Chapter 2

Overview of the Systems Engineering Design Process

2.1 INTRODUCTION

This chapter provides a quick tour of many of the major concepts found in Chapters 6 to 11. This tour is quite valuable as part of an academic course on the engineering of systems because this chapter provides the context for the detailed discussion to follow. However, the advanced reader may wish to skip this chapter.

Section 2.2 addresses the processes for design and for integration and qualification. Included here are definitions for key terms such as system, function, and external system. Section 2.3 describes many of the key concepts of the design and integration processes. Included in the design process are the concepts of operational concept, external system diagram, objectives hierarchy, requirements, functions, items, components, and interfaces. Verification, validation, and acceptance are discussed as part of the integration and qualification process. Finally, Section 2.4 introduces the software product CORE, which is a systems engineering tool used in selected portions of this book to enable the student to practice and learn the many engineering concepts discussed here.

2.2 DESIGN PROCESS

This section begins by defining some key terms that set the stage for discussing the engineering of a system. Then a more detailed discussion of the two legs of the Vee process, design and integration (and qualification), are presented in more detail than in Chapter 1.

2.2.1 Key Terms

As part of this overview of the design process, we must establish some important definitions:

System: set of components (subsystems, segments) acting together to achieve a set of common objectives via the accomplishment of a set of tasks.

System Task or Function: set of functions that must be performed to achieve a specific objective.

Human-Designed System

- A specially defined set of segments (hardware, software, physical entities, humans, facilities) acting as planned
- via a set of interfaces, which are designed to connect the components,
- to achieve a common mission or fundamental objective (i.e., a set of specially defined objectives),
- subject to a set of constraints,
- through the accomplishment of a predetermined set of functions.

System's External Systems: set of entities that interact with the system via the system's external interfaces. Note in Figure 2.1, the external systems can impact the system and the system does impact the external systems. The system's inputs may flow from these external systems or from the context, but all of the system's outputs flow to these external systems. The external systems, many or all of which may be legacy (existing) systems, play a major role in establishing the stakeholders' requirements.

***System's Context*:** set of entities that can impact the system but cannot be impacted by the system. The entities in the system's context are responsible for some of the system's requirements. See Figure 2.1.

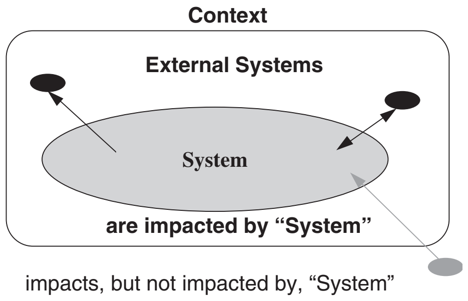

FIGURE 2.1 Depiction of the system, external systems, and context.

2.2.2 Design

As discussed in Chapter 1, design includes decomposition and definition of both the requirements, or statement of the design problem, and the architectures, functional and physical representations of the system. The allocated architecture addresses which physical resources of the system are going to perform which functions, but also includes a mapping of all of the requirements to the physical resources of the system. As well as addressing the system that must be operational for users, the design process should address all relevant systems needed during the life cycle of this system: the development system, of which the systems engineers are part; the manufacturing system, if needed; the deployment system, if needed; any training systems that are needed; a refinement system for system upgrades; and the retirement system, if needed. Finally, the qualification systems for each of these systems need to be addressed.

There are eight functions below that capture the complete systems engineering process. The first two (0a and 0b) are performed at the very beginning and are really part of a general problem solving process. The last six functions (1–6) are the focus of this book and are considered the core repetitive functions of the systems engineering design process:

0a Define the problem to be solved

0b Define and evaluate alternate concepts for solving problem

1. Define the system level design problem being solved
2. Develop the system functional architecture
3. Develop the system physical architecture
4. Develop the system allocated architecture
5. Develop the interface architecture
6. Define the qualification system for the system

All eight of these functions are shown in Table 2.1 with their major inputs and outputs. Chapters 6 through 11 address the last six functions, respectively. As can be seen by the respective inputs and outputs of these functions, these last six functions cannot be conducted in series but have to be concurrent. The resource that performs these functions is the systems engineering team.

The first of the repetitive design functions has to create a definition of the problem being solved for which the next five develop a set of designs (across the system's life cycle). The seven functions that comprise this first design function, defining the design problem (stakeholders' requirements), are:

1. Develop operational concept
2. Define system boundary with external systems diagram
3. Develop system objectives hierarchy
4. Develop, analyze, and refine requirements (stakeholders' and system)

TABLE 2.1 Functions of the Design Process

Design Function	Major Inputs	Major Outputs
Define Problem To Be Solved	Concerns and Complaints by Stakeholders Available Data from Stakeholders	Definitions of Measures of Effectiveness and Desired Ranges Constraints
Develop and Evaluate Alternate Concepts for Solving Problem	Ideas for Concepts from All Interested Parties Objective Hierarchy & Value Parameters for Meta-System	Recommended Concept(s)
Define System Level Design Problem Being Solved	Stakeholders' Inputs Operational Concept	Stakeholders' Requirements
Develop System Functional Architecture	Stakeholders' Requirements Operational Concept	Functional Architecture
Develop System Physical Architecture	Stakeholders' Requirements	Physical Architecture
Develop System Allocated Architecture	Stakeholders' Requirements Functional Architecture Physical Architecture Interface Architecture	Allocated Architecture
Develop Interface Architecture	Draft Allocated Architecture	Interface Architecture
Develop Qualification System for the System	Stakeholders' Requirements Systems Requirements	Qualification System Design Documentation

5. Ensure requirements feasibility
6. Define the test system requirements
7. Obtain approval of system documentation

These seven functions are shown with their major inputs and outputs in Table 2.2. There is an important distinction between the stakeholders' requirements and the system requirements. Stakeholders' requirements are those requirements that the system's stakeholders agree define their needs. As such, the stakeholders' requirements are written in the common language of the stakeholders (e.g., English, Chinese). The system requirements are a translation of the stakeholders' requirements into the appropriate engineering terminology (e.g., foot-pounds, bits, and decibels). Chapter 6 presents more detail about this process.

The detailed processes for developing the functional, physical, allocated, and interface architectures are not presented here because many of the concepts for

TABLE 2.2 Functions of the System-Level Design Problem

Design Function	Major Inputs	Major Outputs
Develop Operational Concept	Stakeholders' Inputs Objective Hierarchy & Value Parameters for Meta-System Recommended Concept	Operational Concept System Concept Input-output traces Meta-system MOEs
Define System Boundary with External Systems Diagram	Operational Concept	System Boundary, System's Inputs and Outputs
Develop System Objectives Hierarchy	Operational Concept Stakeholders' Inputs	System-level Objectives Hierarchy
Develop, Analyze and Refine Requirements (Stakeholders' and System)	Operational Concept System Boundary, Inputs & Outputs Objectives Hierarchy Stakeholders' Inputs	Stakeholders' & Systems Requirements
Ensure Requirements Feasibility	Stakeholders' & Systems Requirements SE Team's Inputs	Design Feasibility
Define the Test System Requirements	Stakeholders' & Systems Requirements Stakeholders' Inputs	Test System Requirements
Obtain Approval of System Documentation	Stakeholders' & Systems Requirements	Stakeholders' & System Requirements Documents

these architectures are not appropriate for this overview. The decomposition of the last function of design, developing the qualification system for the system, is a replication of the design process for the system but with the focus on the elements of the qualification system.

