CHAPTER 5

FUNCTIONAL ANALYSIS AND ALLOCATION

5.1 INTRODUCTION

The purpose of this systems engineering process activity is to transform the functional, performance, interface and other requirements that were identified through requirements analysis into a coherent description of system functions that can be used to guide the Design Synthesis activity that follows. The designer will need to know what the system must do, how well, and what constraints will limit design flexibility.

This is accomplished by arranging functions in logical sequences, decomposing higher-level functions into lower-level functions, and allocating performance from higher- to lower-level functions. The tools used include functional flow block diagrams and timeline analysis; and the product is a functional architecture, i.e., a description of the system—but in terms of functions and performance parameters, rather than a physical description. Functional Analysis and Allocation facilitates traceability from requirements to the solution descriptions that are the outcome of Design Synthesis.

Functions are discrete actions (use action verbs) necessary to achieve the system's objectives. These functions may be stated explicitly, or they may be derived from stated requirements. The functions will ultimately be performed or accomplished through use of equipment, personnel, facilities, software, or a combination.

5.2 FUNCTIONAL ANALYSIS AND ALLOCATION

Functional and performance requirements at any level in the system are developed from higher-level requirements. Functional Analysis and Allocation is repeated to define successively lower-level functional and performance requirements, thus defining architectures at ever-increasing levels of detail. System requirements are allocated and defined in sufficient detail to provide design and verification criteria to support the integrated system design.

This top-down process of translating system-level requirements into detailed functional and performance design criteria includes:

- Defining the system in functional terms, then decomposing the top-level functions into subfunctions. That is, identifying at successively lower levels what actions the system has to do,
- Translating higher-level performance requirements into detailed functional and performance design criteria or constraints. That is, identifying how well the functions have to be performed,
- Identifying and defining all internal and external functional interfaces,
- Identifying functional groupings to minimize and control interfaces (functional partitioning),
- Determining the functional characteristics of existing or directed components in the system and incorporating them in the analysis and allocation,
- Examining all life cycle functions, including the eight primary functions, as appropriate for the specific project,
- Performing trade studies to determine alternative functional approaches to meet requirements, and

- Revisiting the requirements analysis step as necessary to resolve functional issues.

The objective is to identify the functional, performance, and interface design requirements; it is not to design a solution…yet!

Functional Partitioning

Functional partitioning is the process of grouping functions that logically fit with the components likely to be used, and to minimize functional interfaces. Partitioning is performed as part of functional decomposition. It identifies logical groupings of functions that facilitate the use of modular components and open-system designs. Functional partitioning is also useful in understanding how existing equipment or components (including commercial) will function with or within the system.

Requirements Loop

During the performance of the Functional Analysis and Allocation process, it is expected that revisiting the requirements analysis process will be necessary. This is caused by the emergence of functional issues that will require re-examination of the higher-level requirements. Such issues might include directed components or standards that cause functional conflict, identification of a revised approach to functional sequencing, or, most likely, a conflict caused by mutually incompatible requirements.

Figure 5-1 gives an overview of the basic parameters of Functional Analysis and Allocation. The output of the process is the functional architecture. In its most basic form, the functional architecture is a simple hierarchical decomposition of the functions with associated performance requirements. As the architecture definition is refined and made more specific with the performance of the

- **Outputs:**
  - Functional architecture and supporting detail
- **Inputs:**
  - Outputs of the Requirements Analysis
- **Enablers:**
  - Multi-discipline product teams, decision database; Tools & Models, such as QFD, Functional Flow Block Diagrams, IDEF, N2 charts, Requirement Allocation Sheet, Timelines, Data Flow Diagrams, State/Mode Diagrams, Behavior Diagrams
- **Controls:**
  - Constraints; GFE, COTS, & Reusable S/W; System concept & subsystem choices; organizational procedures
- **Activities:**
  - Define system states and modes
  - Define system functions & external interfaces
  - Define functional interfaces
  - Allocate performance requirements to functions
  - Analyze performance
  - Analyze timing and resources
  - Analyze failure mode effects and criticality
  - Define fault detection and recovery behavior
  - Integrate functions

