

**Functional Analysis  
for Facility Engineering Data Modeling  
using the  
PARTitioned eNginering DAta flow  
model (PANDA)**

by

D. H. Douglas Phan

H. Craig Howard

**TECHNICAL REPORT**

**Number 77**

November, 1992

**Stanford University**

# **TABLE OF CONTENTS**

<b>ABSTRACT</b> .....	vii
<b>CIFE TECHNICAL REPORT SUMMARY</b> .....	ix
<b>Chapter 1 — INTRODUCTION</b> .....	1
1.1 What Is Functional Analysis? .....	2
1.2 Why Is Functional Analysis Important? .....	3
1.3 Why Is Functional Analysis Relevant to Our Research? .....	4
1.4 What Is the Research Development Objective Here? .....	4
1.5 Organization of the Report.....	5
<b>Chapter 2 — PROBLEM DEFINITION</b> .....	1
2.1 Required Capabilities .....	7
2.1.1 Representing Participants and Participation .....	7
2.1.2 Representing Processes .....	8
2.1.3 Representing Data & Physical Flows, & Data Generation Operations..	9
2.2 Required Properties .....	11
2.2.1 Being More Formal Than Natural Languages .....	11
2.2.2 Being Graphical .....	11
2.2.3 Being Capable of Producing Highly Readable Schemata.....	11
2.2.4 Being Simple and Easy to Use .....	12
<b>Chapter 3 — BACKGROUND</b> .....	13
3.1 Brief Survey of Process Models.....	14
3.1.1 General Process Models.....	14
3.1.2 In the Domain of Facility Engineering .....	16
3.1.3 Summary and Conclusions of Survey .....	17

3.2 Evaluation of the Five Selected Models .....	17
3.2.1 Summary of the Selected Models .....	17
3.2.2 Evaluation Criteria .....	19
3.2.3 Results .....	20
3.2.4 Discussion .....	24
3.3 Conclusion .....	25
<b>Chapter 4 — THE EXTENDED MODEL: PANDA .....</b>	<b>26</b>
4.1 Overview of PANDA.....	27
4.2 Key Concepts and Graphical Representations .....	30
4.2.1 Partition I: Participants.....	30
4.2.2 Partition II: Process .....	32
4.2.3 Partition III: Data-Material-Products (DMP).....	38
4.2.4 Summary of Concepts and Graphical Representations .....	45
4.3 Rules.....	48
4.3.1 Syntactic Rules.....	48
4.3.1.1 Syntactic Rules For Nodes.....	48
4.3.1.2 Syntactic Rules For Links.....	52
4.3.2 Semantic Rules.....	55
4.4 Schema Transformation Operations.....	56
<b>Chapter 5 — USING PANDA.....</b>	<b>60</b>
5.1 Scope Definition before Functional Analysis .....	61
5.2 Mixed Two-Pass Methodology .....	61
5.3 Guidelines for Using Concepts of PANDA .....	64
5.3.1 Using Participant and Participation.....	64
5.3.2 Functional Decomposition Using Subprocess .....	66
5.3.3 Using Precedence Relationship.....	69
5.3.4 Using Decision and Alternative .....	69

5.3.5 Using Boundary .....	70
5.3.6 Using Data Item and Data Repository .....	72
5.3.7 Using Flow Network .....	73
5.4 Guidelines for Drawing Partitioned Data Flow Diagrams .....	73
5.4.1 Labeling All Nodes .....	73
5.4.2 Numbering Activities and Decision Nodes .....	74
5.4.3 Laying Out Subprocesses .....	75
5.4.4 Annotating the Diagram .....	76
5.5 Validation of Functional Schemata .....	77
<b>Chapter 6 — A CASE STUDY .....</b>	<b>77</b>
6.1 Case Study and Scope of Analysis Involved .....	78
6.2 Functional Decomposition of the Tower Engineering Process .....	78
6.2.1 Breakdown of the Process into Phases .....	78
6.2.2 Breakdown of the Phases into Functions .....	79
6.3 Detailed Description of the Process .....	80
6.3.1 Transmission Line Analysis and Design Phase .....	81
6.3.2 Tower Structural Conceptual Design .....	83
6.3.3 Tower Structural Detailed Design .....	86
6.3.4 Tower Construction Planning .....	89
6.3.5 Tower Construction Execution .....	92
6.3.6 Tower Facility Management .....	94
6.4 Graphical Functional Schemata of the Process .....	95
<b>SUMMARY &amp; CONCLUSIONS .....</b>	<b>116</b>
<b>REFERENCES .....</b>	<b>121</b>
<b>GLOSSARY .....</b>	<b>127</b>

## LIST OF FIGURES

<b>FIGURE 1.3.1:</b>	An Overview of The Primitive-Composite (or P-C) Approach.....	5
<b>FIGURE 3.1:</b>	Work Done in Process Modeling.....	15
<b>FIGURE 4.1.1:</b>	A Sample Partitioned Data Flow Diagram Using PANDA.....	29
<b>FIGURE 4.2.1.1:</b>	Executive vs. Supporting Roles For Participants.....	31
<b>FIGURE 4.2.2.1:</b>	Activities Related by Precedence Relationships.....	32
<b>FIGURE 4.2.2.2:</b>	A Sample Decision in Evaluating a Tower Geometry....	33
<b>FIGURE 4.2.2.3:</b>	An Sample Interference During Tower Construction.....	34
<b>FIGURE 4.2.2.4:</b>	A Sample Subprocess Involving Dimensioning and Laying out Tower Members and Connections.....	35
<b>FIGURE 4.2.3.1:</b>	A Sample Data Repository Node.....	38
<b>FIGURE 4.2.3.2:</b>	Sample Data Items and Data Flow.....	39
<b>FIGURE 4.2.3.3:</b>	An Illustration of a Data Source, Data Repository, Data Item, and Data Generation Operation.....	40
<b>FIGURE 4.2.3.4:</b>	Sample Material, Product and Physical Flow.....	43
<b>FIGURE 4.2.3.5:</b>	An Illustration of Mixed Flow and Flow Network.....	44

## ***LIST OF FIGURES (cont.)***

<b>FIGURE 5.3.1:</b>	Defaulting Conventions for Participation Links.....	64
<b>FIGURE 5.3.3.1:</b>	An Example of Applying Disjunctive Precedence Rule to An Activity.....	69
<b>FIGURE 5.3.5.1:</b>	Boundary Nodes Delimiting and Connecting Subprocesses.....	71
<b>FIGURE 5.3.5.2:</b>	Subprocesses Sharing More Than One Boundary Node.....	72
<b>FIGURE 5.4.1.1:</b>	Node Label.....	74
<b>FIGURE 5.4.2.1:</b>	Numbering Activities and Decision Nodes.....	74
<b>FIGURE 5.4.3.1:</b>	Two Suggested Layouts of Subprocesses.....	75
<b>FIGURE 5.5.1:</b>	A Checklist for Validation of Functional Schemata.....	77
<b>FIGURE 6.2.1.1:</b>	Major Phases of the Tower Facility Engineering Process.....	80
<b>FIGURE 6.2.2.1:</b>	Hierarchical Functional Decomposition of the Phases of the Tower Engineering Process into Functions.....	81

## **LIST OF TABLES**

<b>TABLE 3.2a:</b>	Evaluation of Selected Models in Supporting the Participant Features.....	21
<b>TABLE 3.2b:</b>	Evaluation of Selected Models in Supporting the Process Features.....	22
<b>TABLE 3.2c:</b>	Evaluation of Selected Models in Supporting the Data, Material and Products (DMP) Features.....	23
<b>TABLE 3.2d:</b>	Evaluation of Selected Models in Having the Required Properties.....	24
<b>TABLE 4.2.4.1a:</b>	Concepts and Graphical Representations in Partition I....	45
<b>TABLE 4.2.4.1b:</b>	Concepts and Graphical Representations in Partition II..	46
<b>TABLE 4.2.4.1c:</b>	Concepts and Graphical Representations in Partition III.	47
<b>TABLE 4.3.1.1:</b>	Matrix of Permissible Node Linkages in PANDA Using the Appropriate Link Types .....	54
<b>TABLE 5.3:</b>	Nodes That Are Likely to Be Duplicated.....	65

## **Abstract**

Our research focuses on developing a general approach to the conceptual modeling of facility engineering data and to the data integration support of many participants, phases, and computer applications. Functional analysis is highly relevant to the research. It is particularly important for representing data and developing databases in facility engineering. The development work described here was motivated by the need for a reference model for functional analysis that supports our research goal. Traditionally, the Data Flow model has been the most popular choice for doing general functional analysis of a variety of processes. It provides the basic concepts of data repository, data flow, activity, and data source or sink. However, analyzing complex facility engineering processes places additional requirements on that model.

In this paper, we propose the *PARtitioned eNginEering DAta flow model* (abbreviated as *PANDA*), an extension of the Data Flow model developed to meet those requirements. *PANDA* supports the concepts needed to analyze facility engineering processes, while adhering as much as possible to the simplicity and ease of use of the Data Flow model. *PANDA* adds to the original model the concepts of participant, participation, precedence relationship, decision, alternative, interference, subprocess, boundary, data item, data generation, material or product, physical flow, mixed flow and flow network. In addition to its concepts, *PANDA* has graphical representations that are consistent with those of the original Data Flow model. Moreover, *PANDA* has a unique architecture that includes three partitions: (1) Participants, (2) Process and (3) Data-Material-Products. With *PANDA*, the analyst can functionally decompose a process into many hierarchical levels of description. At the detailed level, the data flow diagram is structured according to the three major partitions. Therefore, the diagram is also called the Partitioned Data Flow diagram, or P-diagram. This architecture helps the analyst organize his or her thinking about a complicated engineering process. It also enables the analyst to produce functional schemata that are highly



readable, both conceptually and graphically. PANDA also provides syntactic and semantic rules that govern the way in which the concepts should be used. Several basic schema transformation operations are provided to enable the analyst to develop a functional schema incrementally. To assist its users, PANDA offers a customized methodology that benefits from the model's partitioned architecture and guides the analyst in applying the model to his or her domain. PANDA also provides guidelines for using specific concepts of the model, validating the resulting functional schemata and drawing the Partitioned Data Flow diagrams of those schemata. In the future, we plan to apply PANDA to other facility engineering domains. The experience gained will help us further enhance the model.

*Keywords: functional analysis, data modeling, facility engineering, reference model, functional decomposition, participants, process, data flow, material, product, functional schema.*

# ***CIFE Technical Report Summary***

## **1. Abstract:**

Building common project information systems has become a viable project management strategy in order to establish effective communication of facility data. Functional analysis is particularly important for developing information systems in facility engineering. Its purpose is to understand what information is used in the activities of a process and how that information is exchanged among those activities. The main objective of the work described here is to develop a reference model for functional analysis of facility engineering processes. This report presents the PArtitioned eNgeering DAta flow model (abbreviated as PANDA), a development to meet the above objective.

## **2. Subject:**

PANDA is an extension of the Data Flow model developed for functional analysis of facility engineering processes. PANDA provides the concepts and graphical representations necessary to analyze those processes. Moreover, PANDA has a unique architecture that includes three partitions: (1) Participants, (2) Process and (3) Data-Material-Products. PANDA also provides syntactic and semantic rules for using its concepts, and several basic schema transformation operations for developing a functional schema incrementally. PANDA offers a customized methodology and guidelines that assist the analyst in applying the model to his or her domain.

## **3. Objectives/Benefits:**

Our research goal is to develop a structured approach to the conceptual modeling of facility engineering data that supports data integration among many participants, life-cycle phases, and computer applications. Functional analysis is highly relevant to our goal. Traditionally, the Data Flow model has been the most popular choice for doing general functional analysis of a variety of processes. However, analyzing complex facility engineering processes places additional requirements on that model. The objective of this work is to develop a reference model for functional analysis of facility engineering processes. We define a reference model as a set of concepts, rules and operations needed to do the job, as well as a methodology and the guidelines for using those concepts, rules and operations. Using such a model, a database designer can study the facility engineering process in which the information system under development will be used. Understanding this process is critical to the successful development of the system.

#### **4. Methodology:**

This development work began with a case study involving electrical utility transmission towers. This case study shows how a facility engineering process works and lead to requirements for developing the reference model that meets the above objective. These requirements were used to guide our development work. The result of the work, PANDA, was also used to model the transmission tower engineering process investigated in the case study.

#### **5. Results:**

PANDA includes the concepts of participant, participation, precedence relationship, decision, alternative, interference, subprocess, boundary, data repository, data item, data source and sink, data flow, data generation, material or product, physical flow, mixed flow and flow network. PANDA has a unique architecture that includes three partitions: (1) Participants, (2) Process and (3) Data-Material-Products. PANDA also provides graphical representations, syntactic and semantic rules, schema transformation operations, a customized methodology, and many guidelines.

The development of PANDA led to the following conclusions: Functional analysis is crucial to the development of information systems in facility engineering. Having a proper model for functional analysis in facility engineering is important. A useful model for functional analysis in facility engineering must have graphical representations and built-in features that would automatically produce highly readable graphical functional schemata. Without these properties, the model might not be used. A methodology and guidelines for using a suggested model improves the likelihood that the model will be used effectively. Support software for functional analysis in facility engineering is definitely needed.

#### **6. Research Status:**

In the future, we plan to apply PANDA to other facility engineering domains. The experience gained will help us further enhance the model. We also see the need for developing computer-aided software engineering tools that can assist functional analysis of facility engineering processes using PANDA. Such a development would be possible when more resources become available.

# **Chapter 1**

---

## **INTRODUCTION**

In this introduction, we first explain what functional analysis is. We point out its importance to facility engineering data modeling and relevance to our research. We then state the specific objective of the work described in this report. Finally, we explain how the report is organized.

### **ORGANIZATION**

- 1.1 What is Functional Analysis?
- 1.2 Why is Functional Analysis Important?
- 1.3 Why is Functional Analysis Relevant to Our Research?
- 1.4 What is the Research Development Objective Here?
- 1.5 Organization of the Report

The design and construction of a facility, whether it is a space station, high-rise building, cable-suspended bridge, or electrical utility transmission tower, typically involves a complex engineering process. This process requires close coordination among the owner, designers, contractors and builders in all phases of the project. The goals are to maintain efficiency and to minimize costs while ensuring high engineering quality. Effective communication of authoritative facility data among the principal players throughout the phases is vital to achieve these goals. Building common project information systems has become a viable project management strategy in order to establish such communication [Froese 92]. In fact, the rapid advancement in computer technology continues to offer more economic means of electronic data storage and more capable tools to build engineering information systems. However, even with increasingly affordable hardwares and powerful softwares, designing and implementing information systems capable of supporting data sharing and communication in facility engineering is still an important and challenging topic of research [Phan 91].

Functional analysis plays a significant role in developing information systems. It concentrates on understanding the process in which the information system under development will be used. Functional analysis is particularly important for representing data and developing databases in facility engineering.

---

## ***1.1 What is Functional Analysis?***

---

Functional analysis is the study of information flow among the activities of a process or processes in an enterprise. Its purpose is to understand what information is used in each activity and how that information is exchanged among the activities. Functional analysis is an important part of developing both databases and applications that operate on the databases [Batini 92]. Functional analysis yields a “functional schema” that shows how a database being developed will be used. This schema can be used to develop a high-level description of the database structure (or “conceptual schema”). The conceptual schema can then be used to develop a computer-processable representation of that structure (or “logical schema”).

Functional analysis is limited to modeling the functional components of processes, namely the activities, and the information flow among the activities. Functional analysis does not consider the social, economic or environmental impacts of the process, the duration and coordination of activities, the allocation of resources, or the costs or quality of products that result from the process. Therefore, functional analysis clearly differs from the process modeling that has been studied in other fields. Consequently, we specifically use the term “functional analysis” instead

of “process modeling.” In this report, we also use the term “analyst” to refer to the person or team who carries out the functional analysis.

---

## ***1.2 Why is Functional Analysis Important?***

---

Generally speaking, functional analysis is important to any effort that requires an understanding of how an enterprise operates and how information is used to support the enterprise’s operation. Specifically, functional analysis aids in representing data and developing databases in facility engineering for the following reasons:

- *Functional analysis is critical to understanding the data needs of the facility engineering process.* By modeling the activities and information flow, functional analysis helps the analyst understand what data is needed, how it is used in the activities, and who the users are. This understanding is critical to representing the data and designing the database in a way that supports that process.
- *Functional analysis can help refine the design requirements and improve the database design.* The design requirements of the database may have been defined prior to functional analysis. However, functional analysis helps the designer understand the data needs of the process as well as set more definitive and specific design requirements. Moreover, functional analysis can lead to a functional schema of the process. The information from this schema can be used directly to design the conceptual schema, thus insuring that the conceptual schema has certain desired design qualities. This information is also valuable to later in mapping from the conceptual schema to the logical schema.
- *Functional analysis can help verify and test the database design.* Using the functional schema, the database designer can verify the completeness of the database: All data used in the process is included in the database, and all activities of the process are supported by operations that manipulate the database [Batini 92]. The designer can select activities shown in the functional schema to define test cases to evaluate other desired qualities of the conceptual schema (e.g., flexibility in supporting different participants of the process).
- *Functional analysis is crucial to the support of data integration in facility engineering.* Facility engineering typically involves multiple project participants, several life-cycle phases, and different computer applications. These determine the principal users of data in the facility engineering process. “Data integration” refers to maximizing the sharing of data representations among these users. Functional analysis aids in identifying these users and

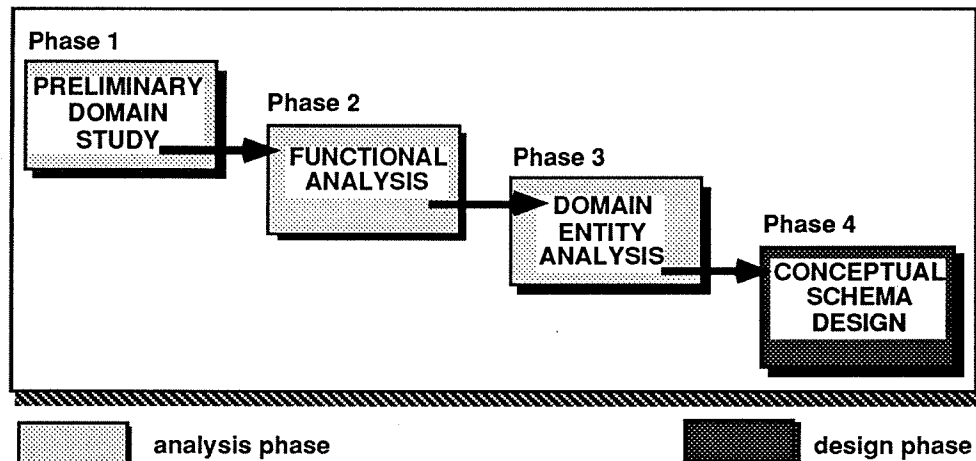
understanding their needs for using and exchanging data. Therefore, functional analysis is crucial to the development data representations that can be shared among users. Moreover, the users would be more likely to accept and use the representations.

- *Functional analysis helps decide which legacy data of the enterprise should be preserved.* For a complex facility engineering process involving multiple large-scale operations in an enterprise, storing every single data might not be economically feasible. In that case, functional analysis provides the big picture of the overall enterprise's operations and aids in deciding which legacy data of the enterprise should be stored and maintained.

### 1.3 Why Is Functional Analysis Relevant to Our Research?

Our research focuses on the conceptual modeling of facility engineering data. (Conceptual modeling is the process of analysis and design leading to the development of conceptual schemata.) Moreover, it emphasizes data integration. Our research goal is to develop a structured approach to the conceptual modeling of facility engineering data that supports data integration among many participants, life-cycle phases, and computer applications. We call our approach the "Primitive-Composite (or P-C) Approach."

Functional analysis is clearly relevant to this research. As Figure 1.3.1 shows, the P-C Approach consists of three analysis phases leading to a conceptual schema design phase; functional analysis is one of the analysis phases.



**FIGURE 1.3.1: An Overview of The Primitive-Composite (or P-C) Approach.** This is an approach to the conceptual modeling of facility engineering data that we are developing. The results of each phase are used directly in the subsequent phases. Our future publications will explain this approach and its four phases in detail.

## ***1.4 What Is the Research Development Objective Here?***

---

*The main objective of the work described here is to develop a reference model for functional analysis that supports our research goal. We define a reference model as a set of concepts, rules and operations needed to do the job, as well as a methodology and the guidelines for using those concepts, rules and operations.*

This development work began with a case study involving electrical utility transmission towers. This case study shows how a facility engineering process works and lead to requirements for developing the reference model that meets the above objective. These requirements are presented in the next chapter.

## ***1.5 Organization of the Report***

---

The remainder of this report is organized into five chapters: problem definition, background, the model, using the model, and a case study. In Chapter 2, we further define the problem by stating the capabilities that the reference model should provide and the properties it should have. We also explain why these capabilities and properties are required. In the first part of Chapter 3, we give a brief survey of existing reference models. In the second part, we evaluate five candidate models, one of which is the Data Flow model. We then present the five models and evaluate them using criteria derived from the required capabilities and properties in Chapter 2. We also justify the selection of the Data Flow model on which we based our extension work. Chapter 4 presents the extended model, the PArtitioned eNginering DAta flow model (or PANDA). We begin with an overview of the model and then introduce its key concepts and graphical representation. We also present its syntactic and semantic rules and basic schema transformation operations. Chapter 5 shows how to use this model, namely by applying its concepts, graphical representations, rules and operations to facility engineering processes and drawing graphical functional schemata. Chapter 6 presents a case study in which the engineering process of transmission towers was modeled using PANDA. We first explain the case study and define the scope of the analysis involved. The tower engineering process is then decomposed into smaller functional units and described in detail. Finally, the graphical functional schemata of the process that result from using PANDA are shown. This report ends with a summary of the five chapters, conclusions about this work, and some acknowledgments.



## **Chapter 2**

---

# **PROBLEM DEFINITION**

We already mentioned that our objective is to develop a reference model for functional analysis that supports facility engineering data modeling. In this chapter, we further define the problem by stating the capabilities and properties that the reference model should have. We also explain why these capabilities and properties are required. These were used to evaluate existing models and guide our development work.

### **ORGANIZATION**

#### **2.1 Required Capabilities**

- 2.1.1 Representing Participants and Participation
- 2.1.2 Representing Processes
- 2.1.3 Representing Data and Physical Flows, and Data Generation Operations

#### **2.2 Required Properties**

- 2.2.1 Being More Formal than Natural Languages
- 2.2.2 Being Graphical
- 2.2.3 Being Capable of Producing Highly Readable Schemata
- 2.2.4 Being Simple and Easy to Use

---

## **2.1 Required Capabilities**

---

*The reference model should have the following capabilities:*

- *the ability to represent the participants and their participation (i.e., the capacities in which they are involved),*
- *the ability to represent complicated, non-linear processes of facility engineering, and*
- *the ability to represent complex data and physical flow within the process as well as the relationships among data as it is generated in the process.*

Each of the three capabilities is associated with a principal characteristic of the facility engineering process. In the following sections, we describe those characteristics and explain why the capabilities are required. We then divide each capability into specific elements, or *features*, that we look for in the needed reference model. In fact, we use these features to evaluate selected models in the next chapter.

### **2.1.1 Representing Participants and Participation**

The model should be capable of representing the participants and their participation (i.e., the capacities in which they are involved) explicitly in the functional schema. This required capability is associated with an important characteristic of the facility engineering process: *Multiple human participants from various disciplines are involved in the process in different capacities.* For example, the process of designing and constructing utility transmission towers involves electrical planners, electrical engineers, structural engineers, structure detailers, fabricators, construction managers and crews.

**JUSTIFICATION OF THIS CAPABILITY** We strongly feel that this capability is needed for the following reasons: First, the participants are as important to the process as the activities or the data. Second, these participants are users of data. By representing them explicitly in the schema, the analyst must automatically think about what data they need and how they use the data. Third, these participants have different needs for the facility data. Representing the participants and their participation in the schema helps verify the correctness of the schema. The designer would ask the question: Which activities and data should be associated with a particular participant? Fourth, when several participants are involved in the same subprocess or activity in different capacities, they need to communicate and exchange data. They can also have conflicts such as in

naming data. With this capability, the analyst can identify areas of the functional schema in which he or she must deal with these issues. For instance, the analyst may resolve naming conflicts by providing clear data definitions in a data dictionary that also results from functional analysis. Last, this capability directly supports data integration among different process participants: Because of it, the functional schema contains information that can be used directly to design a conceptual schema capable of supporting multiple participants.

**FEATURES** This required capability yields the features listed below. We label them with the abbreviation PAR (for participants) followed by a number for later reference.

*PAR-1. Representing the participants in the process.*

*PAR-2. Representing participation or the capacities in which the participants are involved.*

### **2.1 .2 Representing Processes**

The model should be capable of representing processes of facility engineering that are complicated and non-linear. *A facility engineering problem may consist of several subproblems. Solving each problem requires synthesis, analysis and decisions. Consequently, the process can be highly complicated.* The engineering process normally extends throughout the life cycle of the facility. Technically, it can be decomposed into several subprocesses, which can be decomposed further into activities. This is called functional decomposition. For instance, the process of transmission tower engineering can be broken down into six phases that involve programming, conceptual design, detailed design, construction planning, construction execution, and management of the facility. In addition, activities are dependent upon one another. Precedence (i.e., the notion of “do A before B”) is an important type of relationships among activities. This relationship type represents the temporal constraints placed on the execution of the activities. The facility engineering process also includes decision-making activities. These involve analyzing competing alternatives and choosing among them.

*Moreover, the process is not linear.* Subprocesses can take place concurrently since multiple participants are involved. At different times, subprocesses can be “activated,” “suspended,” “resumed” or “terminated.” In addition, design synthesis involves the typical “Propose-Verify-and-Redesign” loop [Chandrasekaran 88], which takes more than one iteration. For example, the structural engineer begins by proposing a preliminary tower geometry in order to initiate the tower loads computation. Once the loads are computed, the geometry is verified and may be redesigned.

**JUSTIFICATION OF THIS CAPABILITY** There is one strong justification for this required capability: Failing to represent a process has a direct negative effect on modeling the data used in that process. In other words, the more accurate the representation of the process is, the more effective the representation of the data will be.

**FEATURES** The following features are derived from this required capability. They are labeled with the abbreviation PRO (for process) followed by a number for later reference.

- PRO-1. Representing processes and activities .*
- PRO-2. Providing a structured way to decompose processes into functional units such as subprocesses and activities.*
- PRO-3. Providing a way to reconnect subprocesses that were analyzed and represented separately to the parent process.<sup>1</sup>*
- PRO-4. Representing the precedence relationships among activities.*
- PRO-5. Representing process non-linearity, including concurrent subprocesses and design iterations.*
- PRO-6. Representing decisions and alternatives.*

### **2.1 .3 Representing Data and Physical Flows, and Data Generation Operations**

The model should be capable of representing complex data and physical flow within the process as well as the relationships among data as it is generated in the process. The facility engineering process is complicated, and the solutions to the overall problem are by no means simple. Synthesizing a solution requires utilizing all the necessary information resources that are available from each of the participants. Typically, data originates from some source. For example, vendors provide catalogs that contain standard parts' data. Data is actually stored in and retrieved from persistent repositories such as those catalogs or electronic databases, companies' design manuals and program input and output files. This data can be used in activities, which in turn produce new data. The new data can then be used in subsequent activities, and so on. A single activity can use several data items or repositories from more than one source. As a result, *the flow of data among activities in the process is complex.*

---

<sup>1</sup> This is the inverse of the above feature, PRO-2.

As the process evolves, the amount and complexity of the data increases over time. During the life-cycle phases of the facility, new data is generated from existing data from several sources. *Therefore, there exist special relationships (e.g., is-derived-from, is-a-version-of, is-stored-into, is-combined-into) between data as it is generated in the process.*

In addition, real-life processes utilize material and products to build the facility. For transmission towers, the fabricated steel parts and assembled tower sections are examples, respectively, of the material and intermediate products used in constructing the tower. Thus, *the physical flow of material and products is also an important part of the process, especially in the construction execution and operations and maintenance phases.*

**JUSTIFICATION OF THIS CAPABILITY** The justifications for this required capability are as follows: First, an important part of functional analysis is representing the flow of data used in the process. Second, functional analysis of real-life facility engineering processes would not be complete without also representing the physical flow of material and products. Third, representing data generation operations in the functional schema provides useful information that can be used directly in the subsequent conceptual schema design. In fact, the way in which data is actually generated in the process should directly affect the way in which it is represented in the database schemata.

**FEATURES** The following features result from this required capability. They are labeled with the abbreviation DMP (for data, material and products) followed by a number for later reference.

- DMP-1. Representing data repositories, which store data.*
- DMP-2. Representing explicitly individual data items that may or may not later be stored in a data repository.*
- DMP-3. Representing complex data flow into and out of activities.*
- DMP-4. Representing data sources and sinks, which are people or things that are the prime originators and receivers of data repositories or data items.*
- DMP-5. Representing the relationships among data as it is generated in the process.*
- DMP-6. Representing material and products that are resources, intermediate results or final results of the process.*
- DMP-7. Representing physical flow of material and products.*

---

## **2.2 Required Properties**

---

*The reference model should have the following properties: It should be more formal than natural languages, graphical, capable of producing highly readable graphical functional schemata, and being as simple and easy to use as possible.* Next, we describe these properties and explain why they are required.

### **2.2.1 Being More Formal Than Natural Languages**

The model should include a finite set of concepts that have clear and precise definitions. This enables the analyst to produce process specifications that are less ambiguous than those using natural languages.

**JUSTIFICATION OF THIS PROPERTY** Without this property, the analyst might produce process specifications that are ambiguous, verbose, inaccurate, or subject to interpretation. Requiring formality of the model is a way to minimize these deficiencies. Formality also increases the potential for developing computer-aided software engineering tools to do functional analysis.

### **2.2.2 Being Graphical**

The model should have graphical representations for creating drawings of the process' functional schema. We call these drawings "graphical functional schemata." This property enables the analyst to produce functional schemata that are pictorial, highly descriptive and also concise.

**JUSTIFICATION OF THIS PROPERTY** Without this property, the model might not be used. As [Ceri 86] pointed out, the popularity of models such as the Data Flow model [Gane 79, Batini 92] and the Structured Analysis Design Technique (SADT) [Ross 77a] attests to the importance of having graphical representations for the model. Moreover, [Ross 77b] demonstrated that using SADT's graphical representation, the analyst can generate process blueprints similar to those of building structures. Process blueprints provide the participants with a structured and precise means of communicating about the process.

### **2.2.3 Being Capable of Producing Highly Readable Schemata**

Graphical representations have been proposed in a number of existing models. However, the model should also have built-in features that automatically yield highly readable graphical

functional schemata. This property enables even a novice analyst to read the schemata. The schema readability is measured not only in terms of how well the analyst can interpret it (graphical readability), but also in terms of how well he or she can comprehend it (conceptual readability).

**JUSTIFICATION OF THIS PROPERTY** Without this property, the analyst would have more freedom to draw in anyway he or she chooses, but could produce graphical functional schemata for complex processes that would be very difficult to read and comprehend. We have seen many poor graphical schemata. Having graphical representations is simply not enough. Rather, the model needs built-in features that enable the analyst to automatically produce highly readable graphical functional schemata. (For instance, our solution, PANDA, automatically divides a graphical functional schema into three partitions. These partitions greatly enhance both the graphical and conceptual readability of the schema.)

#### ***2.2.4 Being Simple and Easy to Use***

The model should be as simple as possible. This will make it easier to use. It should be designed so that a novice analyst, with minimal learning, could apply its basic features. An experienced analyst would use the more advanced features in order to be more skillful and efficient.

**JUSTIFICATION OF THIS PROPERTY** Without this property, the model might not be used. In fact, [Ceri 86] gave several reasons why many suggested models have not been used. The most critical reason was the difficulty in learning and using a intrinsically complicated model.