The design process, as presented here, is not a formal process in the sense that success can be proved, designs can be proved to be correct, and so forth. Some researchers have developed formal processes, primarily in software engineering. These formal processes have not succeeded in software development and are relatively rare in the engineering of systems. An example of such a formal process for engineering design is Suh's [1990] axiomatic design process. Suh defines two major concepts—functional requirements and design parameters. He posits two axioms: (1) independence axiom: maintain the independence of the functional requirements and (2) information axiom: minimize the information content of the design.

While Suh introduces hierarchical decomposition in his axiomatic process, there is not sufficient richness of concepts in his process to handle the complexity of the engineering issues associated with the development of a system. First, as will be discussed in Chapter 6, functional requirements are a derived entity that have no inherent meaning to the stakeholders; input and output requirements are statements that relate to stakeholders needs. Second, Suh's process does not provide a sufficient process to develop and enable validation of the requirements. Finally, the interaction of functions, components, and interfaces, as described in this book but is missing in some richness in Suh's approach, is needed to deal with the generation and analysis of design options, as well as guide the qualification of the system in terms of the stakeholders' needs.

2.2.3 Integration and Qualification

The second half of the Vee model, integration and qualification, is primarily a bottom-up process that comprises integrating the most basic building blocks of the system and verifying that these lower level components meet the specifications, or sets of requirements, that were developed for them during design. However, before this integration and verification process begins, a validation of the requirements development process should take place to attempt to demonstrate that the low-level design solutions are still consistent with the stakeholders' needs. At the end of qualification come the important steps of validating that the system that has been designed and verified does in fact agree with the operational concept and is acceptable to the stakeholders. The stakeholders include the bill payer, the users, the maintainers and supporters,

TABLE 2.3 Functions of the Integration and Qualification Process

Design Function	Major Inputs	Major Outputs
Conduct Early Validation	Stakeholders' Inputs Operational Concept Stakeholders' Requirements Derived Requirements	Validated Requirements Validated Operational Concept
Conduct Integration and Verification Testing	Configuration Items (CIs) Components Derived Requirements	Verification Testing Document Verified Components and System
Conduct System Validation Testing	Verified System Stakeholders' Requirements Stakeholders' Inputs	Validation Testing Document Validated System
Conduct System Acceptance Testing	Validated System Acceptance Test Plan Stakeholders' Inputs	Acceptance Testing Document Accepted System

the manufacturers, the trainers, the deployers, the refiners, and the retirers. The integration and qualification process can be divided into four segments:

1. Conduct early validation
2. Conduct integration and verification testing
3. Conduct system validation testing
4. Conduct system acceptance testing

Table 2.3 presents these four functions that comprise integration and qualification with their major inputs and outputs. Each of these functions is described in more detail in Chapter 11.

2.3 KEY SYSTEMS ENGINEERING CONCEPTS

This section provides a detailed discussion of key constructs for design and integration. An operational concept, external system diagram, objectives hierarchy, and requirements are the essential elements of the definition of the design problem. Functions and items comprise the functional (or logical) architecture. Components are the building block for the physical architecture. Interfaces combined with the functional and physical architectures are key to understanding the allocated architecture. Qualification is verification, validation, and acceptance during integration. Figure 2.2 provides an entity relation diagram showing many of these concepts (or entities) and their relationships.

2.3.1 Operational Concept

An *operational concept* is a vision for what the system is (in general terms), a statement of mission requirements, and a description of how the system will be used. A set of scenarios describes how the system will be used by defining the system's interaction with other systems. For example, the operational concept for an elevator system may begin with a description of cars that carry people and equipment moving vertically in shafts. The mission requirements would discuss the desired times that passengers would wait from the time they requested service until they arrived at their destination. Next, the operational concept would use scenarios such as those in Table 2.4 to define how the elevator would be used during the elevator's operational phase.

2.3.2 External Systems Diagram

Defining the boundaries of a system is critical but often neglected. An *external systems diagram* is used to establish the bounds of the system and communicate the results of this bounding process. This diagram can be created from the

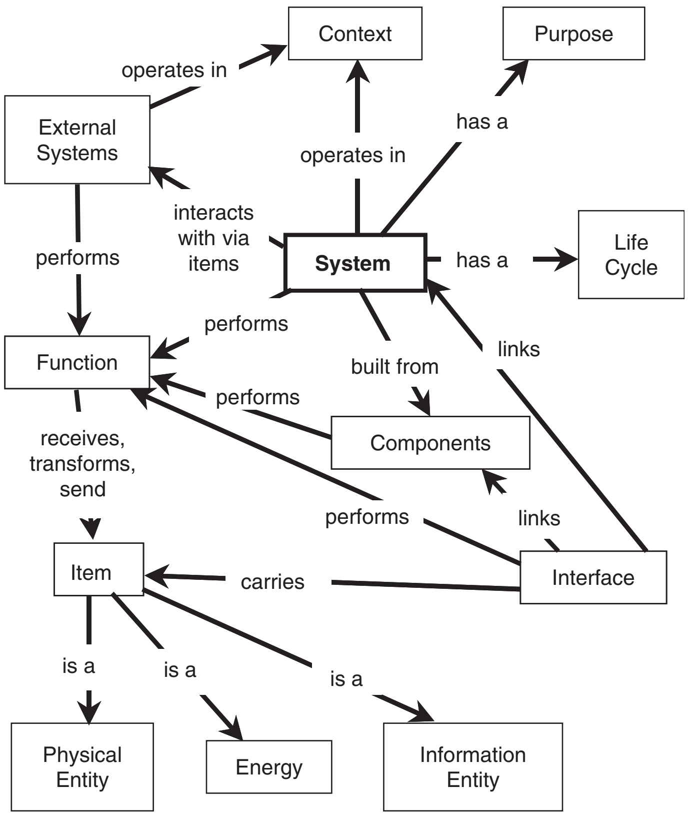

FIGURE 2.2 Many of the concepts and their relationships.

scenarios in the operational concept for the system and should he completely consistent with those scenarios.

Figure 2.3 shows an external systems diagram for the operational phase of the elevator using IDEF0, which is an acronym based on the following word phrase: Integrated Definition for Function Modeling. (IDEF0 will be presented in detail in the next chapter.) The systems are shown as arrows entering the bottom of each box, their functions are verb phrases in the boxes, and their inputs, controls, and outputs are shown as labels on lines entering from the left, entering from the top, and exiting from the right of the boxes, respectively. Many other process and dynamic modeling techniques can be used to draw and represent the system's boundaries, some of which are presented in Chapter 12.