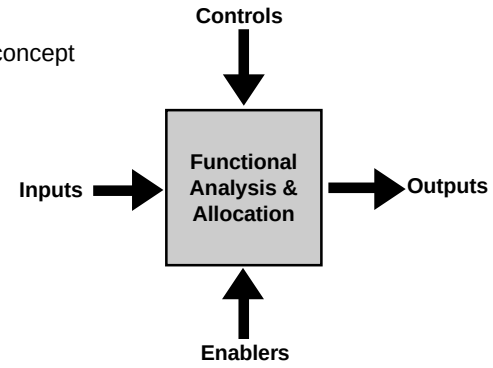

Figure 5-1. Functional Analysis and Allocation

activities listed in Figure 5-1, the functional architecture becomes more detailed and comprehensive. These activities provide a functional architecture with sufficient detail to support the Design Synthesis. They are performed with the aid of traditional tools that structure the effort and provide documentation for traceability. There are many tools available. The following are traditional tools that represent and explain the primary tasks of Functional Analysis and Allocation (several of these are defined and illustrated beginning on page 49):

- Functional flow block diagrams that define task sequences and relationships,
- IDEF0 diagrams that define process and data flows,
- Timeline analyses that define the time sequence of time critical functions, and
- Requirements allocation sheets that identify allocated performance and establish traceability of performance requirements.

5.3 FUNCTIONAL ARCHITECTURE

The functional architecture is a top-down decomposition of system functional and performance requirements. The architecture will show not only the functions that have to be performed, but also the logical sequencing of the functions and performance requirements associated with the functions. It also includes the functional description of existing and government-furnished items to be used in the system. This may require reverse engineering of these existing components.

The functional architecture produced by the Functional Analysis and Allocation process is the detailed package of documentation developed to analyze the functions and allocate performance requirements. It includes the functional flow block diagrams, timeline sheets, requirements allocation sheets, IDEF0 diagrams, and all other documentation developed to describe the functional characteristics of the system. However, there is a basic logic to the functional architecture, which in its preliminary form is presented in the example of Figure 5-2. The Functional Analysis and Allocation process would normally begin with the

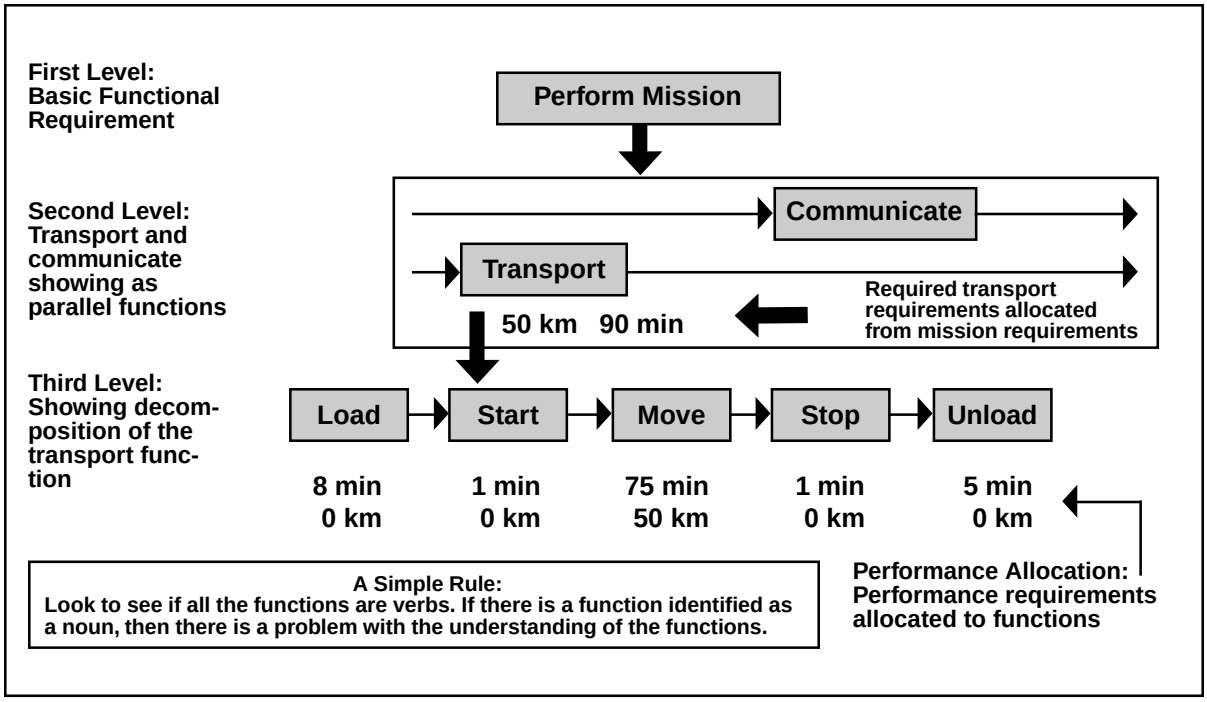

Figure 5-2. Functional Architecture Example

IPT drafting such a basic version of the architecture. This would generally give the IPT an understanding of the scope and direction of the effort.