Key concepts for this chapter: *reference model, functional analysis, functional schema, participants, participation, process, functional decomposition, data flow, physical flow, data generation operations, graphical functional schema, schema readability (conceptual and graphical).*

## **Chapter 3**

---

# **BACKGROUND AND EVALUATION OF SELECTED MODELS**

This chapter has two parts. In the first half, we give a brief survey of existing reference models. We introduce models proposed for general purposes and for the domain of facility engineering. In the second part, the five candidate models selected in this study are evaluated. First, the five models are summarized, and the evaluation criteria that come from the required capabilities and properties in the previous chapter are presented. We then discuss the results of the evaluation. This chapter concludes by justifying the model chosen for this research.

### **ORGANIZATION**

- 3.1 Brief Survey of Process Models
  - 3.1.1 General Process Models
  - 3.1.2 In the Domain of Facility Engineering
  - 3.1.3 Summary and Conclusions of Survey
- 3.2 Evaluation of the Five Selected Models
  - 3.2.1 Summary of the Selected Models
  - 3.2.2 Evaluation Criteria
  - 3.2.3 Results
  - 3.2.4 Discussion
- 3.3 Conclusion



---

### **3.1 Brief Survey of Process Models**

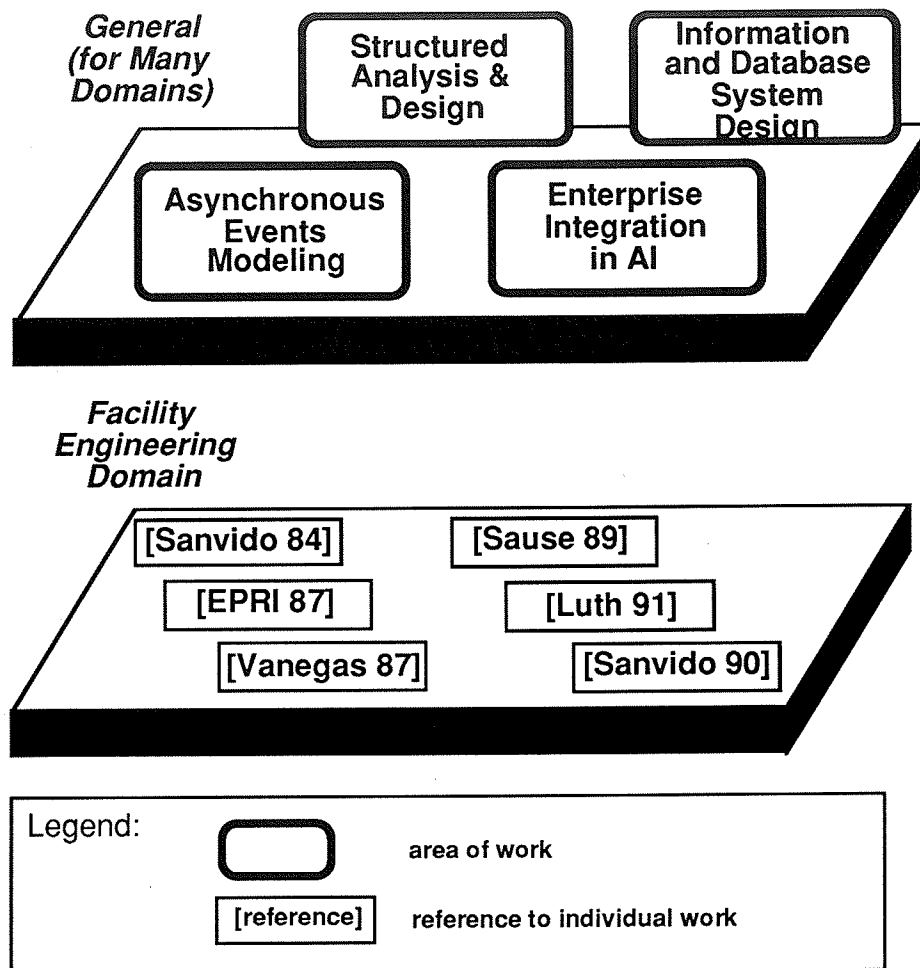
---

In the introduction, we explained the difference between functional analysis and process modeling. The objective of this work is to develop a reference model for functional analysis that support facility engineering data modeling. However, to survey the current state of knowledge, we wish to review existing models that have been proposed for all process modeling purposes. For convenience, we call these “process models.”

#### **3.1.1 General Process Models**

*The general process models are those that can be applied to many domains. They were not developed to meet the specific requirements of any one particular domain. A large number of these models have been proposed for different purposes. As Figure 3.1 shows, those that are relevant to this study stemmed from many areas of research and development:*

1. Software engineering has provided a number of process models. In particular, the *structured analysis and design* approach has contributed the popular Data Flow model (and its variations) [Gane 79, Yourdon 79, De Marco 82] and the Structured Analysis Design Technique (SADT) [Ross 77a, 77b].
2. Since the sixties, numerous process models have been proposed under the heading “Information System Design” [Lockemann 86]. They include the Information Systems Work and Change Analysis (ISAC) approach [Lundeberg 82], the Conceptual Information Analysis Methodology (CIAM) [Gustavsson 82] and the Integrated Computer Aided Manufacturing Definition (IDEF) methodologies [Bravoco 85a, 85b, Mayer 92], including IDEF0 and IDEF3, to name a few. Other models were developed specifically for “Database System Design.” They include the DATAID Database Design Methodology [Ceri 86], Nijssen’s Information Analysis Method (NIAM) [Verheijen 82] and the Active and Passive Component Modelling (ACM/PCM) [Brodie 82] to name a few. In addition, a number of so-called Conceptual Modeling Languages (CML), defined in [Borgida 85], are primarily used for conceptual database design, but also make possible the formal specifications of processes. These specifications can be automatically interpreted and verified. These languages include TAXIS [Mylopoulos 80], Galileo [Albano 85] and Requirements Modeling Language (RML) [Greenspan 86].



**FIGURE 3.1: Work Done in Process Modeling.** General process models have been introduced from many different areas. They differ in terms of origin, development objectives, concepts, methodology, formalization, etc. In facility engineering, less work has been done. Most of what has been done puts more emphasis on accurately depicting the process than on providing a reference model for analyzing and representing the process.

- According to [Lockemann 86], the *theory of nets* [Peterson 77] has resulted in a number of models that have a firm mathematical underpinning. The strength of these models is in modeling *asynchronous process events*. The models include Petri Nets [Peterson 77], Information Control Nets (ICN) [Ellis 79] and the Information Management Language Inscribed High-Levels Petri Nets (IML) [Richter 82]. Petri Nets is a powerful and mathematical model that has wide applications in the area of process modeling. In fact, ICN and IML are applications as well as extensions of Petri Nets to different domains.

4. More recently, research and development on “Enterprise Integration” in the field of Artificial Intelligence (AI) has focused on acquiring an understanding of how an enterprise operates and how information plays a role in supporting the enterprise’s operation. Work in this area is under way at several universities [Fox 92, Jagannathan 92, Srinivasan 92] and industry-funded research centers [Billmers 92, Bradshaw 92, Grosos 92, Gruber 92] across the country. The general direction here is to explore advanced AI techniques to support enterprise modeling, automation and integration.

Another class of process models comes from social system theories that strongly emphasize the human organizational aspects of real-life processes. These models are not considered relevant to this study.

A more complete listing and review of reference models in the above areas may be found in [Olle 83], [Ceri 86], [Lockemann 86], and [Chadha 91].

### **3.1.2 In the Domain of Facility Engineering**

*Less work of this nature has been done in facility engineering.* The work done in this domain is presented here in order of development.

[Sanvido 84] developed a “construction process model” that is primarily a management and control structure for optimizing construction performance at the site. The Plant Information Network (PIN) from the Electric Power Research Institute (EPRI) [EPRI 87] provides a comprehensive, descriptive documentation of a typical electrical power plant engineering process throughout the plant’s life-cycle. It uses an activity listing, an activity process diagram showing the activities’ precedence relationships, and the Entity-Relationship data model to model the process data. [Vanegas 87] presents a descriptive model for the early phases of building design, coupled with some underlying concepts and graphical symbols. Its objective is to integrate design and construction during those phases. [Sause 89] proposes a “model” for structural engineering design processes. It is mainly a description of a cognitive design process. The Integrated Building Process Model (IBPM) [Sanvido 90] delineates a prototypical engineering process for buildings that enclose between 75,000 and 250,000 square feet of usable floor space. The essential functions considered are managing, planning, design, construction and operations. The model uses the general IDEF0 (Integrated Computer Aided Manufacturing Definition) model [Bravoco 85a, 85b]. [Luth 91] provides a verbal description of the facility engineering life-cycle typical of high-rise commercial office buildings. The life-cycle consists of functions (i.e., groups of activities) that center around interactions between the owner, designers and construction managers.

### 3.1.3 Summary and Conclusions of Survey

A large number of general process models have been introduced from various areas of research and development. They differ vastly in terms of development objectives and emphasis, underlying concepts, representational means, methodology, degree of formalization and software support tools. They also address various analysis and design tasks and phases in the information system life-cycle. Not all of them are reference models as defined in our introduction. As [Lockemann 86] points out, standardization to these models has not been done. Integration of these models and their software support tools to provide the user with a comprehensive working set of models and tools is still needed.

By contrast to the general process models, less work has been done in facility engineering. most of what has been done emphasized the accurate depiction of the process itself rather than the development of a reference model for analyzing and representing the process. Indeed, the majority of the work are functional schemata or verbal descriptions of facility engineering processes and not reference models as defined in the introduction. In other words, no “reference model” had been developed for facility engineering processes.

---

## 3.2 Evaluation of the Five Selected Models

---

### 3.2.1 Summary of the Selected Models

We selected five models for evaluation, based on three main reasons. First, all five models were selected in an earlier review by [Ceri 86] and again by [Batini 92]. Second, these five models are representative of the first three categories of general process models discussed in Section 3.1.1. Finally, we believed them to be the strongest candidates for this study.

This section summarizes these models by highlighting their key concepts (shown in italics). The literature referenced explains the models in detail.

**DATA FLOW MODEL** [Gane 79, Yourdon 79, De Marco 82, Batini 92]— This model is the simplest and by far the most popular of the five models. It has at least three variations: [Gane 79], [Yourdon 79] and [De Marco 82]. The one considered here comes from [Gane 79] and is reinforced later in [Batini 92]. The model covers the fundamental concepts of *process*, *data flow*, *data repository*, and *data source or sink*. It also provides graphical representations of these concepts. The resulting graphical functional schema is called a Data Flow diagram.

**STRUCTURED ANALYSIS DESIGN TECHNIQUE (SADT)** [Ross 77a, 77b] <sup>¥</sup> — This model advocates *structured decomposition* of processes into functional units. Its two distinctive fundamental concepts are *activity* and *information flow*. Activities are represented as boxes, and information flows are shown as arrows. Moreover, it introduces the notion of *roles* of information flow. Depending on its position relative to the activity, the information flow can play an *input*, *output*, *control* or *mechanism role*. In addition, the accompanying Structured Analysis (SA) language includes some forty features that can be used to generate blueprint-like process diagrams.

**INFORMATION SYSTEMS WORK AND CHANGE ANALYSIS (ISAC)** [Lundeberg 82] — This model was developed for analyzing activities, and information as well as information management problems and needs (or *change analysis*). It uses *activity graphs (A-graphs)* that are similar to data flow diagrams. An A-graph involves *sets*, *activities* and *flows*. There are *real sets* (of people or material), *message sets* (of documents or information) and *composite sets* (of people, material and messages). Therefore, the model includes *real or physical flows* of real sets and *mixed flows* of composite sets, in addition to *message flows*. Moreover, an A-graph has a defined structure. It uses *boundaries* to delimit the portion of the information system being considered. The inputs are placed above the boundaries, and the outputs below the boundaries. Arrows are used only to indicate a flow from bottom to top; by default, other flows do not require arrows.

**PETRI NETS** [Peterson 77] — This model uses the concepts *token*, *state* and *transition*. Tokens flow through the network and highlight the activated states as “active” (or “marked”). A transition is fired when its input states have at least one token. The effect of firing the transition is to remove one token from each of the input states of the transition and place one token into each of the output states. A transaction can be represented as a separate net. The model is based on a solid mathematical theory. In fact, the model was developed to describe discrete-event systems with asynchronous and concurrent events. However, it has a large number of extensions, subclasses and applications ranging from modeling hardware (i.e., large, powerful computers), software (e.g., operating systems, information systems), and formal languages to modeling production assembly lines.

---

<sup>¥</sup> The IDEF0 (*Integrated Computer Aided Manufacturing Definition*) model [Bravoco 85a, 85b] is similar to this model. On the other hand, the IDEF3 model [Mayer 92] is among the latest additions to the set of IDEF conceptual tools. IDEF3 provides a structured method for capturing the description of how a particular system or organization operates.

**INFORMATION CONTROL NETS (ICN)** [Ellis 79] — This model is an extension of Petri Nets used for modeling large office information systems. An information control net captures the notions *procedures*, *activities*, *precedence constraints*, and *resources* (including *information repositories*) in a graphical format. *Initiation arcs* and *termination arcs* indicate the beginning and end respectively of a procedure. Parallel processing of asynchronous activities can be shown by placing *AND nodes* before those activities. Dashed lines represent activities reading and storing data from and to information repositories. The model also distinguishes between short-term and long-term repositories. It includes two other important control structures: *decision* and *ramification*. A decision invokes only one activity that follows a single-decision alternative, whereas ramification can initiate more than one activity in parallel or at different times. The model has a formal mathematical underpinning. It can be used for analyzing real-life office automation, reorganization and streamlining.

### 3.2.2 Evaluation Criteria

We used four main criteria to evaluate the models. The first criterion comes from the required capabilities stated in Section 2.1 of the previous chapter. The last three criteria come from the required properties in Section 2.2. All these criteria are listed below in order of their relative importance:

1. *Supporting the features of the required capabilities*: The ordinal measurements for this criterion are simply “YES” and “NO.”
2. *Having graphical representations*: The ordinal measurements for this are “YES” and “NO.”
3. *Graphical readability and conceptual readability of resulting graphical functional schemata*: The ordinal measurements for this are “LOW,” “MEDIUM” and “HIGH.”
4. *Simplicity and ease of use*: The ordinal measurements for this are “LOW,” “MEDIUM” and “HIGH.”
5. *Formality*: [Ceri 86] classified specification languages into three categories: (1) *informal languages* such as natural languages, (2) *semi-formal* or *formatted* languages, which have known keywords, syntax and semantics but produce non-executable specifications, and (3) *formal* languages that allow automatic interpretation and checking of the specifications for errors such as incompleteness, inconsistencies and conflicts. To simplify the criterion and

keep it consistent with the other criteria, the corresponding measurements for this criterion are “LOW,” “MEDIUM” and “HIGH.”

### **3.2.3 Results**

Tables 3.2a to 3.2c summarize the evaluation using the first criterion. They are arranged according to the three required key capabilities of the needed reference model. This is done intentionally to show how the models perform in each category. Table 3.2d summarizes the evaluation using the last four criteria.

**TABLE 3.2a: Evaluation of Selected Models in Supporting the Participant Features**

<i>Required Features</i>	<i>Data Flow Model</i>	<i>SADT</i>	<i>ISAC</i>	<i>Petri Nets</i>	<i>Information Control Nets</i>
PAR-1: Participants in the process	YES (but indirectly as data sources or sinks)	NO	YES (but indirectly using real sets and real flows)	NO	NO
PAR-2: Participation (or the capacities in which the participants are involved)	NO	NO	NO	NO	NO



**TABLE 3.2b: Evaluation of Selected Models in Supporting the Process Features**

<i>Required Features</i>	<i>Data Flow Model</i>	<i>SADT</i>	<i>ISAC</i>	<i>Petri Nets</i>	<i>Information Control Nets</i>
PRO-1: Processes and activities  (Most models provide this fundamental feature.)	YES (By representing activities and processes as sets of activities)	YES (By representing activities and processes as sets of activities)	YES (By representing activities and processes as sets of activities)	YES (By representing activities in terms of states & transactions)	YES (By representing activities in terms of states & transactions)
PRO-2: Hierarchical structure for functional decomposition of processes	NO ( [Batini 92] adds top-down design strategy.)	YES (By advocating structured decomposition)	YES (By using reference code for activities)	YES (By modeling a transaction as a separate net)	YES
PRO-3: Reconnection of subprocesses that are analyzed and represented separately from the parent process	NO (But, [Batini 92] suggests "boundary equivalence")	YES	YES (By using reference code for activities)	YES	YES
PRO-4: Activities' precedence relationships ("do A before B")	NO	NO	NO	YES (By transaction firing rules)	YES (More explicit than Petri Nets)
PRO-5: Process non-linearity	NO	NO	NO	YES	YES (Using AND nodes)
PRO-6: Decisions and alternatives	NO (But activities can be designated as decisions.)	NO (But activities can be designated as decisions.)	NO (But activities can be designated as decisions.)	NO (But activities can be designated as decisions.)	YES (By using decision & ramification points)

**TABLE 3.2c: Evaluation of Selected Models in Supporting the DMP Features**

<i>Required Features</i>	<i>Data Flow Model</i>	<i>SADT</i>	<i>ISAC</i>	<i>Petri Nets</i>	<i>Information Control Nets</i>
DMP-1: Data repositories	YES	NO (No distinction of temporary and stored data)	YES (Message sets, no distinction of temporary and stored data)	NO	YES (As an extension to Petri Nets)
DMP-2: Data items represented explicitly	NO (Only implicitly as annotations to data flows)	NO	NO	NO	NO
DMP-3: Data flows into & out of activities (Most models provide this.)	YES ( Inputs & outputs of activities are denoted by flow directions.)	YES (It defines roles of flows as input, output, control or mechanism.)	YES	YES	YES
DMP-4: Data sources and sinks	YES (Sources and sinks are interface with the system.)	NO	NO	NO	NO (One can argue the other way with resources.)
DMP-5: Data generation relationships	NO	NO	NO	NO	NO
DMP-6: Material & products	NO	NO	YES (As real sets and real flows)	NO	YES (As an extension of Petri Nets)
DMP-7: Physical flow	NO	NO	YES (As real sets and real flows)	NO	YES (As an extension of Petri Nets)

**TABLE 3.2d: Evaluation of Selected Models in Having the Required Properties**

<i>Required Properties</i>	<i>Data Flow Model</i>	<i>SADT</i>	<i>ISAC</i>	<i>Petri Nets</i>	<i>Information Control Nets</i>
Formality	MEDIUM	MEDIUM	MEDIUM	HIGH	HIGH
Graphical representations for the model	YES	YES	YES	YES	YES
Graphical readability and conceptual readability of resulting graphical functional schemata	MEDIUM	MEDIUM	LOW	MEDIUM	MEDIUM
Simplicity and ease of use	HIGH	LOW	LOW	MEDIUM	MEDIUM

### 3.2.4 Discussion

Let us consider the first criterion. The Data Flow model and ISAC are strong performers in terms of supporting the participant features. The former represents the participants indirectly as data *sources* and *sinks*, while the latter uses *real sets* to represent people and material. The other three models do not support these features. Petri Nets and its spin-off, Information Control Nets (INC), are clearly the winners in terms of supporting the process features. They are undoubtedly powerful tools for modeling discrete-event processes, especially those with asynchronous and concurrent events. The Data Flow model covers the basic features of activities and processes, but falls short of the other process features. SADT and ISAC can be seen as attempts to enhance that model by adding more process features. The Data Flow model is a solid performer in terms of supporting the data features. Indeed, it was developed primarily for modeling data flow. ISAC and SADT contribute by introducing new concepts to represent information flows. ISAC introduces real sets and real flows. SADT suggests the roles of various information flows and the relative positioning of graphical elements to depict these roles. Petri Nets is the weakest model in this category. However, its extension, ICN, makes several improvements in the representation of information.

Turning to the other four criteria (i.e., formality, graphical representations, schema readability and simplicity), Petri Nets and ICN are the most formal models. They both have strong mathematical underpinnings and can produce verifiable and executable process specifications. The other models are considered semi-formal. The Data Flow model is by far the simplest and easiest model to use. It can produce readable graphical functional schemata. On the other hand, functional schemata using SADT and ICN are either difficult to read or overwhelmed by available features. All of the five models have graphical representations, but none have built-in features to automatically produce highly readable functional schemata.

---

### **3.3 Conclusion**

---

The above results indicate that all five models offer advantages as well as shortcomings, given the criteria considered. However, the Data Flow model is the best all-around performer. First, it covers all three categories (participants, process and data), though it does not provide all the features required. Second, the model provides graphical representations, which are a key to its popularity. Third, it is simple and thus easy to learn and use. All three reasons explain why it is the most widely used model in functional analysis, as [Batini 92] points out. For the same reasons, *the Data Flow model is by far the most appropriate tool for this study*. However, extensions to the model are clearly needed in order to incorporate the missing features.

In reviewing a number of information system design models and tools, [Lockemann 86] expresses the following frustration: "... no author seems willing to take what is available and build on top of it to achieve a better degree of integration and interchangeability of techniques." *In this case, we decided to use this popular model and extended it to meet the necessary requirements. Doing so is much more efficient than coming up with a totally new model. The extended model described in the next chapter is called the PArtitioned eNginEering DAta flow model (PANDA).*

Key concepts for this chapter: <i>process models, general process models, Data Flow model, PArtitioned eNginEering DAta flow model (or PANDA).</i>
--

## Chapter 4

---

# THE EXTENDED MODEL: PANDA

---

In the previous chapter, we evaluated five candidate models. As a result of this evaluation, we selected the Data Flow model and decided to extend it to meet the necessary requirements. In this chapter, we present the extended model, the *PARTitioned eNginEering DATA flow model* (or *PANDA*). We first give an overview of the model, then focus on its key concepts, graphical representations, rules and operations. In the next chapter, we will show how to use the model, namely by applying its concepts, graphical representations, rules and operations to facility engineering processes and drawing graphical functional schemata. In this chapter, we show examples taken from our case study in the domain of electrical utility transmission towers.

### ORGANIZATION

- 4.1 Overview of PANDA
- 4.2 Key Concepts & Graphical Representations
  - 4.2.1 Partition I: Participants
  - 4.2.2 Partition II: Process
  - 4.2.3 Partition III: Data-Material-Products (DMP)
  - 4.2.4 Summary of Concepts and Graphical Representations
- 4.3 Rules
  - 4.3.1 Syntactic Rules
    - 4.3.1.1 Syntactic Rules for Nodes
    - 4.3.1.2 Syntactic Rules for Links
  - 4.3.2 Semantic Rules
- 4.4 Schema Transformation Operations

## 4.1 Overview of PANDA

---

*PANDA has a multi-leveled partitioned "architecture."* This unique architecture is designed to create a planned, structured layout of the data flow diagrams. With PANDA, the analyst can functionally decompose a process into several subprocesses, which in turn include many activities. Therefore, a process might have many hierarchical description levels. As Figure 4.1.1 shows, the data flow diagram at the detailed level has three major partitions:

- *Participants:* This partition presents the people involved in the process in different capacities. These capacities are clearly annotated on the diagram.
- *Process:* This partition presents the process and its subprocesses and activities. Boundary nodes clearly delimit subprocesses and depict their states as "activated," "suspended," "resumed" or "terminated." Decisions and alternatives are part of the process description. Interferences are special occurrences that disrupt a smooth execution of the process. Activities, decisions, interferences and boundaries can have precedence relationships with each other.
- *Data-Material-Products* (abbreviated as DMP): This partition presents the data items, data repositories and data flow. It also shows the material and products and their physical flow. The data flow and physical flow are graphically represented as networks of data, material and products circulating in and out of activities. This partition shows the relationships (e.g. is-derived-from, is-a-version-of, is-stored-into) among data items and data repositories as they are generated in the process. The partition also presents the people or things that are originators or receivers of data items or data repositories.

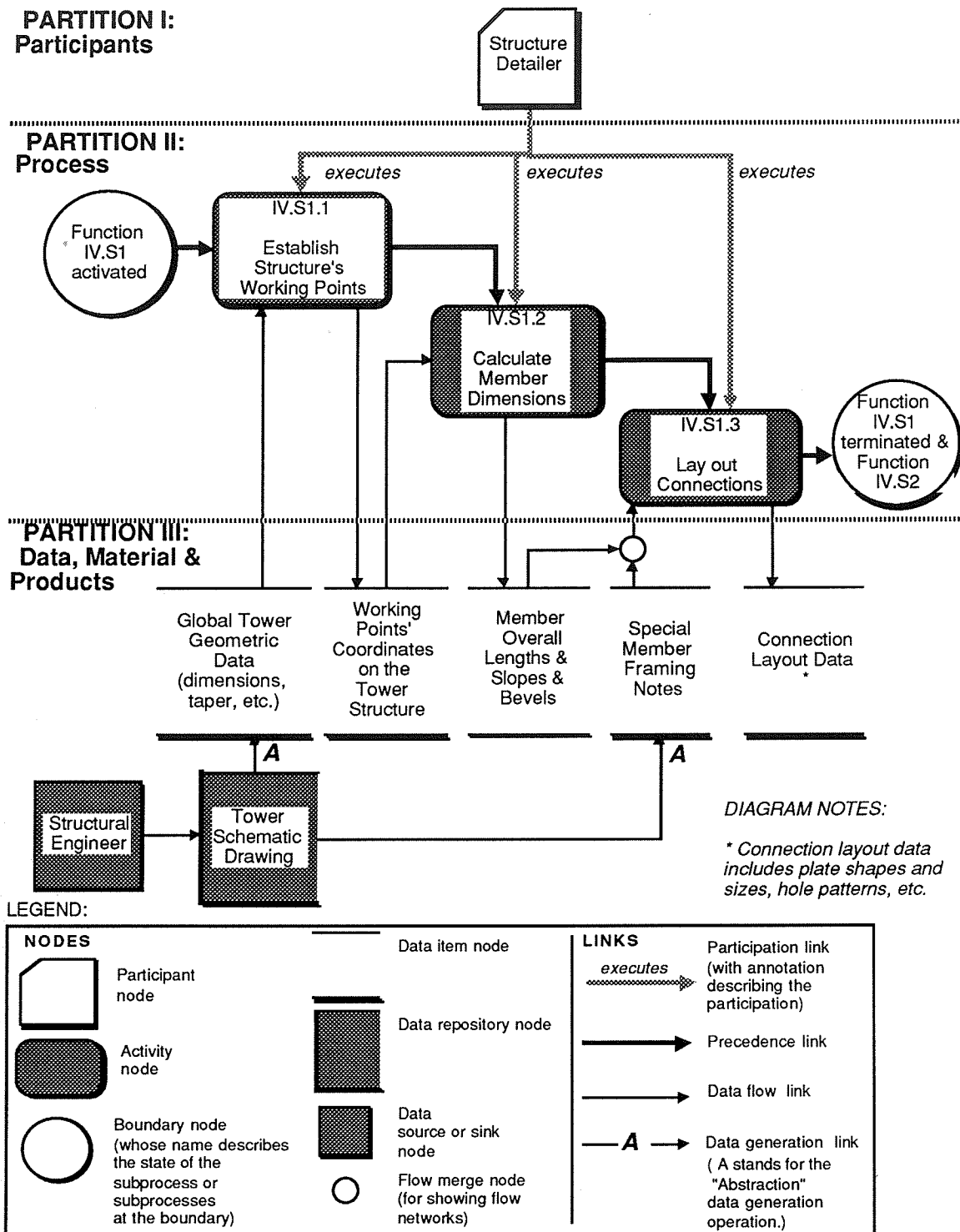
*This unique architecture serves two main purposes:*

- *It helps the analyst to organize his or her thinking about a complicated engineering process.* To be specific, it helps the analyst focus his or her attention to different aspects of the process when he or she is doing the functional analysis. As we explain in the next chapter, the methodology accompanying the model also profits from this architecture, by giving different priorities to the partitions at different stages of analysis.
- *It enhances both the conceptual readability and graphical readability of the process' functional schema.* Each partition encloses only the concepts that are relevant to it. With

these built-in partitions of the schema, users can easily review different aspects of the process: the participants; subprocesses and activities; or data, material and products.

In addition to its concepts, PANDA has associated graphical representations that are consistent with those of the original Data Flow model shown in [Batini 92]. As a result, the analyst can draw graphical functional schemata of a process. Due the three built-in partitions, such a schema is also called a *Partitioned Data Flow diagram* (or *P-diagram*). Figure 4.1.1 illustrates a sample P-diagram. In this example concerning transmission tower engineering, the structure detailer establishes the so-called “working points” using the tower geometry data from the tower schematic drawing that is provided by the structural engineer. These working points are points of reference on the tower structure used to calculate the member dimensions in the next step. As a result, the member lengths, slopes and bevels are determined and then used to do the next activity, laying out the connections. Special notes about member framing from the tower schematic drawing are also used in this activity. The resulting connection layout data includes the connection plates’ shape, size, hole pattern, etc.

PANDA also provides syntactic and semantic rules that govern the way in which the concepts should be used. Several basic schema transformation operations are also provided to enable the analyst to develop a functional schema incrementally. A methodology accompanying the model guides the analyst in applying the concepts to his or her problem domain. Moreover, PANDA provides additional guidelines for using specific concepts of the model and drawing Partitioned Data Flow diagrams.



**FIGURE 4.1.1: A Sample Partitioned Data Flow Diagram Using PANDA.** This diagram illustrates an example of dimensioning members and laying out connections in transmission tower engineering. The diagram is divided into three major partitions, which enhance its readability. The shaded elements and data flow links are part of the original Data Flow model [Batini 92]. Others are part of our extension.



---

## 4.2 Key Concepts and Graphical Representations

---

The concepts of PANDA are arranged according to the three major partitions. In the following three sections, we introduce these concepts and present their graphical representations. Important definitions and ideas are highlighted in italics. In many instances, we use the same example of transmission tower engineering introduced in Figure 4.1.1.

### 4.2.1 Partition I: Participants

This partition includes two concepts: participant and participation. These were not included in the original Data Flow Model [Batini 92] and are part of our extension.

**PARTICIPANT** *A participant represents a class of personnel that takes part in activities of a process. Each participant can be involved in more than one activity in the process.* For instance, the structure detailer, structural engineer and electrical engineer are three different participants in the transmission tower engineering process. In the example shown in Figure 4.1.1 above, the structural detailer is involved in three activities: establishing the tower structure's working points, calculating member dimensions, and laying out connections. This concept supports Feature PAR-1 defined in Chapter 2.

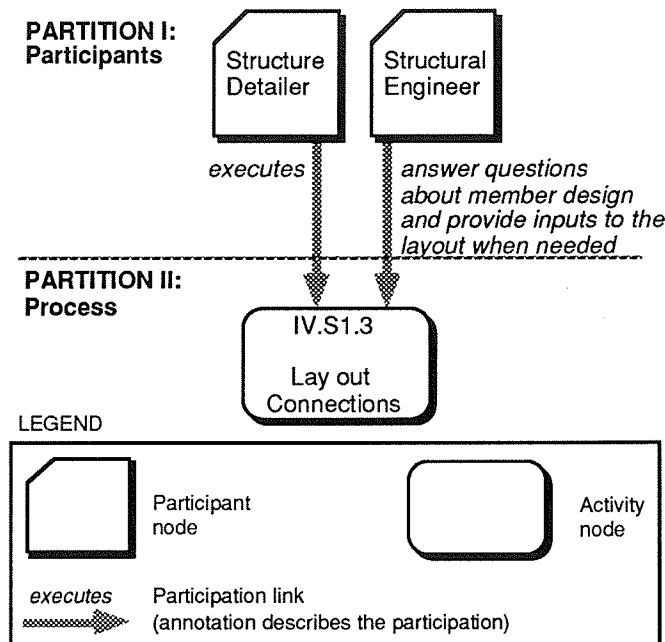
*The graphical representation of this concept is the participant node.* In Figure 4.1.1, there is one participant node labeled "Structural Detailer".

**PARTICIPATION** *This concept represents the capacity in which a participant is involved in an activity of a process.* Referring back to the example in Figure 4.1.1, the structural detailer actually carries out each of the three activities. This concept supports Feature PAR-2 defined in Chapter 2.

There are two possible predefined *roles* for participants: actually carrying out an activity and being directly responsible for its successful completion (*executive role*), or being involved in other indirect capacities and having no direct responsibility or authority (*supporting role*). These roles are mutually exclusive: A participant can assume either an executive role or a supporting role in an activity.