On the basis of this diagram, the system is bounded by the definition of all of the inputs (and controls) that enter the system, as well as the outputs that the system must produce. The characteristics of these inputs and outputs can be described, thus finishing the definition of the boundary of the system.

TABLE 2.4 Sample Operational Concept Scenarios for an Elevator

1) Passengers (including mobility, visually and hearing challenged) request up service, receive feedback that their request was accepted, receive input that the elevator car is approaching and then that an entry opportunity is available, enter the elevator car, request a floor, receive feedback that their request was accepted, receive feedback that the door is closing, receive feedback about the floors at which the elevator is stopping, receive feedback that an exit opportunity is available at the desired floor, and exit the elevator with no physical impediments.
2) Passenger enters the elevator car, as described in 1, but finds an emergency situation before an exit opportunity is presented, and notifies the police or health authorities using communication equipment that is part of the elevator. Elevator maintenance personnel create an exit opportunity.
3) A maintenance person needs to repair an individual elevator car; the maintenance person places the elevator system in ‘‘partial maintenance’’ mode so that the other cars can continue to pick up passengers while the car(s) in question is (are) being diagnosed, repaired, and tested. After completion, the maintenance person places the elevator system in ‘‘full operation’’ mode.

2.3.3 Objectives Hierarchy

The *objectives hierarchy* of a system contains a hierarchical representation of the major performance, cost, and schedule characteristics that the stakeholders will use to determine their satisfaction with the system. For example, stakeholders evaluate an elevator system on the basis of the time spent waiting from their arrival at the elevator until they are delivered at their desired floor. Stakeholders are also concerned about the quality of the ride and the availability of the elevator services. The stakeholder, who is responsible for the building in which the elevator is located, is concerned about the monthly operating cost of providing the elevator services. See Figure 2.4 for a sample objectives hierarchy for the operational phase of the elevator.

2.3.4 Requirements

Requirements were defined in Chapter 1. Chapter 6 will address how to use the operational concept and external systems diagram, as well as the objectives hierarchy, to develop the stakeholders' requirements. For now, the four categories of requirements (input/output, technology and system-wide, tradeoff, and test) are assumed; these definitions will be expanded and motivated in Chapter 6.

1. *Input/output requirements*: include (a) inputs, (b) outputs, (c) external interface constraints, and (d) functional requirements.
2. *Technology and system-wide requirements*: consist of requirements that address (a) technology to be incorporated into the system, (b) the suitability (or ‘‘-ilities’’) of the system, (c) cost of the system, and (d) schedule issues (e.g., development time period, operational life of the system).

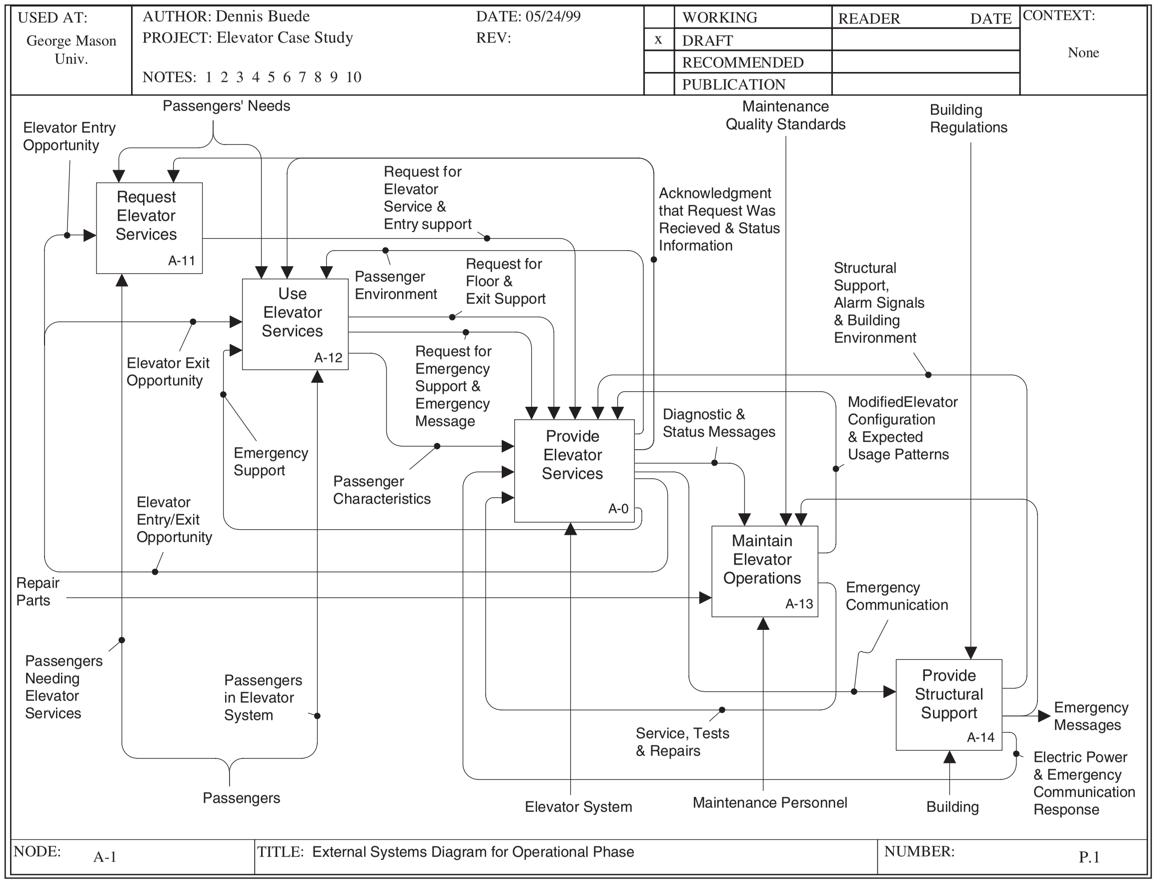
USED AT:
George Mason Univ.
AUTHOR: Dennis Buede
PROJECT: Elevator Case Study
NOTES: 1 2 3 4 5 6 7 8 9 10
DATE: 05/24/99
REV:
WORKING
x DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
None
Passengers' Needs
Elevator Entry Opportunity
Request Elevator Services
A-11
Use Elevator Services
A-12
Request for Elevator Service & Entry support
Passenger Environment
Request for Floor & Exit Support
Acknowledgment that Request Was Recieved & Status Information
Maintenance Quality Standards
Building Regulations
Structural Support, Alarm Signals & Building Environment
Elevator Exit Opportunity
Request for Emergency Support & Emergency Message
Provide Elevator Services
A-0
Diagnostic & Status Messages
ModifiedElevator Configuration & Expected Usage Patterns
Emergency Support
Passenger Characteristics
Elevator Entry/Exit Opportunity
Maintain Elevator Operations
A-13
Emergency Communication
Repair Parts
Passengers Needing Elevator Services
Passengers in Elevator System
Provide Structural Support
A-14
Emergency Messages
Electric Power & Emergency Communication Response
Service, Tests & Repairs
Passengers
Elevator System
Maintenance Personnel
Building
NODE: A-1
TITLE: External Systems Diagram for Operational Phase
NUMBER: P.1

FIGURE 2.3 External systems' diagram for operational use of an elevator.

