes in designs, requests from NC-programming for changes in process plans, and requests from workstations for changes in process plans, fixtures, workpieces, setups, or NC-programs

E.4 Data Needs in the SPC

This section looks at the data needs of the subsystems of the SPC. The STEP Data Repository, and Network Communications subsystems specified in [Fowler] are not expected to require any STEP data for operation (the Data Repository will <u>store</u> STEP data), so they are not discussed below.

Much data used in the SPC will conform to the SPC APs, but some will not, either because a standard format is not required or because a non-AP standard is to be used. NC-program data, in particular, will not conform to any AP, since there are already standards for high-level NC-program languages, and low level languages are machine-specific.

The SPC will be doing data access continually. The SPC may be used to test STEP data access concepts, such as the STEP Data Access Interface (SDAI), but that is independent of APs, and this section does not deal with data access.

In the intermediate term, the SPC may run using a schedule and real-time control systems for the cell and the workstations. This will require data for commands, command status, orders, order status, station status, and schedules. It is not expected that this data will be in STEP format.

In the long run, the SPC could include a subsystem for controlling data preparation. If such a system is added, the design, process planning, and equipment programming systems will have to be able to accept data preparation commands and return status messages regarding the commands. It is not expected such data would be in STEP format.

The remainder of this section summarizes the table which follows. Except as specified otherwise, all types of information listed below are expected to be prepared in machine-readable form according to an AP for that type of information for all the purposes listed in the table. Some types of information will be subject to AIC requirements rather than AP requirements, but these are not differentiated below.

In this appendix “elementary B-Rep means a boundary representation with only elementary curves (lines and conics) and surfaces (plane, sphere, cylinder, cone, torus). “Advanced B-Rep” means elementary B-Rep plus intersection curves, b-spline curves, and b-spline surfaces.

E.4.1 Design

E.4.1.1 Essential in First Implementation

The design system takes in design information in any non-STEP human-readable form and in machine-readable forms as follows: nominal shape in elementary B-Rep form, tolerances, materials, and surface finishes. The design system produces machine-readable output in the same forms.

E.4.1.2 Desirable Soon

The system takes in nominal shape information in several additional forms: facetted (polyhedral) B-Rep, advanced B-Rep, implicit form features, and explicit form features. Other types of information can also be handled: design change request, drafting, presentation, FEM, and some treatments.

E.4.1.3 Desirable in Long Run

The system takes in nominal shape information as: csg, octree, or partial shape. The system can handle functional specifications, kinematics and assembly analysis.

E.4.2 Process Planning

Each process plan will conform to two APs, one domain-independent, and one domain-dependent.

E.4.2.1 Essential in First Implementation

The process planning system takes in: nominal shape information in elementary B-Rep form, tolerances, materials, surface finishes, and resource descriptions (tooling [cutters, probes, and holders for cutters and probes], workpieces, and fixtures). The system produces material removal volumes, setups, and process plans for 3-axis machining and inspection as output. It can also accept these plans as input.

E.4.2.2 Desirable Soon

The system takes in nominal shape information in several additional forms: facetted (polyhedral) B-Rep, advanced B-Rep, implicit form features, and explicit form features. The system handles process plans for a turning center, for workstations, and for the cell. The system handles some treatments and change requests for design, process plan, material removal volume, and setup.

The system uses data about stock, but not in STEP format. Data about stock is used by the process planning system for generating a document for the stock room which tells how to make a workpiece from stock. When the stock room cuts a piece of stock for use in making a part, it is called a workpiece thereafter.

E.4.2.3 Desirable in Long Run

The system takes in nominal shape information as csg or octree.

E.4.3 Equipment Programming

E.4.3.1 Essential in First Implementation

The equipment programming system takes in: nominal shape information in elementary B-Rep form, tolerances, materials, surface finishes, resource descriptions (tooling [cutters, probes, and holders for cutters and probes], workpieces, and fixtures), material removal volumes, setups, and process plans for 3-axis machining and inspection. The system produces NC-programs for 3-axis machining and inspection as output. The NC-programs are written in high-level or machine-specific languages, but are not in STEP format.

E.4.3.2 Desirable Soon

The system takes in nominal shape information in several additional forms: facetted (polyhedral) B-Rep, advanced B-Rep, implicit form features, and explicit form features. The system prepares NC programs for turning. The system produces changes requests for designs, process plans, material removal volumes, and setups.

E.4.3.3 Desirable in Long Run

The system takes in nominal shape information as csg or octree.

E.4.4 Cell Control

E.4.4.1 Essential in First Implementation

The cell is not needed in the first implementation.

E.4.4.2 Desirable Soon

The cell handles general process plans, workstation plans, and cell plans.

E.4.5 Machining Workstation

E.4.5.1 Essential in First Implementation

The workstation takes in NC-programs, setups, and resource descriptions (tooling [cutters, probes, and holders for cutters and probes], workpieces, and fixtures).