In the example shown in Figure 4.1.1 above, the structural detailer plays the executive role. On the other hand, the structural engineer can assist the detailer in laying out the connections by answering questions about the member design or providing inputs to the layout. The structural engineer then plays the supporting role, as Figure 4.2.1.1 illustrates.

In PANDA, each participation is represented graphically by a participation link from a participant node to an activity node in the next partition. The link is annotated with a clear description of the participation (i.e., capacity in which the participant is involved).

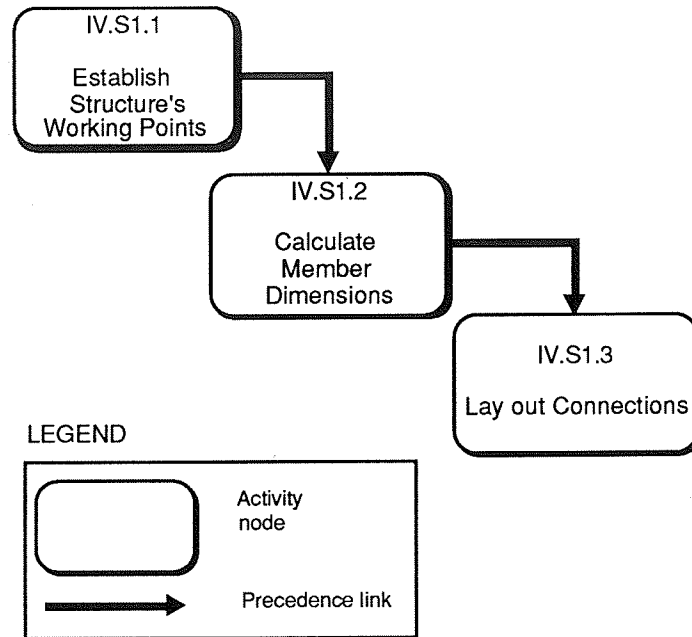


**FIGURE 4.2.1.1: Executive vs. Supporting Roles For Participants.** In this case, the detailer plays the executive role and the structural engineer plays the supporting role.

#### 4.2.2 Partition II: Process

This partition includes the following concepts: activity, precedence relationships, decision and alternative, interference, subprocess, boundary, and process non-linearity. Of these, only activity was included in the original Data Flow model [Batini 92]. The others are parts of our extension.

**ACTIVITY** As mentioned earlier, a process can be decomposed into smaller functional units. An activity is defined as “an organizational unit of a process for performing a specific task” [Webster 86]. It is considered to be the smallest functional unit of the process. Indeed, a process includes many activities. This concept supports Feature PRO-1. Its graphical representation is the activity node. Figure 4.2.2.1 illustrate some sample activities.



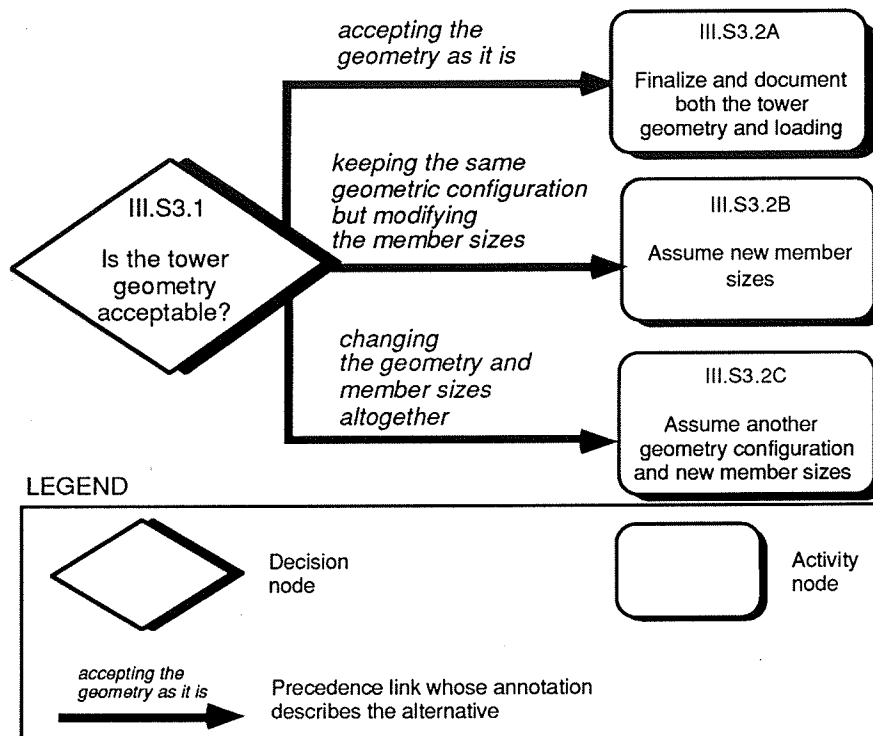
**FIGURE 4.2.2.1: Activities Related by Precedence Relationships.** An activity is an organizational unit of a process for performing a specific task. Activities have precedence relationships with each other. There consecutive activities that involve dimensioning members and laying out connections in transmission tower engineering are shown here. This figure is a portion of Figure 4.1.1.

**PRECEDENCE RELATIONSHIPS** Activities have precedence relationships. These relationships place explicit temporal constraints on the execution of the activities. *An activity, A, precedes (or has precedence over) another activity, B, if B cannot be started until A finishes. A is the predecessor activity, B is the successor activity.* In Figure 4.2.2.1, establishing the working points of the tower structure precedes calculating member dimensions, which precedes laying out connections. The concept of precedence relationships supports Feature PRO-4.

*The graphical representation here is the directed precedence link. Each connects a pair of activity nodes. Its arrow goes from the predecessor to the successor.*

**DECISION AND ALTERNATIVE** Making decisions is essential to solving facility engineering problems. Decisions can be seen as a special class of activities. *Indeed, a decision is a special activity that involves answering a preponderant question by considering one or more alternatives and choosing among them.* In Figure 4.2.2.2, the structural engineer decides whether the current configuration of a tower geometry is acceptable. There are three alternatives: accepting the geometry as it is, keeping the same configuration but modifying the member sizes, or changing the geometry and member sizes altogether. Each alternative produces a unique solution. If the engineer selects the second alternative for instance, he or she proceeds to assume new member sizes and redo the structural analysis. This concept supports Feature PRO-6.

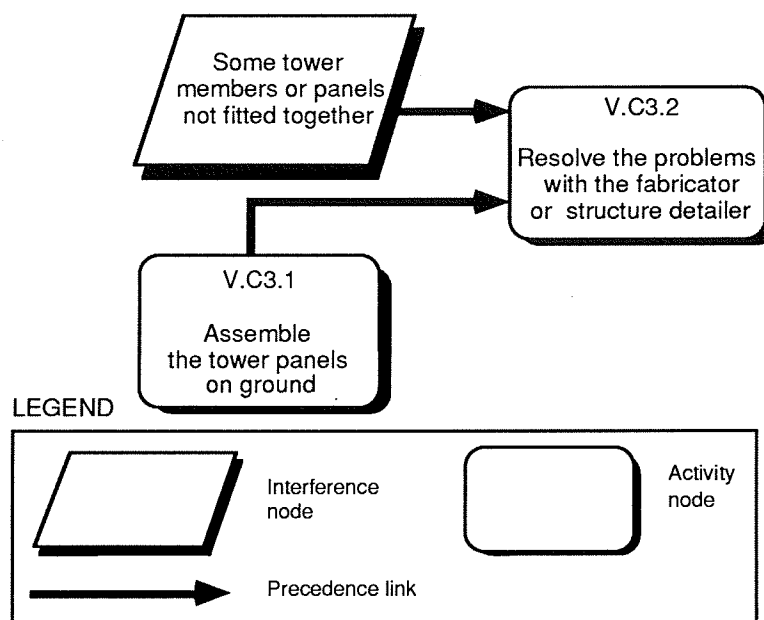
As illustrated in Figure 4.2.2.2, *a decision is graphically represented as a decision node. The alternatives are represented as annotations to the precedence links coming out of the decision nodes.* Each link points to an activity node representing the action that needs to be carried out when choosing the alternative. Therefore, like other activity nodes, a decision node can be connected to an activity node by precedence links.



**FIGURE 4.2.2.2: A Sample Decision in Evaluating a Tower Geometry .** A decision involves answering a preponderant question by considering many alternatives and choosing among them. Each alternative leads to a different activity. Decision nodes can be connected to activity nodes by precedence links.

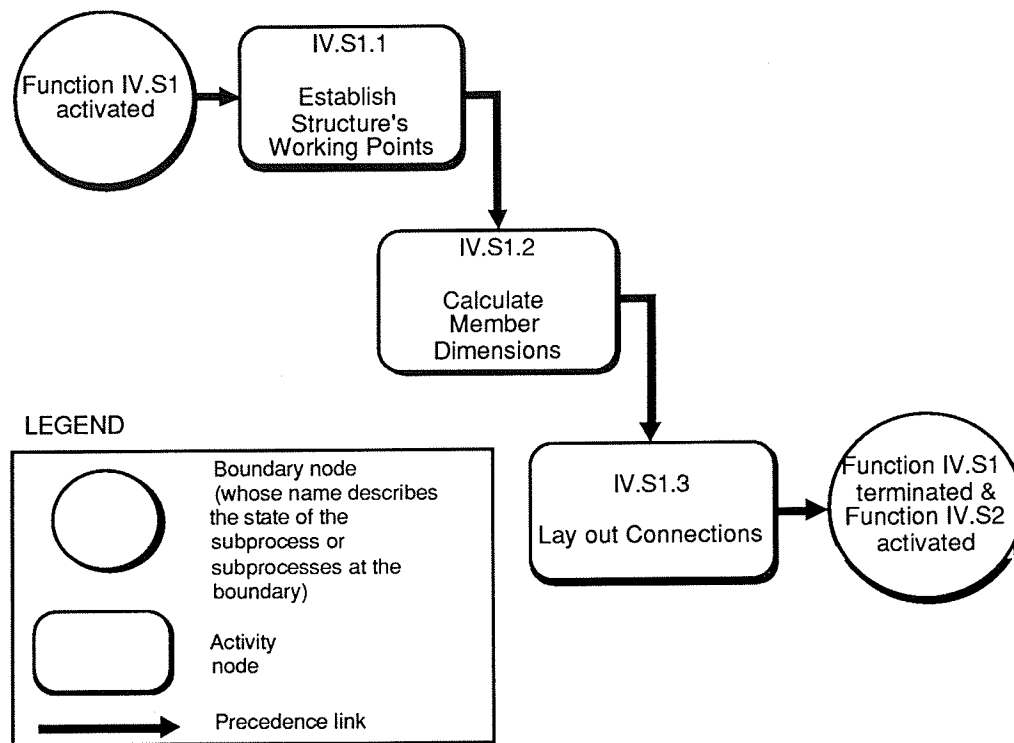
**INTERFERENCE** An interference is a special occurrence that interrupts the successful execution of the process. This concept does not come from the list of required features but is included in the model since real-life engineering processes normally experience interferences. Examples in transmission tower engineering include missing loading conditions on the tower design request, errors in detailed drawing during fabrication, members that do not fit together during the structure's assembly and erection, etc. An interference leads to some activities required to remedy the situation. It can be short-term, long-term or indefinite. It can also put the current process into a "suspended" or "terminated" state. It can create a loop that takes the process back to some earlier activities.

As illustrated in Figure 4.2.2.3, an interference is graphically represented as an interference node. Interference nodes can be connected to activity nodes and even decision nodes by precedence links.



**FIGURE 4.2.2.3: A Sample Interference During Tower Construction.** When the construction crew assembles the tower panels and discovers that some tower panels do not fit together, the crew informs the fabricator or structure detailer to resolve the problem. This may be due to fabrication or detailing errors. The interference causes a delay in tower construction.

**SUBPROCESS** Activities, decisions and interferences that are related can form a subprocess. Subprocesses are the intermediate functional units of a process. Indeed, hierarchical top-down functional decomposition provides a powerful mechanism that can be used to analyze complex real-life processes. It allows the analyst to define different levels of description about the process with increasing amounts of detail. For example, the transmission tower engineering process is first decomposed into six major stages of development. Each stage is then decomposed into several functional areas, which consists of many activities. The concept of subprocess can effectively support this functional decomposition. The analyst can use it to represent all the intermediate functional units (in this case, the development stages and functional areas) to which the process is decomposed. Figure 4.2.2.4 extends Figure 4.2.2.1 and shows that the three related activities form a subprocess that occurs in the construction planning stage of the tower engineering process.



**FIGURE 4.2.2.4: A Sample Subprocess Involving Dimensioning and Laying out Tower Members and Connections.** This occurs in the tower construction planning stage. The subprocess is labeled “Function IV.S1.” It has three activities and two boundaries. Its activity and boundary nodes are connected by precedence links. The boundary nodes’ name describes the state of the subprocess or subprocesses at the boundary. (Working points are points of reference on the tower structure that are used to calculate the dimensions of the tower members. This figure is also taken directly from Figure 4.1.1.)

Formally, a subprocess is defined as a fixed set of activities, decisions, interferences and delimiting boundaries, which have precedence relationships to each other. (Boundaries are explained next.) This definition can be recursive. A subprocess can include individual activities, decisions and interferences, or other child subprocesses. Thus, the definition of a subprocess covers the intermediate functional units of a process as well as the process itself. A process is the encompassing subprocess that has no parent. As a result, the concept of subprocess as defined here supports both features PRO-1 and PRO-2.

As shown in Figure 4.2.2.4, *the graphical representation of a subprocess is of the subprocess' components: activities, decisions, interferences and boundaries.* Alternatively, the analyst can draw a box in dashed lines that encloses the subprocess and label it.

**BOUNDARY** Each subprocess has a beginning and an end. Depending on how the subprocess is designed, it can also come into contact with other activities or subprocesses. For example, the subprocess shown in Figure 4.2.2.4 comes into contact with the subprocess that immediately follows it. *A boundary marks the beginning or end of a subprocess, or the borderline between a subprocess and another activity or subprocess. Each subprocess has at least two boundaries.*

In actuality, boundaries serve three main purposes: (1) they delimit subprocesses; (2) they connect subprocesses; and (3) they allow subprocesses to be mapped back to their parent process or subprocess. The analyst can examine and represent subprocesses separately. He or she can produce a graphical functional schema for each subprocess, but must delimit the subprocess with clear boundaries. These must have explicit names. The subprocess is also connected by these boundaries to those activities or subprocesses with which it comes into contact. Subprocesses on separate graphical functional schemata can be reconnected to form the parent process by matching the boundaries with the same name. Consequently, the concept of a boundary supports both features PRO-2 and PRO-3.

As illustrated in Figure 4.2.2.4, *boundaries are graphically represented as boundary nodes. The name of the boundary node describes the state of the subprocess or subprocesses at the boundary. The possible states are: "activated" (or "started"), "suspended," "resumed" and "terminated" (or "ended").* For example, the subprocess in Figure 4.2.2.4 is delimited by two boundary nodes. One of these is labeled "Function IV.S1 terminated and Function IV.S2 activated."

The four node types (i.e., activity, decision, interference, boundary nodes) defined above can be seen as special types of the so-called "process nodes" in the original Data Flow model [Batini 92]. Therefore, to reiterate, any two of these node types can be connected to each other by precedence links.

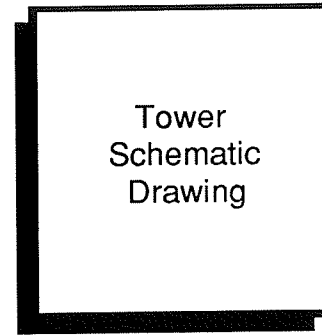
**PROCESS NON-LINEARITY** First, this concept is built on the concept of precedence relationships discussed earlier. In fact, precedence links are not only used to show linear processes. *Each node in this partition can have more than one outgoing precedence link with other nodes.* This allows the analyst to represent parallel activities or subprocesses of non-linear processes. *Moreover, by naming its boundary nodes, a subprocess can be designated as “activated,” “suspended,” “resumed” or “terminated.”* Subprocesses that are suspended during the execution of other subprocesses can be represented. As an example, in the tower construction execution stage, the fabricator procures the needed raw material and suspends fabricating parts. The material supplier then starts gathering and delivering the ordered raw material. Parts fabrication can resume after the material is delivered. As a result, multiple concurrent subprocesses and activities conducted by different participants can be delineated. *Reciprocally, any node can have more than one incoming precedence link from other nodes.* This allows the analyst to represent iterative subprocesses such as the common “Propose-Verify-and-Redesign” loop [Chandrasekaran 88] in design synthesis. For example, the structural engineer begins by proposing a preliminary tower geometry to initiate the tower loads computation. Once the loads are computed, he or she verifies and possibly redesigns the geometry. The loop continues until the engineer obtains a satisfactory solution. The concept of process non-linearity supports Feature PRO-5.



### 4.2.3 Partition III: Data-Material-Products (DMP)

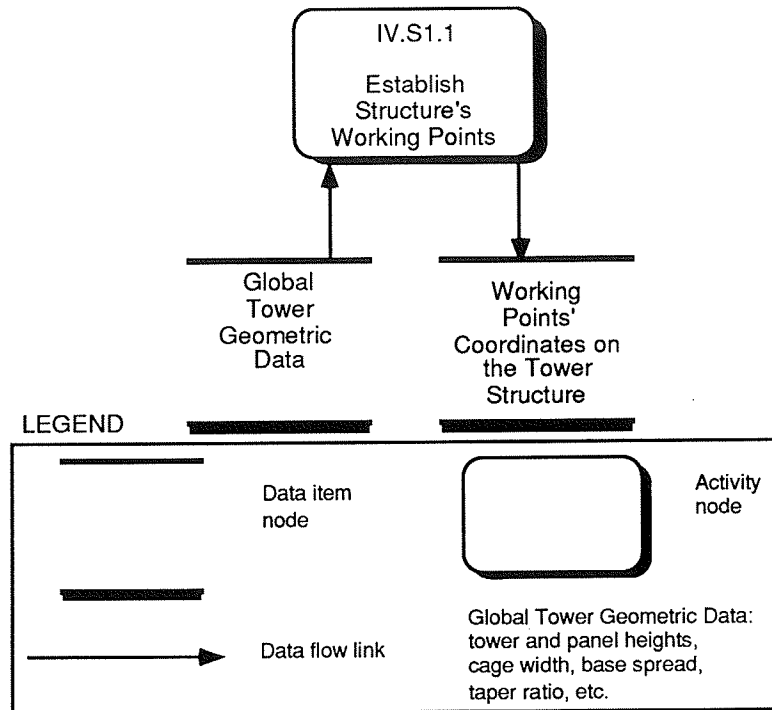
This partition includes the following concepts: data repository, data item, data flow, data source or sink, data generation, material or product, physical flow, mixed flow, and flow network. Only data repository, data flow, and data source or sink were included in the original Data Flow model [Batini 92]. The others are part of our extension.

**DATA REPOSITORY** In an enterprise, data plays an essential role: It provides valuable information for conducting daily operations, solving critical problems and making important decisions. Therefore, data is commonly stored and retrieved when needed. *A data repository is a permanent storage of data in paper or electronic format. Examples include files, permanent records, look-up tables, paper or electronic forms, electronic databases, vendors' standard parts catalogs, standard design codes, companies' design manuals, engineering drawings, and program input and output printouts.* This concept supports Feature DMP-1. *Its graphical representation is the data repository node, as shown in Figure 4.2.3.1.*



**FIGURE 4.2.3.1: A Sample Data Repository Node.**

**DATA ITEM** *A data item represents a single piece of data or a collection of data that is created by activities of the process and used as input by other activities. A data item may or may not later be stored into a data repository.* In Figure 4.2.3.2 on the next page, the activity of establishing the tower structure's working points uses the global tower geometric data as input. As a result, it yields the working points' coordinates. This concept supports Feature DMP-2. It was not included in the original Data Flow model. In that model, data items can be shown only as annotations (of data flow links) in the data flow diagram. On the contrary, *data items are represented explicitly in PANDA and have their own graphical representation, the data item node, as shown in Figure 4.2.3.3. The justification is that explicit representation of data items makes it easier to show in detail where data comes from, how it is combined for use in the activities, how it gets generated from other data, and where it finally goes. This information can greatly benefit the conceptual schema design.*

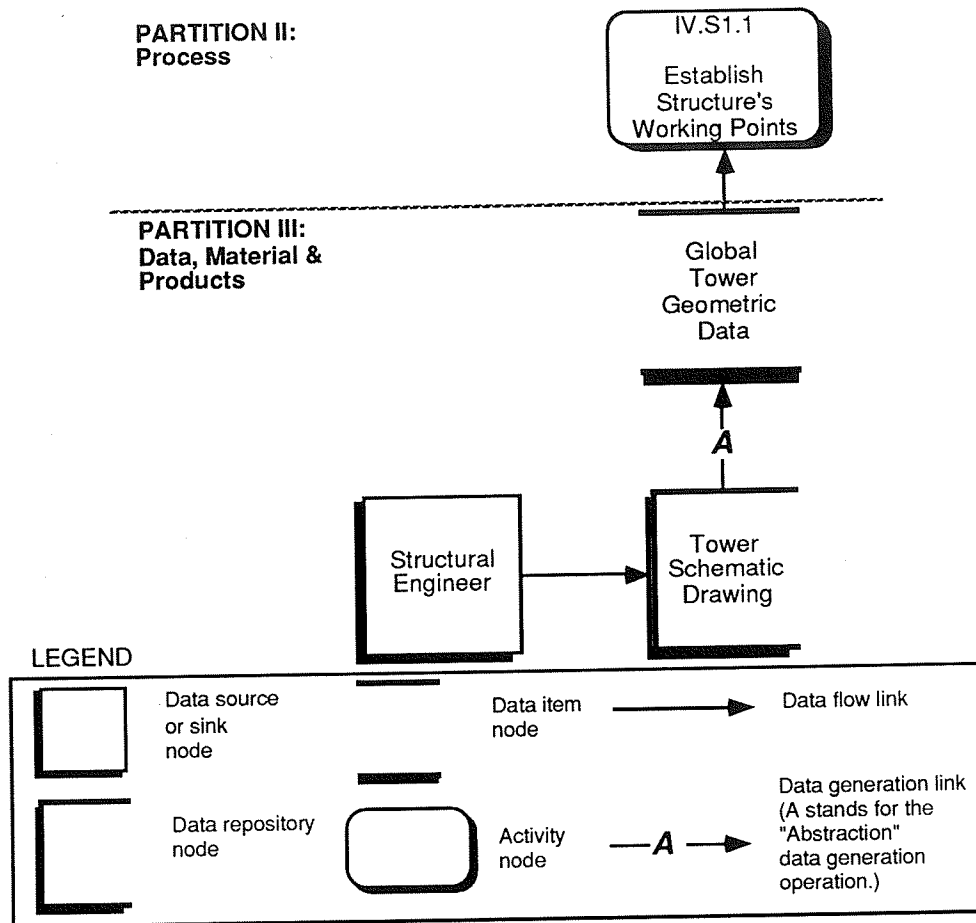


**FIGURE 4.2.3.2: Sample Data Items and Data Flow.** Data items are individual data or collections of data that are created by activities of the process and used as input by others. Data flow denotes that a data item flows into or out of an activity. (This figure is taken from Figure 4.1.1.)

**DATA FLOW** As mentioned above, activities can use existing data as input and can create new data. *The concept of data flow indicates that a data item or data repository flows into or out of an activity.* This concept supports Feature DMP-3.

As shown in Figure 4.2.3.1, data flow is graphically represented by the directed data flow link between a data item node or data repository node and an activity node. The direction of the arrow indicates whether the data item or data repository serves as input to or output from the activity.

**DATA SOURCE OR SINK** Data repositories or data items generally originate from sources known as “data sources,” and go to destinations called “data sinks.” The example in Figure 4.2.3.3 combines and extends those shown in Figures 4.2.3.1 and 4.2.3.2. In this example, the structural engineer provides the tower schematic drawing, which contains the global tower geometric data used to establish the structure’s working points. The structural engineer is a data source, the drawing is a data repository, the global tower geometric data is a data item, and establishing working points is an activity. As another example, the structure detailer generates the erection bill of materials and bundling list and gives those results to the construction manager who



**FIGURE 4.2.3.3: An Illustration of a Data Source, Data Repository, Data Item, and Data Generation Operation.** The structural engineer provides the tower schematic drawing, which contains the global tower geometric data used to establish the structure's working points. Here, the structural engineer is a data source, and the tower schematic drawing is a data repository. The data flow link between the two corresponding nodes simply indicates that the drawing comes from the engineer. (It implies no activity or subprocess). The global tower geometric data is a data item, which is an abstraction of the data contained in the drawing. (This figure is taken from Figure 4.1.1.)

initiates the tower construction. The erection bill of materials and bundling list is a data repository, and the construction manager is a data sink. A data source or sink represents the person or thing that are the prime originator or receiver of data repositories or data items. This concept supports Feature DMP-4. Certain versions of the Data Flow model, such as the one presented in [Batini 92], call data sources and sinks interfaces and treat them as if they were external to the system under consideration.

*The graphical representation here is the data source or sink node. Data source or sink nodes can be connected to data source nodes or data item nodes by data flow links.*

**DATA GENERATION** During the life-cycle phases of the facility, new data is generated from existing data from several sources. *The concept of data generation captures the special relationships that exist between data items or repositories as the process unfolds.* This concept enables the analyst to better show how data is actually generated and evolved in the process. This concept supports Feature DMP-5.

*As shown in Figure 4.2.3.3, data generation is graphically represented as directed data generation links. These are special links that exist only among data items and data repositories. The arrow direction goes from the source data to the resulting data. The link is also annotated with an abbreviation showing the type of data generation relationship involved. The six main types of data generation relationships are: is-abstracted-to, is-derivation-of, is-previous-version-of, is-stored-into, is-combined-into and is-presented-in.*

*is-abstracted-to* Relationship Type (Type A, abbreviated as A) — A relationship of this type indicates that a data item or repository of origin is used to draw a certain data item or repository of interest and suppress others. This usually involves reducing the volume of the base data. As Figure 4.2.3.3 shows, the tower schematic drawing is used to abstract the global tower geometric data item needed to establish the working points.

*is-derivation-of* Relationship Type (Type D, abbreviated as D) — A relationship of this type indicates that an existing data item or repository is used to derive or compute a new data item or repository. For example, the data of the tower's loading conditions is used to derive the load cases' data, which in turn is used to derive the individual loads' data. Indeed, generating the tower loads data involves a series of successive computations.

*is-previous-version-of* Relationship Type (Type V, abbreviated as V) — A relationship of this type indicates that an existing data item or repository is modified to create a new version of it. This involves changing the value of the data item or the content of the data repository. For example, during tower construction, the detection of missing or erroneous data in the tower detail drawing requires the revision of the drawing and creation of a corrected version.

*is-stored-into* Relationship Type (Type S, abbreviated as S) — A relationship of this type indicates that an existing data item is placed into permanent storage (i.e., into a data repository). For example, data items of the detailed fabrication features of the tower parts are transcribed into the tower detailed drawing (data repository) for long-term storage.

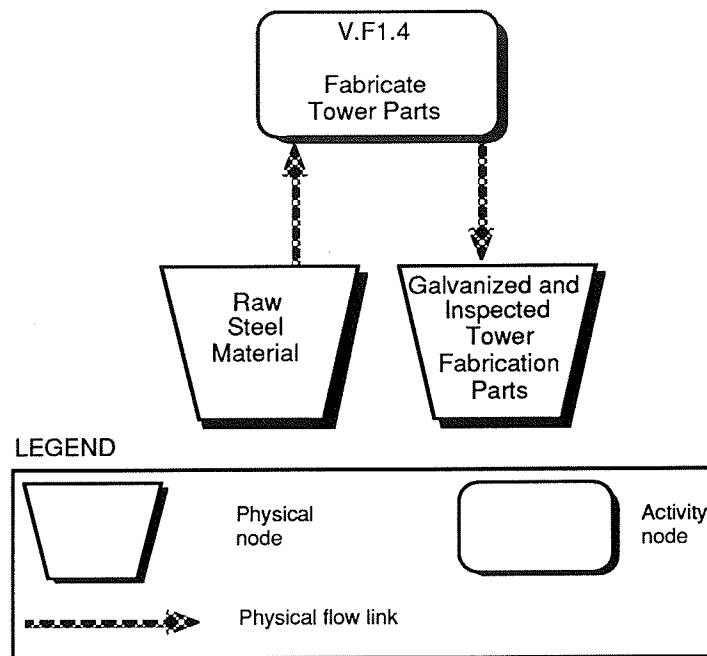
*is-combined-into* Relationship Type (Type C, abbreviated as C) — A relationship of this type indicates that an existing data item or repository is put together with other available data items or repositories to form a new data item or repository. For example, the activity of generating a tower schematic drawing requires combining data items that describe the cross-section and general dimensions of the tower and the structural systems and members.

*is-presented-in* Relationship Type (Type P, abbreviated as P) — A relationship of this type indicates that an existing data item or repository is presented under a different format in another data item or repository. For instance, generating a tower schematic drawing involves presenting data of the tower geometry, loading conditions, member sizes, and typical member framing in a special standardized format.

In real-life engineering processes, generating new data involves performing several of the operations described above at the same time. For example, generating a tower detailed drawing involves deriving, combining, presenting and storing several data items. For more examples, the reader can refer to the graphical functional schemata of the tower engineering process in Chapter 6.

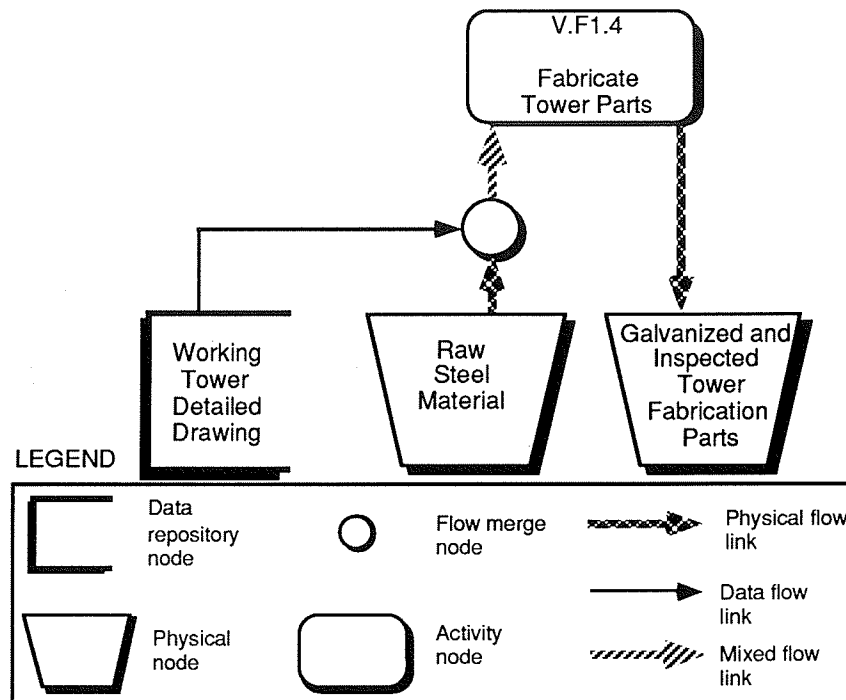
**MATERIAL OR PRODUCT** Typically, a real-life facility engineering process utilizes many physical resources and yields tangible products. These are as important to the process as the data itself. In the example shown in Figure 4.2.3.4, raw steel material is used to fabricate tower parts that will be assembled to construct the tower. *The concept of material or product represents the resources, as well as intermediate or final results, of the process.* This concept supports Feature DMP-6. *Its graphical representation is the physical node.*

**PHYSICAL FLOW** Physical flow maps a material or product to an activity, as data flow maps a data item or data repository to an activity. The term “physical flow” was introduced in the ISAC model [Lundeberg 82]. *The concept of physical flow here indicates that a material or product flows into or out of an activity.* This concept supports Feature DMP-7. *Physical flow is graphically represented by the directed physical flow link between a physical node and an activity node, as shown in Figure 4.2.3.4.* The direction of the arrow indicates whether the material or product is a resource or result of the activity.