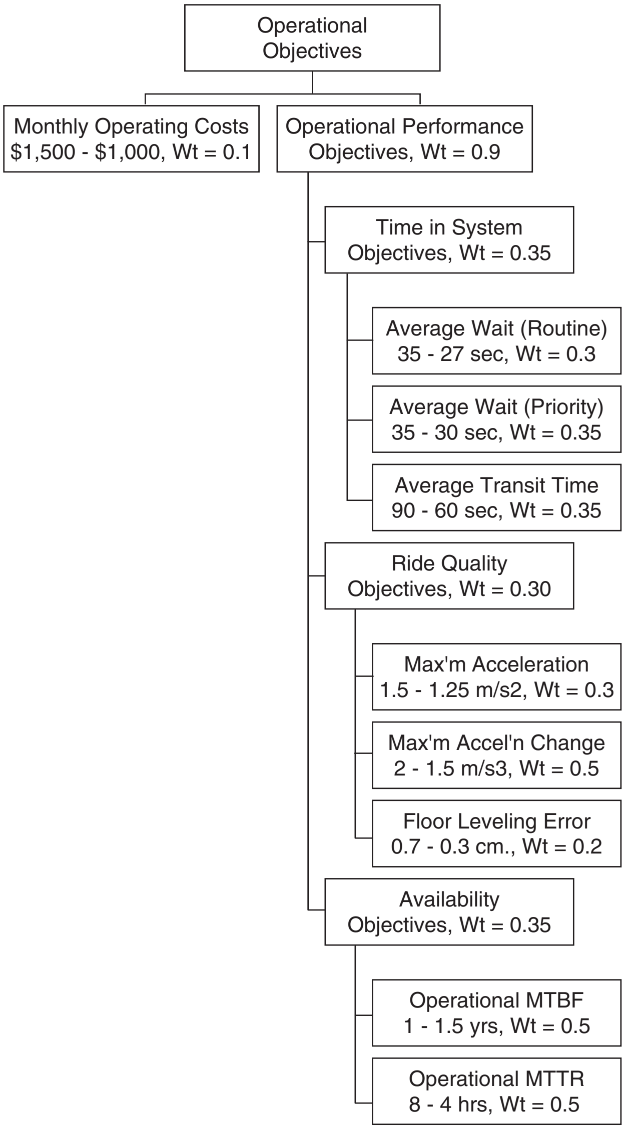
Operational Objectives
Monthly Operating Costs $1,500 - $1,000, Wt = 0.1
Operational Performance Objectives, Wt = 0.9
Time in System Objectives, Wt = 0.35
Average Wait (Routine) 35 - 27 sec, Wt = 0.3
Average Wait (Priority) 35 - 30 sec, Wt = 0.35
Average Transit Time 90 - 60 sec, Wt = 0.35
Ride Quality Objectives, Wt = 0.30
Max'm Acceleration 1.5 - 1.25 m/s2, Wt = 0.3
Max'm Accel'n Change 2 - 1.5 m/s3, Wt = 0.5
Floor Leveling Error 0.7 - 0.3 cm., Wt = 0.2
Availability Objectives, Wt = 0.35
Operational MTBF 1 - 1.5 yrs, Wt = 0.5
Operational MTTR 8 - 4 hrs, Wt = 0.5

FIGURE 2.4 Fundamental objectives hierarchy for operational phase of elevator.

3. *Trade-off requirements*: are algorithms to enable the engineers of the system to conduct (a) performance trade offs, (b) cost trade offs, and (c) cost–performance trade offs.
4. *System qualification requirements*: have four primary elements: (a) how the test data for each of the first categories of requirements will be obtained, (b) how the test data will be used to determine that the real

TABLE 2.5 Sample Elevator Requirements for the Operational Phase

- 4.3 Stakeholders' Requirements
  - 4.3.5 Operational Phase Requirements.
    - 4.3.5.1 Input/Output Requirements.
      - 4.3.5.1.1 Input Requirements.
        - 4.3.5.1.1.1 Emergency Support Inputs.
          - 4.3.5.1.1.1.1 The system shall support manual overrides.
          - 4.3.5.1.1.1.2 The elevator system shall allow passengers with a designated pass key to assume complete control of an elevator car.
      - 4.3.5.1.2 Output Requirements.
        - 4.3.5.1.2.1 Passenger Environment Outputs.
          - 4.3.5.1.2.1.1 The elevator car shall have adequate illumination.
      - 4.3.5.1.4 Functional Requirements.
        - 4.3.5.1.4.1 The elevator shall accept passenger requests and provide feedback.
        - 4.3.5.1.4.2 The elevator shall move passengers between floors safely and comfortably.
        - 4.3.5.1.4.3 The system shall control elevator cars efficiently.
        - 4.3.5.1.4.4 The system shall enable effective maintenance and servicing.
    - 4.3.5.2 System-wide & Technology Requirements.
      - 4.3.5.2.1 The system MTBF shall be greater than 1 year. The design goal is 1.5 years. Failure is defined to be a complete inability to carry passengers.
      - 4.3.5.2.2 The system MTTR shall be less than 8 hours. The design goal is 4 hours. Repair means the system is returned to full operating capacity.

system conforms to the design that was developed, (c) how the test data will be used to determine that the real system complies with the stakeholders' requirements, and (d) how the test data will be used to determine that the real system is acceptable to the stakeholders.

These requirements categories are relevant to each phase of the system's life cycle discussed above.

For now consider Table 2.5 to be an example of the input/output and system-wide requirements for the operational phase of an elevator. Note that these examples of requirements are shown in an outline or hierarchical format. Also note that the real requirements — those statements that start with "The Elevator system shall…"— are at the bottom of the hierarchy. Every entry of the hierarchy that has another level below it is not really a requirement, but a group of requirements.

2.3.5 Functions

A *function* is a transformation process that changes inputs into outputs. A system is modeled as having a single, top-level function that can be decomposed into a hierarchy of subfunctions. The system's top-level function transforms all of the inputs to the system into all of the outputs of the system. See Figure 2.5 for the elevator example using IDEF0. This system function can be taken directly from the external systems diagram.

