

Introduction to the AP233 Committee Draft

What Systems Engineering Is

Systems Engineering is “An interdisciplinary collaborative approach to derive, evolve, and verify a life cycle balanced system solution that satisfies customer expectations and meets public acceptability. (IEEE 1220-1984)”

“Systems Engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem. (INCOSE)”

The complete problem spans:

- Stakeholder Needs
- Requirements
- The Enterprise and Competitors
- Concept
- Design
- Design Optimization by Trade Studies
- Design Validation and Verification
- Manufacture
- Product Validation and Verification
- Test
- Maintenance
- Disposal
- Training and Support
- Cost and Schedule
- Program Management

“Systems Engineering integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept to production to operation. Systems Engineering considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs. (INCOSE)”

Systems Engineering makes extensive use of specification of all interfaces, performance analysis, and trade studies to ensure that the system is near optimal for and will be accepted by stakeholders and the marketplace. It is applicable to anything for which a well defined boundary can be established whether mechanical, electrical, electronic, information, chemical, biological, social or organizational.

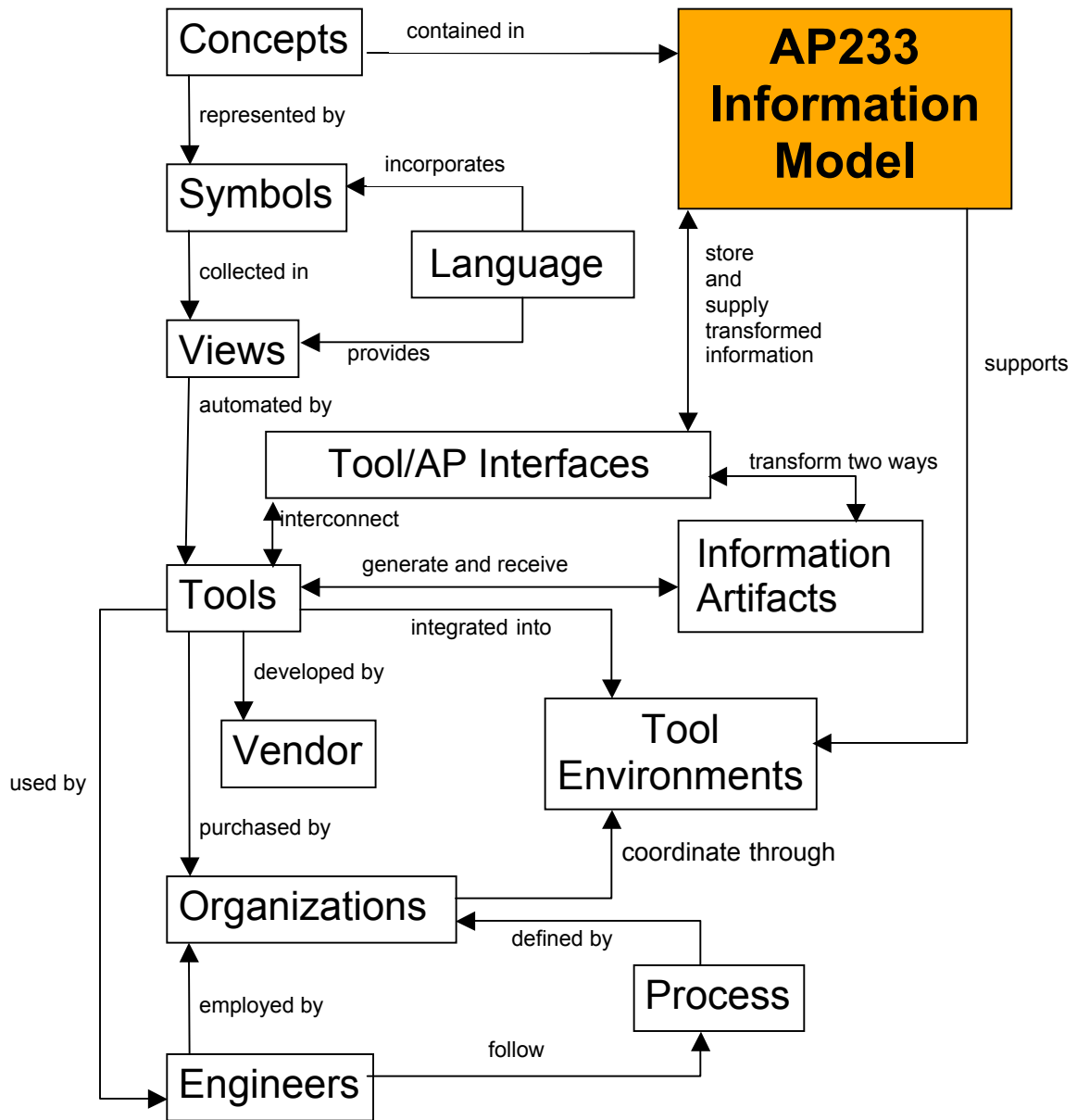


Figure 1. AP233 Information Model

What AP233 Is

AP233 is an information model that captures the concepts used in Systems Engineering, as Figure 1 shows. These concepts may be represented by a variety of symbols and each set of such symbols can be combined into an number of different views. A particular set of symbols and views constitutes a language. Tools automate the use of such a language to represent and model systems. For any language a number of different tools may exist and they may not be able to exchange data even using the same language unless they have compatible schema for storing that language. Thus a single set of concepts results in a plethora of languages and tools.

The tools are used by engineers. Frequently it is the organization that establishes the process, a specification of the views and development order that shall be followed, that the engineers shall follow. Two organizations using the same toll may or may not use equivalent views in doing their work.

In general there is no single tool that can support all the work that must be done so the tools are most efficient when put in an environment that enables them to exchange the information artifacts that the engineers produce using the tools. Often different tools incorporated in a single environment cannot exchange information. Very often the organizations in a supply chain or working together on a single project as a consortium will have different environments incorporating different sets of tools. Yet all members of the supply chain or consortium need to efficiently share data rigorously and at low cost.

The application protocol, like AP233 for Systems Engineering, provides a solution to this problem. Each tool needs an interface that interconnects the tool with the AP. The interface transforms the data artifacts from the tool for correct storage in the AP and transforms data supplied by the AP for use by the tool.

In this way AP233 supports information exchange among organizations even though their environments contain different tools. AP233 is the glue for organizations working together in a consortium or a supplier chain.

What AP233 Contains

Figure 2 shows the basic capabilities of AP233. It is divided into two main parts, Program Management and System Requirements/Design. Major capabilities are listed in