Functional Architecture Example

The Marine Corps has a requirement to transport troops in squad-level units over a distance of 50 kilometers. Troops must be transported within 90 minutes from the time of arrival of the transport system. Constant communication is required during the transportation of troops. Figure 5-2 illustrates a preliminary functional architecture for this simple requirement.

5.4 SUMMARY POINTS

Functional analysis begins with the output of requirements analysis (that is, the identification of higher-level functional and performance requirements). Functional Analysis and Allocation consists of decomposition of higher-level functions to lower-levels and then allocation of requirements to those functions.

- There are many tools available to support the development of a Functional Architecture, such as: functional-flow block diagrams, timeline analysis sheet, requirements allocation sheet, Integrated Definition, and others.
- Use of the tools illustrated in this chapter is not mandatory, but the process they represent is:
  - Define task sequences and relationships (functional flow block diagram (FFBD)),
  - Define process and data flows (IDEF0 diagrams),
  - Define the time sequence of time-critical functions (timeline analysis sheets (TLS)), and
  - Allocate performance and establish traceability of performance requirements (requirements allocation sheets (RAS)).

SUPPLEMENT 5-A

FUNCTIONAL FLOW BLOCK DIAGRAM

The purpose of the functional flow block diagram (FFBD) is to describe system requirements in functional terms.

Objectives

The FFBD is structured to ensure that:

- All life cycle functions are covered.
- All elements of system are identified and defined (e.g. prime equipment, training, spare parts, data, software, etc.).
- System support requirements are identified to specific system functions.
- Proper sequencing of activities and design relationships are established including critical design interfaces.

Characteristics

The FFBD is functionally oriented—not solution oriented. The process of defining lower-level functions and sequencing relationships is often referred to as functional decomposition. It allows traceability vertically through the levels. It is a key step in developing the functional architecture from which designs may be synthesized.

Figure 5-3 shows the flow-down structure of a set of FFBDs and Figure 5-4 shows the format of an FFBD.

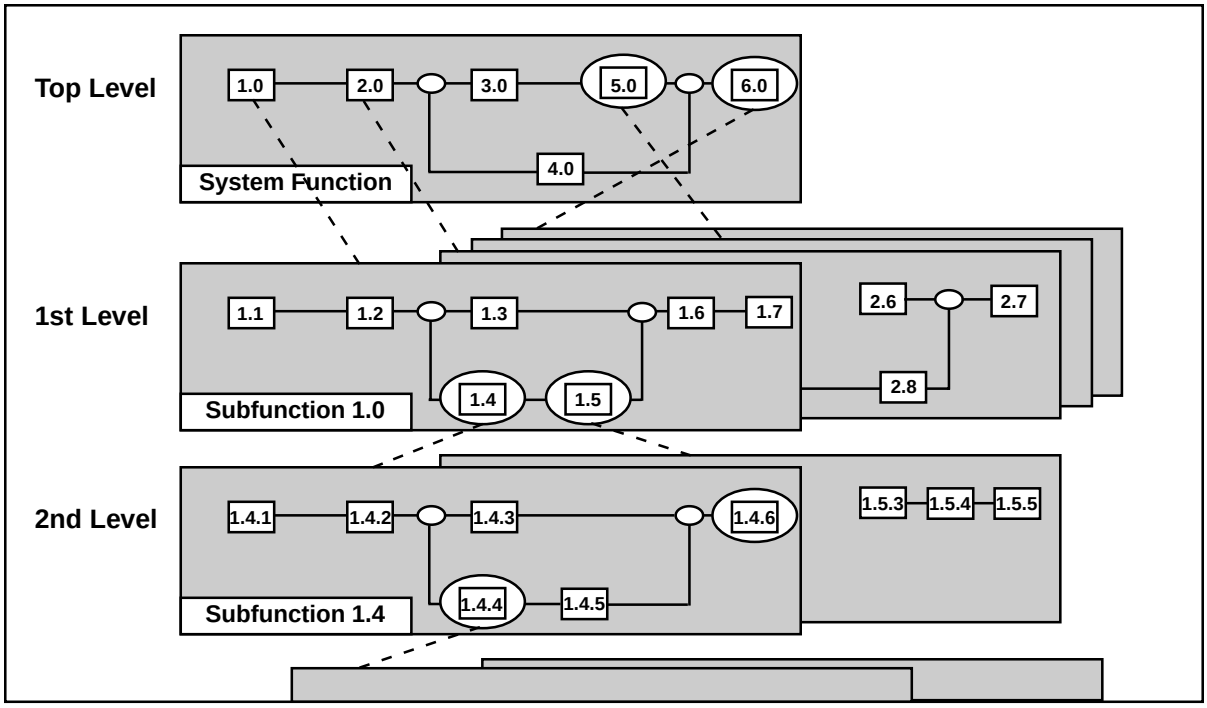

Figure 5-3. FFBD Traceability and Indenture

Key FFBD Attributes

Function block: Each function on an FFBD should be separate and be represented by single box (solid line). Each function needs to stand for definite, finite, discrete action to be accomplished by system elements.