The top-level function is decomposed into subfunctions as part of the development of the functional architecture. Each subfunction transforms a subset of the inputs from the outside (plus some other internally generated inputs) into a subset of the outputs (plus some other internally generated outputs). This decomposition cannot be found in a book or dictated by the stakeholders; the decomposition is a product of the engineers of the system and is part of the architectural design process that is attempting to solve the design problem established by the requirements. The decomposition can be carried as deeply as needed to define the transformations that the system must be able to perform. Figure 2.6 shows the first-level decomposition of the elevator system function using IDEF0.

2.3.6 Items

Items are the inputs that are received by the system, the outputs that are sent by the system to other systems, and the inputs that are generated internally to the system and sent to other parts of the system to assist in the transformation process for which the system is responsible. Items can be physical entities that have mass and energy, or items can be information that are somehow transformed into items with mass or energy to be transmitted from one physical element to another. The external inputs and outputs of the elevator are shown in Figure 2.5. Both the external and internal items of the elevator, at the first level of functional decomposition, are shown in Figure 2.6.

2.3.7 Components

A *component* of a system is a subset of the physical realization (and the physical architecture) of the system to which a subset of the system's functions have been (will be) allocated. A component could be the integration of hardware and software, a specific piece of hardware, a specific segment of the system's software, a group of people, facilities, or a combination of all of these. As with the requirements and functions, there is often a hierarchical structure to the components that comprise the system. The first level decomposition of the elevator into components is shown in Figure 2.7.

2.3.8 Interfaces

An *interface* is a connection resource for hooking to another system's interface (an external interface) or for hooking one system's component to another (an internal interface). Interfaces have inputs, produce outputs, and perform functions. An interface can be as simple as a wire or conveyor belt or as sophisticated as a global communication system (which is a system in its own right).

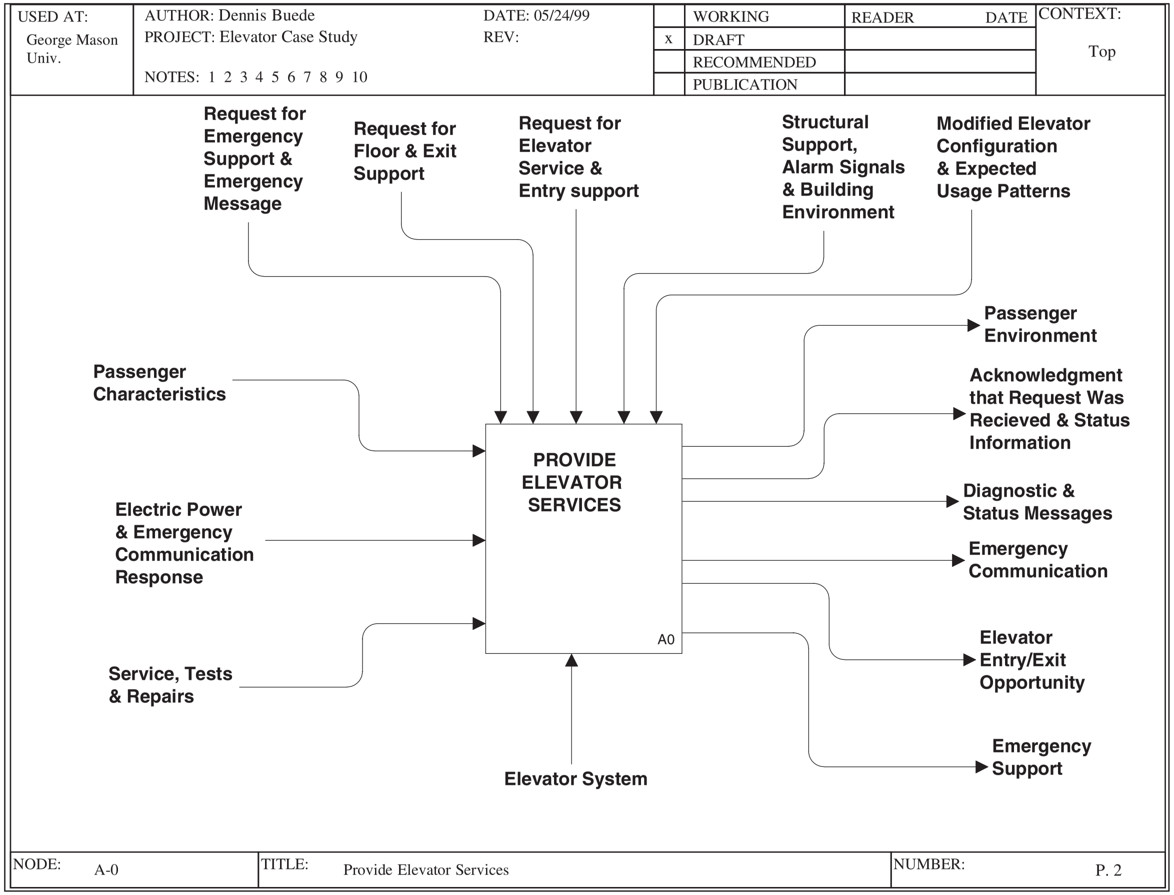
USED AT:
George Mason Univ.
AUTHOR: Dennis Buede
PROJECT: Elevator Case Study
NOTES: 1 2 3 4 5 6 7 8 9 10
DATE: 05/24/99
REV:
WORKING
x DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
Top
Request for Emergency Support & Emergency Message
Request for Floor & Exit Support
Request for Elevator Service & Entry support
Structural Support, Alarm Signals & Building Environment
Modified Elevator Configuration & Expected Usage Patterns
Passenger Characteristics
Electric Power & Emergency Communication Response
Service, Tests & Repairs
PROVIDE ELEVATOR SERVICES
A0
Elevator System
Passenger Environment
Acknowledgment that Request Was Recieved & Status Information
Diagnostic & Status Messages
Emergency Communication
Elevator Entry/Exit Opportunity
Emergency Support
NODE: A-0
TITLE: Provide Elevator Services
NUMBER:
P. 2

FIGURE 2.5 Top-level function for the elevator.