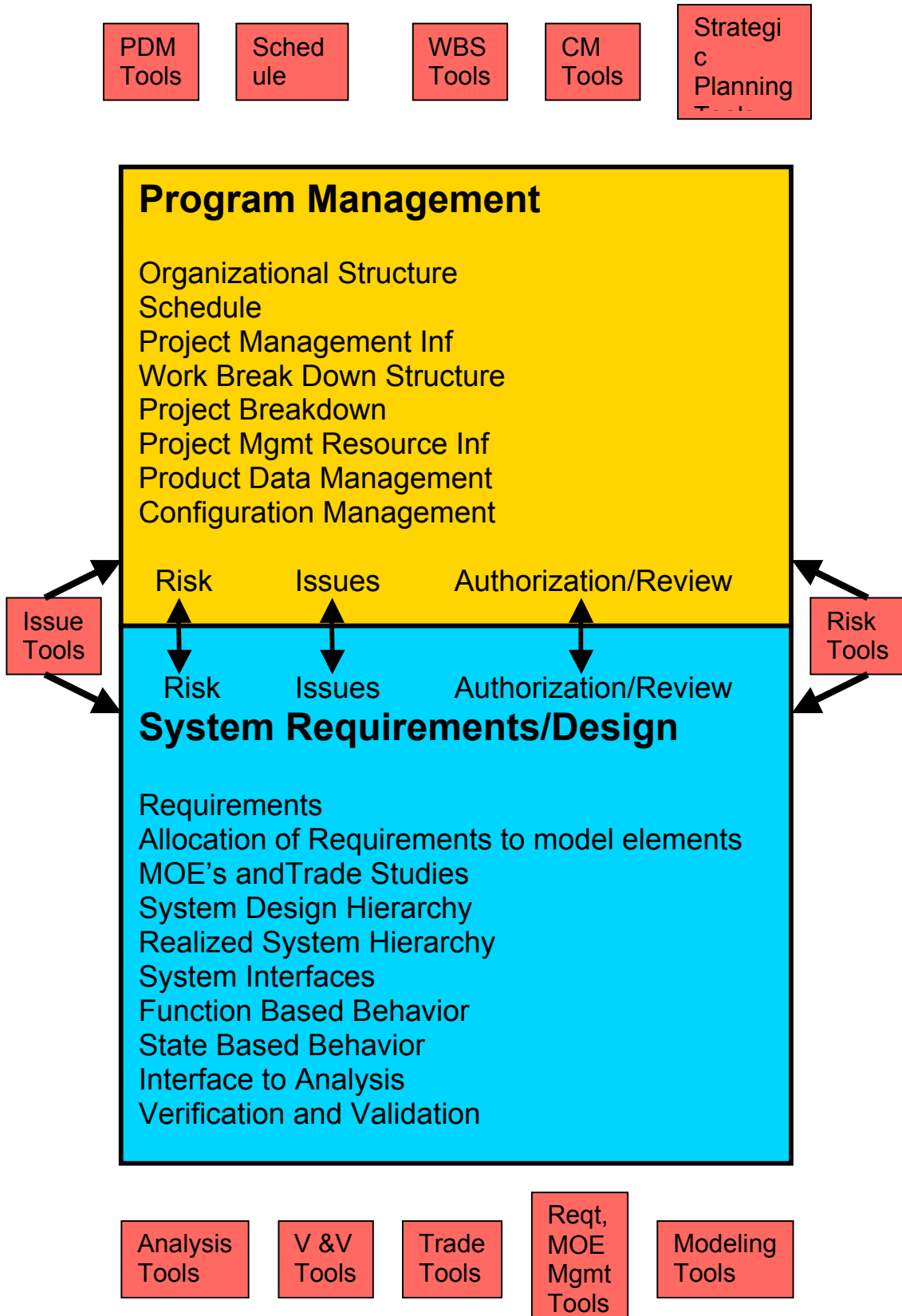


Figure 2 AP233 Content and Associated Tools

each box. The relevant kinds of tools that are applicable surround each box. The two main parts of the work are interconnected by risk analysis, issue resolution, and management authorization and review. Where these capabilities share concepts, i.e. where they are semantically equivalent, there can be exchange of information artifacts among any of the tools through the tool interfaces and the AP233 information model.

The same transfer is possible where there is semantic identity with downstream design, manufacture, test, and maintenance tools. There are many ISO 10303 Application Protocols (AP's) for a variety of industrial businesses such as, geometry, circuit board and chip development, ship building, building construction, furniture design and manufacture, maintenance, etc. A large number of different tools are used by these businesses and interface through their own AP's. All of these AP's have been made modular and share, as much as possible, the same modules. This approach is vital for connecting systems engineering tools to downstream tools and it makes possible the use of AP233 as an integrating AP across all the others.

The shared modularity has also required that the AP233 modules use naming conventions that were developed a decade or more ago by developers in other disciplines. The consequence is that AP233 contains names in the detailed EXPRESS code that will seem foreign to practicing systems engineers. It also means that many modules will use a different common module for widely needed representation of entities like number or a weighting factor. Thus the reader who looks at a particular module may not see there an entity that he knows belongs there because it resides in a shared module and is connected to the module he is reading. Efficient use of the EXPRESS language also means that the connection to the common module may go through a chain of classes and subclasses; it may be very indirect for the reader.

For the reasons above, this introduction does not attempt to show the reader the details of the EXPRESS code. Rather this introduction attempts to:

1. Describe AP233 as a container of concepts shared by tools
2. List number of the capabilities AP233 supports
3. Show kinds of tools that can interface with one another through AP233
4. Show the major mappings of capabilities onto the modules that constitute the information module.

Those who wish to investigate the modules in detail can use this Introduction as a guide to that adventure. Figure 3 shows a top level view of the module hierarchy. It is important to be aware that the boxes under a higher box do not necessarily completely compose it. Rather there may be additional things in the box above. This is not intuitive to those accustomed to hardware development, but it is a result of the software languages.