Function numbering: Each level should have a consistent number scheme and provide information concerning function origin. (E.g., top level—1.0, 2.0, 3.0, etc; first indenture (level 2)—1.1, 1.2, 1.3, etc; second indenture (level 3)—1.1.1, 1.1.2, 1.1.3, etc.) These numbers establish identification and relationships that will carry through all Functional Analysis and Allocation activities and facilitate traceability from lower to top levels.

Functional reference: Each diagram should contain a reference to other functional diagrams by using a functional reference (box in brackets).

Flow connection: Lines connecting functions should only indicate function flow and not a lapse in time or intermediate activity.

Flow direction: Diagrams should be laid out so that the flow direction is generally from left to right. Arrows are often used to indicate functional flows.

Summing gates: A circle is used to denote a summing gate and is used when AND/OR is present. AND is used to indicate parallel functions and all conditions must be satisfied to proceed. OR is used to indicate that alternative paths can be satisfied to proceed.

GO and NO-GO paths: **"**G" and "bar G" are used to denote "go" and "no-go" conditions. These symbols are placed adjacent to lines leaving a particular function to indicate alternative paths.

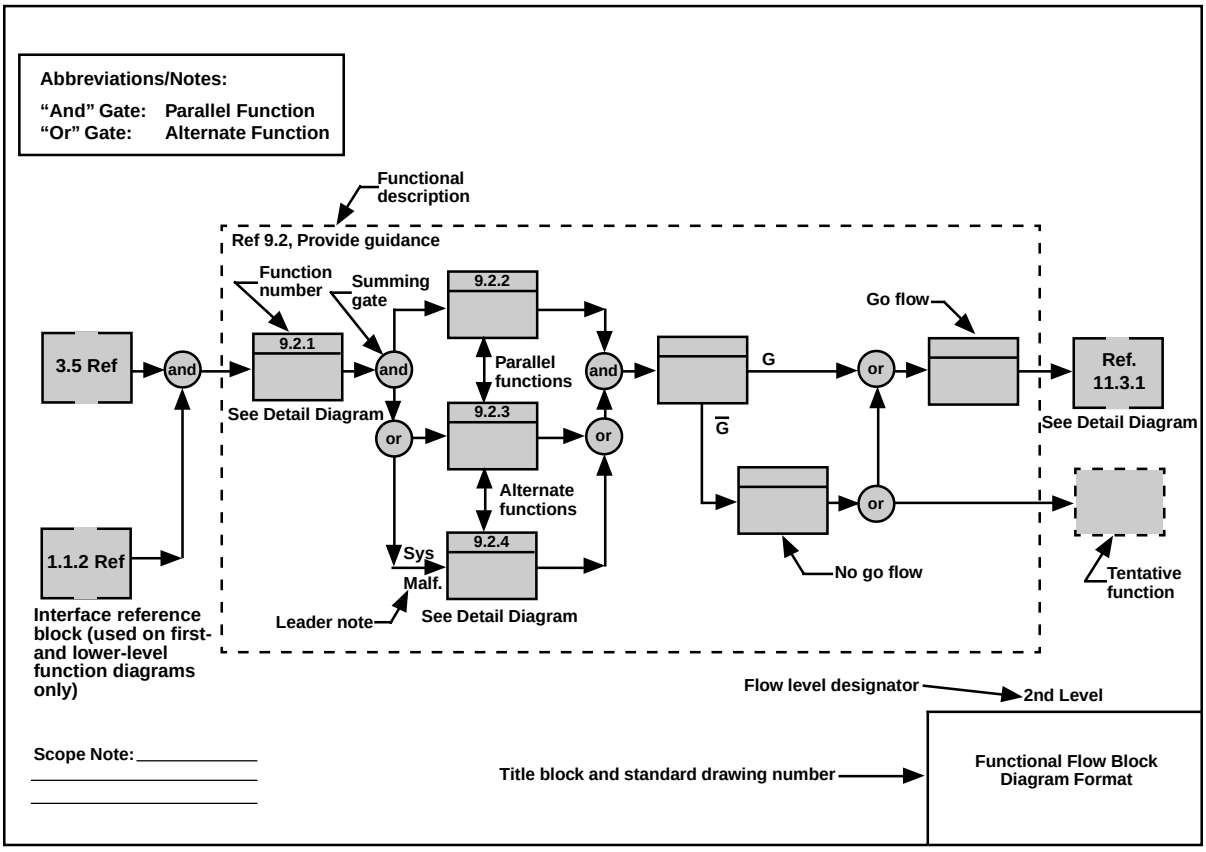

Figure 5-4. Functional Flow Block Diagrams (FFBD) Format

SUPPLEMENT 5-B

IDEF0

Integration Definition for Function Modeling (IDEF0) is a common modeling technique for the analysis, development, re-engineering, and integration of information systems; business processes; or software engineering analysis. Where the FFBD is used to show the functional flow of a product, IDEF0 is used to show data flow, system control, and the functional flow of life cycle processes.