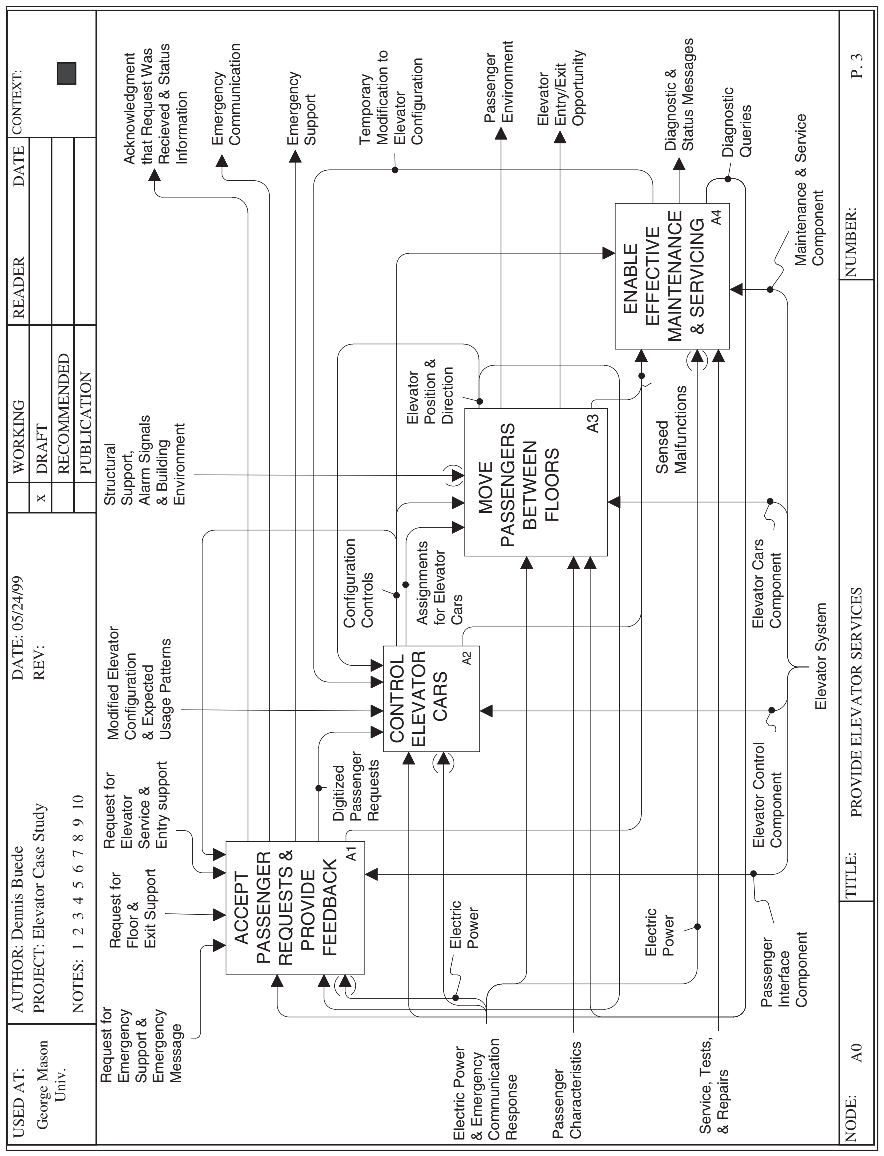
USED AT: George Mason Univ.
AUTHOR: Dennis Buede
PROJECT: Elevator Case Study
DATE: 05/24/99
REV:
NOTES: 1 2 3 4 5 6 7 8 9 10
WORKING
x DRAFT
RECOMMENDED
PUBLICATION
READER
DATE
CONTEXT:
ACCEPT PASSENGER REQUESTS & PROVIDE FEEDBACK
A1
CONTROL ELEVATOR CARS
A2
MOVE PASSENGERS BETWEEN FLOORS
A3
ENABLE EFFECTIVE MAINTENANCE & SERVICING
A4
Request for Emergency Support & Emergency Message
Request for Floor & Exit Support
Request for Elevator Service & Entry support
Modified Elevator Configuration & Expected Usage Patterns
Structural Support, Alarm Signals & Building Environment
Acknowledgment that Request Was Recieved & Status Information
Emergency Communication
Emergency Support
Temporary Modification to Elevator Configuration
Passenger Environment
Elevator Entry/Exit Opportunity
Diagnostic & Status Messages
Diagnostic Queries
Maintenance & Service Component
Digitized Passenger Requests
Configuration Controls
Assignments for Elevator Cars
Elevator Position & Direction
Sensed Malfunctions
Electric Power
Electric Power
Electric Power & Emergency Communication Response
Passenger Characteristics
Service, Tests, & Repairs
Passenger Interface Component
Elevator Control Component
Elevator Cars Component
Elevator System
NODE: A0
TITLE: PROVIDE ELEVATOR SERVICES
NUMBER:
P. 3

FIGURE 2.6 First-level decomposition of the elevator system function.

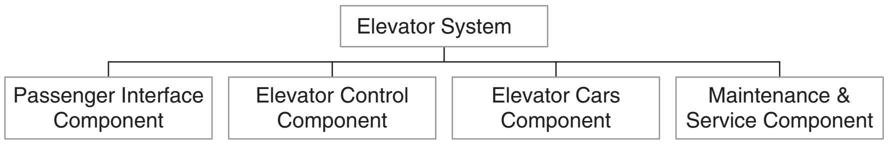
Elevator System
Passenger Interface Component
Elevator Control Component
Elevator Cars Component
Maintenance & Service Component

FIGURE 2.7 Physical architecture of the elevator system.

2.3.9 Verification

As discussed in Chapter 1, verification addresses whether the system was built right. In practical terms, *verification* is the determination that each configuration item (CI), component, and the system meets the requirements for it during the design phase. These requirements will be input/output or technology/system-wide requirements. Inspection, testing, analysis and simulation, or demonstration can establish this verification. Inspection and testing are most common at the CI level. Demonstration and analysis/simulation are most common at the system level.

2.3.10 Validation

Validation addresses whether the right system has been developed. Validation gets back to the illusive needs of the stakeholders. One aspect of validation that should be performed during the design phase of development attempts to demonstrate the design problem is evolving properly from a high-level statement in the operational concept that correctly reflects the needs of the stakeholders to scenarios, stakeholders' requirements, system requirements, component requirements, and CI requirements. Stated another way this early *validation of the design problem* demonstrates as completely as possible that the design problem as defined by a large set of requirements for all of the CIs is the same design problem as reflected in the operational concept and the minds of the stakeholders.

Validation during the integration and qualification phase demonstrates that the system that was designed and has been integrated meets the needs of the stakeholders as defined by the operational concept. *Validation of the system* stops short of the needs of the stakeholders because that will be addressed by the acceptance of the design by the stakeholders.

2.3.11 Acceptance

The final step of integration and qualification is *acceptance* by the stakeholders; do the stakeholders feel that the system as designed is acceptable? This conclusion allows the stakeholders to compare the system to their own needs and decide whether they will accept the system. The job of the engineers of the system during acceptance testing is to construct the proper set of test activities and test equipment and facilities to provide the information needed by the stakeholders for making their decision.