**FIGURE 4.2.3.4: Sample Material, Product and Physical Flow.** Raw steel material is used to fabricate tower parts that will be assembled to build the tower. Physical flow shows whether the material or product is a resource or a result of the activity.

**MIXED FLOW** In the facility construction execution stage, an activity can use not only data, but also material and products. Extending the example in Figure 4.2.3.4, Figure 4.2.3.5 shows that fabricating the tower parts requires raw steel material as well as the tower detailed drawing. *The concept of mixed flow indicates that a data item or repository and a material or product flow together into or out of an activity. Although this concept was not included in the list of required features, it is essential for showing how data, material and products can be used in a single activity of a typical engineering process. The graphical representation here is the directed mixed flow link.* The term “mixed flow” was introduced in the ISAC model [Lundeberg 82].



**FIGURE 4.2.3.5: An Illustration of Mixed Flow and a Flow Network.** A detailed drawing and raw steel material are used to fabricate tower parts. Mixed flow indicates that data, material and products together flow into an activity. Flow networks can represent complex data flow, physical flow, or in this case, mixed flow.

**FLOW NETWORK** *The concept of a flow network represents an aggregated way of showing complex data flow among activities: Several different data items and data repositories can be used in a single activity in order to generate more data. This concept applies to physical flow as well as to mixed flow. This concept supports Feature DMP-3.*


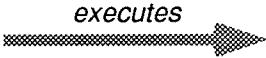
*A flow network can be shown graphically using flow merge nodes, as shown in Figure 4.2.3.5. Each node represents a point where the data flow, physical flow, or mixed flow among activities merge.*

Building flow networks of data with flow merge nodes can be highly beneficial. First, the analyst can view such networks as instruments to think about how data is generated and used by subprocesses and activities. Second, the analyst can significantly improve the graphical as well as the conceptual readability of the functional schema, especially when complicated engineering data flows are involved (see the drawings shown in Chapter 6). Moreover, the analyst can build flow networks of material and products (especially in the construction execution phase) or also mixed flow networks of data, material and products.

#### 4.2.4 Summary of Concepts and Graphical Representations

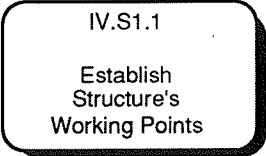

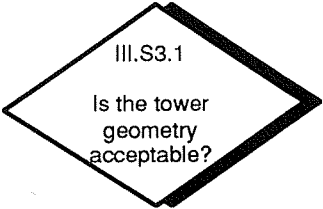
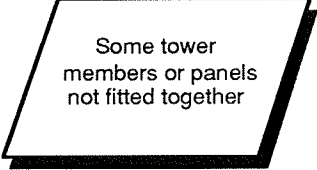
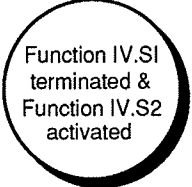
Tables 4.2.4.1a to c summarize the key concepts and graphical representations of PANDA. The concepts are shown in the table in the same order in which they were introduced in the previous section.

**TABLE 4.2.4.1a: Concepts and Graphical Representations in Partition I.** Both concepts shown below are part of our extension to the original Data Flow model as shown in [Batini 92].

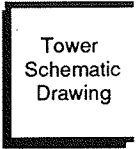








<i>CONCEPTS</i>	<i>GRAPHICAL REPRESENTATIONS</i>
<i>Participant</i>	Participant Node 
<i>Participation</i>	Participation Link 



**TABLE 4.2.4.1b: Concepts and Graphical Representations in Partition II.** Only “Activity” comes from the original Data Flow model as shown in [Batini 92]. The other concepts are part of our extension.

<b>CONCEPTS</b>	<b>GRAPHICAL REPRESENTATIONS</b>
<i>Activity</i>	Activity Node 
<i>Precedence Relationships</i>	Directed Precedence Link 
<i>Decision</i>	Decision Node 
<i>Interference</i>	Interference Node 
<i>Subprocess</i>	Its graphical representation is of the activities, decisions, interferences and boundaries that make up the subprocess.
<i>Boundary</i>	Boundary Node 
<i>Process Non-linearity</i>	<ul style="list-style-type: none"> <li>• Any of the nodes in Partition II can have more than one outgoing or incoming links.</li> <li>• By naming its boundary nodes, a subprocess can be designated as “activated” (or “started”), “suspended,” “resumed,” and “terminated” (or “ended”).</li> </ul>

**TABLE 4.2.4.1c: Concepts and Graphical Representations in Partition III.** Only “Data Repository,” “Data Flow,” “Data Source or Sink” come from the original Data Flow Model [Batini 92]. The others are part of our extension.

<i>CONCEPTS</i>	<i>GRAPHICAL REPRESENTATIONS</i>	
<i>Data Repository</i>	Data Repository Node	
<i>Data Item</i>	Data Item Node	
<i>Data Flow</i>	Directed Data Flow Link	
<i>Data Source or Sink</i>	Data Source or Sink Node	
<i>Data Generation</i>	Directed, Annotated Data Generation Link	 <p>(Annotation is the abbreviation of the type of data generation operation involved.)</p>
<i>Material or Product</i>	Physical Node	
<i>Physical Flow</i>	Directed Physical Flow Link	
<i>Mixed Flow</i>	Directed Mixed Flow Link	
<i>Flow Network</i>	Flow Merge Node	

---

## **4.3 Rules**

---

In addition to its concepts and graphical representations, PANDA has a number of syntactic and semantic rules. The concepts and graphical representations provide the basic elements used to represent processes and draw their graphical functional schema. However, there are specific constraints on the way in which these elements should be used. The syntactic rules ensure that the nodes are connected properly, that the links are used in the right places, and consequently, that the resulting schema conforms to our definition of the model. In other words, these rules form the underlying grammar of the model. Moreover, a syntactically valid schema may not represent a process that would make sense in real life. Therefore, the semantic rules further constrain the use of the elements in order to produce meaningful schemata. In the following sections, these rules are presented in the same order in which the concepts were introduced. We first state each rule and then explain why it makes sense.

### **4.3.1 Syntactic Rules**

#### **3.3.1.1 Syntactic Rules For Nodes**

##### ***All Nodes in Any of the Three Partitions***

- *No node can be connected to itself.* Each node type represents a concept of sufficient granularity such that there would not be self-connected nodes. For instance, an activity node connected to itself would not make any sense.

##### ***Participant Nodes (PARTITION I)***

- *A participant node must be connected to at least one activity or decision or interference node in Partition II.* Indeed, a participant must be involved in some activity, decision or interference that occurs in the process. Otherwise, the participant must not be represented in the schema.
- *A participant node cannot be connected to any boundary nodes in Partition II or to any other nodes in Partition I or III.* In this model, we define a participant as a category of personnel that takes part in the execution of the process. Therefore, connecting a participant node in any other way would not make sense. For instance, a participant node cannot be connected to another participant node. By definition, a participant has no direct relationships with the data, material and products represented in Partition III.

**Activity Nodes and Decision Nodes (PARTITION II)** The following syntactic rules apply to both activity and decision nodes.

- *An activity or decision node must be connected to at least another node of any type in Partition II. Otherwise, the activity or decision is not part of any subprocess or related to any other event in the process. We see a process as an interconnected network of related events rather than a collection of disconnected random events.*
- *An activity or decision node can be connected to one or more nodes in Partitions I or III, as long as they are not data source or sink nodes. An activity or decision can involve participants represented in Partition I, and any elements in Partition III (e.g., data item, data repository) other than data sources and sinks. A data source or sink represents a prime originator or receiver of data and has nothing to do with an activity or decision.*

**Interference Nodes (PARTITION II)**

- *An interference node must be connected to at least: (a) one activity or decision node or (b) another interference node in Partition II. In the second case, at least one node in a series of connected interference nodes must be connected to an activity or a decision node. Indeed, an interference must lead to an activity or a decision that is required to remedy the situation. Or, it must lead to another interference. In that case, a chain of interferences must eventually lead to some activity or decision that would solve the problem. Otherwise, the interference node must not be included in the schema.*
- *An interference node cannot be connected to any nodes in Partition III. By definition, an interference does not use or produce any data, material or product represented in Partition III.*
- *An interference node can be connected to all nodes in Partitions I or II. By definition, an interference can indirectly involve a participant represented in Partition I. As mentioned above, it can lead to an activity or decision, or to another interference. It can even lead to a subprocess whose beginning is marked by a boundary.*

### **Boundary Nodes (PARTITION II)**

- *A boundary node must be connected to at least: (a) another one activity or decision or (b) another boundary node in Partition II.* In fact, a boundary node must mark the beginning of a subprocess that starts with an activity, decision or child subprocess (i.e., boundary node of the child). Or, it must mark the termination of a subprocess that ends with an activity, decision or a child subprocess. Otherwise, it would not make sense to include the boundary node in the schema.
- *A boundary node cannot be connected to any nodes in Partitions I or III. However, it can be connected to interference nodes in Partition II.* According to its definition, a boundary has nothing to do with participants represented in Partition I, or to data, material or products represented in Partition III. As mentioned before, an interference can lead to a subprocess marked by a boundary.

### **Data Repository Nodes and Data Item Nodes (PARTITION III)**

- *A data repository or data item node must be connected to at least: (a) one activity or decision node in Partition II or (b) one flow merge node in Partition III.* Indeed, a data repository or data item must be used in some activity or decision in the process. If it is not, it must be merged with other data repositories and data items, the result of which must then be used in some activity or decision. Otherwise, there is no reason to represent the data repository or data item in the schema.
- *A data repository or data item node cannot be connected to any nodes in Partition I, to any interference or boundary nodes in Partition II, or to any physical nodes in Partition III.* By definition, participants represented in Partition I have no direct relationship with data repositories and data items. Interferences and boundaries do not use or create data repositories and data items. Material and products represented in Partition III have no direct relationships with data repositories and data items, although data flow can merge with physical flow to create mixed flow.
- *A data repository or data item node can be connected to one or more data repository and data item nodes, or to data source or sink nodes in Partition III.* Data repositories and data items can be generated from other data repositories and data items. Also, they can flow from some data sources and to some data sinks.

### **Data Source or Sink Nodes (PARTITION III)**

- *A data source or sink node must be connected to at least one data repository or data item node in Partition III. A data source or sink must provide or receive a data repository or data item. Otherwise, there is no reason to represent it in the schema.*
- *A data source or sink node cannot be connected to any other node type in Partition III or to any nodes in Partitions I or II. By definition, a data source or sink represents the originator or receiver of data items or data repositories. Therefore, it has nothing to do with any other concept of the model. It is not even related to another data source or sink.*

### **Physical Nodes (PARTITION III)**

- *A physical node must be connected by physical flow links to at least: (a) one activity or decision node in Partition II or (b) one flow merge node in Partition III. An explanation similar to that for the first rule for data repository and data item nodes given above applies here.*
- *A physical node cannot be connected to any nodes in Partition I, or to any interference or boundary nodes in Partition II, or to any nodes in Partition III other than flow merge nodes. By definition, material and products have no direct relationships with participants represented in Partition I. They are not used or produced by interferences and boundaries represented in Partition II. They have no direct relationships with other data items or repositories, or other material and products. They can only be merged into flow networks that include other data, material and products.*

### **Flow Merge Nodes (PARTITION III)**

- *A flow merge node must be connected to at least: (a) one activity or decision node in Partition II or (b) one flow merge node in Partition III. In the second case, at least one node in a series of connected flow merge nodes must be connected to an activity or decision node in Partition II. Indeed, a flow network must eventually be used in some activity or decision of the process or be merged with another network. In the second case, the result must eventually be used in some activity or decision. Otherwise, the network must not be represented in the schema. Instead, a single data flow or physical flow should be used.*

- *A flow merge node cannot be connected to any nodes in Partition I, to any interference or boundary nodes in Partition II, or to any data source or sink nodes in Partition III. According to its definition, a flow network has nothing to do with participants represented in Partition I, interferences and boundaries represented in Partition II, or data sources and sinks represented in Partition III. However, it may consist of flow of several data repositories, data items, and material and products. It may also combine with another network.*
- *A flow merge node can be connected to data repository, data item, and physical and flow merge nodes in Partition III. A flow network may consist of flow of several data repositories, data items, and material and products. It may also combine with another network.*

### **3.3.1.2 Syntactic Rules For Links**

- *Each link of any type must connect exactly two distinct nodes.*
- *A precedence link can connect nodes of any type in Partition II. Activity, decision, interference and boundary nodes in Partition II are special types of the process node introduced in the original model. Nodes of all four types can be temporally ordered using precedence relationships.*
- *A data flow link can connect: a data source node or sink to a data repository or data item node in Partition III; a data repository or data item node to a flow merge node in Partition III; two flow merge nodes in Partition III; a data repository or data item node in Partition III to an activity or decision node in Partition II; a flow merge node in Partition III to an activity or decision node in Partition II. These possibilities for using the data flow link come directly from the way in which the concept of data flow is defined in the model. Any other possibility would not conform to this definition.*
- *A data generation link can connect: two data repository nodes in Partition III; two data item nodes in Partition III; a data repository node and a data item node in Partition III. By definition, a data generation link indicates that a data item or data repository is generated by abstraction, derivation, versioning, etc. from another data item or data repository. Therefore, such a link would make sense only in the possibilities that are listed here.*

- *A physical flow link can connect: a physical node and a flow merge node in Partition III; two flow merge nodes in Partition III; a physical node in Partition III and an activity or decision node in Partition II; a flow merge node in Partition III and an activity or decision node in Partition II. All of these possibilities come directly from the definition of physical flow. Any other would violate this definition.*
- *A mixed flow link can connect: two flow merge nodes in Partition III; a flow merge node in Partition III and an activity or decision node in Partition II. Each link must connect only two nodes. Again, these possibilities come from the definition of mixed flow. Any other would not invalidate this definition.*

We summarize the material on syntactic rules for nodes and links in the matrix shown in Table 4.3.1.1 on the next page. In addition, this matrix shows all the permissible ways in which a directed link is used to connect a source node to a destination node.



**TABLE 4.3.1.1:** *Matrix of Permissible Node Linkages in PANDA Using the Appropriate Link Types.* This matrix summarizes all the permissible ways to link nodes in a Partitioned Data Flow diagram and the appropriate link types to use. The column headings across the table designate the types of nodes from which a directed link can originate. The row headings on the left hand side of the table designate the types of nodes to which a directed link can go. A filled slot in the matrix indicates that a node of type designated by the column heading can be linked to a node of the type designated by the row heading. The type of link that should be used is designated by the slot entry. An empty slot indicates that no linkage is permissible. Due to space constraints, we use the following abbreviations: *Repository* for data repository, *Item* for data item, participation for a participation link, precedence for a precedence link, data flow for a data flow link, generation for a data generation link, physical for a physical flow link, flophys/mix for a data flow or physical flow or mixed flow link. All of these links are directed links.

Partition From Node / To Node	I				II				III			
	Participant	Activity	Decision	Interference	Boundary	Repository	Item	Flow Merge	Physical	Source / sink		
<i>Participant</i>												
<i>Activity</i>	participation	precedence	precedence	precedence	precedence	data flow	data flow	flophys/mix	physical			
<i>Decision</i>	participation	precedence	precedence	precedence	precedence	data flow	data flow	flophys/mix	physical			
<i>Interference</i>	participation	precedence	precedence	precedence	precedence							
<i>Boundary</i>		precedence	precedence	precedence	precedence							
<i>Repository</i>		data flow	data flow			generation	generation			data flow		
<i>Item</i>		data flow	data flow			generation	generation			data flow		
<i>Flow Merge</i>		flophys/mix	flophys/mix			data flow		flophys/mix	physical			
<i>Physical</i>		physical	physical									
<i>Source / sink</i>						data flow	data flow					

### 4.3.2 Semantic Rules

In this section, we present the semantic rules of the model. The first two precedence rules apply to all nodes (i.e., activity, decision, boundary and interference) in Partition II. In these rules, the so-called “process event” refers to an activity, decision or interference, or a boundary that marks the initiation of a subprocess. The semantic rules are:

- **Conjunctive Precedence Rule** — *A process event cannot occur unless all its precedent events are completed.* This rule elaborates the connotation of the precedence relationships for those nodes that have more than one incoming precedence links. The other nodes from which the precedence links originate represent the precedent events.
- **Disjunctive Precedence Rule** — *If the node representing a process event is marked with the special symbol “+” enclosed in a circle, then the event can occur as soon as one of its precedent events has been completed.* This rule presents an exception to the above conjunctive precedence rule. Indeed, it represents a special type of precedence relationships in which only one of the predecessors must finish before a successor can begin. This rule is needed to cover the special cases in which the situation described by the rule applies. For example, an activity can have two incoming precedence links, one of which loops back from an activity that happens downstream. The special symbol comes from the original Data Flow model [Gane 79].
- **Bounded Subprocess Rule** — *Each subprocess must contain at least one activity and must be delimited by at least two distinct boundary nodes. One of the boundary nodes must have an “activated” state and another must have a “terminated” state.* Therefore, all subprocesses must eventually be terminated. This rule ensures that each subprocess is defined properly and has a beginning and end. As a result, the overall process always has a beginning and end.
- **State Closure Rule** — *A subprocess that enters an “activated” or “resumed” state must eventually reach a “terminated” or “suspended” state; similarly, a subprocess that enters a “suspended” state must eventually reach a “resumed” or “terminated” state.* This rule extends the previous rule by specifying the possible states that can occur in between the beginning and the end of a subprocess. With this rule, we assume that a subprocess is initiated and eventually terminated or suspended. When suspended, it is resumed or terminated later. When resumed, it is terminated or suspended again. A subprocess that is

suspended and resumed over and over again must eventually reach a terminated state, by the previous rule. In short, this rule ensures that no subprocess has open ends in the schema.

---

## 4.4 Schema Transformation Operations

---

The ultimate goal of functional analysis is to represent processes in an enterprise and produce their functional schemata. Developing functional schemata of processes, especially those in facility engineering, is a complicated and time-consuming task. As [Batini 92] pointed out, the analyst typically begins with an initial schema and takes several iterations to further develop and refine it. Each iteration includes many steps, which in turn can involve one incremental transformation of the schema.

PANDA provides a number of basic atomic operations that support those incremental transformations. (They are actually “types” of operations, for the purpose of generality and conciseness. A type can have many variations.) These operations have been customized for this model. In addition, their definition closely follows the syntactic and semantic rules stated in the previous sections. This ensures that the analyst can produce valid and meaningful functional schemata using these operations. These operations are used in the methodology accompanying the model that is presented in the next chapter. They are labeled for future reference. The label consists of a roman numeral such as I, II, III indicating the applicable partition and an abbreviation such as C for Creation (i.e., creating new nodes) and M for Modification (i.e., modifying the way in which the nodes are connected). The abbreviation is followed by an integer indicating the ordering of the operation.

The basic schema transformation operations are:

- ***I-C1: Creating a Participant*** — This operation creates a participant node in Partition I. It draws the new node and links it to an existing activity, decision or interference node in Partition II using a participation link. The link’s annotation describes the participation (i.e., capacity in which the participant is involved).
- ***I-M1: Adding a Participation Link*** — This operation adds a participation link connecting an existing participant node in Partition I to another activity, decision or interference node in Partition II.
- ***I-M2: Modifying a Participant*** — This operation changes the name or the participation’s description of an existing participant node in Partition I, or reconnects that node to a

different activity, decision or interference node in Partition II using a new participation link. This operation has three variations: modifying name, modifying participation, and modifying an existing node connectivity.

- **II-C1: Creating a Subprocess** — This operation creates a subprocess in Partition II. It draws one activating boundary node, one activity node, and one terminating boundary node and connects them in that order using precedence links.
- **II-C2: Creating a Functional Unit in an Existing Subprocess** — This operation creates a functional unit and adds it to a subprocess already defined. The unit can be an activity, decision, interference, or even a child subprocess. The corresponding new node (or nodes in the case of subprocess) are drawn and then connected to one existing node in Partition II. The connection is done using precedence links and following the appropriate syntactic rules for activity, decision, interference and boundary nodes.
- **II-M1: Doing Functional Decomposition** — This operation spawns an existing activity node at one level of functional decomposition, say Level A, to a child subprocess at the next lower level, Level B. The subprocess at Level B is created in the same way as the one described in Rule II-C1. The decomposed activity at Level A is also changed to a subprocess by adding boundary nodes that delimit the subprocess. (Both subprocesses have boundary nodes with identical names for latter reconnection.)
- **II-M2: Modifying a Functional Unit** — This operation changes the name of a node (i.e., activity, decision, interference or boundary) in Partition II, or reconnects that node to a different node using a new link. The appropriate syntactic rules for nodes in Partition II and links apply. This operation has two main variations: modifying name and modifying an existing node connectivity.
- **III-C1: Creating a Data Repository or Data Item** — This operation creates a data repository or data item node in Partition III. It draws the new node and connects it to one activity or decision node in Partition II or to a flow merge node in Partition III using a data flow link.
- **III-M1: Adding a Data Flow Link from a Data Item or Repository to a Process Node** — This operation adds a data flow link connecting an existing data item or data repository node in Partition III to an activity or decision node in Partition II or another flow merge node in Partition III.

- **III-M2: Adding a Data Generation Link** — This operation adds a data generation link connecting an existing data item or data repository node in Partition III to another data item or data repository node. The operation has two variations: generating a data repository or data item.
- **III-C3: Creating a Data Source or Sink** — This operation creates a data source or sink node in Partition III. It draws the new node and connects it to one data item or data repository node in Partition III using a data flow link.
- **III-M3: Adding a Data Flow Link from a Data Source or Sink to a Data Item or Repository** — This operation adds a data flow link connecting an existing data source or sink node in Partition III to another data item or data repository node.
- **III-C4: Creating a Material or Product** — This operation creates a physical node in Partition III. It draws the new node and connects it to one activity or decision node in Partition II or to one flow merge node in Partition III using a physical flow link.
- **III-M4: Adding a Physical Flow Link** — This operation adds a physical flow link connecting an existing physical node in Partition III to an activity or decision node in Partition II or another flow merge node.
- **III-C5: Creating a New Flow Network** — This operation creates a new network from several existing flows that have a common destination. It has three variations: creating a data flow network, physical flow network, or mixed flow network. By adding a flow merge node and a data flow link, the first variation combines flows from data item or data repository nodes into a network. The new flow merge node is connected to the data item or repository nodes. The new data flow link goes from the flow merge node to the destination node. Similarly, the second variation combines flows from physical nodes into a network by adding a flow merge node and physical flow link. The third variation combines several flows, some from data item or data repository nodes and others from physical nodes, by adding a flow merge node and a mixed flow link.
- **III-M5: Merging with an Existing Flow Network** — This operation merges a flow or flow network that already exists with another existing flow network. It differs from the above operation: No new flow network is created here. This operation has three variations: merging a data flow, a physical flow or another flow network. The first variation merges a data flow from a data item or data repository node with a flow merge node by connecting

the first node to the latter node. The second variation is similar to the first, except that it involves a physical node and a flow merge node. The third variation is identical, except that it involves two flow merge nodes. These variations can be used over and over again to build elaborate flow networks. In all three cases, the type of the link coming out of the flow merge node must be checked using the syntactic rules for links.

- ***II-M6: Modifying a Node in Partition III*** — This operation changes the name of a node (i.e., data item, data repository, data source or sink, or flow merge node) in Partition III, or reconnects that node to a different node using a new link. The appropriate syntactic rules for nodes in Partition III and links apply. This operation has three main variations: modifying name and modifying an existing node connectivity.

Key concepts for this chapter: *partitioned data flow architecture, participant, participation, process, activity, precedence relationships, decision, alternative, interference, subprocess, boundary, state, data item, data repository, data flow, data source or sink, data generation, material or product, physical flow, mixed flow, flow network, node, link, Partitioned Data Flow diagram (or P-diagram), rule, schema transformation operation.*

## **Chapter 5**

---

# **USING PANDA**

We begin this chapter by explaining how the scope of functional analysis of a process can be defined. We then present a methodology for using the model. Next, we give guidelines for using specific concepts of the model. We provide additional guidelines on how to draw Partitioned Data Flow diagrams. Finally, we suggest a check list that can be used to validate the resulting functional schemata.

### **ORGANIZATION**

- 5.1 Scope Definition before Functional Analysis
- 5.2 Mixed Two-Pass Methodology
- 5.3 Guidelines for Using Concepts of PANDA
  - 5.3.1 Using Participant and Participation
  - 5.3.2 Functional Decomposition Using Subprocess
  - 5.3.3 Using Precedence Relationship
  - 5.3.4 Using Decision and Alternative
  - 5.3.5 Using Boundary
  - 5.3.6 Using Data Item and Data Repository
  - 5.3.7 Using Flow Network
- 5.4 Guidelines for Drawing Partitioned Data Flow Diagrams
  - 5.4.1 Labeling All Nodes
  - 5.4.2 Numbering Activities and Decision Nodes
  - 5.4.3 Laying Out Subprocesses
  - 5.4.4 Annotating the Diagram
- 5.5 Validation of Functional Schemata

---

## **5.1 Scope Definition before Functional Analysis**

---

The life cycle of a facility extends from its initial programming through its design, construction, maintenance, operation, retrofitting and even demolition. The engineering process that extends over this life cycle is long-term and quite involved. *Before functional analysis of the process begins, the scope of that analysis must be defined. This is crucial to ensuring satisfactory results. Defining the scope involves:*

- *Stating the breadth of the analysis effort:* Making general statements about which disciplines, participants, phases, data and data sources and sinks are to be included in and excluded from the analysis. For example, in the case study presented in the next chapter, we cover all phases of the electrical utility transmission tower engineering process except the last tower facility management phase, which we are not interested in.
- *Stating the depth of the analysis effort:* Specifying as much as possible the extent to which the analyst should carry out the analysis of the above categories. This also allows the analyst to put different emphases on various parts of the analysis. For instance, in the case study, we analyze the tower structural design, construction planning and construction execution phases in greater detail than the transmission line analysis and design phase. In tower structural design, we focus on designing new tower structures rather than retrofitting existing structures.

The resulting specifications will guide the analyst throughout the entire effort.

---

## **5.2 Mixed Two-Pass Methodology**

---

This methodology guides the analyst in applying PANDA to different facility engineering processes. *The methodology uses a mixed two-pass strategy, which involves top-down functional decomposition and bottom-up functional refinement in two successive passes.* [Batini 92] presents different strategies for functional analysis using the original Data Flow model. However, the strategy presented here is customized for use with PANDA. This methodology also takes advantage of PANDA's partitioned architecture by giving different priorities to the partitions at different stages of analysis.



The methodology is as follows:

PASS I: TOP-DOWN FUNCTIONAL DECOMPOSITION WITH PROCESS-PARTICIPANTS-DMP PRIORITY The first pass involves top-down functional decomposition of the process, with the following order of priority: (1) process, (2) participants, and (3) data, material and products. It includes the following steps:

- 1.1 Produce the top-level skeleton functional schemata of the phases and functions of the facility engineering process:* First, identify the major phases of the process and then the functions of which they are constituted. Section 5.3.2 provides detailed guidelines for doing this. Using PANDA, produce one or more *skeleton functional schemata* for those phases and functions. Use subprocesses to represent the phases and activities to represent functions at this top level. Use mainly the schema transformation operations II-C1 and II-C2 (defined in Section 4.4) to create subprocesses and add functional units (i.e., an activity, decision, interference, or even child subprocess) to those subprocesses. Use the graphical representation in PANDA to produce the Partitioned Data Flow diagrams (or P-diagrams) of these schemata.
- 1.2 Perform top-down functional decomposition using the skeleton functional schemata with highest priority on Partition II (Process):* Using the above skeleton functional schemata, successively decompose the functions into their component activities. (The analyst has complete control over the total number of functional decomposition levels.) Use the schema transformation operations II-C1, II-C2 to create subprocesses and add functional units and especially, operation II-M1 to do functional decomposition. Because of the potentially overwhelming complexity of the process at this point, concentrate mainly on the process (i.e., phases, functions and activities). Identify the key participants, when possible. The analyst does not have to be very specific about, or thorough with, the elements of Partition III. The order of priority here is Partition II, Partition I and Partition III.
- 1.3 Stop functional decomposition at the level of activities and augment the other partitions:* Stop functional decomposition at the level of individual activities. Section 5.3.2 provides a rule of thumb for doing this. Augment Partition I (Participants) and Partition III (Data-Material-Products) as much as possible by specifying the details to be included in those partitions. For instance, be more specific about the participants and their participation, the input and output data of the activities, the data repositories used, the potential flow networks, the data generation operation performed, and the data sources and sinks needed. Use mainly the schema transformation operations I-C1 and III-C1 to C5.

In this pass, the analyst does not have to complete Partitions I and III.

PASS II: BOTTOM-UP FUNCTIONAL REFINEMENT WITH DMP-PARTICIPANTS-PROCESS PRIORITY  
The second pass involves bottom-up functional refinement of the process, with the priority placed on: (1) data, material and products, (2) process, and (3) participants, in that order. It includes the following steps:

- 2.1 *Revise low-level functional schemata and complete Partition III (Data-Material-Products) and Partition I (Participants):* Revise each functional schema by first reviewing, and making necessary changes, and then completing Partition III of the schema. Then, using Partition III, revise Partition II by reviewing it and modifying it if necessary. Go back and forth between these two partitions until no more changes are needed. Using Partition II, review, modify and complete Partition I. Use the schema transformation operations II-C1 to C2, I-C1, III-C1 to C5 to add new nodes to the three partitions and especially, I-M1, III-M1 to M5 to modify Partitions I and III.
- 2.2 *Take an inventory of the important data sources and sinks, data repositories and data items involved in the process and trace them:* First, take an inventory of all key elements that are involved in the process. Those elements include data sources and sinks, data repositories, data items and participants. Then, trace back and see whether and where those elements are represented in the functional schemata produced from the previous pass. If they have not already been included, revise the appropriate functional schemata to include them in Partition III. This may involve revising Partitions II and I of those functional schemata.
- 2.3 *Carry out another step similar to 2.2 but for the participants.* First, take an inventory of all participants involved in the process. Then, trace back and see whether and where those participants are represented in the functional schemata. If they have not already been included, revise the appropriate functional schemata to include them in Partition I. This may involve revising Partitions II and III of those functional schemata.
- 2.4 *Revise the higher level functional schemata using a bottom-up refinement approach:* Use the functional schemata modified in the previous step to revise the functional schemata of higher levels, including those skeleton schemata produced in Step 1.1. The objective here is to integrate the low-level functional schemata and produce consistent functional schemata at all levels.

All three partitions should be completed by the end of this second pass.

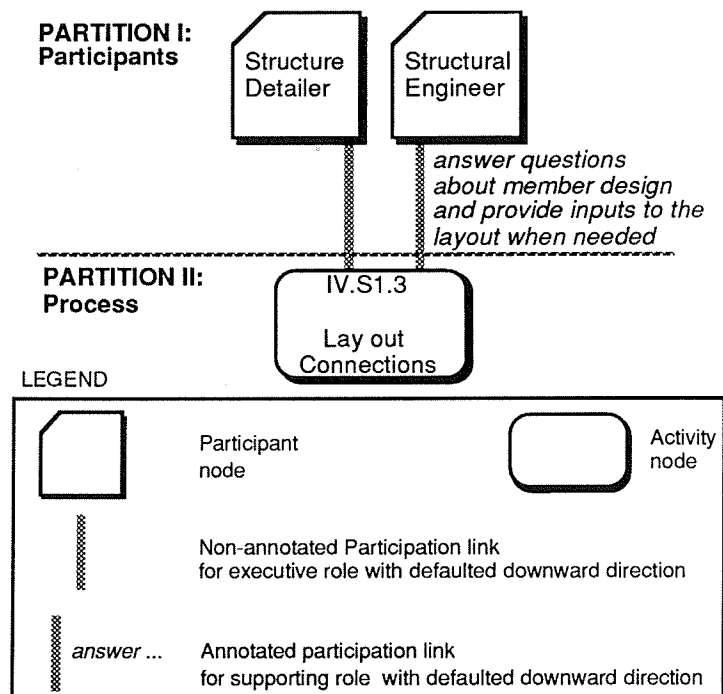
### 5.3 Guidelines for Using Concepts of PANDA

The following sections provide more detailed guidelines for using specific concepts of PANDA. In addition to the above methodology, the typical analyst should follow these guidelines when he or she designs the functional schema. The analyst should see these as suggestions rather than requirements.