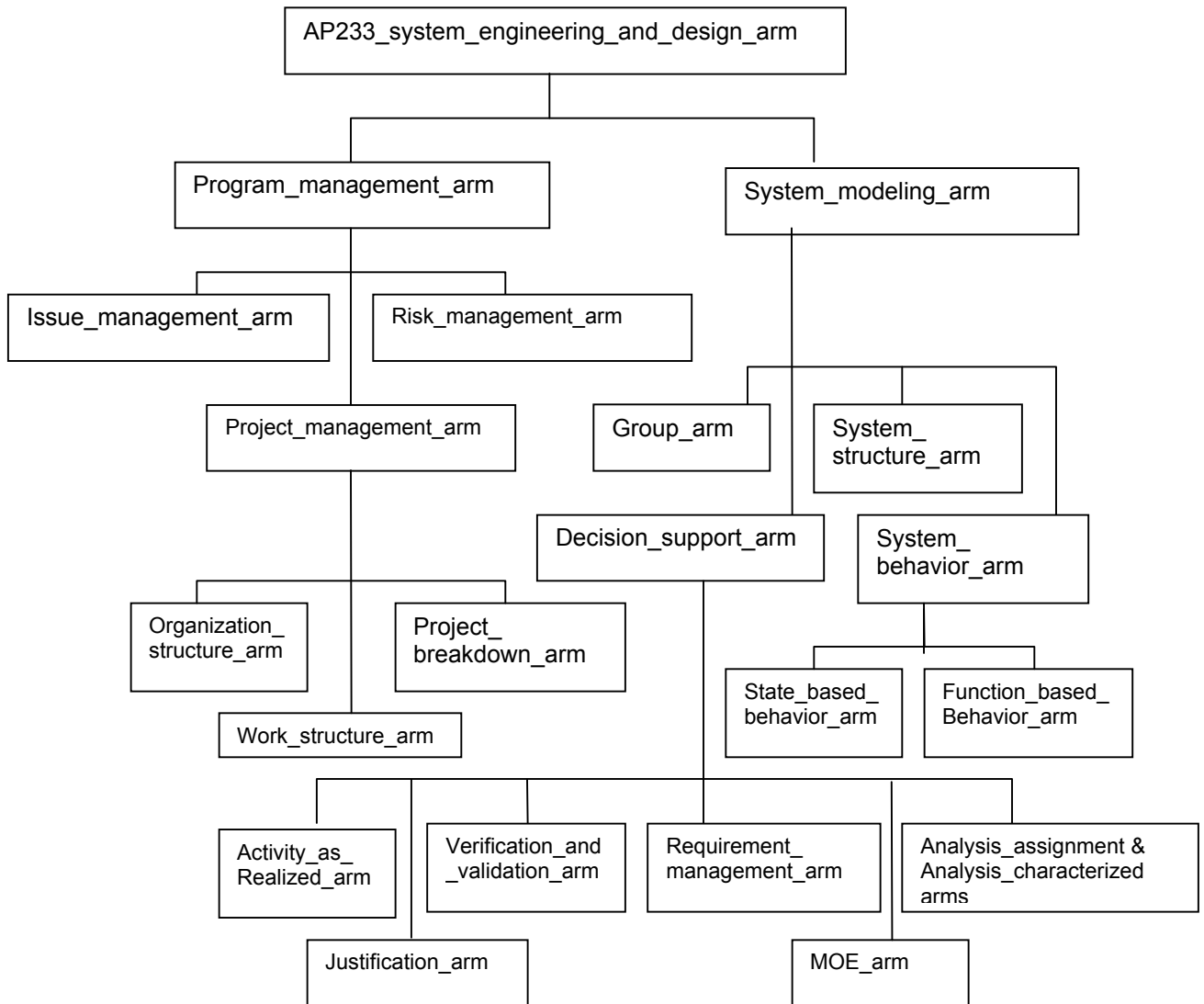


Figure 3. Top Level Hierarchy of AP233 Modules

The capabilities of Figure 2. map into the modules of Figure 3. are shown in Table 1. Be aware that these are not leaf cell modules. There may be structure below them, and they may connect to modules above and below through type extensions. This makes reading the code or the graphic EXPRESS-G diagrams difficult, but it is the nature of the language.