2.4 INTRODUCTION TO SysML

As described in Chapter 1, SysML is a visual modeling language for conducting systems engineering. Sections 2.2 and 2.3 have introduced you to some of the key terms and basic process for performing Traditional, Top-Down Systems Engineering (TTDSE). Figure 2.8 shows the processes of TTDSE associated with the left-hand side of the Vee model on the left hand side of the figure. Included with these process elements of TTDSE are modeling activities that should utilize a graphical modeling language. The right-hand side of Figure 2.8 shows the diagram types of SysML. Double-headed arrows in Figure 2.8 show which modeling elements from TTDSE are addressed by which diagram types of SysML. Naturally there are many to many-to-many and many-to-one relations shown. There are also three modeling activities in TTDSE that are not supported by SysML (requirements taxonomy; creativity; and the morphological box for physical architecture alternatives and risk and trade studies). Each of these topics is addressed in Chapters 6, 8, and 13, respectively. Also,

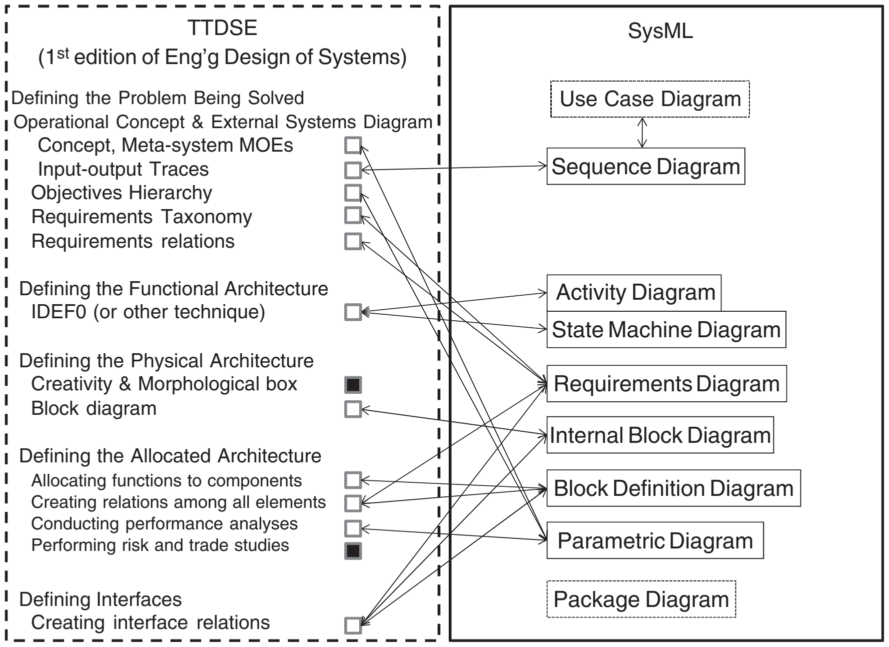
TTDSE
(1st edition of Eng'g Design of Systems)
Defining the Problem Being Solved
Operational Concept & External Systems Diagram
Concept, Meta-system MOEs
Input-output Traces
Objectives Hierarchy
Requirements Taxonomy
Requirements relations
Defining the Functional Architecture
IDEF0 (or other technique)
Defining the Physical Architecture
Creativity & Morphological box
Block diagram
Defining the Allocated Architecture
Allocating functions to components
Creating relations among all elements
Conducting performance analyses
Performing risk and trade studies
Defining Interfaces
Creating interface relations
SysML
Use Case Diagram
Sequence Diagram
Activity Diagram
State Machine Diagram
Requirements Diagram
Internal Block Diagram
Block Definition Diagram
Parametric Diagram
Package Diagram

FIGURE 2.8 Comparison of TTDSE and SysML.

there are two diagram types (use case and package) in SysML that are not related to the elements of TTDSE from the first edition of this book; these two diagram types will be covered in Chapter 3.

This figure should convey to you that SysML is a nearly complete and well-designed, interconnected set of visual modeling diagrams that enable a substantial improvement in model-based systems engineering. All elements of the design process that we cover in this book are covered. Since the test system is another system, SysML can be equally applied to the design and operation of the test system, just as it can be applied to the design of the systems engineering system of engineers, domain experts, technologists, and managers.

2.5 USE OF CORE (SYSTEMS ENGINEERING TOOL)

Part of the educational material provided with this book is an academic version of a system engineering tool called CORE. (You may download the academic version of CORE from http://www.vitechcorp.com.) The rest of this chapter provides an overview of concepts embedded in CORE. Instructions for using the tool can be obtained in the user's manual and guided tour available with CORE.

At its simplest, CORE is comprised of classes (e.g., requirements, functions, items), examples or elements of those classes (e.g., specific requirement), and relations. The most basic user activities of CORE are entering and editing elements of the classes and establishing relations between elements of classes. Other important activities include viewing products of the design data, saving your work, and obtaining reports that document the design contained in the database. The automated tutorial demonstrates these functions for a collection management system as a sample problem.

2.5.1 Classes

Table 2.6 lists the classes in CORE 2.0. These classes contain both the major elements of the systems engineering design process that are discussed in this chapter as well as a number of supporting classes. For a given system, the job of the engineers of the system is to define specific elements of the system for each of these major classes (e.g., stakeholders' requirements, functions, components, and items).

2.5.2 Relations

As part of the engineering design process, requirements must be related to functions and components using the specify relation, functions allocated to components, and inputs and outputs assigned to interfaces. Table 2.7 defines the relations available in CORE to define relationships within the design and integration classes. These relations are fully compatible with the mathematical

TABLE 2.6 Systems Engineering Classes in CORE

Class	Definition of the Class
Category	A general purpose element that can be used to represent such concepts as Version Number, Element Classification, etc.
Component	A physical entry that can represent the system; a subsystem or further decomposition of the system, including a configuration of a component; or an external system or the meta-system.
Defined Term	An acronym or special term that needs to be defined as part of the requirements process.
Document	A source/authorization for information from stakeholders entered into the system description database or reported from the database.
Domain Set	The number of iterations or replications in a control structure.
Function	A process that accepts one or more inputs (items) and transforms them into outputs (items). A function should have a completion criterion for each exit.
Interface	The logical connection between parts of the system's architecture.
Issue	A problem (as well as its resolution) with any element in the system's design.
Item	Physical entities or data that flows within and between functions. An item is an input to or output from a function.
Link	The physical implementation of an interface.
Requirement	A requirement extracted from the source documentation for a system, or a refinement of a higher level requirement. Requirements should be refined until only a single, testable statement of a system's feature remains.
Resource	A characteristic (e.g., power, channels, instructions processed per second) of one or more components that are used, captured, or generated and can be depleted during the operation of the system.
Risk	The uncertainty of attaining/achieving a product performance level or program milestone.
State/Mode	Sometimes used as the highest level functional breakout to define a set of functions that system performs at a point in time, e.g., startup, normal operation, recovery operations, shut down.
Verification Requirement	The requirement to be met by the qualification system, the level at which it must be met, the method of qualification, and current qualification status.

definition of relations in Chapter 4 and can be depicted graphically by directed graphs as discussed in Chapter 5. For each relation there is an opposing relation that reverses the direction of the relation. Table 2.7 shows the relations (and their opposing relations in parentheses) and defines each relation by identifying which class is on the left side (tail of the arrow) and which class is on the right side (head of the arrow).