IDEF0 is capable of graphically representing a wide variety of business, manufacturing and other types of enterprise operations to any level of detail. It provides rigorous and precise description, and promotes consistency of usage and interpretation. It is well-tested and proven through many years of use by government and private industry. It can be generated by a variety of computer graphics tools. Numerous commercial products specifically support development and analysis of IDEF0 diagrams and models.

IDEF0 is a model that consists of a hierarchical series of diagrams, text, and glossary cross-referenced to each other. The two primary modeling components are: functions (represented on a diagram by boxes), and data and objects that interrelate those functions (represented by arrows). As shown by Figure 5-5 the position at which the arrow attaches to a box conveys the specific role of the interface. The controls enter the top of the box. The inputs, the data or objects acted upon by the operation, enter the box from the left. The outputs of the operation leave the right-hand side of the box. Mechanism arrows that provide supporting means for performing the function join (point up to) the bottom of the box.

The IDEF0 process starts with the identification of the prime function to be decomposed. This function is identified on a "Top Level Context Diagram," that defines the scope of the particular IDEF0 analysis. An example of a Top Level Context Diagram for an information system management process is shown in Figure 5-6. From this diagram lower-level diagrams are generated. An example of a derived diagram, called a "child" in

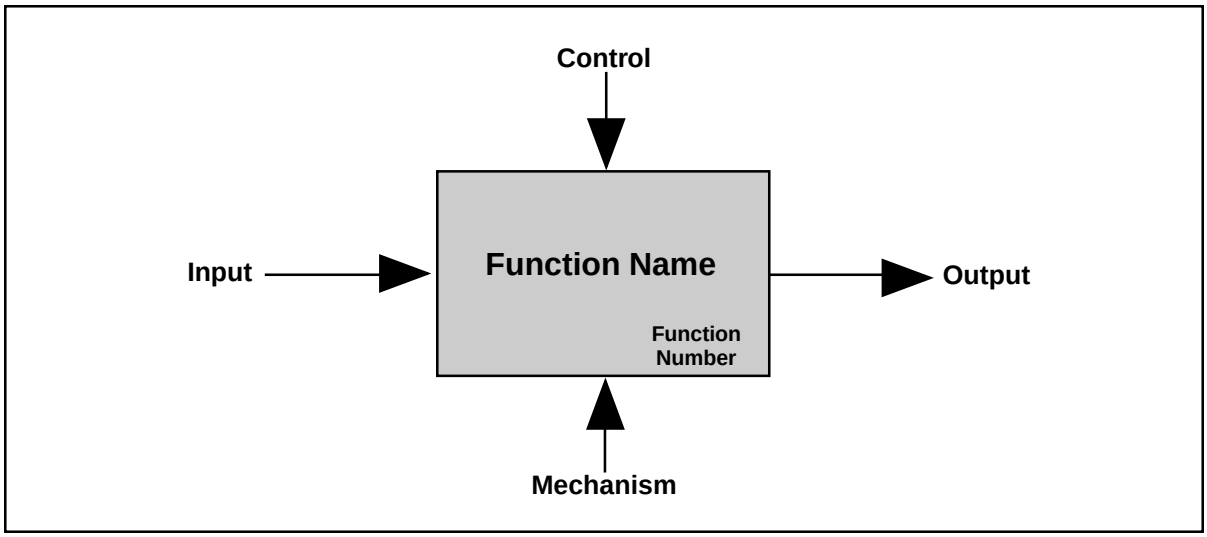

Figure 5-5. Integration Definition for Function Modeling (IDEF0) Box Format

IDEF0 terminology, for a life cycle function is shown in Figure 5-7.