Program Management Capabilities	Module Name
Organization Structure	Organization_structure_arm
Project Management Information	Project_management_arm
Project Management Resource Information	Organization_structure_arm and Work_structuer_arm
Schedule	Project_breakdown_arm
Project Breakdown	Project_breakdown_arm
Configuration Management	Product_data_management_arm under the System_structure_arm
Work Breakdown	Work_structure_arm
Product Data Management, PDM	Product_data_management_arm under the System_structure_arm
Issues	Issue_management_arm
Risk	Risk_management_arm
System Requirements/Design Capabilities	Module Name
Requirements	Requirement_management_arm
Allocation of Requirements to Model Elements	Requirement_management_arm
MOE's and Trade Studies	MOE_arm
Interface to Analysis	Analysis_assignment_arm Analysis_characterized_arm
State Based Behavior	State_based_behavior_arm
Function Based Behavior	Function_based_behavior_arm
System Design Hierarchy	Next_assembly_usage in the Assembly_structure_arm under the System_structure_arm
Realized System Hierarchy	Next_assembly_usage in the Assembly_structure_arm under the System_structure_arm
System Interfaces	System_structure_arm & its subclass Interface_arm
Verification and Validation	Verification_and_validation_arm

Table 1. Mapping of capabilities to AP233 Modules

Within the System_structure_arm there are several modules that are used to create collections of things that may be hierarchies or networks (directed graphs) of information useful to the engineer. The superclass of these is the Product_breakdown_arm that can be used for any special purpose. It has three subclasses: the System_breakdown_arm, the Zonal_breakdown_arm, and the Physical_breakdown_arm. It is critical for the reader to note that these names have come from much earlier SC/4 work and not from the systems engineering community. The System_breakdown arm does not capture a breakdown of the system, but rather a special subset of systems for specialized purposes. The Physical_breakdown_arm is not the breakdown for the system into real parts made out of atoms. Rather it is a special collection. The

zonal_breakdown_arm is a breakdown into geometric regions of a system. This approach is often used in ship and aircraft development.

Introduction for Decision_Support_Arm

Figure 3. shows the hierarchy for the Decision_support_arm.

1. An Interface to analysis

It contains three arms related to an interface to analysis. They are: External_analysis_representation_arm, Analysis_assignment_arm, and Analysis_characterized_arm. These arms store the data a tool user needs to specify the equation he wishes to have solved, where in the systems models to pick up the needed parameter values, and where to return the computed values. This capability underlies all of performance calculation, trade studies, and verification and validation. It is the quantitative heart of systems engineering and AP233.

2. It contains the arm that supports documentation for the justification of decisions.

3. It contains the arm for the data concerning verification and validation.

4. It contains the Requirement_management_arm that supports information rights, documents, requirement sources, classifications, effectivity, and requirement assignment.

5. Versioning and version relationships are needed not only by requirements, but also by many other elements of the total system model. Consequently they are defined once for an entity called "Product" and then inherited from that entity through the software mechanism of subclasses. The best way to think about Product is as Thing or Anything. In consequence, The versioning and version relationships are in an arm called Requirement_ID_and_version that inherits from arms Product_version_arm and then from Product_view_definition_arm.

6. The concepts of tracing among entities and collecting them into groups is also pervasive. Consequently the traceability and collection of requirements is supported by the Requirement_view_relationship_arm that inherits from the Product_view_definition_relationship_arm.

7. Trade studies are supported by the MOE arm. It directly contains MOE's, the direction of optimization (maximize or minimize) and the regularization function. Because the application of numeric weights to a collection of things is a general need in many places in AP233, weights are connected via the EXPRESS interconnection mechanism of type extensions. A similar mechanism is used to relate MOE's to the activity that is performed to generate the trade study. For similar reasons the traceability among MOE's and their collection into groups is obtained by making them a subclass of Requirement. This has the advantage of using the EXPRESS language in a compact way to get the traceability and collection features for MOE's, but also to allow derived requirements for a particular design solution to trace to the MOE's and the trade study that caused its selection.