TABLE 2.7 Common Systems Engineering Relations

Relation (Opposing Relation)	Definition	Application
"is part of"; examples include refines (requirements) decomposes (functions, items) built from (components) ("is a superset of")	The left side is a subset of the right side. For example, the requirement on the left side incorporates the one on the right, the function on the left side is decomposed by the one on the right.	Hierarchies of requirements, functions, items, components.
"input to" ("inputs")	The item on the left side is an input to the function on the right side.	Development of the functional architecture
"output from" ("outputs")	The item on the left side is an output from the function on the right side.	Development of the functional architecture
"triggers" ("triggered by")	The item on the left side triggers the activation of the function on the right side.	Development of the functional architecture
"defined by" ("defines")	The state/mode on the left side is defined by the function on the left side.	Development of the functional architecture
"built from" ("built in")	The left side (system or component) is comprised of the system or component on the right side.	Development of the physical architecture
"exhibited by" ("exhibits")	The state/mode on the left side is exhibited by the component on the right side.	Development of the physical architecture
"allocated to" ("performs")	The function on the left side is being assigned to the component on the right side for the purpose of execution.	Development of the operational architecture
"specifies" ("specified by")	The requirement on the left side specifies the function, state/mode, item, component, interface or link on the right side.	Development of the functional and operational architectures
"transfers" ("transferred by")	The link on the left side transfers the item on the right side.	Development of the interface architecture
"comprises" ("comprised of")	The link on the left side comprises the interface on the right side.	Development of the interface architecture

(*Continued*)

TABLE 2.7. Continued

Relation (Opposing Relation)	Definition	Application
"connects to" ("connected to")	The link on the left side connects to the component or system on the right side.	Development of the interface architecture
"joins" ("joined to")	The interface on the left side joins the component on the right side.	Development of the interface architecture

One subtlety that has been ignored so far is the relating of requirements to functions and items, or the system. Input/output requirements are defined in such a way that each such requirement should be directly relatable to both specific functions and items. Technology and system-wide requirements are those requirements that cannot be related to specific functions or items but must be satisfied by the system. As a result each input/output requirement is traced to (or specifies) the lowest level function that receives the relevant input or produces the relevant output, all of the functions that are above that function in the functional decomposition, and the item directly relevant to that requirement. (Note that the third category of input/output requirements is function requirements; these requirements specify the top-level system function because they define the decomposition of that function.) Similarly, each technology and system-wide requirement specifies the system. Relating requirements to functions and the system through the specify relation is important because this activity initiates the process of creating a set of requirements for the system to satisfy and provides the material for subsets of the requirements to be associated with specific components. These subsets of requirements become the specifications that each CI design team must meet. (Note that the requirements related to functions ultimately are assigned to the system and its components when each function is allocated to one or more components for execution.)

2.5.3 Documents

CORE enables you to design your document. However, the outline of the document that will be used throughout this book is the System Description Document (SDD), which can be found under the reports available from version 1.2 of CORE. This SDD outline (see Table 2.8) contains an initial section in which a general description of the system would be provided; the operational concept would be found here if CORE captured this material. The stakeholders' requirements are found in Section 2. Section 3 of the SDD is for the requirements that are constraints; the approach taken here is to include all

TABLE 2.8 Outline of System Description Document

1. Primary System/Component Description
2. Originating (Stakeholders') Requirements
3. Design Constraints
4. Performance Requirements
5. Issues & Decisions
6. Risks
7. Functional Behavior Models
8. Item Dictionary
9. Resources
10. Components
11. Interfaces
12. Requirements Traceability Matrix

requirements as part of the stakeholders' requirements so Section 3 can be deleted. The section on performance objectives is for the objectives hierarchy. Section 5 enables the systems engineering team to capture the design issues and decisions, which are usually addressed as part of the allocated architecture. Risk management is addressed in Section 6; this is the place that key uncertainties are defined and the potential impact of bad outcomes defined. The functional architecture is defined in Section 7 as both a process model and behavioral model; CORE uses an N^2 model for the system's process and a function flow block diagram model for the system's behavior (both of these models are covered in Chapter 12). The item dictionary (or data model) is found in Section 8. The physical architecture is defined in Section 9, and the interfaces that are derived from the combination of the item dictionary and the physical architecture are defined in Section 10 of the SDD. The logical and physical interfaces developed by the system engineering team are described in Section 11. Section 12 provides a cross reference of requirements with the selected test methods that will be used for qualification. Section 13 provides a requirements specification matrix that associates functions and components with the stakeholders' requirements.

2.6 SUMMARY

This chapter has given definitions and provided discussions on the most important concepts in the engineering of systems. The operational concept of the system provides the theme for the system as viewed by the stakeholders and defines scenarios depicting how its users will employ the system and how the system will interact with other systems. The external systems diagram defines the interaction in terms of inputs and outputs with other systems and is consistent with the operational concept. The objectives hierarchy of the system lays out the performance, cost, and schedule objectives that the stakeholders have for the system; this objectives hierarchy provides a satisfaction index for the stakeholders for alternate system designs. The requirements of the system provide constraints and performance ranges for the system in terms of inputs and outputs, and its system-wide and technology-related characteristics. The requirements also state the trade offs that the stakeholders are willing to make in the development of the system and the constraints and performance ranges associated with testing the system. These first four concepts deal with defining the design problem.

Three additional concepts (functions, components, and interfaces) are part of the design process. Functions are those activities performed by the system (and all other systems) to transform inputs into outputs. Components are the resources of the system that perform the system's functions. Interfaces connect components; external interfaces connect components of the system to components of other systems, and internal interfaces connect components of the system, to each other.

Throughout this entire process, from the operational concept through requirements, it is important to remember that the engineers have to concern themselves with more than just the operational system that the users of the system want, but also the systems relevant to every stage of the life cycle of the system (e.g., the development system, manufacturing system, the retirement system).

This chapter described how to read an IDEF0 diagram; IDEF0 is a process modeling technique that will be described in detail in the next chapter and used throughout this book. The software product CORE that will be used extensively in this book was described in terms of its classes and the relationship between those classes for systems engineering. CORE's data structure is based upon a data modeling technique called entity–relationship (ER) diagrams and is discussed in more detail in Chapter 12.

PROBLEMS

2.1 Use the requirements in Table 2.5 to define the elevator's requirements. Use the IDEF0 models in Figure 2.4 and 2.5 to define the functional decomposition of the elevator system and to identify the external inputs and outputs, as well as those that are internally generated and consumed. Use the organization chart in Figure 2.6 to define the physical decomposition of the elevator components.

Enter all of the above information on the elevator as the system into CORE: enter the requirements shown in Table 2.5, enter the functions shown in Figure 2.4 and 2.5, enter the items shown in Figure 2.5, and enter the components shown in Figure 2.6 as elements of the corresponding classes. Then establish the relevant relations associated with Table 2.7. These relations include hierarchies for the requirements and functions as well as the system with its components. Use the "specify" relation to connect the requirements to the appropriate function, item or system, designate items as inputs or outputs of the relevant functions and allocate each function to the system or appropriate component.