E.4.5.2 Desirable Soon

The workstation takes in: designs (in elementary B-Rep, implicit form feature, and explicit form feature form), material removal volumes, and process plans for itself. The workstation produces change requests for design, process plan, material removal volumes, and setup.

E.4.6 Inspection Workstation

E.4.6.1 Essential in First Implementation

The workstation handles setups, and resource descriptions (tooling [cutters, probes, and holders for cutters and probes], workpieces, and fixtures). The workstation handles NC-programs.

If inspection is not done using an NC-program, then the shape, tolerance, and surface finish data which equipment programming would have handled will be handled in the inspection workstation, instead (this is not reflected in the table).

E.4.6.2 Desirable Soon

The workstation takes in: designs (in elementary B-Rep, implicit form feature, and explicit form feature form), material removal volumes, and process plans for itself. The workstation produces change requests for design, process plan, material removal volumes, and setup.

DATA USE IN STEP PRODUCTION CELL (1 of 2)

	design	process planning	equipment program-ming	cell control	machining work-station	inspection work-station
ESSENTIAL IN FIRST IMPLEMENTATION						
nominal shape						
elementary B-Rep	***x***	***x***	***x***			
material removal volumes		***x***	***x***			
setup		***x***	***x***		***x***	***x***
tolerances	***x***	***x***	***x***			
surface finishes	***x***	***x***	***x***			
materials	***x***	***x***	***x***			
process plans						
general		***x***	***x***			
3-axis machining		***x***	***x***			
inspection		***x***	***x***			
physical resources						
tooling		***x***	***x***		***x***	***x***
workpieces		***x***	***x***		***x***	***x***
fixtures		***x***	***x***		***x***	***x***
DESIRABLE AFTER FIRST IMPLEMENTATION COMPLETE (in addition to above)						
nominal shape						
elementary B-Rep					*x*	*x*
material removal volumes					*x*	*x*
facetted B-Rep	*x*	*x*	*x*			
implicit form features	*x*	*x*	*x*		*x*	*x*
explicit form features	*x*	*x*	*x*		*x*	*x*
advanced B-Rep	*x*	*x*	*x*			
drafting	*x*					
presentation	*x*					
FEM	*x*					
process plans						
general				*x*		
turning		*x*	*x*			
workstation		*x*		*x*	*x*	*x*
cell		*x*		*x*		
change requests						
design	*x*	*x*	*x*		*x*	*x*
process plan		*x*	*x*		*x*	*x*
material removal volumes		*x*	*x*		*x*	
setup		*x*	*x*		*x*	*x*
treatments	*x*	*x*				

DATA USE IN STEP PRODUCTION CELL (2 of 2)

	design	process planning	equipment program-ming	cell control	machining work-station	inspection work-station
DESIRABLE IN LONG RUN (in addition to above)						
nominal shape						
csg	**x**	**x**	**x**			
octree	**x**	**x**	**x**			
partial shape	**x**					
functional specifications	**x**					
kinematics	**x**					
assembly analysis	**x**					
ESSENTIAL, BUT NOT IN STEP FORMAT						
NC-programs			x		x	x
DESIRABLE, BUT NOT IN STEP FORMAT						
physical resources						
stock		x				
manufacturing commands				x	x	x
manufacturing command status				x	x	x
orders				x	x	x
order status				x	x	x
station status				x	x	x
schedules				x	x	x
data preparation commands	x	x	x	x		
data preparation status	x	x	x	x		
inspection reports	x	x	x			x

F AP Classification Using N-Spaces

One classification method which may be useful is to view the scope of an AP as covering some portion of an N-dimensional space. N is the number of axes deemed required. Although N may be any integer, more than 3 axes cannot be visualized, and fewer than 3 is not enough, so 3-space is generally used. The general approach is illustrated in Figure 8, which shows the situation for N=3. The axes of the space will each represent some attribute that products can have. The set of axes should be orthogonal, in the sense that it is possible to move a point in the N-space parallel to any one axis (i.e. change values on that axis) without changing values along any other axis.

Some of the axes may represent continuous variables, such as time or mass, while others may be discrete (or nearly so), such as engineering discipline. To make use of N-spaces, it is convenient to divide them into N-cells. In order to do this, any axes representing continuous variables must be divided into discrete parts. Time, for example, might be divided into years. It is also useful to set limits on any unbounded variable in order to deal with an N-block in N-space, rather than an infinite hunk of N-space. Since each edge of the N-block is divided into a finite number of pieces, the N-block itself will be divided into a finite number of N-cells. In Figure 8, for example, attribute A has a range of 5 parts, B a range of 7 parts, and C a range of 4 parts, giving a total of 140 cells.

A point in N-space describes a particular situation. Points in N-space do not necessarily have counterparts in the real world, however, as not every possible situation occurs. The “volume” of an N-cell is typically a meaningless concept, since the unit of “volume” is the product of the units on the axes, which (unlike ordinary volume units such as cubic meters) has no mapping to the physical world.