An associated technique, Integration Definition for Information Modeling (IDEF1x), is used to supplement IDEF0 for data intensive systems. The IDEF0 standard, Federal Information Processing Standards Publication 183 (FIPS 183), and the IDEF1x standard (FIPS 184) are maintained by the National Institute of Standards and Technology (NIST).

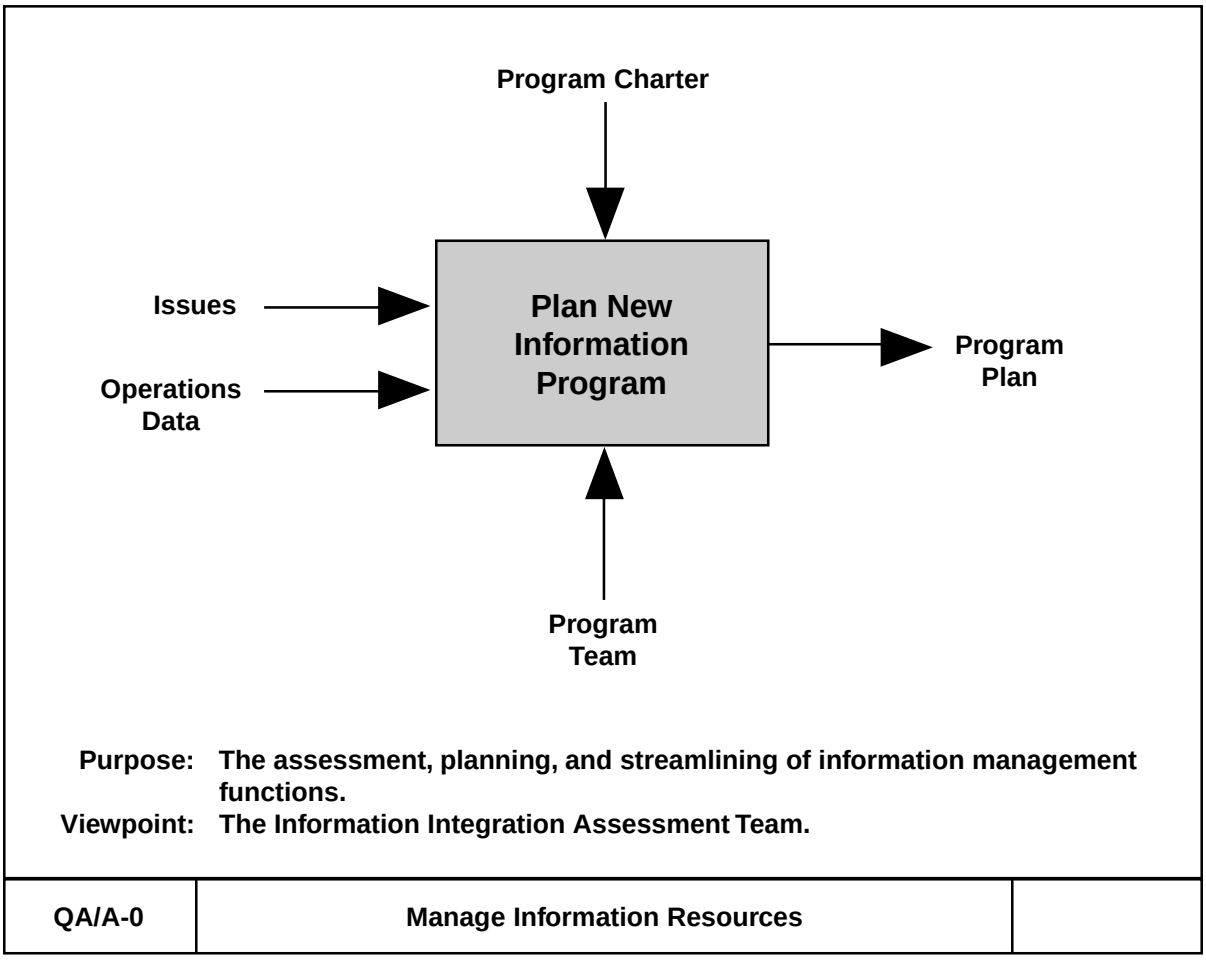

Figure 5-6. Top-Level Context Diagram

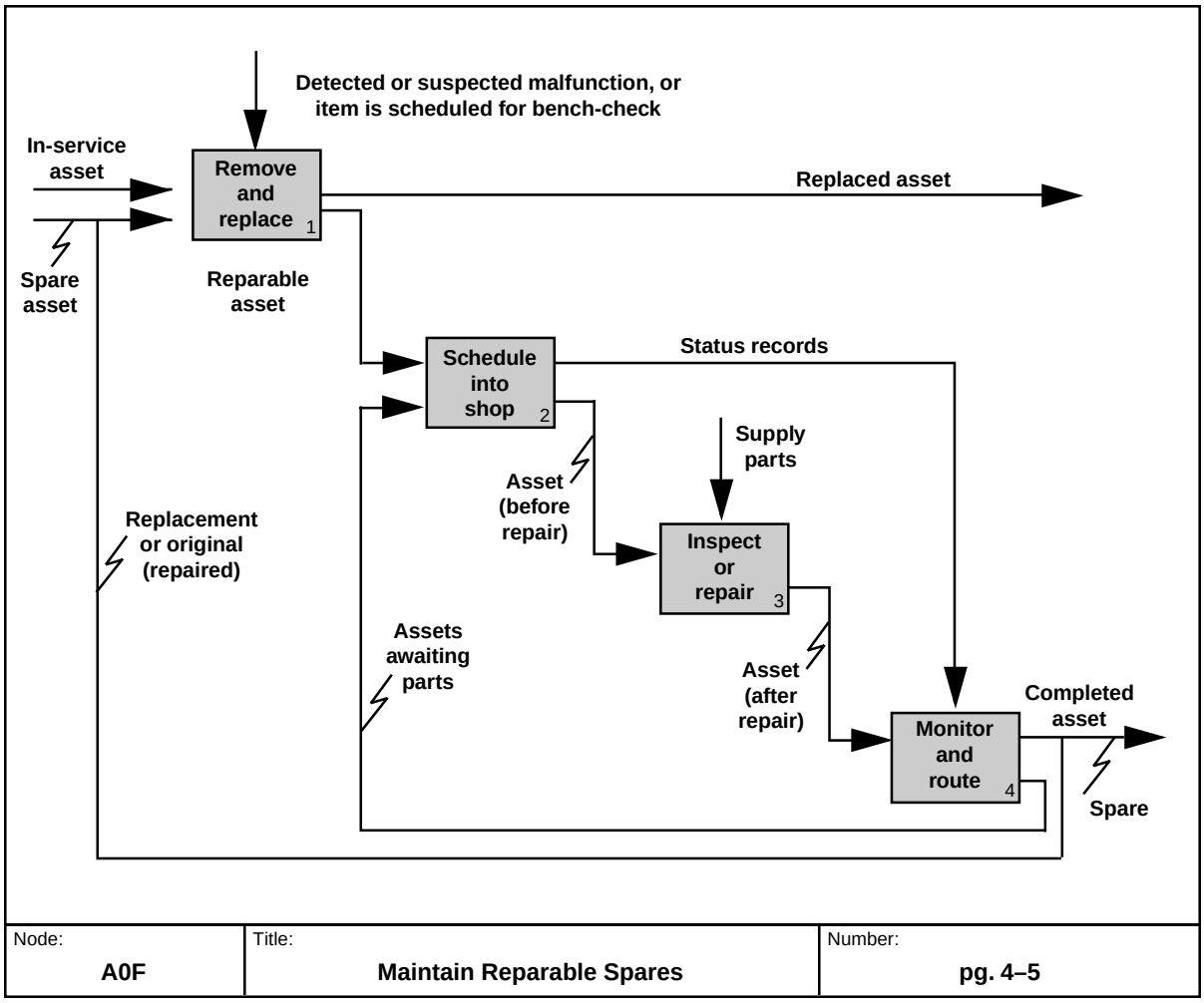

Figure 5-7. IDEF0 Diagram Example

SUPPLEMENT 5-C

TIMELINE ANALYSIS SHEETS

The timeline analysis sheet (TLS) adds detail to defining durations of various functions. It defines concurrency, overlapping, and sequential relationships of functions and tasks. It identifies time critical functions that directly affect system availability, operating time, and maintenance downtime. It is used to identify specific time-related design requirements.

The TLS includes purpose of function and the detailed performance characteristics, criticality of function, and design constraints. It identifies both quantitative and qualitative performance requirements. Initial resource requirements are identified.

Figure 5-8 shows an example of a TLS. The time required to perform function 3.1 and its subfunctions are presented on a bar chart showing how the timelines relate. (Function numbers match the FFBD.)