#### 5.3.1 Using Participant and Participation

*Defaulting*, an idea first suggested in the work in ISAC A-Graphs [Lundeberg 82], is used both to reduce the amount of work an analyst has to do and to improve the graphical readability of the resulting P-diagram. There are two conventions for defaulting here:

1. *No annotation is needed for the executive role, as illustrated in Figure 5.3.1. However, for the supporting role, the link must be annotated with a clear description of the capacity in which the participant is involved.* A link from a participant node to an activity node without annotation means that the participant is carrying out the activity.



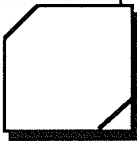


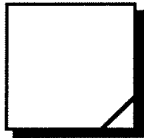

**FIGURE 5.3.1: Defaulting Conventions for Participation Links.** There are two: No annotation is needed for the executive role, and all participation links assume a downward direction. (The reader can refer to Figure 5.2.1 for comparison.)

To review, there are two possible roles for participants: an executive role (carrying out an activity), or a supporting role (being involved in other indirect capacities). Both types of roles use the same graphical representation, the participant link. However, the annotation of the link from the participant node to the activity node differs in the two cases.

2. Any link between a participant node in Partition I and an activity node in Partition II assumes a downward direction. The opposite direction is not relevant here. Partition I (of participants) is always shown above Partition II (of process) in the P-diagram.

In addition, the analyst can duplicate participant nodes in the P-diagram to minimize the number of link crossings and improve the graphical readability of the diagram. Duplicated nodes must be clearly denoted using the standardized graphical notations in Table 5.3. These notations were inspired by Gane's version of the Data Flow model [Gane 79].

**TABLE 5.3:** *Nodes that Are Most Likely to Be Duplicated.* These nodes include participant, data repository, data item, data source or sink, material or product nodes.

<i>Concept</i>	<i>Graphical Representation</i>
<i>Participant</i>	Duplicated Participant Node 
<i>Data Repository</i>	Duplicated Data Repository Node 
<i>Data Items</i>	Duplicated Data Item Node 
<i>Data Source or Sink</i>	Duplicated Data Source or Sink Node 
<i>Material or Product</i>	Duplicated Physical Node 

### **5.3.2 Functional Decomposition Using Subprocess**

Hierarchical top-down functional decomposition provides a powerful mechanism for analyzing complicated processes, as explained in Section 4.2.2. It allows the analyst to gradually reveal the details of the process. The concept of subprocess in PANDA can effectively support this decomposition. Using this concept, the analyst can represent all the intermediate functional units to which the process is decomposed.

**IDENTIFYING PHASES** To begin, decompose the overall process into several phases. *A phase is defined here as a subprocess that corresponds to a major identifiable stage of development in the facility life cycle.* For example, the tower facility engineering process can be divided into many phases: Tower Structural Conceptual Design, Tower Structural Detailed Design, and Tower Construction Planning, to name a few. Diagram 0 in Chapter 6 shows all phases of the tower engineering process.

Use the following guidelines to identify the phases of a process:

- *As a short cut, consult a domain expert.* A domain expert can usually identify a phase in terms of the following: time, people involved, place, work involved, goal, important decisions made and end results. A goal is a long-term purpose toward which a major effort in the project is directed.
- *Otherwise, look for major milestones in the process where a transfer of responsibilities and important deliverables between participants from two major disciplines takes place.* The dividing lines between the phases are usually associated with these turning points. For example, at the end of the Tower Structural Detailed Design phase, the chief structural engineer hands over the structure's schematic drawing to the construction manager, who then takes charge and initiates the Tower Construction Planning phase.
- *Consider the informational differences between various periods in the process.* The informational differences between the phases normally lie in the amount and granularity of detail of the facility description. For example, the Tower Structural Conceptual Design phase produces the global geometric data of the tower structure (i.e., tower height, panel heights, cage width, base spread, etc.) and the general geometric data of the systems and members in the tower body and arms. By contrast, the Tower Structural Detailed Design phase adds a lot more detailed design data. This data includes member sizes (e.g., angle L 8

x 8 x 1/2), dimensions, cross-sectional properties, analysis end conditions, stresses, deflections, reaction loads, and number and pattern of bolt holes.

- *Finally, consult existing work on standard process description for different facility types.* For instance, [Luth 91] suggests three broad categories of phases: planning, execution and operation. Here is a short list of suggested references. [Vanegas 87] describes the early phases of building design. [Sanvido 84] concentrates on the later construction process. [EPRI 87] documents a life-cycle engineering process for electrical power plants. [Sanvido 91] shows a life-cycle descriptive model of buildings. [Luth 91] provides a life-cycle process description for high-rise commercial office buildings. [Phan 92] describes the phases of electrical transmission tower engineering. The next chapter in this report summarizes that description.

**IDENTIFYING FUNCTIONS OF A PHASE** A function is a group of coherent activities that together help achieve a distinct objective or short-term purpose. Diagram 1 of Chapter 6 shows the functions of the Transmission Analysis and Design phase. A phase can be decomposed into several functions using the following guidelines:

- *Identify functions as groups of coherent activities that together help achieve a distinct objective or short-term purpose.* For example, the Tower Structural Conceptual Design phase can be divided into three functions: Preliminary Load Computation, Geometry Configuration and Conceptual Design Refinement. The activities are the organizational units for performing the function.
- *Divide the major tasks of the phase according to the discipline involved.* A function is normally carried out by experts of one discipline.
- *Examine the project control hierarchy.* [Luth 91] suggests that divisions of a process are subprocesses that interact, communicate and also impose constraints on one another. A project control hierarchy is needed to anticipate and coordinate these subprocesses. Such a hierarchy exists at three levels: global, regional and local. At the global level, [Luth 91] classifies functions into three general categories: owner, design and construction. The breakdown in the control structure at the next lower levels may help identify the functions of the process.

- *Finally, keep each function small—no larger than a single sheet on which the schema is drawn. Larger functions that span several sheets of drawing need to be re-examined and possibly decomposed further.*

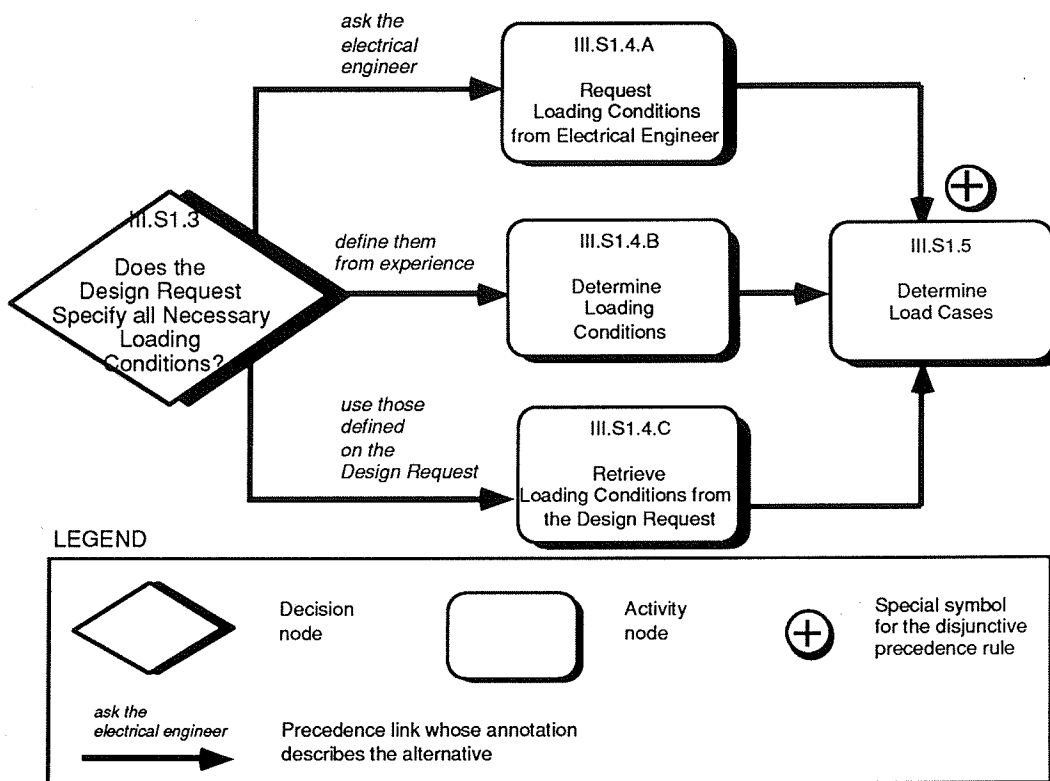
**IDENTIFYING ACTIVITIES OF A FUNCTION** Finally, decompose functions into several activities. Diagram 1.1 in Chapter 6 shows the activities of the first function of the Transmission Line Analysis and Design phase. As rules of thumb, stop the functional decomposition when one of the following becomes true:

- *Further decomposition would require specifying domain knowledge about the design.*
- *Further decomposition would not reveal new information of interest about the participants involved or data used.*
- *A single experienced designer can carry out the task involved in a reasonable, acceptable number of man-hours.*
- *A single software module can be built to handle the activity.*

For example, to further decompose "Determine Tower Loads" in the tower conceptual design requires knowing how to select the proper equations needed to compute wind and wire tension loads, and how to choose the values for various load factors for the given tower location, setting and loading conditions. Further, a spreadsheet program can be built to compute the loads.

### 5.3.3 Using Precedence Relationship

In general, a precedence link from a process node, A, to another process node, B, in Partition II indicates that B cannot begin until A finishes. However, when B has several predecessors, the analyst can apply the disjunctive precedence rule by marking B with the special symbol “+” enclosed in a circle. This means that B can start as soon as one of its predecessors was completed. Without this symbol, the conjunctive precedence rule prevails: B cannot start until all of its predecessors are completed. Figure 5.3.3.1 gives an example of applying the disjunctive precedence rule to an activity.



**FIGURE 5.3.3.1: An Example of Applying Disjunctive Precedence Rule to An Activity.** In this example, a decision involves several alternatives, which lead to the same downstream activity designated II.S1.5. However, this activity can start as soon as any of its predecessors is completed.

### 5.3.4 Using Decision and Alternative

When using the concept of decision and alternative, the analyst should use these guidelines:

- Each alternative should lead to a unique activity. An alternative that combines other alternatives should be defined separately and lead to its own activity.

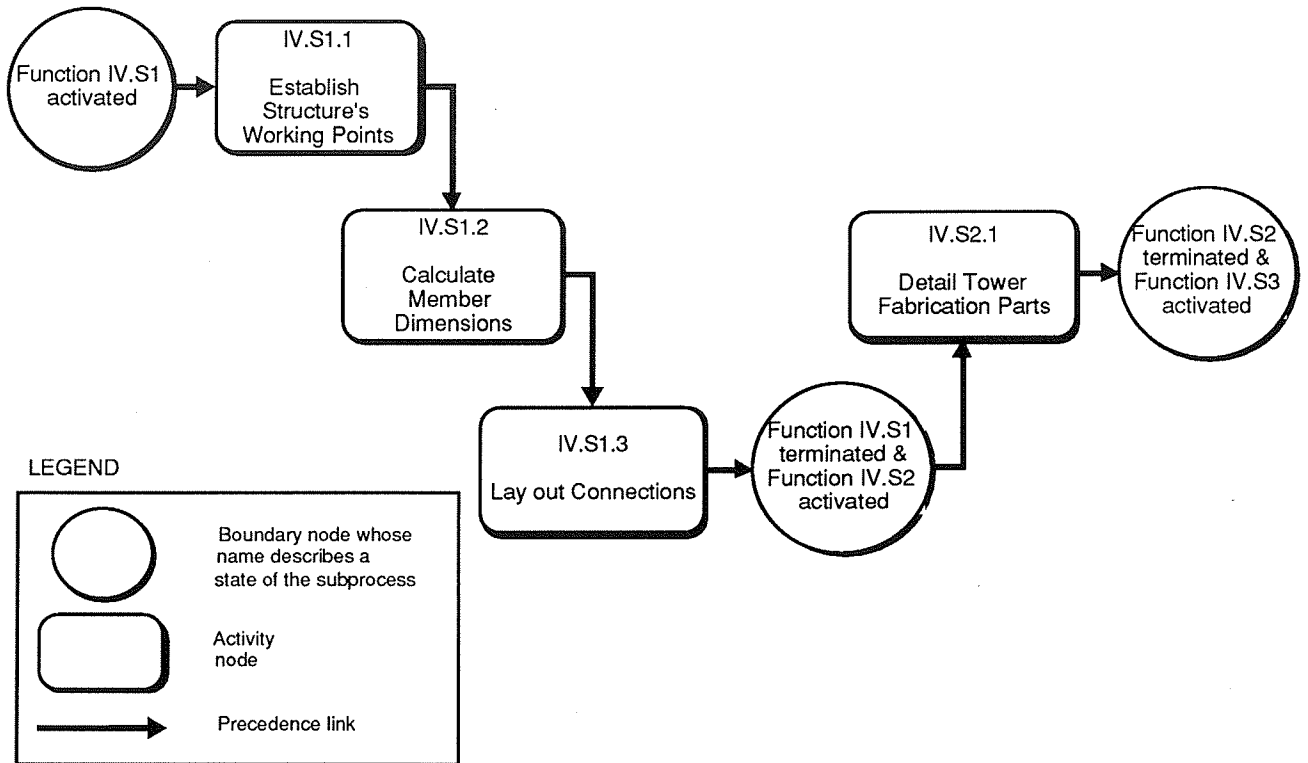


- When several alternatives for the same decision all lead eventually to the same activity (as shown in Figure 5.3.3.1), the analyst should carefully consider which precedence rule (i.e., conjunctive or disjunctive) applies and use the correct rule.

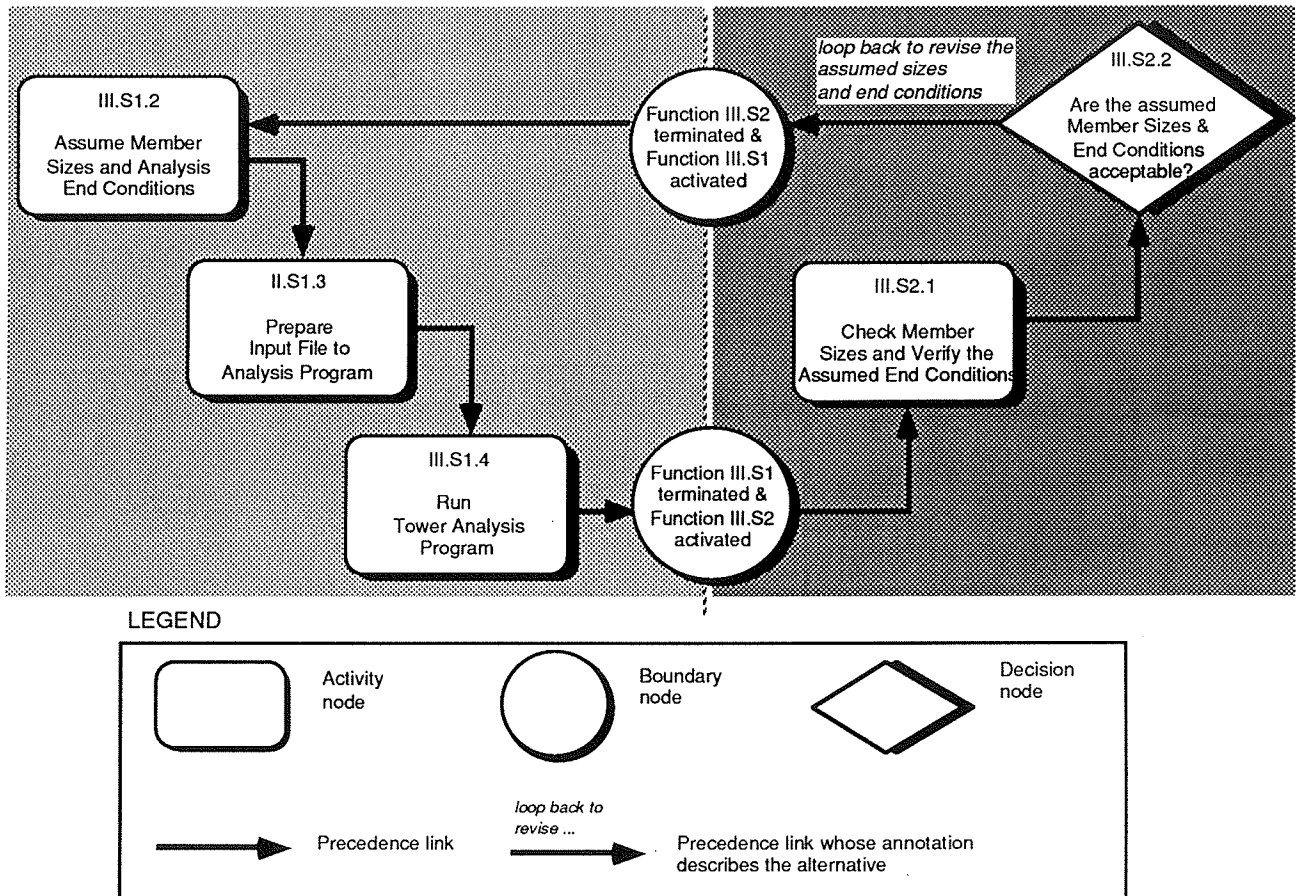
### **5.3.5 Using Boundary**

When using the concept of boundary, the analyst should follow these guidelines:

- The analyst should always use a boundary node to mark the beginning or end of a subprocess.
- He or she can also use boundaries to connect a subprocess to a single activity or join two consecutive subprocesses that are in contact. In the second case, the adjoining subprocesses share a common boundary node, as illustrated in Figure 5.3.5.1.
- The analyst should name the boundary node by describing the states of the subprocess or subprocesses at that boundary.
- When using boundary nodes, the analyst should pay close attention to the bounded subprocess rule and state closure rule. The first rule ensures that each subprocess is defined properly and has a beginning and end. The second rule extends the first rule by making sure that every subprocess must eventually terminate and thus has no open ends in the schema. Both rules are explained in Section 4.3.2.



**FIGURE 5.3.5.1: Boundary Nodes Delimiting and Connecting Subprocesses.** The first subprocess, designated IV.S1, consists of three activities and is delimited by the boundary node on the left and that in the middle. The second of these boundary nodes connects that subprocess to another subprocess on the right. The second subprocess, designated IV.S2, has one activity.



**FIGURE 5.3.5.2: Subprocesses Sharing More Than One Boundary Node.**

A subprocess may have more than two boundary nodes. It can share one or more boundary nodes with another subprocess with which it is in contact. This normally occurs when the process involves a design loop, as illustrated in Figure 5.3.5.2.

### 5.3.6 Using Data Item and Data Repository

There are two general guidelines here:

- The analyst can represent data items that are transient or will be stored later. With data item nodes, the analyst can represent a single data item or a collection of related data items. The latter case can be used to improve the graphical readability of the P-diagram. In that case, the name of the node lists the data items in the collection.
- As with participant nodes, the analyst can duplicate data item or data repository nodes in the P-diagram. The objective is to minimize the number of link crossings and improve the graphical readability of the diagram. Similarly, the analyst can duplicate data source or sink

nodes. All duplicated nodes must be clearly denoted using the graphical notations in Table 5.3. The name of the duplicated node can mention when the original node was introduced.

### **5.3.7 Using Flow Network**

The following convention is adopted for flow networks:

*Links among flow merge nodes or links between them and other node types in Partition III are generally uni-directional, and their arrows should be clearly shown. Undirected links are permitted but assumed to be bi-directional.*

Thus, the analyst has the ability to show bi-directional flows when needed. These flows can help reduce the number of links to be drawn.

---

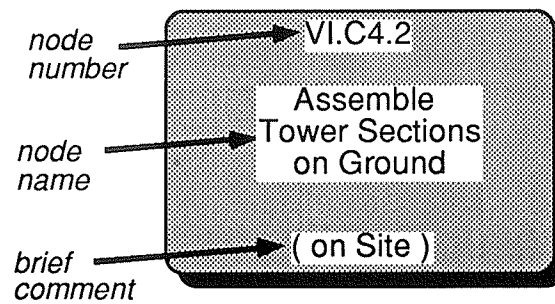
## **5.4 Guidelines for Drawing Partitioned Data Flow Diagrams**

---

### **5.4.1 Labeling All Nodes**

As in the original Data Flow model [Gane 79, Yourdon 79], the following convention applies to labeling all nodes in the P-diagram. As illustrated in Figure 5.4.1.1, a node label includes:

- *A node number* — This is required for all activity or decision nodes and optional for all other nodes. The numbering of activity and decision nodes allows the analyst to show the process' functional decomposition and to trace that decomposition through all the functional schemata.
- *A node name* — This is required for all nodes. The name should express the meaning or content of the node. For example, the name of an activity node explains the work involved, whereas the name of a data item node specifies its content. However, the name of boundary nodes should describe the state of the subprocess or subprocesses at the boundary.
- *A brief comment shown in parentheses*. This comment is optional for all nodes. For activity nodes, the comment may describe the physical location where the activity is performed.

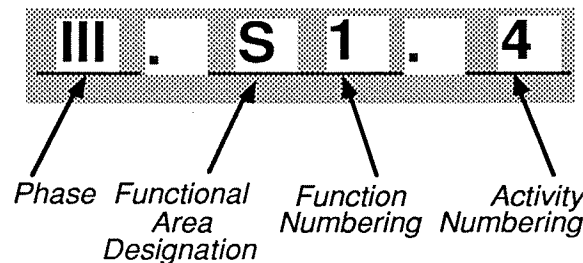


**FIGURE 5.4.1.1: Node Label.** The node number is optional for all nodes except activity and decision nodes. The node name is required for all nodes. The brief comment is optional for all nodes.

### 5.4.2 Numbering Activities and Decision Nodes

Each activity node and decision node should have a node number, which indicates the phase and function to which it belongs. The following node numbering scheme, illustrated in Figure 5.4.2.1, is suggested: Indicating a phase using a Roman numeral such as I, II, III, etc. Indicate a function by designating the pertinent functional area and then denoting with a number the order of the function in the phase to which it belongs.

In facility engineering, the possible functional areas are Architecture, Structural Engineering, Geotechnical Engineering, Foundation Engineering, Electrical Engineering, Mechanical Engineering, Construction, Environmental Engineering, Project Management, Material Supply, Fabrication, etc. The functional area designation is a selected abbreviation of one of those functional areas. Activities are indicated using .1, .2, .3, .4, etc. If Activity 1 is later decomposed, the component activities will be numbered 1.1, 1.2, 1.3, etc. Parallel subprocesses are indicated using alphabet characters such as A, B, C, etc. As an example, the activity “Run Tower Structural Analysis” can be labeled as III.S1.4, indicating that it is the fourth activity in the first function of Phase III. This activity involves the structural engineering discipline.



**FIGURE 5.4.2.1: Numbering Activities and Decision Nodes.** Phases are indicated using Roman numerals. Functions are indicated by a designation of the functional area followed by a number. Functional area designation is a selected abbreviation of the functional area involved such as S for Structural Engineering. Activities are indicated using .1, .2, .3, etc.

### 5.4.3 Laying Out Subprocesses

The analyst can use different layouts to graphically represent a subprocess. These layouts are not available in the original Data Flow model and can greatly enhance the graphical readability of the resulting P-diagram. Figure 5.4.3.1 illustrates two possible layouts. The staircase layout places the functional units of a subprocess as descending or ascending steps of a staircase. A change in the direction of two consecutive staircases translates into a change in subprocesses in the engineering process. The cascade layout places all functional units of the subprocess on the same line. However, consecutive subprocesses are laid on different lines, creating a cascading effects.

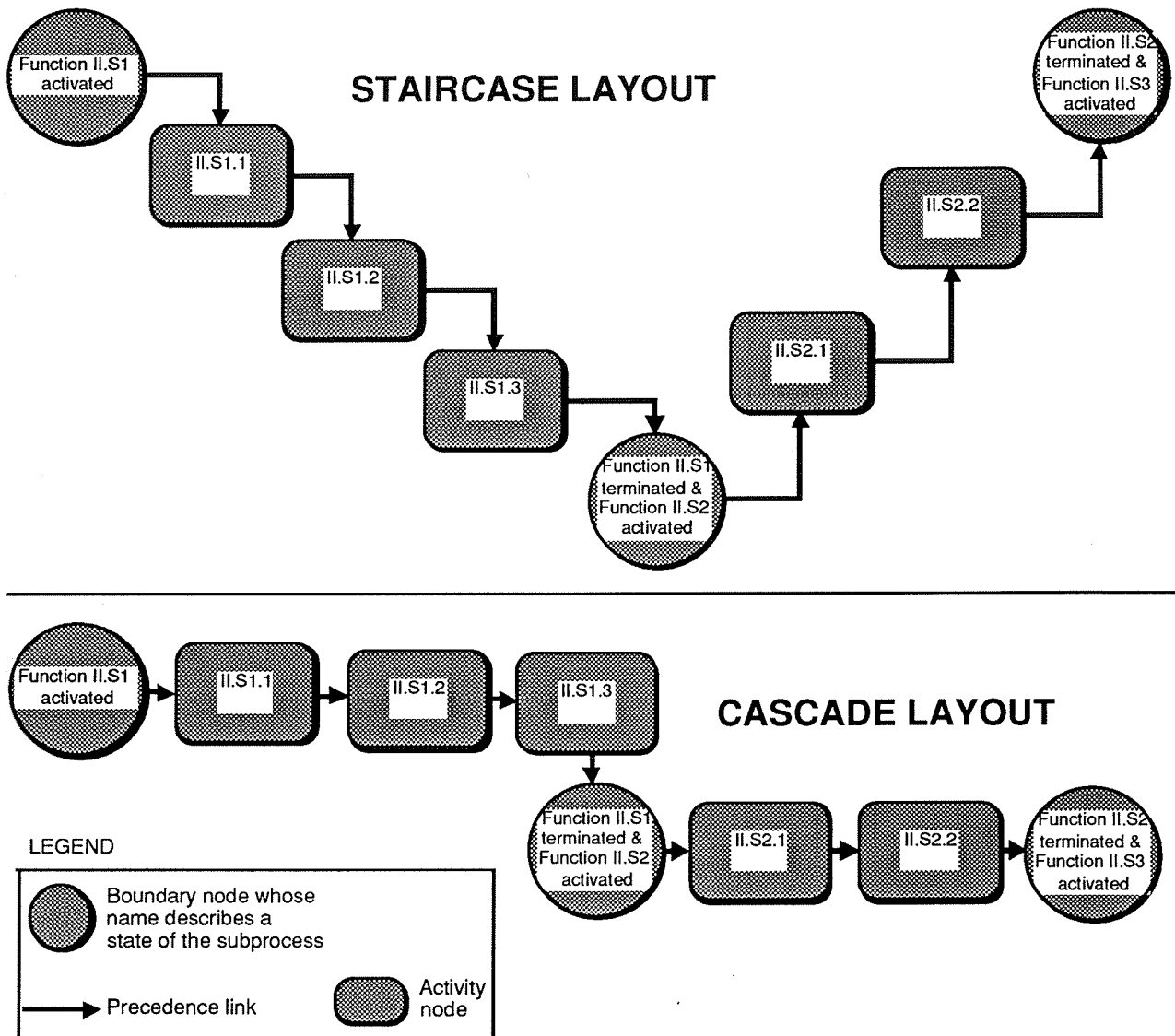


FIGURE 5.4.3.1: Two Suggested Layouts of Subprocesses.

#### **5.4.4 Annotating the Diagram**

Annotation can also enhance the P-diagram's descriptiveness and enable it to communicate more information. Annotation relies on verbal description to complement the graphical elements of the diagram. There are three types of notes<sup>†</sup> for doing this:

1. *Direct Link Notes:* Annotate the links in the diagram by showing the note directly next to the link. This annotation scheme applies to participation, precedence, data flow and data generation links.
2. *Referenced Notes:* Annotate a node or link in the diagram using a reference number or symbol that points to a separate note displayed at a conventional place (e.g., the lower right corner) in the diagram.
3. *Diagram Notes:* Annotate the diagram using general notes that apply to the whole diagram. Like reference notes, diagram notes should be collected and placed at a conventional place in the diagram.

---

### **5.5 Validation of Functional Schemata**

---

Figure 5.5.1 on the next page shows a checklist that the analyst or other team members can use to validate the resulting functional schemata. The checklist helps ensure that the functional schemata is complete, minimal and correct. The first four items in the checklist verify completeness of the schema; the fifth item verifies for minimality; and the remaining items verify for correctness. Schema completeness, minimality and correctness are defined in [Batini 92].

---

<sup>†</sup> *These note types are similar to notes, footnotes and metanotes in the Structured Analysis Design Technique (SADT) [Ross 77b].*

1. Are all phases, functions and key activities in the process represented in at least one of the functional schemata?
2. Can the activity be further decomposed? (Using the rule of thumb in Section 5.3.2, answer this question for each of the activities in all the schemata.)
3. For each of the activities or decisions in all of the functional schemata:
  - a. Are all the participants in the activity and their roles included in the schema?
  - b. Are all the precedence relations of the activity included in the schema?
  - c. Are all the input and output data of the activity included in the schema?
  - d. Are all the data sources and sinks of the input and output data of the activity included in the schema?
  - e. Are the precedence rules (i.e., conjunctive precedence or disjunctive precedence) defined in Section 4.3.2 applied correctly to each activity, decision or interference node in the schema?
4. Are all the interferences that normally occur during the process included in the schema?
5. Are any participants, data items, data repositories, and data sources and sinks represented in more than one place? If so, are they clearly marked as duplicated nodes using the graphical representation in Table 5.3.
6. Does the bounded subprocess rule (defined in Section 4.3.2) apply to every subprocess in the schema? In other words, does every subprocess have at least one activity, and is it delimited by at least two boundary nodes? Are all subprocesses eventually terminated?
7. Does the state closure rule (defined in Section 4.3.2) apply to every subprocess in the schema? In other words, does every subprocess that enters an activated or resumed state reach a terminated or suspended state, and does every subprocess that enters an suspended state reach a resumed or terminated state?

**FIGURE 5.5.1: A Checklist for Validation of Functional Schemata.**

Key concepts for this chapter: *scope of analysis, mixed two-pass methodology, guidelines, schema validation, completeness, minimality, correctness.*



## **Chapter 6**

---

# **A CASE STUDY: ELECTRICAL UTILITY TRANSMISSION TOWER FACILITY ENGINEERING**

In this chapter, we first explain the case study and define the scope of analysis involved. We then introduce the hierarchical functional decomposition of the tower engineering process and describe the process in detail. Each phase is explained in terms of time, place, participants, work involved, goal, key terms and concepts, functional decomposition, and end results. Finally, we present the graphical functional schemata of the process that result from using PANDA.

### **ORGANIZATION**

- 6.1 Case Study and Scope of Analysis Involved
- 6.2 Functional Decomposition of the Tower Engineering Process
  - 6.2.1 Breakdown of the Process into Phases
  - 6.2.2 Breakdown of the Phases into Function
- 6.3 Detailed Description of the Process
  - 6.3.1 Transmission Line Analysis & Design Phase
  - 6.3.2 Tower Structural Conceptual Design
  - 6.3.3 Tower Structural Detailed Design
  - 6.3.4 Tower Construction Planning
  - 6.3.5 Tower Construction Execution
  - 6.3.6 Tower Facility Management
- 6.4 Graphical Functional Schemata of the Process

---

## **6.1 Case Study and Scope of Analysis Involved**

---

Electrical utility transmission towers are large lattice structures used for transmitting electrical power. The tower facility engineering process extends throughout the tower life cycle, from the initial need analysis to the possible final demolition of the structure. The process includes several stages of development. It typically involves multiple participants from various disciplines, including electrical engineering, structural engineering, fabrication and construction management. Computer applications are used to automate certain design functions of the process, such as structural analysis and member design. In a complex engineering process of this nature, the database designer must thoroughly understand the process in order to effectively model the data used. Therefore, tower facility engineering is an ideal domain for applying PANDA.