The N-block and N-cells might be used for AP classification by considering the activities that occur in the situation represented by each cell or the data used by those activities. In Figure 8, for example, a single cell has been highlighted in gray. Its projections on the sides of the N-block are also shown in gray.

It is often useful to consider what happens in entire subspaces of the N-block, where some variables have fixed values, but some others vary over their entire ranges. In Figure 8, for example, a subspace may be one-dimensional or two dimensional. A two dimensional subspace is a slice of the N-block such as the one whose sides are shown cross-hatched on Figure 8. There is no general algorithm, however, for deciding which subspaces are interesting.

It does not appear to be useful to try to locate entities from STEP information models in N-space (as has been done on page 32 of [Kirkley], for example) because the usefulness of an entity is not generally limited to one point or one cell in N-space. Rather, an entity may be expected to be useful in entire subspaces of N-space, or possibly in some collection of cells which does not comprise a subspace, and may not even be connected.

In determining whether an N-space or a collection of N-spaces might be useful for classifying APs, the axes that might be used must be determined, and sets of mutually orthogonal axes need to be considered.

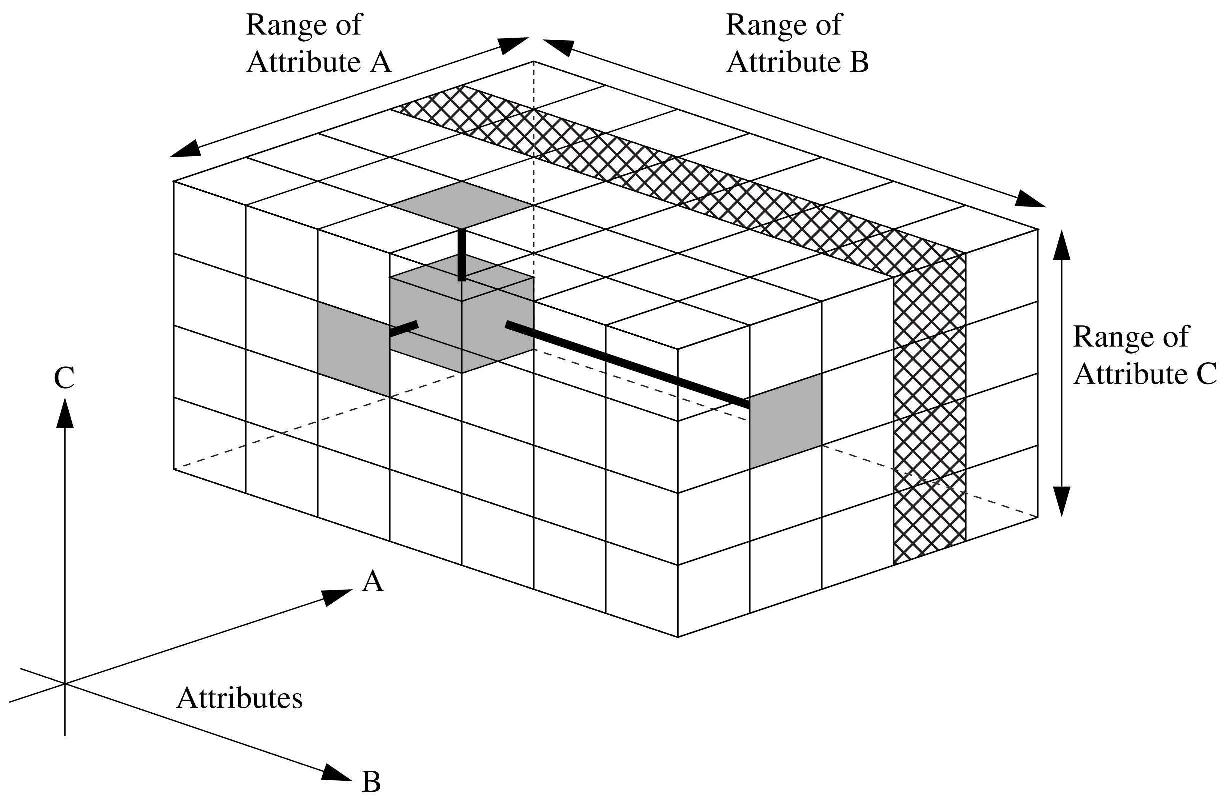

Figure 8. N-space

N-spaces have several drawbacks:

1. The number of cells becomes unmanageable very quickly as the number of axes increases. Six axes with ten subdivisions each yields a million cells, for example.

2. An N-Space with more than three axes cannot be drawn or visualized readily.

3. Some axes, such as “Type of Product Data” in Figure 6, have no natural order. Depending upon the choice of ordering, an AP may cover a single block in 3-space, or it may cover several separate blocks. Being split into parts gives an impression of fragmentation that may be purely an artifact of the choice of an ordering.

4. In the case of AP classification, the choice of axes is not clear-cut; no single choice stands out as the right choice.