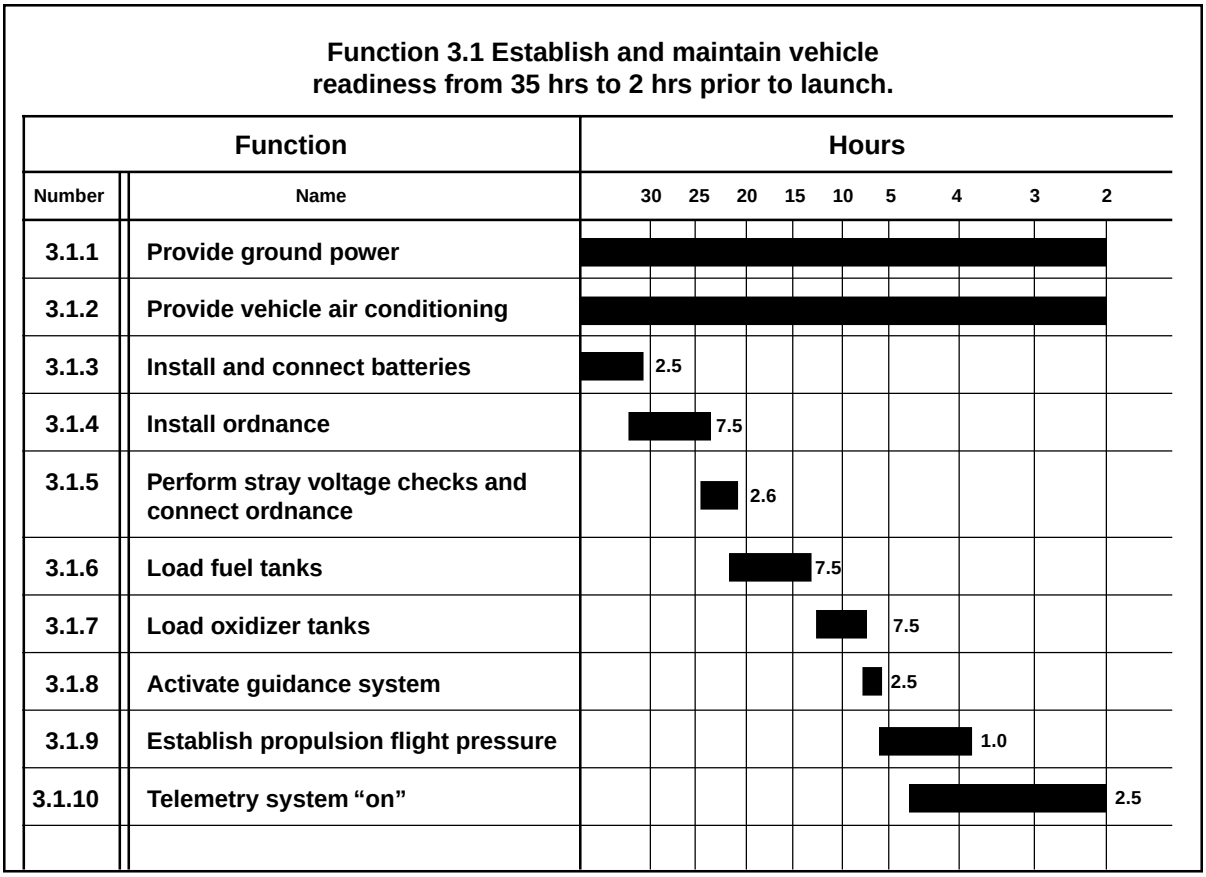

Figure 5-8. Time Analysis Sheet

SUPPLEMENT 5-D

REQUIREMENTS ALLOCATION SHEET

The Requirements Allocation Sheet documents the connection between allocated functions, allocated performance and the physical system. It provides traceability between Functional Analysis and Allocation and Design Synthesis, and shows any disconnects. It is a major tool in maintaining consistency between functional architectures and designs that are based on them. (Function numbers match the FFBD.)

Requirements Allocation Sheet	Functional Flow Diagram Title and No. 2.58.4 Provide Guidance Compartment Cooling	Equipment Identification		
Function Name and No.	Functional Performance and Design Requirements	Facility Rqmnts	Nomen-clature	CI or Detail Spec No.
2.58.4 Provide Guidance Compartment Cooling	The temperature in the guidance compartment must be maintained at the initial calibration temperature of +0.2 Deg F. The initial calibration temperature of the compartment will be between 66.5 and 68.5 Deg F.			
2.58.4.1 Provide Chilled Coolant (Primary)	A storage capacity for 65 gal of chilled liquid coolant (deionized water) is required. The temperature of the stored coolant must be monitored continuously. The stored coolant must be maintained within a temperature range of 40–50 Deg F. for an indefinite period of time. The coolant supplied must be free of obstructive particles 0.5 micron at all times.			

Figure 5-9. Requirements Allocation Sheet (Example)