In this case study<sup>†</sup>, we examine the entire process of engineering a transmission tower. However, we put less emphasis on the tower facility management that occurs after the tower is constructed. This facility management involves many scenarios in which various activities are carried out to keep the tower operational. It is difficult to cover all of those scenarios given our time and resource constraints. We also analyze the initial electrical design stage of the process in less detail than some other stages. In tower structural design, we focus on designing new tower structures rather than retrofitting existing structures.

---

## **6.2 Functional Decomposition of the Tower Engineering Process**

---

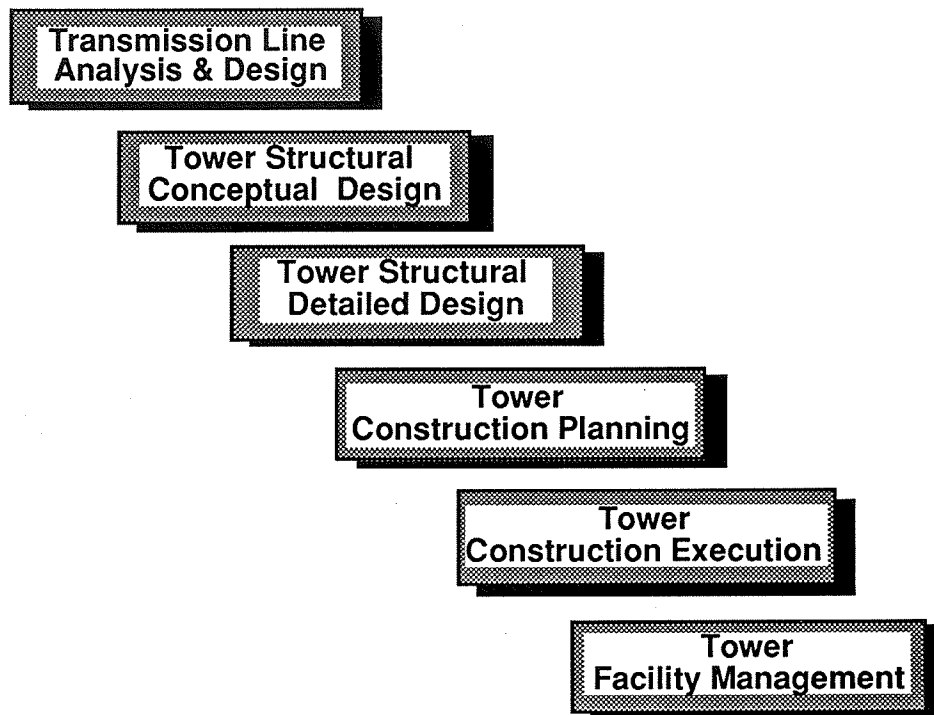
### **6.2.1 Breakdown of the Process into Phases**

A process can be decomposed into several phases, as described in Section 5.3.2. A phase is a subprocess that corresponds to a major identifiable stage of development in the facility life cycle. Indeed, it can be identified by domain experts in terms of time, people involved, place, work involved, goal, important decisions made, and end results.

The tower facility engineering process involves programming, design, construction and long-term management of the facility. As shown in Figure 6.2.1.1, the entire process can be broken down into the following phases: (1) *Transmission Line Analysis and Design*, (2) *Tower Structural Conceptual Design*, (3) *Tower Structural Detailed Design*, (4) *Tower Construction Planning*, (5) *Tower Construction Execution* and (6) *Tower Facility Management*.

---

<sup>†</sup> The information presented in this case study was provided and verified by engineers of a local utility company and also came from our own work experience in this area.

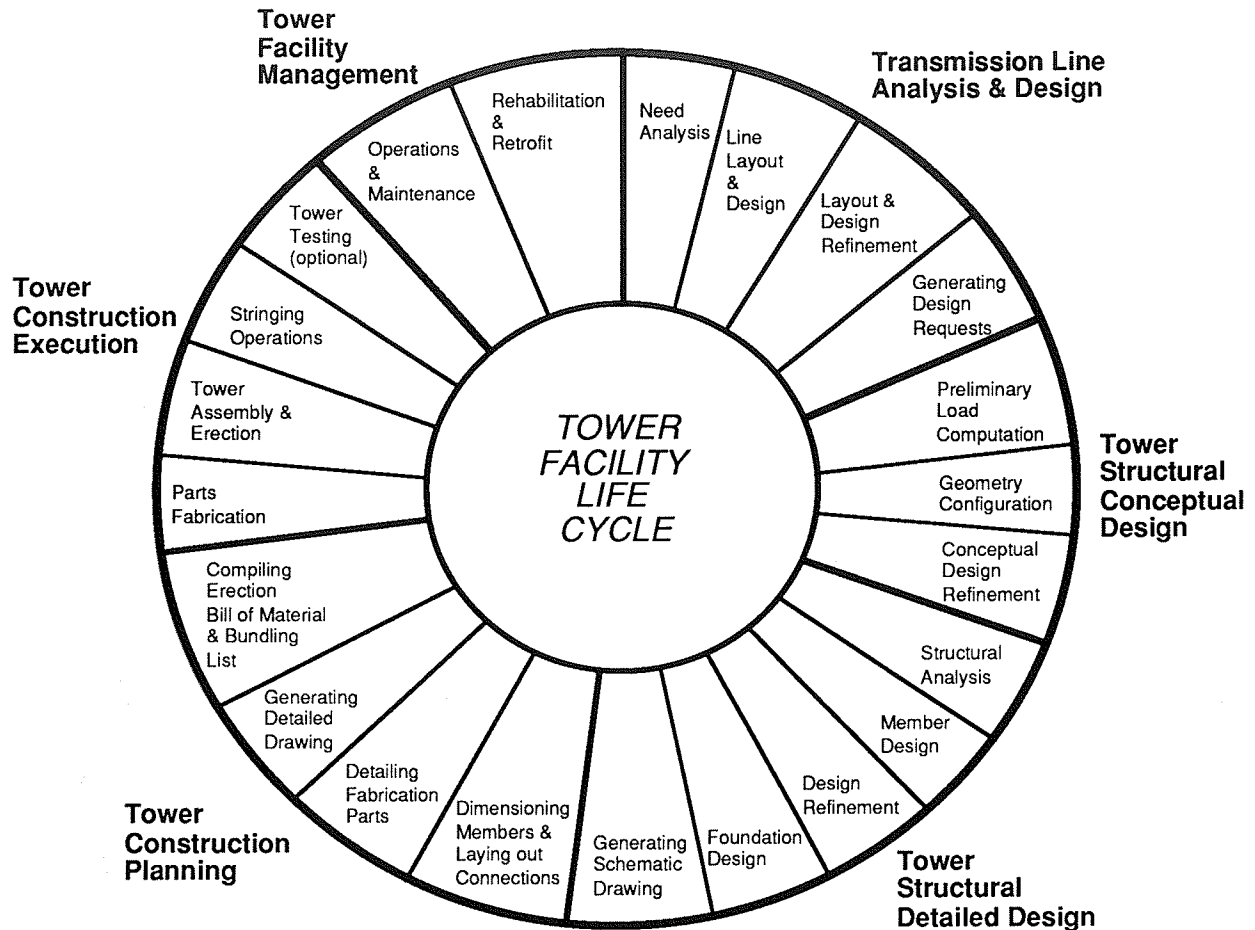


**FIGURE 6.2.1.1: Major Phases of the Tower Facility Engineering Process.**

Phase 1 here corresponds to the programming of the facility, Phases 2 and 3 to the design, Phase 4 and 5 to the construction, and Phase 6 to the management. By and large, this breakdown is consistent with the one for commercial high-rise office buildings as defined in [Luth 91]. However, since transmission towers are less complicated than buildings, their life cycle is much simpler. This breakdown is customized for towers to accurately depict their life cycle.

### **6.2.2 Breakdown of the Phases into Functions**

Each phase can then be divided into two or more functions. A function is a group of coherent activities that together help achieve a distinct objective or short-term purpose. Figure 6.2.2.1 on the next page summarizes the functions into which the six phases of the tower engineering process can be decomposed.



**FIGURE 6.2.2.1: Hierarchical Functional Decomposition of the Phases of the Tower Engineering Process into Functions.** The functions are labeled within the sections of the circle. The phases are labeled outside of the circle. The heavier lines mark the beginning and end of each phase and help identify the functions that belong to that phase.

### 6.3 Detailed Description of the Process

In this section, we describe in detail all six phases of the tower engineering process. We use the following format consistently for all phases: (1) general description (time, key project participants, place, work involved and goal), (2) key terms and concepts, (3) description of the functions to which the phase is decomposed and (4) end results of the phase. This format reflects the way in which a domain expert would identify a particular phase.

As an alternative, the reader can review the graphical functional schemata of the process using PANDA that are shown in the next section. Those schemata present a concise, pictorial description of the process.

### **6.3.1 Transmission Line Analysis and Design Phase**

**GENERAL DESCRIPTION** This phase occurs at the beginning of the process when a need or economic opportunity for electrical utility services is perceived. The key participants are electrical engineers and structural engineers. This phase occurs at the engineers' workplace. It involves analyzing the perceived need and laying out and designing a new transmission line. The goal is to achieve an economic solution that fills the need and has an acceptable cost.

**KEY TERMS AND CONCEPTS** The following terms and concepts are used in this phase:

*Design Request* — An official document that requests the analysis or design of an existing or new tower type. It also specifies all the requirements needed for designing the tower type.

*Tower Design Requirements* — Statements of what is required of the tower's design (and even construction). For transmission towers, design requirements vary with the tower type cover electrical clearances, loading, strength and serviceability, constructibility, cost, right-of-way and tower dimensions.

**FUNCTIONS** This phase consists of four functions:

- *Need Analysis:* The electrical engineer analyzes the perceived need for electrical utility services within a region. She considers the current electricity demand and supply in the region. She projects the region's future growth and the increase in its electricity consumption. The engineer then decides whether a new transmission line is needed. If so, he or she determines how much electrical power should be transmitted and what type of conductors should be used to transmit that power. This allows the engineer to determine the transmission line voltage that is necessary.
- *Line Layout and Design:* The engineer sets the direction of the transmission line, considering the geography and topology of the terrain. He then designs the types of towers needed to support the line. Specifically, the engineer determines the global attributes of those tower types, including the tower function classification, line angle, tower setting, etc. He also sets the location and orientation of individual tower structures used in the line.
- *Line Layout and Design Refinement:* After the line is laid out and designed, the electrical engineer approximates its total cost. She decides whether the existing line layout and design needs to be refined to obtain a lower-cost solution. At this point, the structural

engineer may get involved by suggesting ways to produce tower structures that are lighter and thus lower cost. The cost of individual structures contributes to the total cost of the transmission line. The electrical engineer may take several iterations before reaching an acceptable solution. She then analyzes the cost/value ratio of the line and decides on its economic feasibility. When the electrical engineer decides that building the line is economically feasible, she finalizes the layout and design.

- *Generating Design Requests for Tower Types:* If the line is to be built, the electrical engineer then generates an official *design request* for each tower type used in the line. The design request contains all the information necessary to design the tower type, including its *design requirements*.

**RESULTS** The end results of this phase are: (1) the layout data of the transmission line and (2) the design requests of the tower types. The line layout data includes the location, orientation and tower settings of the towers used in the transmission line. For each tower type, the design request contains the design requirements and global attributes of the tower type. The design requirements help define the constraints on the tower design and construction in subsequent phases. The global attributes describe:

- tower function classification, line angle, static-wire spans and conductor-wire spans,
- tower electrical characteristics, including voltage, number of circuits, circuit arrangement and minimum static shield angle,
- the types and properties of the electrical equipment the tower carries, specifically conductor and static types, conductor and static tensions, the number of conductor and static wires per phase, and the types of insulators and hardware.

A sketch showing a typical tower setting in the line may also be included in the design request.

### 6.3.2 Tower Structural Conceptual Design

**GENERAL DESCRIPTION** This phase occurs as soon as the structural engineers receive the design request. The key participants are structural engineers, and this phase occurs at their workplace. It involves calculating the loads applied to the tower structure and configuring a tower geometry. The goal is to obtain a tower geometry that both meets the requirements (including the loading requirements) and will result in an economical design.

**KEY TERMS AND CONCEPTS** The following terms and concepts are used in this phase:

*Loading Condition* — A description of a scenario in which a tower structure would be subjected to external environmental loadings. For utility transmission towers, the environmental loadings can be gravity, wind, ice, temperature, ground motion, impact forces on the structure, conductor-wire tension (under normal conditions, broken wire conditions and construction and maintenance conditions), etc.

*Load Case* — A particular way in which the loading condition might occur. For example, a hurricane extreme loading condition with 100 mile-per-hour winds may produce several different load cases, each corresponding to a different wind direction: perpendicular to the transverse face of the tower, perpendicular to the conductor wires, at 45 degrees to the conductor wires, at every 15-degree increment from the tower bisector, etc.

*Load* — An external force applied to a structure in a certain load case. A description of a load includes its magnitude, direction, location, type (i.e., axial, moment, torsion) and form (i.e., concentrated, linear, per area).

*Load Tree* — A schematic representation of loading from a load case on a diagram of the structure. A conventional load tree shows all the load vectors from that load case at the location where they are applied on the structure. Alternatively, a combined load tree shows a resultant load vector at each significant location on the structure.

*Tower Structural Systems* — For transmission towers, the four major types of structural systems are (1) leg systems, (2) lacing systems, (3) arm systems and (4) redundant systems. Leg, lacing and arm systems are the *primary systems* of the structure that resist loading. Redundant systems are the *secondary systems* that mainly increase the stiffness and reliability of the primary load-resisting systems.

*Member* — A conceptualized component of a system that serves a particular function. *Primary members* (e.g., leg and lacing members) are members of primary systems, whereas *secondary members* (e.g., redundant members) belong to secondary systems.

**FUNCTIONS** This phase consists of three functions: (1) Preliminary Load Computation, (2) Geometry Configuration and (3) Conceptual Design Refinement. These functions are highly interdependent: Loads on the tower structure cannot be computed until a preliminary tower geometry is obtained, and a tower geometry cannot be generated until the loads are computed. This underlines the complexity of the tower design synthesis. In practice, the engineer uses his or her design experience as well as an iterative approach to find a solution. The detailed description of the functions is as follows:

- *Preliminary Load Computation:* Given the tower electrical clearances, the structural engineer first approximates the spatial arrangement of the static level, the conductor levels and the tower body. At this point, she needs a preliminary geometry of the tower in order to compute the loads. She may use the geometry of a similar tower type as a starting point. Alternatively, the engineer may use rules of thumb or past design experience to roughly configure a new geometry. She also determines all the necessary *loading conditions* if they were not completely specified in the design request. Using the loading conditions, she defines the principal *load cases*. For each load case, the engineer then calculates the *loads* and generates a *load tree* (either conventional or combined).
- *Geometry Configuration:* Having generated the load trees, the structural engineer can calculate the optimum base spread (i.e., the larger dimension at the base of the tower) that can sustain the tower loading. Next, he determines the cross-sectional shape of the tower. Following standard practice, the shape can be either square or rectangular. This decision depends on the way loading is applied to the structure and on the way the structure would distribute that loading. The engineer then determines the load paths and the tower structural systems suitable for those load paths. These systems include the leg, arm, lacing and redundant systems of the tower structure. To select the bracing pattern of the redundant systems, the engineer must give great attention to the overall stability of the system. Geometrically unstable redundant systems can cause premature failure of the primary members. In designing all those structural systems, the engineer also works out the complete details of the tower geometry and topology. In short, the geometry of a tower type is determined by several factors, including the number of circuits, the electrical clearances, the type of insulators (i.e., their length and maximum transverse displacement), the amount



of vertical deflection of the conductor wires at the attachment points, and the economy that the engineer is trying to achieve.

- *Conceptual Design Refinement:* After one iteration of the above functions, the structural engineer goes back and calculates the load trees based on the existing geometry. She then refines the existing geometry using those load trees. This process continues until a satisfactory geometry is obtained. The aesthetic impact of the tower may be the final consideration in configuring its geometry.

**RESULTS** This phase generates a large amount of data. First, it produces data that describes the individual loads of the load trees from different load cases (i.e., magnitude, direction, location, type and form). Second, this phase produces data of the tower geometry, including: (1) global geometric data of the tower (i.e., tower height, panel heights, cage width, base spread, taper ratio, extension heights, bend line elevation, etc.) and (2) detailed geometric data of the systems and members in the tower body and the static and conductor arms. The second set of data includes spatial data (i.e., coordinates, orientation, spatial envelope dimensions), geometric data (i.e., shape, dimensions, etc.) and topological data (i.e., topological representation and connectivities). The structural engineer also knows about the functions of the tower's systems and members. (These functions are resisting loads, implementing load paths, transferring loads, supporting members, bracing members, etc.) Unfortunately, this knowledge may not be represented in any format or documented in any source.

### 6.3.3 Tower Structural Detailed Design

**GENERAL DESCRIPTION** This phase occurs when the structural engineer has a satisfactory tower geometry and is in a position to design the structure. The key participants are structural engineers and foundation engineers (civil engineers who specialize in the field of foundation engineering). This phase occurs at the workplace or workplaces of these participants. It involves carrying out the detailed design of the structure and communicating the design information, by means of drawings and written specifications, to those who will detail, fabricate and erect the structure. Given the tower geometry determined during the previous phase, the goal is to obtain a light-weight structure that meets all the strength and serviceability requirements.

**KEY TERMS AND CONCEPTS** The following terms and concepts are used in this phase:

*Tower Anchoring Devices* — Devices used to connect the structure to the foundation. There are two common types: (1) base shoes used with anchor bolts and (2) stub angles. A *base shoe* is a welded assembly that consists of an angle and a base plate. *Anchor bolts* are special bolts used to anchor the tower structure to the foundation. A *stub angle* is a special angle member that is embedded in the concrete foundation and bolted to the tower legs.

*Schematic Drawing* — A drawing of the tower type's structure that the structural engineer produces at the end of the detailed design phase. The drawing is intended to communicate the tower design information to the detailer, fabricator and construction crew. Generating a schematic drawing involves putting the data generated in the required presentation format, and making any special notes or specific details to the detailer, fabrication and construction crew.

*Element* — An analysis component of a system that corresponds to a particular member of the system.

**FUNCTIONS** This phase consists of five functions: (1) Structural Analysis, (2) Member Design, (3) Design Refinement, (4) Foundation Design and (5) Generating Schematic Drawing. The first three functions are highly interdependent: The structural analysis cannot be carried out until the member sizes are known, and the member design cannot be done until the members' stresses and deflections from the analysis are obtained. The design refinement involves iterating over the preceding two functions to improve the design. In practice, the structural engineer also uses his or her design experience to find a solution. A detailed description of these functions is as follows:

- *Structural Analysis:* The structural engineer decides on the material (e.g., steel) and material grade (e.g., A-36) of the tower members. (Transmission towers today are built out of steel or aluminum.) The engineer also assumes the member sizes and analysis element (i.e., truss, beam, column, beam-column, etc.). To do this, the structural engineer uses his or her design experience and the data used in the design of similar tower structures in the past. He or she then carries out a structural analysis of the tower. Since transmission towers are highly indeterminate structures, they are analyzed using commercial finite-element analysis programs. The engineer also uses the loads and the tower geometry from the previous phase to prepare data for the input file of the analysis program. By running the analysis, the engineer obtains the behavior of the structure under the specified load cases. The output data includes the stresses, deflections and end reactions of the members.
- *Member Design:* Using the data generated by the analysis, the structural engineer checks the assumed member sizes for strength and serviceability. The members' capacity must sustain the controlling stresses and deflections from all load cases. The engineer considers the strength and serviceability requirements of standard design codes such as [ASCE 71], [ANSI 82], [SAE 88] and [AISC 89]. If a member fails, she tries another size until a satisfactory size is obtained. She also verifies the end conditions—i.e., the analysis model—of the members assumed in the structural analysis. A re-analysis is necessary if there is a large discrepancy between the members and elements assumed and those designed. Then, for each structural member, the engineer also determines the number, pattern and diameters of bolt holes required to transmit the member's internal loads. This information will be used to lay out the connections between members and to specify the fabrication details in the next phase.
- *Design Refinement:* Third, after one iteration of structural analysis and member design, the structural engineer might iterate over those two functions in order to design a structure that uses smaller member sizes and thus has a lighter weight. This function of the process is called design refinement rather than design optimization because its objective is to find a "good enough" solution rather than the best solution. Such a solution would meet all design constraints, would have acceptable accuracy in the analysis, and could be found in a reasonable time using the available human and computational resources. Also note that in this function, as in the previous two functions, loading, strength and serviceability, constructibility and cost requirements are carefully considered.

- *Foundation Design:* Last, using the controlling reaction loads at the tower base from all load cases, the foundation engineer designs the tower foundation. This involves designing the concrete foundation as well as the *anchoring devices* of the tower.
- *Generating Schematic Drawing:* The engineer produces a *schematic drawing* of the structure to communicate the information to the detailer, fabricator and construction crew.

**RESULTS** This phase results in a large amount of design information. This information includes the member sizes (e.g., angle L 8 x 8 x 1/2), dimensions, cross-sectional properties, analysis end conditions, stresses, deflections, and reaction loads, and the number and pattern of bolt holes. Moreover, the schematic drawing of the tower type displays the following:

- (1) drawing title, which includes the designation and version number of the drawing as well as the designation of individual sheets,
- (2) a schematic diagram of the tower structure showing its geometry and topology,
- (3) global tower geometric data such as tower height, panel heights, cage width, base spread, taper ratio, extension heights, bend line elevation, etc.
- (4) typical sizes (e.g., L 8 x 8 x 1/2) of the leg, lacing and redundant members,
- (5) loading conditions for which the structure is analyzed and designed, and even references to the load calculation,
- (6) general notes to the detailer about specifications, bolt sizes, steel material grades, member framing, drawing layout, etc., and
- (7) any specific details regarding member framing, member splices, static and conductor wire connections, tower-to-foundation connections, etc.

This schematic drawing is used in the next phase.

### 6.3.4 Tower Construction Planning

**GENERAL DESCRIPTION** This phase occurs once there is a finished schematic drawing that can be used to work out all the necessary construction details. The key participants are structure detailers. However, when detailing problems (e.g., missing design information, uncommon member sizes) arise, the detailers cooperate with the structural designers to resolve the problem. This phase occurs at the workplace of the detailers, who can be either company or contract employees. The phase involves developing all the fabrication and erection details in preparation for tower construction. The goal is to plan the next construction phase in order to minimize errors and maximize efficiency.

**KEY TERMS AND CONCEPTS** The following terms and concepts are used in this phase:

*Fabrication Part* — A single piece to be fabricated and used in the construction of the structure. It corresponds to a member or connection in the structure. For transmission towers, the part can be an angle member or a connection plate. However, a member or connection may have more than one fabrication part.

*Fabrication Feature* — A single specification of how a fabrication part should be made out of raw material. There are numerous fabrication features. For transmission tower members, fabrication features include dimensions, the hole pattern, hole sizes, edge preparation, edge clipping, gage line, etc.

*Mark Number* — An identification mark printed or stamped on a fabrication part.

*Working Point* — A point of reference that is selected from the tower structure's schematic diagram and used to work out all detailed dimensions of the members.

*Detailed Drawing* — A drawing of the tower type's structure produced by the detailer at the end of the construction planning phase. The drawing is intended to communicate to the fabricator and construction crew the detailed information necessary to fabricate and construct the tower in the subsequent phase. Generating a detailed drawing involves putting the data generated in the required presentation format, specifying the fabrication features of the tower members, and making any special notes or specific details to the fabrication and construction crew.

**FUNCTIONS** This phase consists of three functions:

- *Dimensioning Members and Laying out Connections:* From the schematic drawing, the detailer establishes the main *working points* of the tower structure. Using these working points, he or she calculates the overall length, slope and bevel of all members. To start laying out the connections, the detailer reviews any special notes about member framing on the schematic drawing. Having established the member lengths, she lays out each connection. The layout data generated for the connection includes its plate shape and size, hole pattern, and required ringfill and bolt length. The detailer also determines the member clearances that are needed to avoid interferences. In the next function, these clearances are used to determine the exact lengths of the fabrication parts that correspond to the members.
- *Detailing Fabrication Parts:* For each member or connection, the detailer determines the number of *fabrication parts* needed. He gives each fabrication part a unique *mark number*. He then details the fabrication part by specifying its fabrication features, such as dimensions, number of holes out, hole pattern, hole diameter, edge preparation, edge clipping, gage line, etc. To do this, the detailer uses the member sizes, steel material grades and any special notes from the schematic drawing. He also uses the connections' layout data from the preceding function.
- *Generating Detailed Drawing:* The detailer first generates the bolt schedule, which includes all the hardware (bolts, nuts, ringfills, etc.) needed to assemble the tower. She then computes the raw (or *black*) and total (or *galvanized*) weights of the basic tower and extensions. Last, she puts together the *detailed drawing* of the tower, which is a detailed graphical and textual representation of the tower structure. It includes all the information necessary to fabricate and construct the structure. As the final result of the design and construction planning, the detailed drawing communicates both the design information and the designer's intention to the fabricator and field construction crew. Therefore, the drawing must be unambiguous, readable, concise and complete as possible. Moreover, almost all detailed drawings currently used are paper-based.‡

---

‡ Better computer-aided design tools can significantly improve this function by automating the generation of detailed drawings in electronic form. Such drawings are a more error-free and cost-effective means of communicating design information.

- *Compiling Erection Bill of Material and Bundling List:* The detailer compiles a detailed list of all fabrication parts. He groups these pieces into separate bundles, taking into account their size, length, quantity and weight. The purpose of this function is to facilitate shipment from the fabrication shop to the site, as well as site handling by the construction crew. The result is the *erection bill of material and bundling list* that shows by bundles all fabrication parts of the tower structure.

**RESULTS** All design data for the tower structure is generated by the end of this phase. The results are the detailed drawing and the erection bill of material and bundling list. The detailed drawing displays the following information:

- *Erection diagrams*, which include a foundation setting plan and detailed sketches of the basic tower and extensions. These sketches indicate the mark numbers of all fabrication parts as well as bolt counts and bolt lengths at each connection.
- *A bolt schedule*, which lists all the hardware (bolts, nuts, ringfills, etc.) needed to assemble the tower.
- Loading conditions similar to those shown on the schematic drawing.
- Tower weights, including the weights of the basic tower and the tower used with different extensions. Each weight is the sum of the galvanized weight of the tower and the weight of the hardware.
- Details of fabrication parts, which include mark number, quantity, size, dimensions, material type and grade, fabrication features and any special fabrication and erection notes.

The erection bill of material itemizes the fabrication parts, whereas the bundling list shows how these pieces should be grouped together. These two results are usually combined into one deliverable. Together, they show by bundles the mark number, quantity, shape (e.g., L-shape for angle), size (e.g., 8 x 8 x 1/2), length, and approximate weight—since the true weight can only be known accurately after fabrication is completed—of all fabrication parts. It also includes any additional written remarks about these pieces.

Together, the detailed drawing and the erection bill of material and bundling list are a comprehensive, but not complete, representation of the tower design data.

### 6.3.5 Tower Construction Execution

**GENERAL DESCRIPTION** This phase occurs as soon as the fabricator who successfully bid the contract for tower fabrication receives the approved purchase order as well as the detailed drawing of the tower type. The key participants are the fabricator, the material supplier, the construction manager and field construction crew. This phase occurs at the fabricator's shop and at the site where the tower is to be erected. However, the electrical engineers and the structural designers of the tower type are also involved in inspecting the construction work at different stages. In addition, they provide their expertise in solving problems that arise during construction. If prototype testing of the tower type is to be carried out, structural engineers are heavily involved in developing the test specifications and monitoring the test procedure. This phase involves constructing the tower structure, as planned in the previous phase. (Actually, several structures of the same tower type may be installed in one transmission line). The goal is to minimize errors, damage and cost overruns during construction.

**KEY TERMS AND CONCEPTS** The following terms and concepts are used in this phase:

*Black (or Raw or Ungalvanized) Steel Parts* — Steel fabrication parts right after fabrication and before galvanization. The weight of all black steel pieces in the tower is called the *black weight*.

*Galvanized Steel Parts* — Black steel fabrication parts after being subjected to a galvanization process. Galvanization uses zinc coating to protect steel against weather corrosion. During galvanization, the black steel pieces are subjected to extensive surface preparation and prefluxing and are then immersed in molten zinc.

**FUNCTIONS** This phase consists of four functions:

- *Parts Fabrication:* The fabricator reviews the detailed drawing and the erection bill of material and bundling list. If she detects any errors, inconsistencies or complications (e.g., material grade is not available at the time), she contacts the detailer or possibly the structural designer and works out the problems. After having a working detailed drawing, the fabricator then prepares material take-off lists. At this point, the fabricator procures the necessary raw material from warehouses or steel mills. Fabrication begins after the material supplier delivers the raw material as ordered. Today, fabrication is done using advanced computer-aided manufacturing technology and in particular, numerically controlled programmable equipment. After making all the parts, the fabricator cleans them up to



remove mill scale, dirt and grease. Then, he galvanizes these *black steel parts* using an time-consuming process. He carefully inspects the *galvanized steel parts* for uniformity, appearance and defects. Finally, he bundles and ships them to the site.

- *Supplying Raw Steel Material:* The material supplier gathers the raw steel material as ordered by the fabricator and then makes the delivery. Parts fabrication resumes as soon as the material was delivered.
- *Tower Assembly and Erection:* The construction crew opens the bundles at the site where the tower is to be erected. They assemble as much of the tower on the ground as possible: tower arms, extensions, cage and panels below the bend line. Problems generally arise during the tower assembly. Member fit, for instance, is a major concern. Parts that do not fit together create complications and slow down the construction process. Revisions to the tower's detailed drawing might be required to avoid similar problems in the future. (Constructibility knowledge should be used early in the tower design and detailing to eliminate problems of this nature.) Next, the construction crew lifts the assembled sections by crane, put them in place and connect them. They then lift the assembled tower, place it onto its stub angles or base shoes, and anchor it to the foundation. (Note that small tower structures can be completely assembled and then lifted onto the foundation, whereas larger structures are usually erected section by section.)
- *Stringing Operations:* After erecting enough towers in the transmission line, the crew carries out stringing operations to install the insulators and static and conductor wires on the towers. The wires are then pulled up to the cable tension specified in the original Design Request and in the detailed drawing.

The tower testing function is optional and not discussed here.

**RESULTS** This phase results in the tower structures constructed and the transmission line installed. The basic difference between this phase and the building construction execution phase in [Luth 91] is that the construction sequence and methods used are different for the two kinds of structures.

### 6.3.6 Tower Facility Management

**GENERAL DESCRIPTION** This phase occurs after the tower is constructed. As with buildings, it corresponds to the remaining period in the tower facility life cycle. Depending on the specific work involved, the participants are the people who have been involved in the design and construction of the tower. Therefore, this phase can occur at the engineers' workplace or in the field. Simply put, this phase involves managing the constructed tower. The goal is to make sure that the tower is operational and thus serve its functions throughout its life span.

**FUNCTIONS** This phase consists of two functions: (1) Operation and Maintenance and (2) Rehabilitation and Retrofitting. These two functions are independent of one another and can be concurrent. A detailed description of these functions follows:

- *Operations and Maintenance:* This function includes post-construction activities aimed at improving or maintaining the conditions of the tower in order to make it operational. For example, a wide-flange member may be added on each side of the tower cage at each level of conductor arms. This addition enables a construction worker to stand on it and work on the tower. These post-construction activities can also be aimed at protecting the tower structure against weathering effects, structural failures, acts of God, sabotage, etc. Member repair also falls into this category. However, the activities here do not change the purposes for which the tower was designed and constructed. In addition, they do not involve major re-analysis or re-design of the tower structure.
- *Rehabilitation and Retrofit:* In contrast to the above function, this function includes post-construction activities aimed at changing the purposes for which the tower was designed and constructed. These activities generally involve re-analysis and/or re-design of the tower structure. Re-analysis and re-design may be necessary for a number of reasons, including: new design code requirements, different construction methods of the tower structure, addition of new electrical equipment to the structure, a different type of foundation and/or foundation construction methods, a different geographic location, a different electrical facility usage of the tower (i.e., usage of the tower with a different voltage, line angle, tower function classification, static and conductor wires, other electrical equipment, etc.).

**RESULTS** The result of this phase is an operational tower. The basic difference between this phase and the building facility management phase in [Luth 91] is that operation and maintenance are defined here as one coherent function instead of two separate functions.

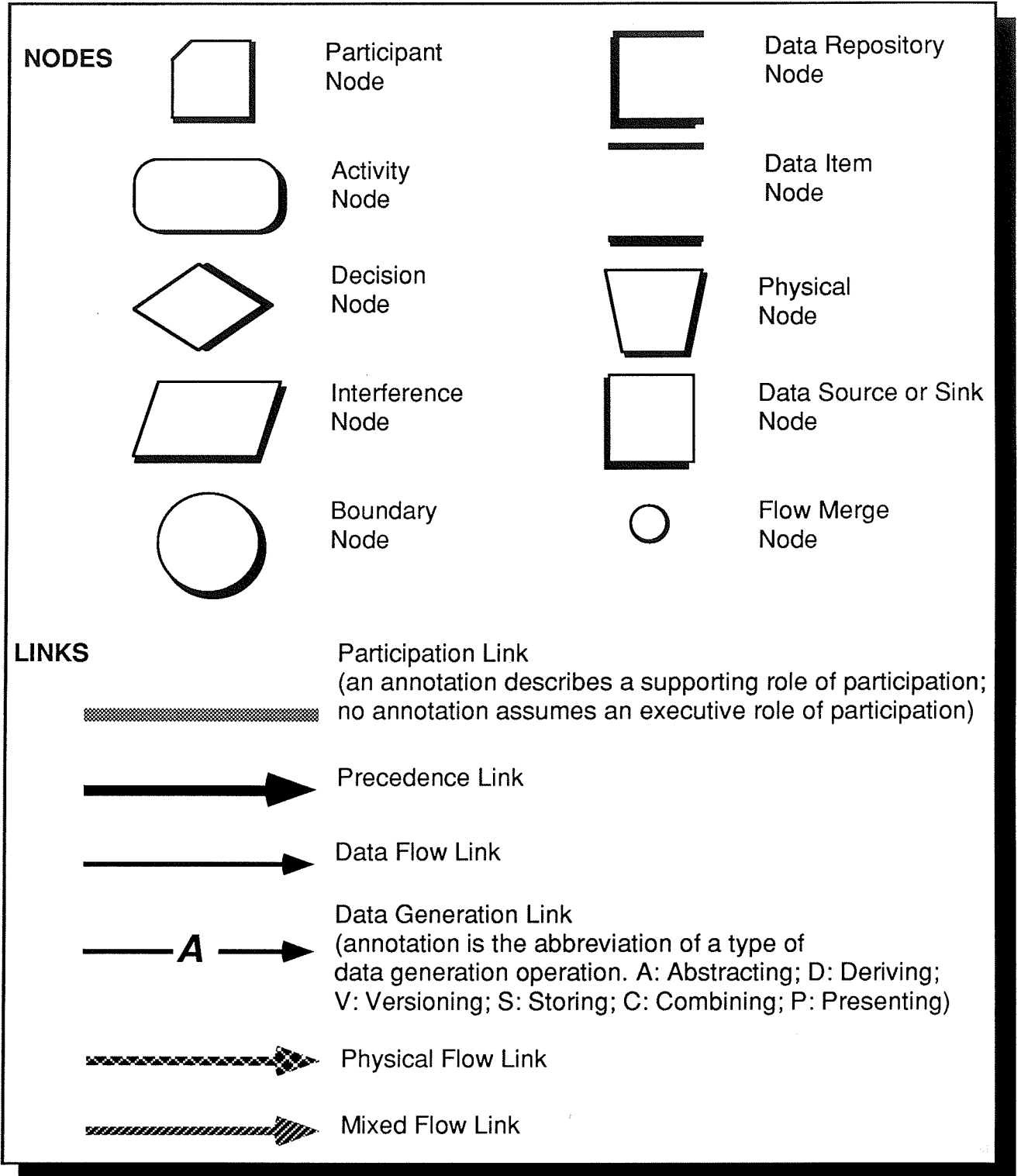
---

#### ***6.4 Graphical Functional Schemata of the Process***

---

This section presents a series of Partitioned Data Flow diagrams (or P-diagrams) that are the graphical function schemata of the tower engineering process described in the preceding section. Due to space constraints, we first show a legend and diagram notes for all the P-diagrams that follow. The legend includes the graphical symbols for the concepts of PANDA. Diagram notes are general notes that applies to an entire diagram that specifically refers to them. Only reference notes (i.e., annotations of nodes or links using a reference number or symbol) are shown directly on the diagram. Then, the first diagram illustrates the process' highest-level skeleton functional schema with all six phases. The remaining diagrams show the more detailed functional schemata of the first five phases. As explained in the beginning, the last tower facility management phase is not shown here.

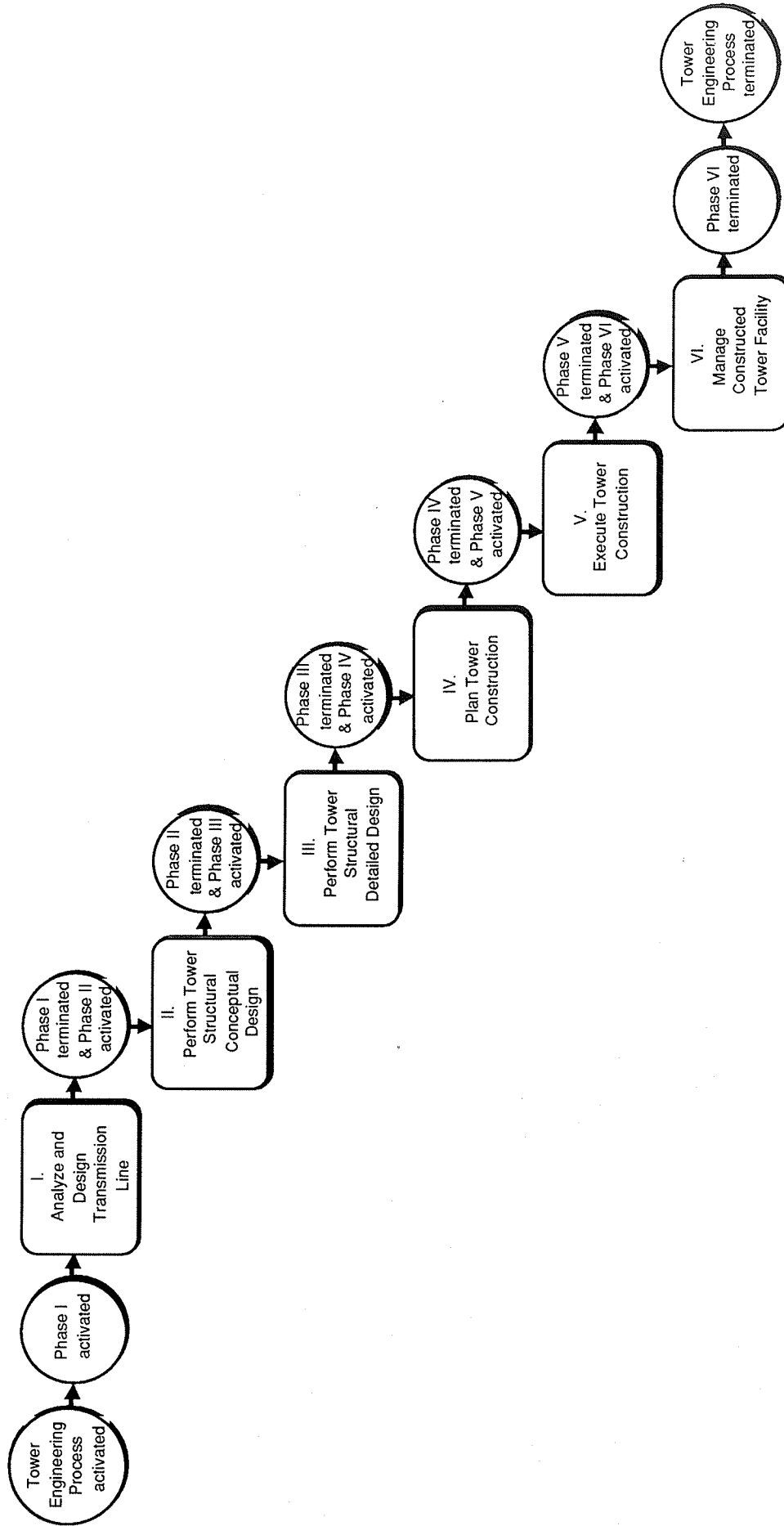
## LEGEND FOR ALL THE PARTITIONED DATA FLOW DIAGRAMS THAT FOLLOW



## **DIAGRAM NOTES FOR ALL THE PARTITIONED DATA FLOW DIAGRAMS THAT FOLLOW**

- A. This diagram is the highest-level skeleton graphical functional schema of the process. It illustrates the breakdown of the process into several major phases. This schema results from the first step, 1.1, of the methodology presented in Section 5.2 of Chapter 5. At this level, the three partitions do not show the details about the participants, activities, data, material and products. Successive top-down functional decomposition of this skeleton schema would reveal those details. Indeed, the diagrams that follow this would give more details for the individual phases and their functions and activities.
- B. This diagram is an intermediate skeleton graphical functional schema of a phase. It illustrates mainly the breakdown of a phase into several functions. This schema results from the first step, 1.1, of the methodology presented in Section 5.2. At this level, the three partitions still do not reveal all the details about the participants, activities, data, material and products. Also, the schema does not necessarily show all the design loops and iterations that could possible occur in this phase. In fact, the diagrams that follow this would reveal all of the aforementioned details for the individual functions of this phase.
- C. This diagram is a detailed skeleton graphical functional schema of one or more functions. It illustrates the breakdown of the function or functions into several activities. It shows all the details in three partitions: (1) Participants, (2) Process and (3) Data, Material and Products. This schema results from completing all the steps in the two passes of the methodology presented in Section 5.2.

**PARTITION II: PROCESS**



**DIAGRAM 0: Highest-level Skeleton Graphical Functional Schema of the Electrical Utility Transmission Tower Facility Engineering Process. (See Diagram Note A)**

PARTITION II: PROCESS

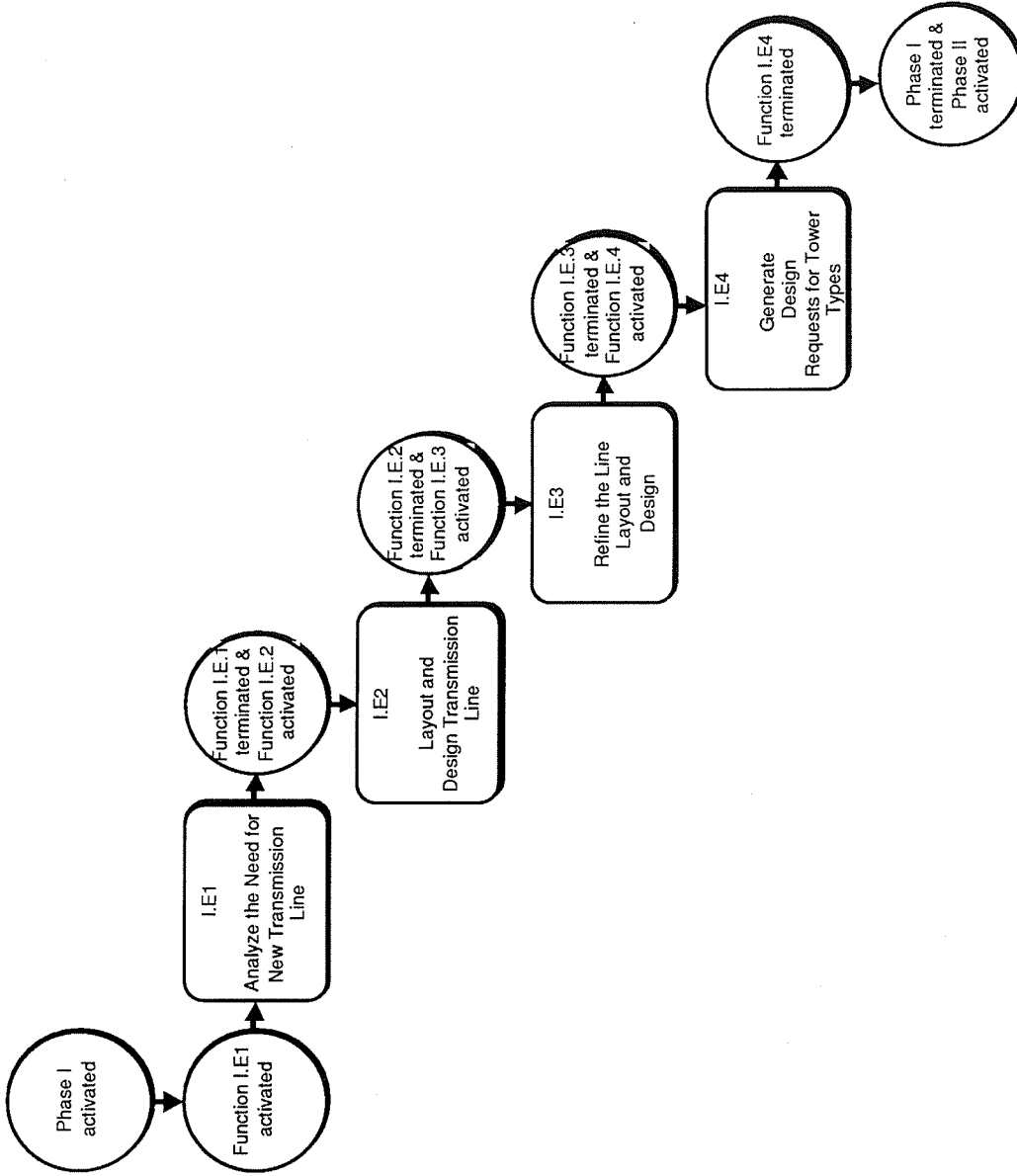


DIAGRAM 1: Intermediate Skeleton Graphical Functional Schema of Phase I, Transmission Line Analysis and Design.  
(See Diagram Note B)

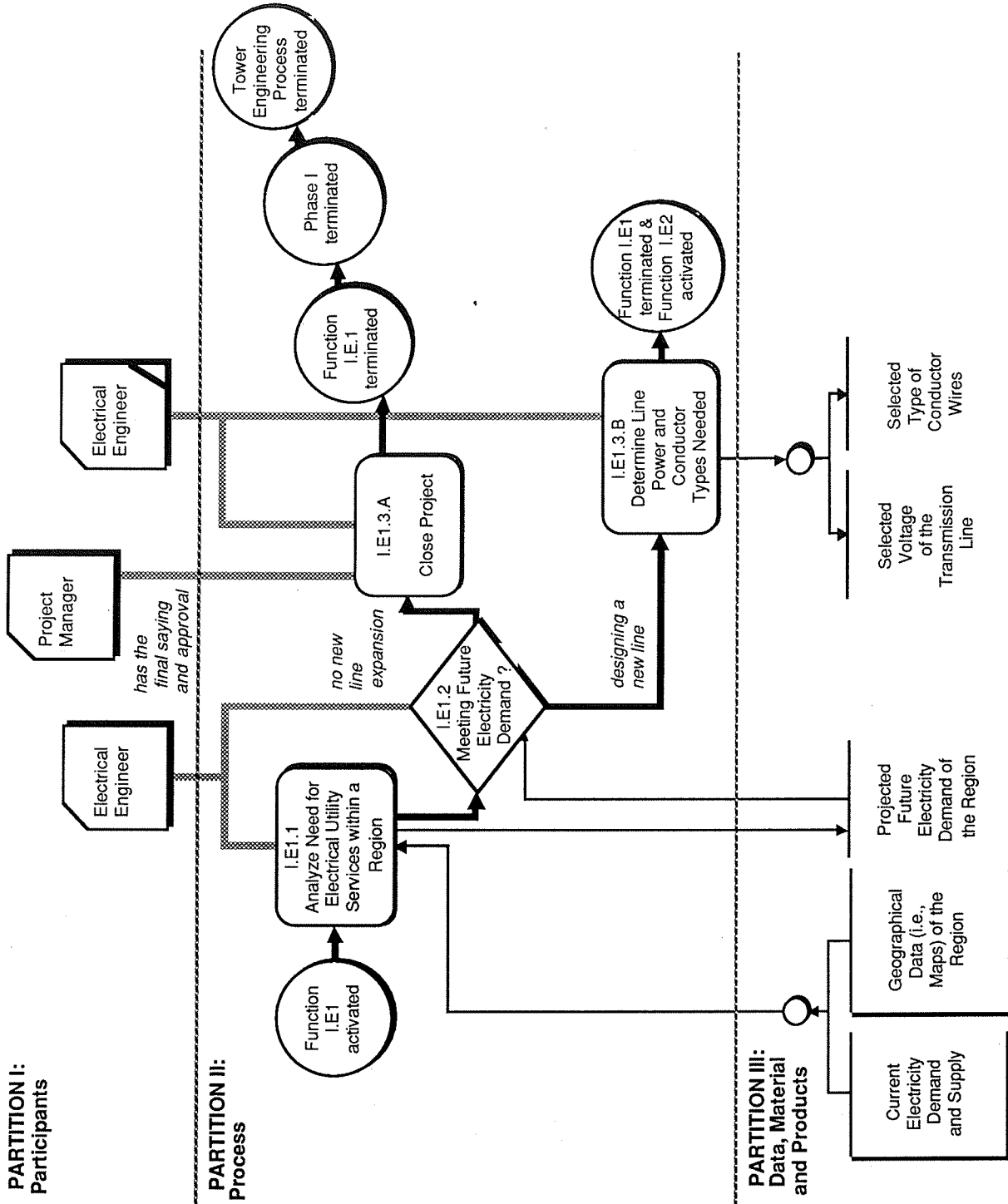


DIAGRAM I.1: Function I.E1, Need Analysis, of Phase I (Transmission Line Analysis and Design). (See Diagram Note C)



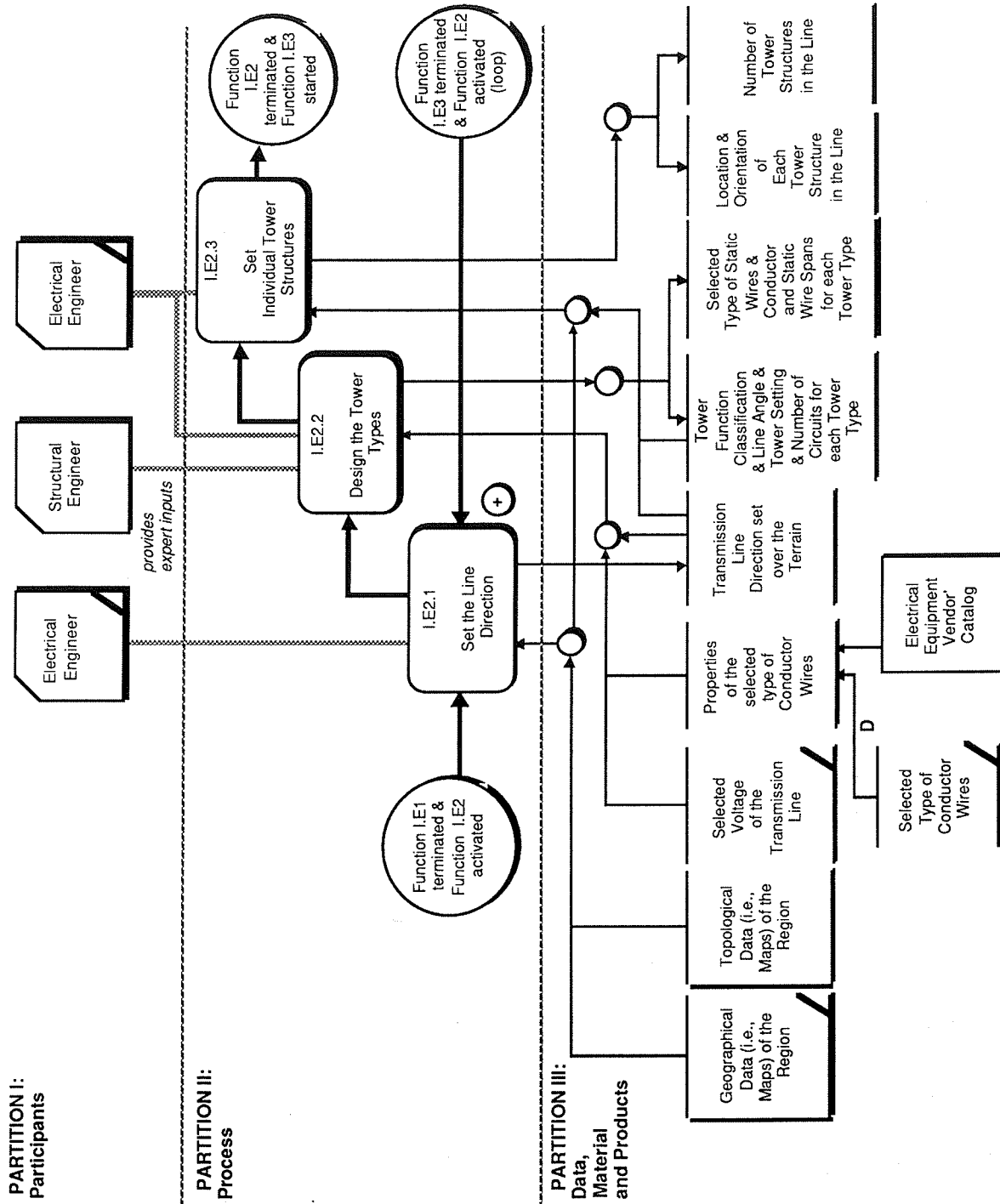
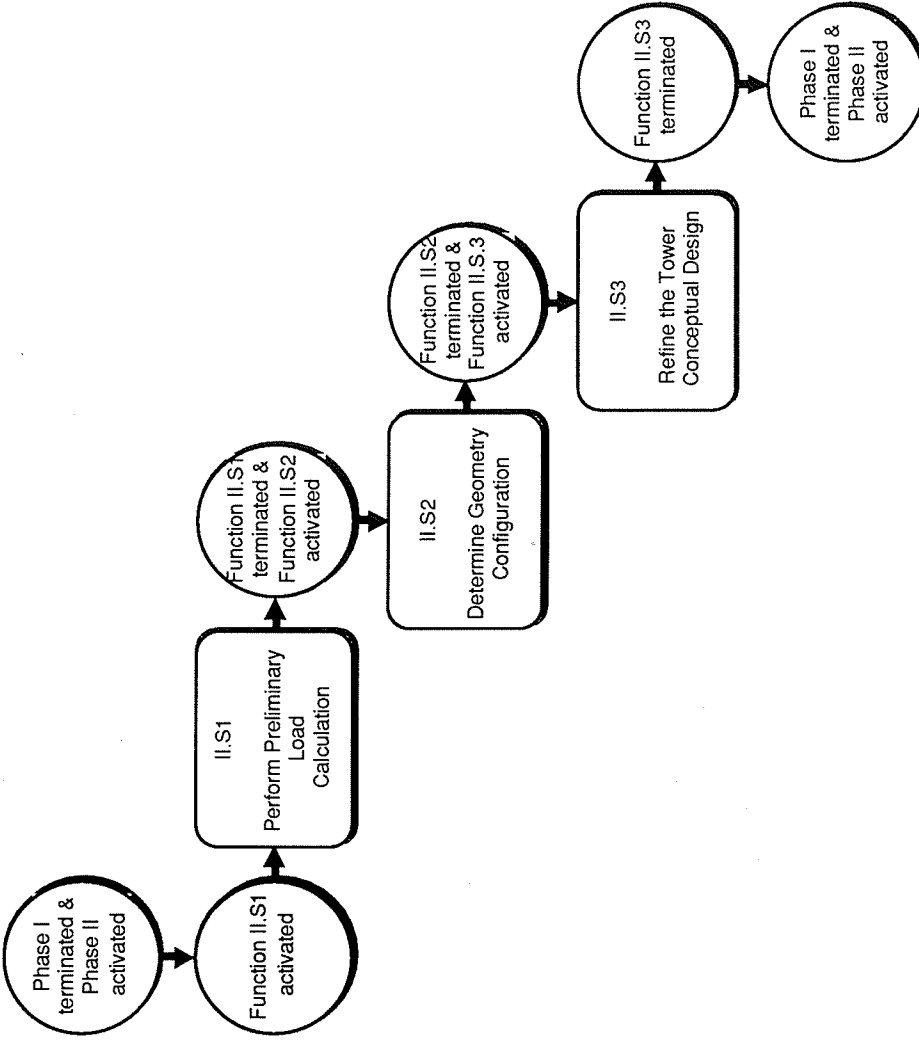


DIAGRAM 1.2: Function I.E2, Line Layout and Design, of Phase I (Transmission Line Analysis and Design).  
(See Diagram Note C)



**PARTITION II: PROCESS**



**DIAGRAM 2: Intermediate Skeleton Graphical Functional Schema of Phase II, Tower Structural Conceptual Design.**

(See Diagram Note B)

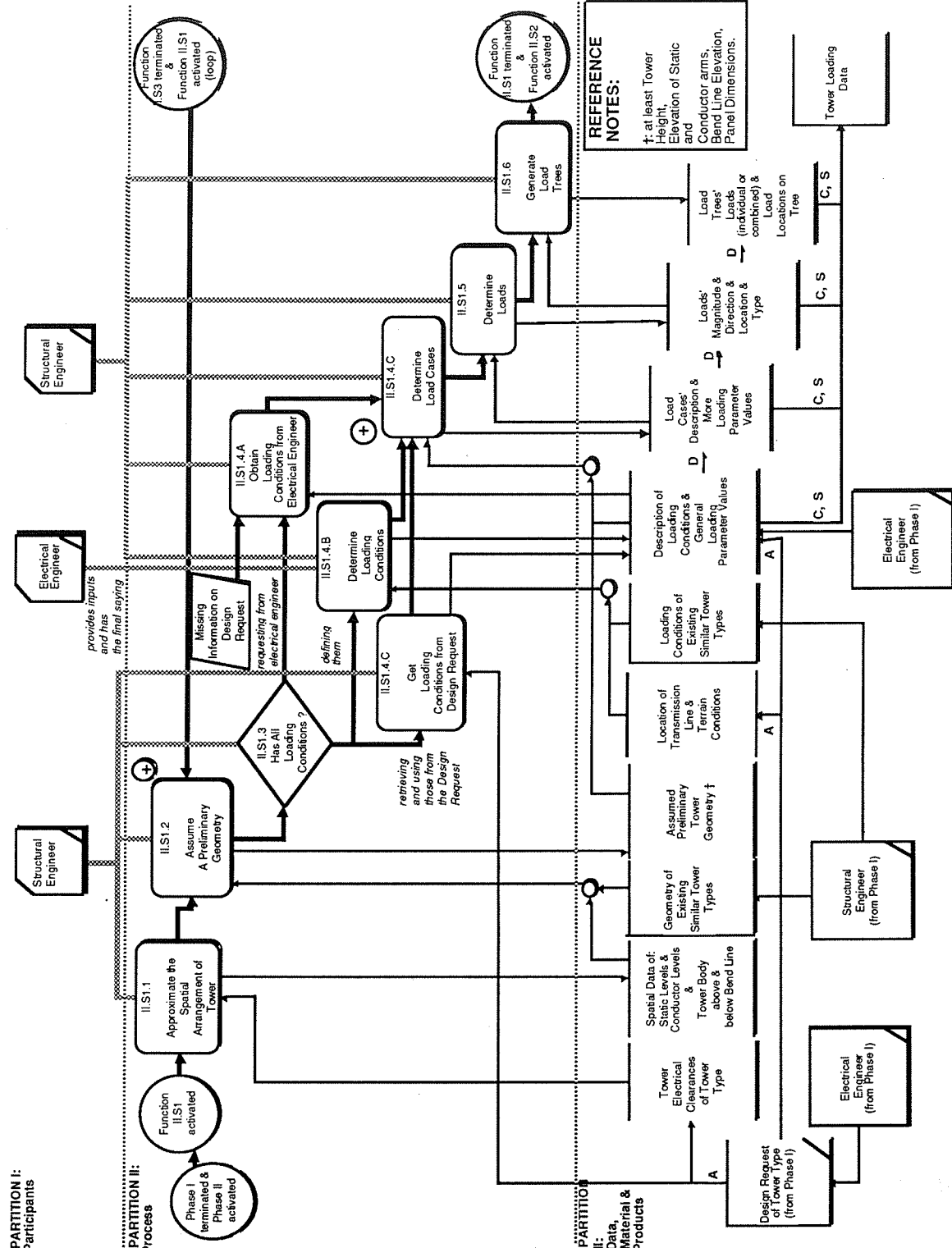


DIAGRAM 2.1: Function II.S1, Preliminary Load Computation, of Phase II (Tower Conceptual Structural Design). (See Diagram Note C)

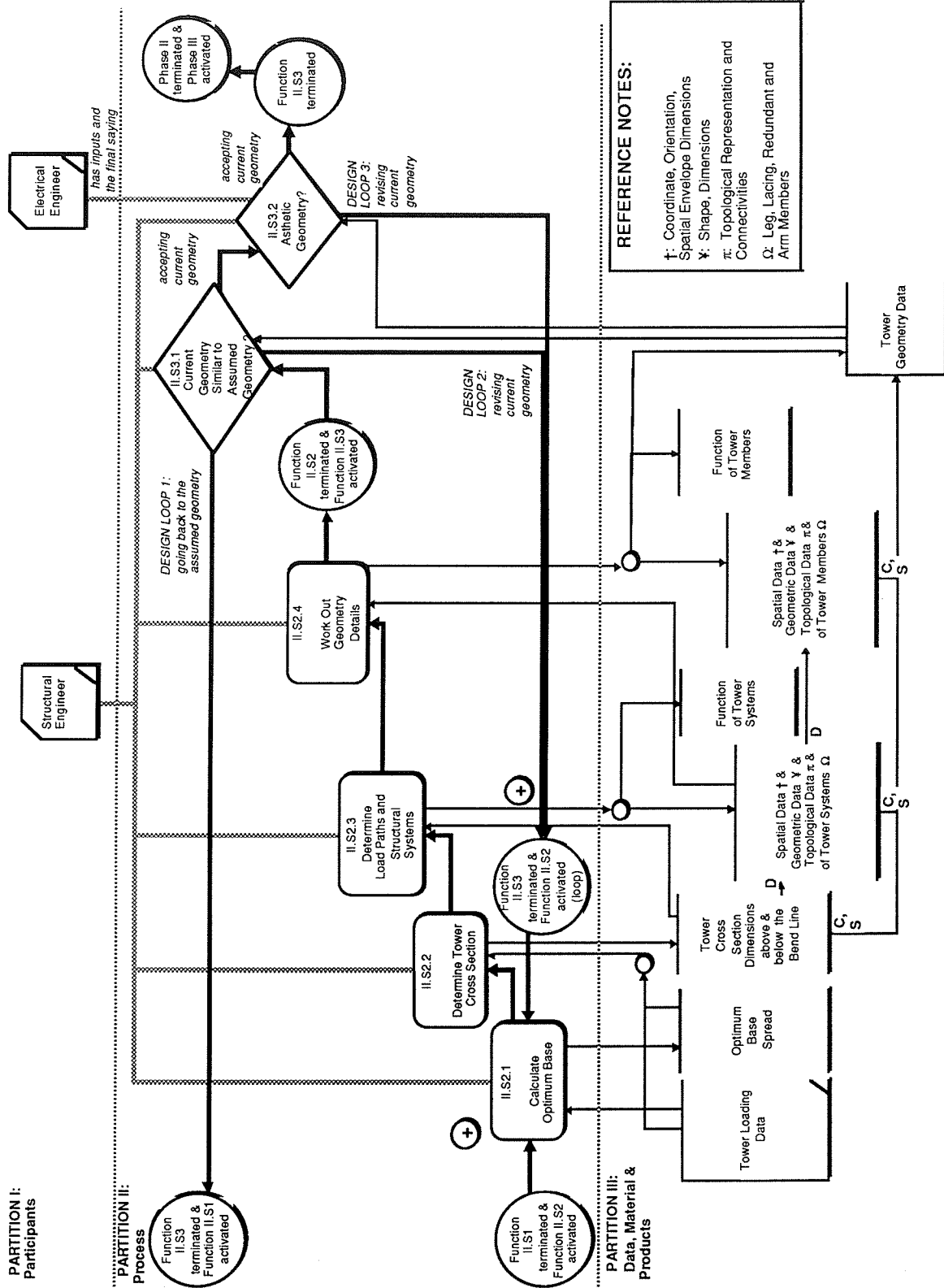


DIAGRAM 2.2: Function II.S2, Geometry Configuration, and Function II.S3, Conceptual Design Refinement (Phase II).

(See Diagram Note C)

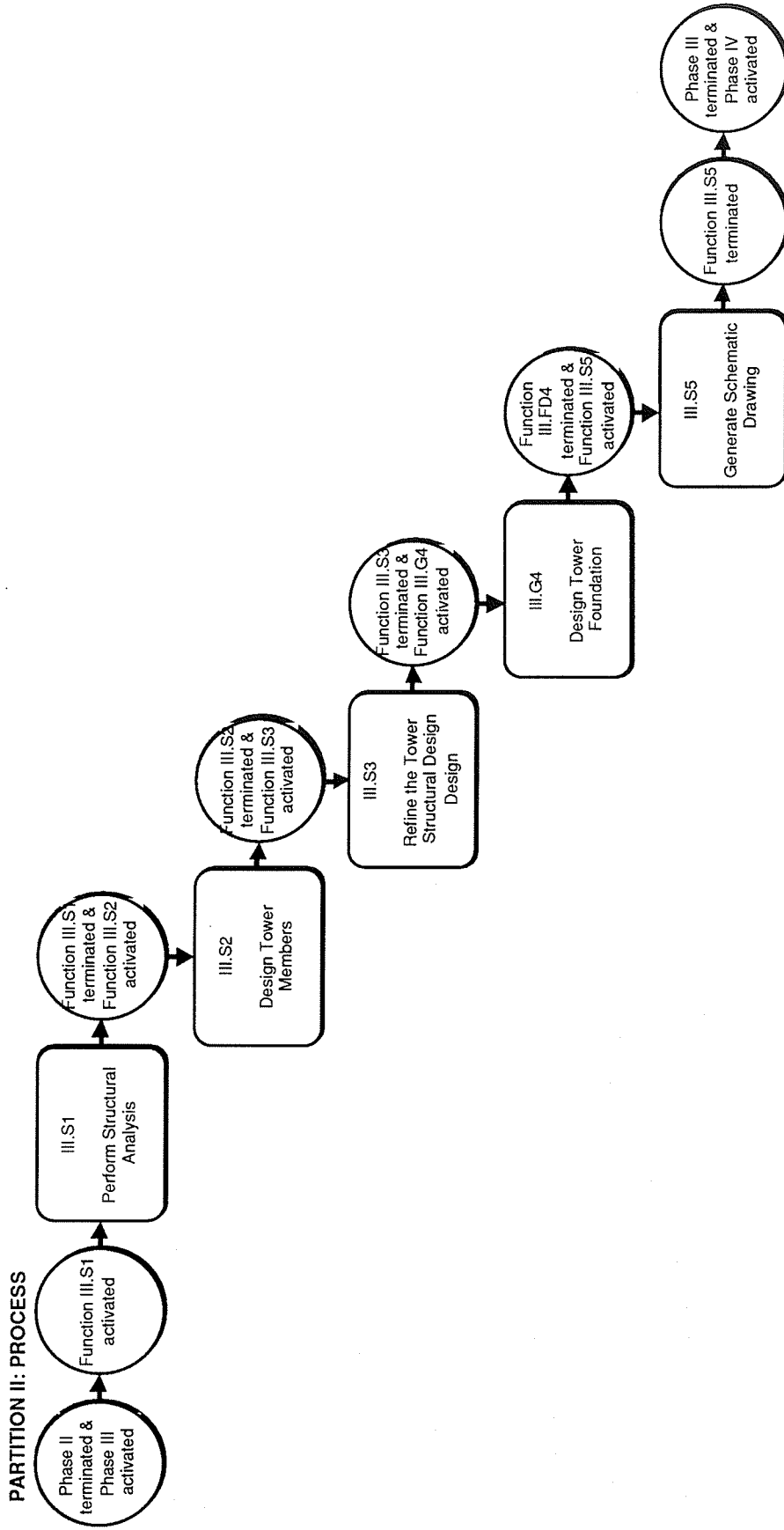


DIAGRAM 3: Intermediate Skeleton Graphical Functional Schema of Phase III, Tower Structural Detailed Design. (See Diagram Note B)

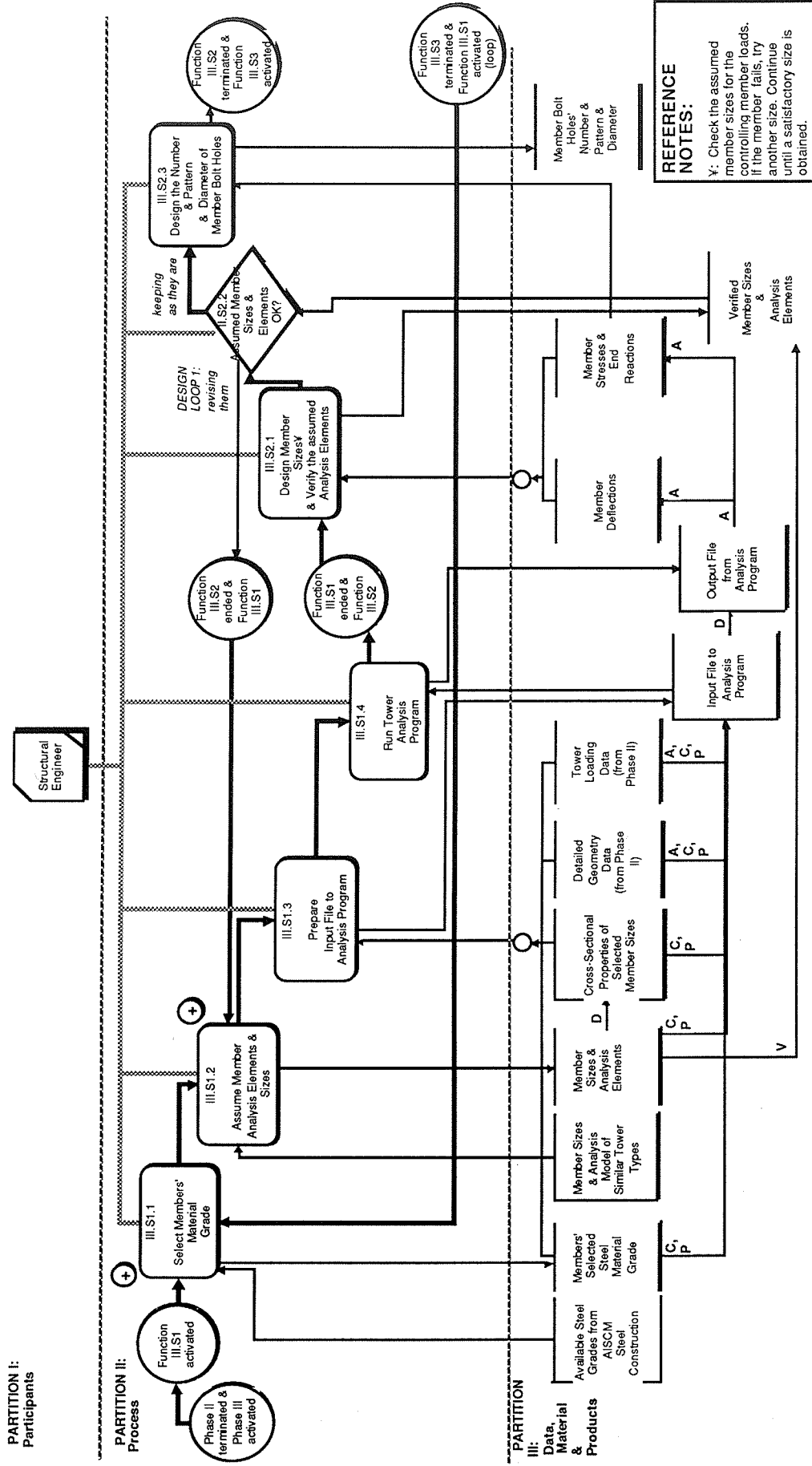
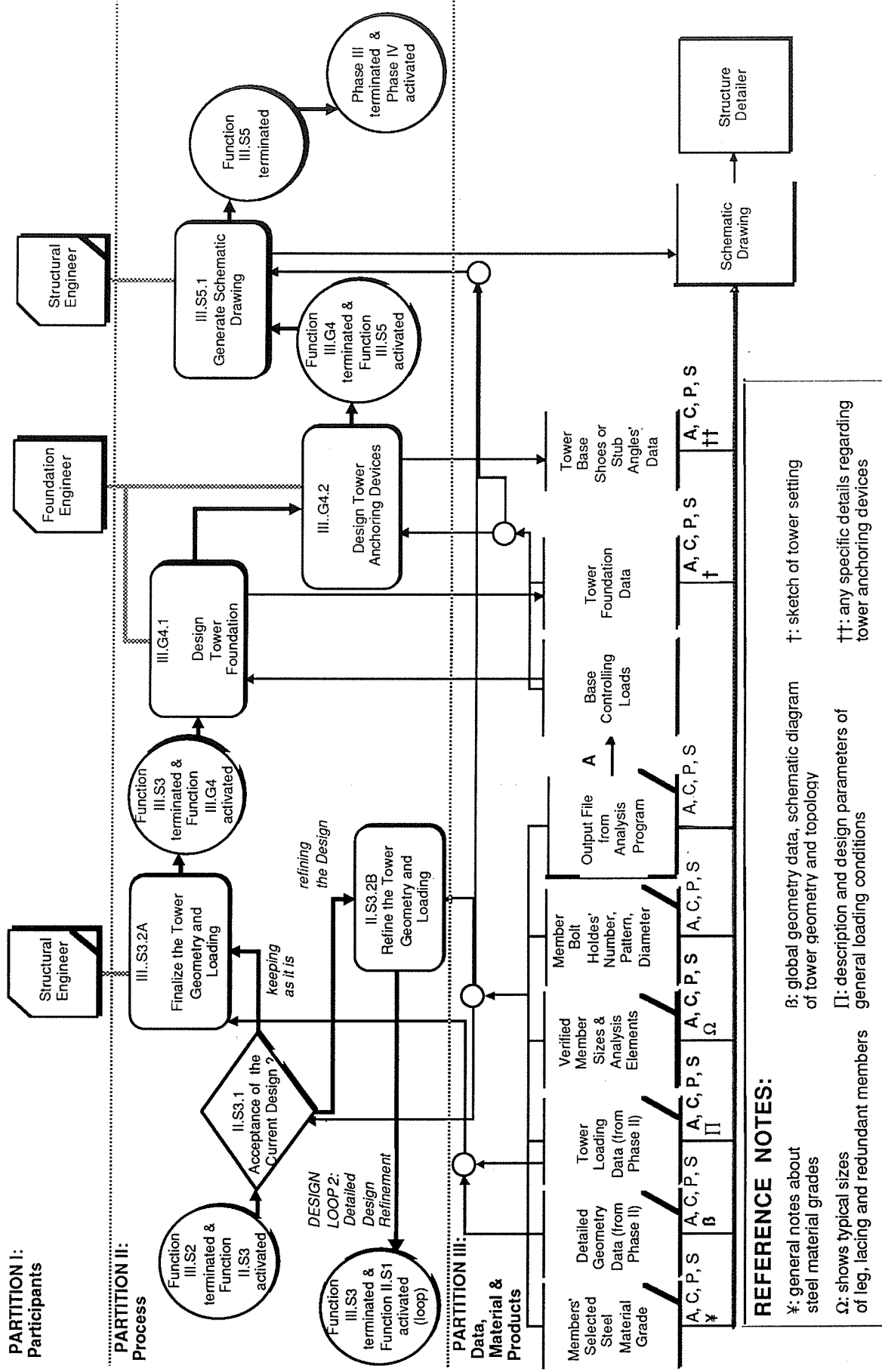


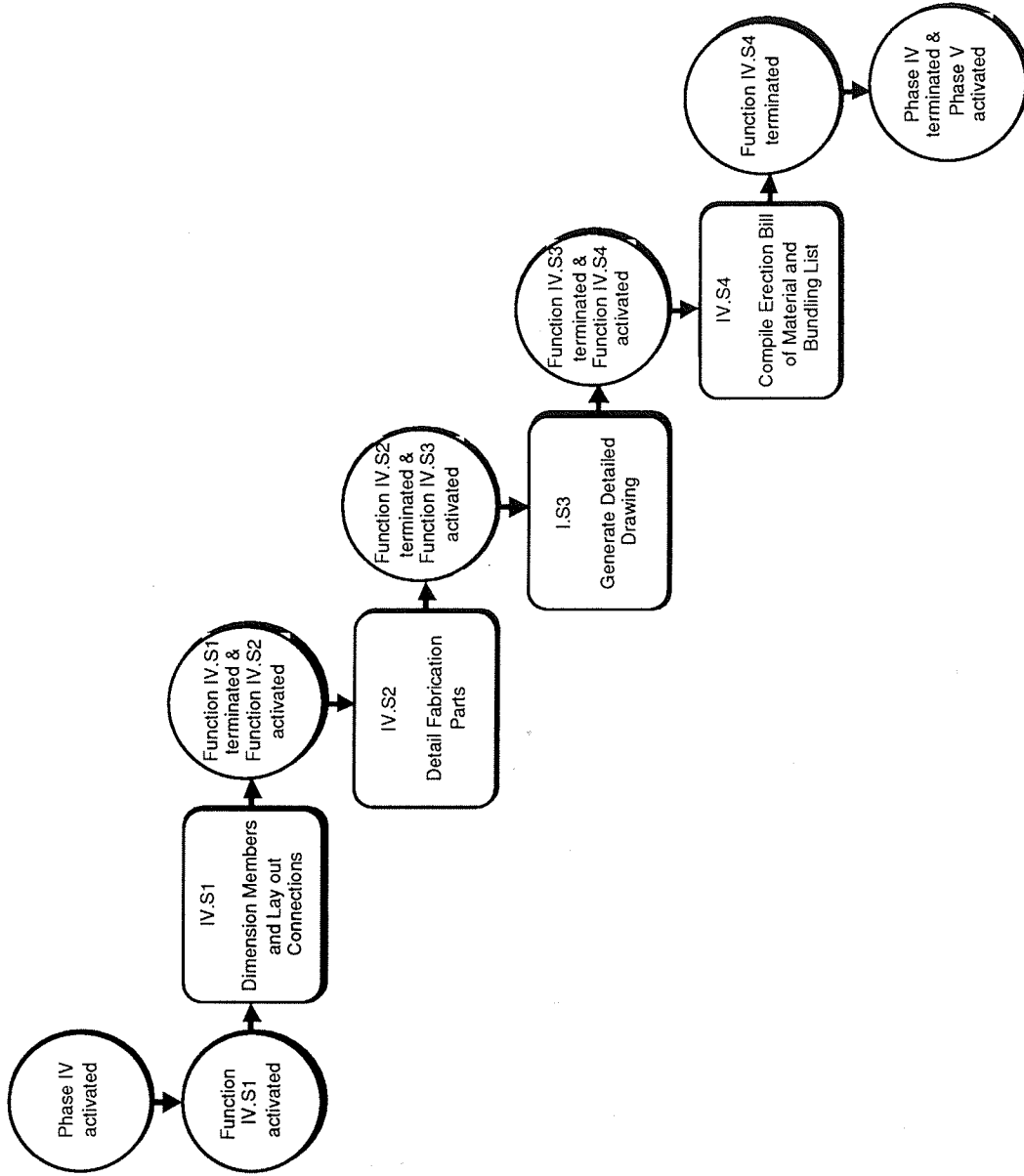
DIAGRAM 3.1: Function III.S1, Structural Analysis, and Function III.S2, Member Design, of Phase III (Tower Structural Detailed Design). (See Diagram Note C)



**DIAGRAM 3.2: Function III.S3, Design Refinement, and Function III.FD4, Foundation Design, and Function III.S5, Generating Schematic Drawing, of Phase III (Tower Structural Detailed Design).** (See Diagram Note C)



**PARTITION II: PROCESS**



**DIAGRAM 4: Intermediate Skeleton Graphical Functional Schema of Phase IV, Tower Construction Planning.**  
(See Diagram Note B)

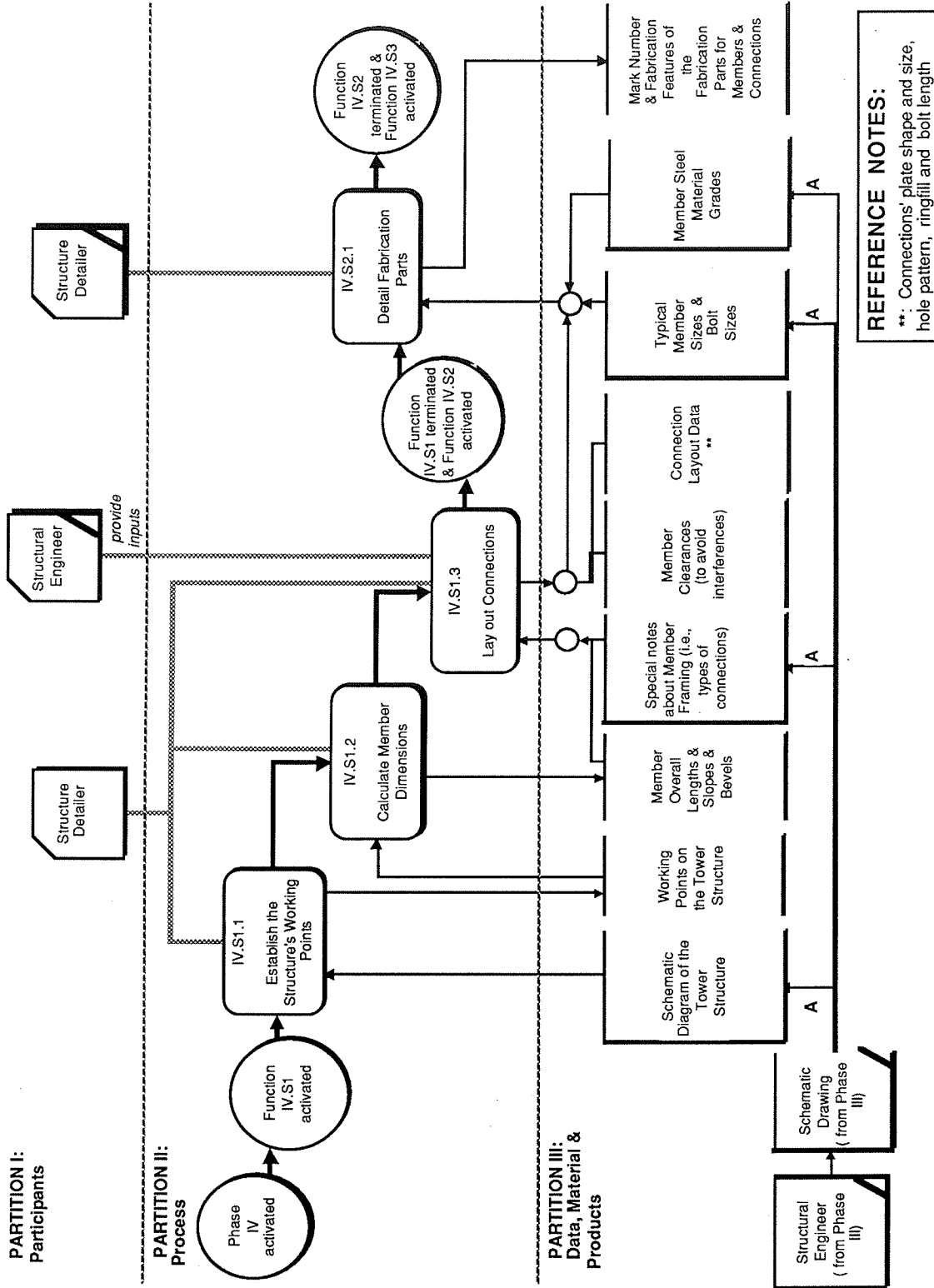


DIAGRAM 4.1: Function IV.S1, Dimensioning Members and Laying out Connections, and Function IV.S2, Detailing Fabrication Parts, of Phase IV (Tower Construction Planning). (See Diagram Note C)

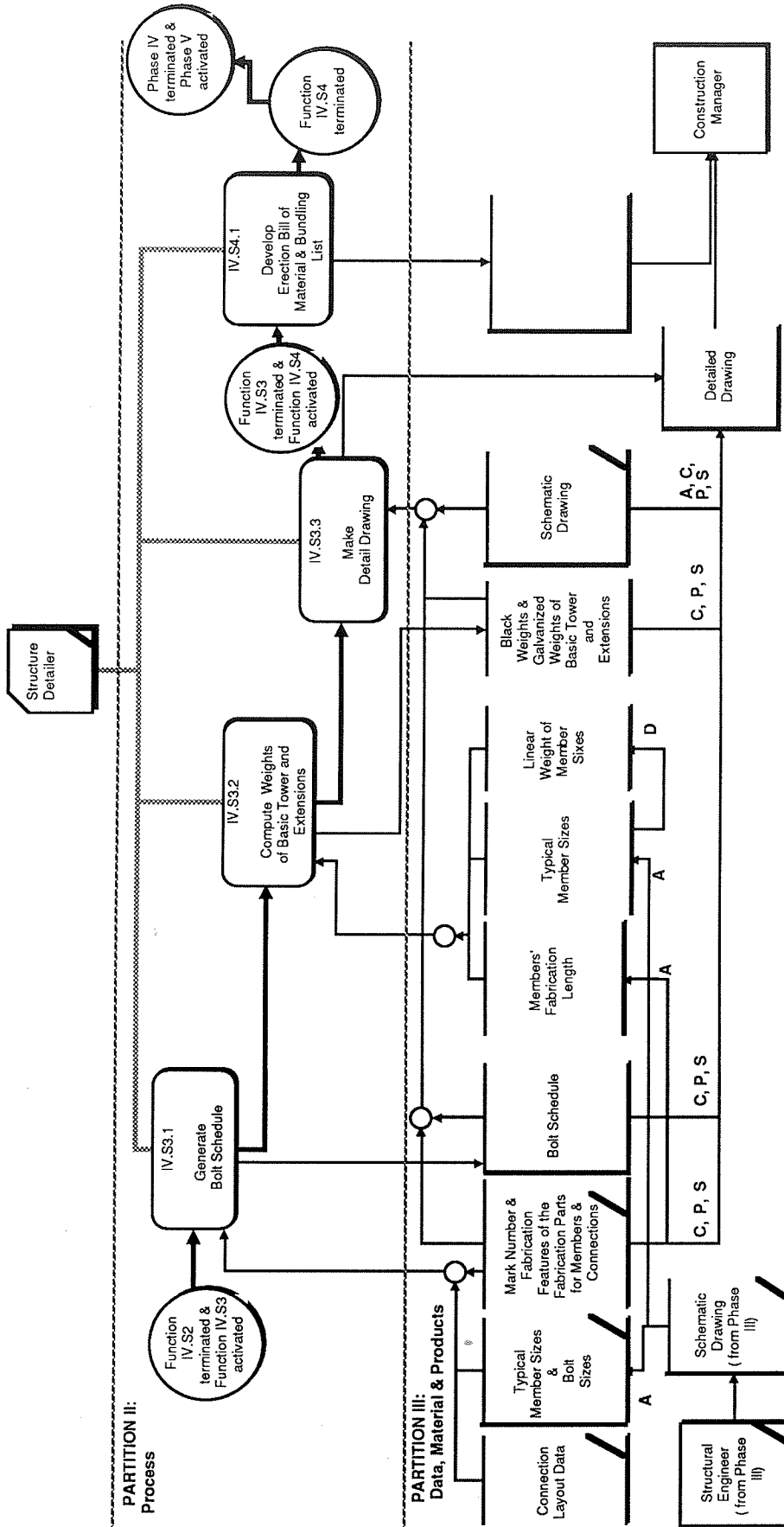
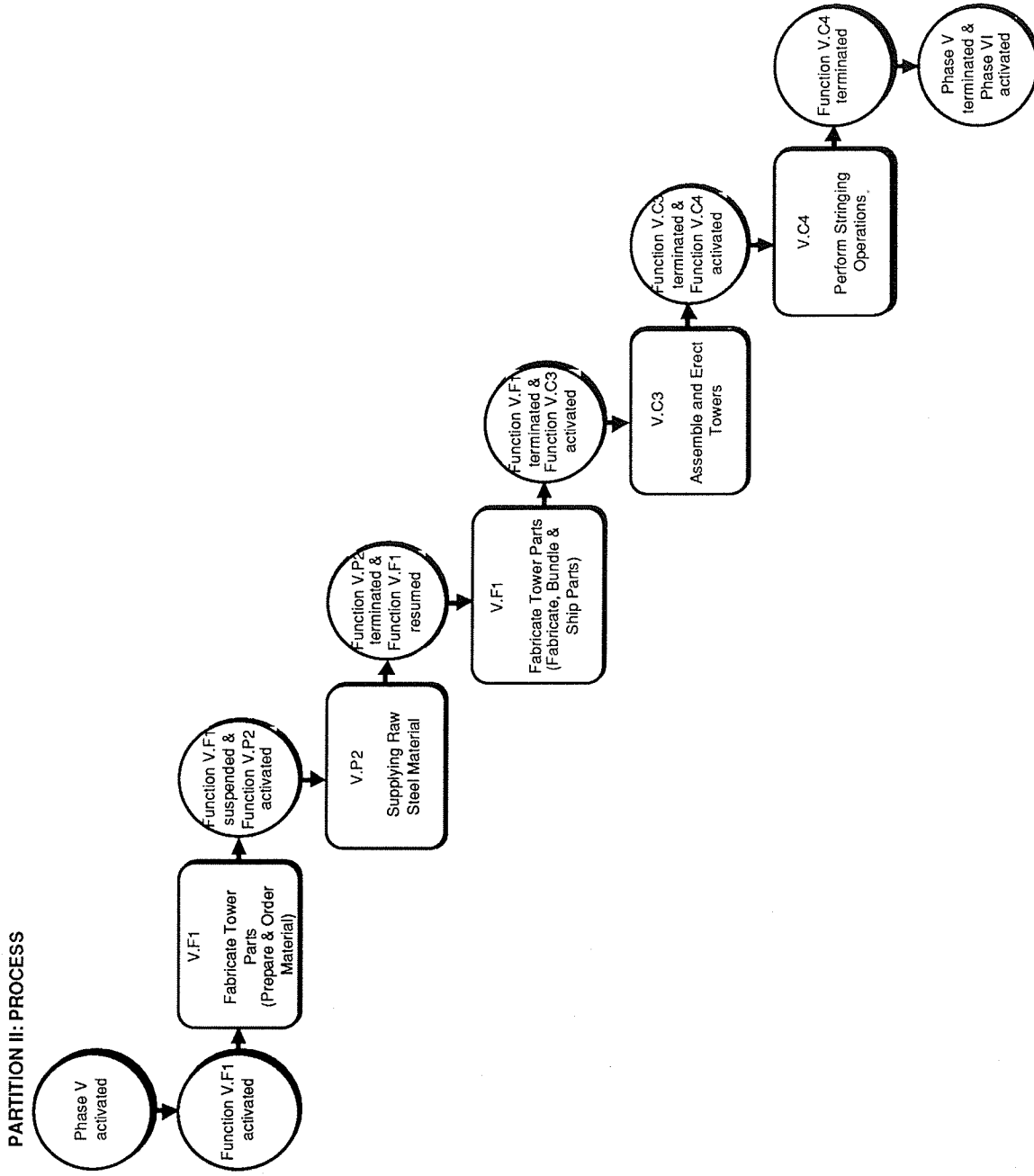


DIAGRAM 4.2: Function IV.S3, Generating Detailed Drawing, and Function IV.S4, Compiling Erection Bill of Material and Bundling List, of Phase IV (Tower Construction Planning). (See Diagram Note C)



**DIAGRAM 5: Intermediate Skeleton Graphical Functional Schema of Phase V, Tower Construction Execution.** (See Diagram Note B)

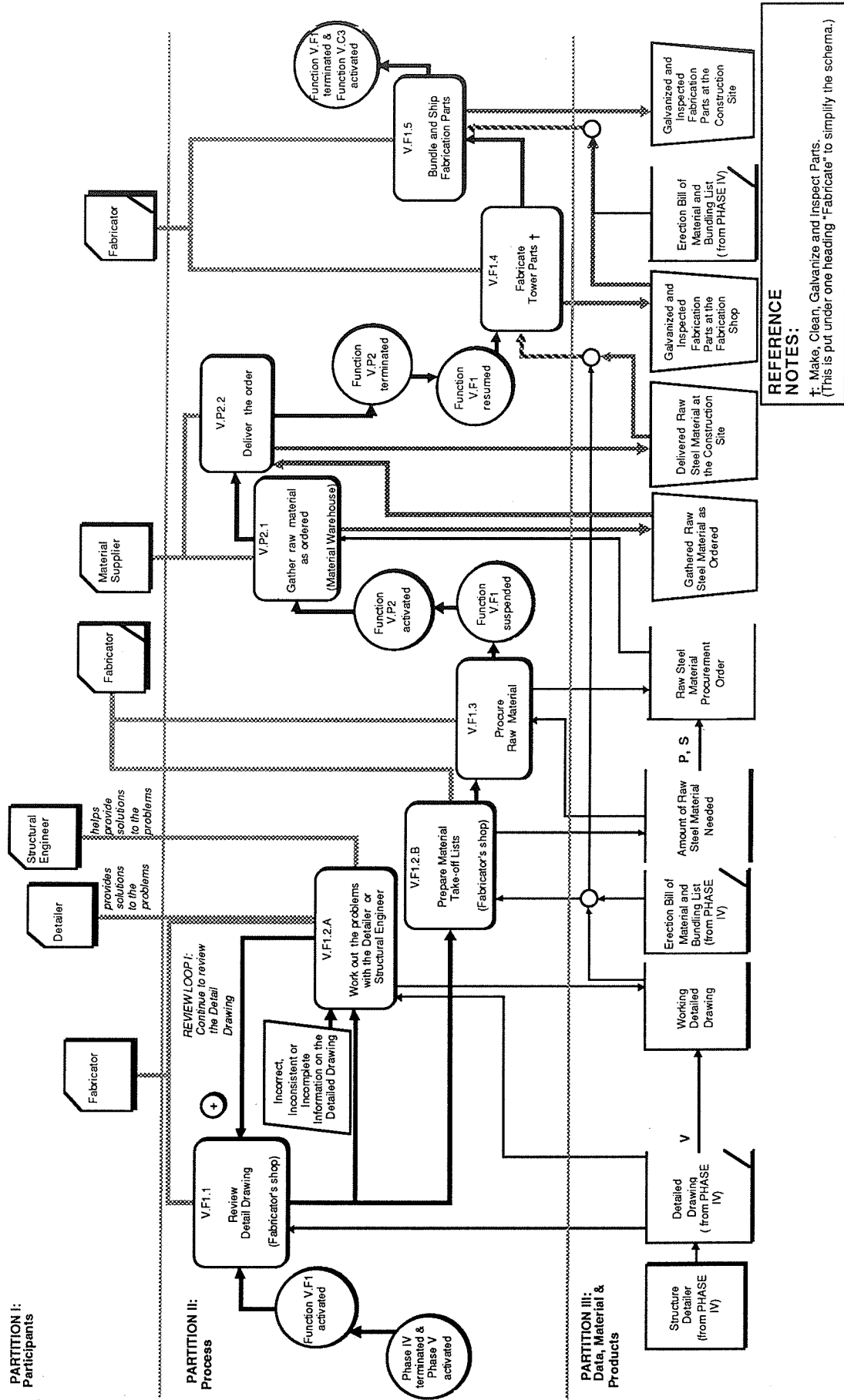


DIAGRAM 5.1: Function V.F1, Parts Fabrications, and Function V.P2, Supplying Material, of Phase V (Tower Construction Execution). (See Diagram Note C)

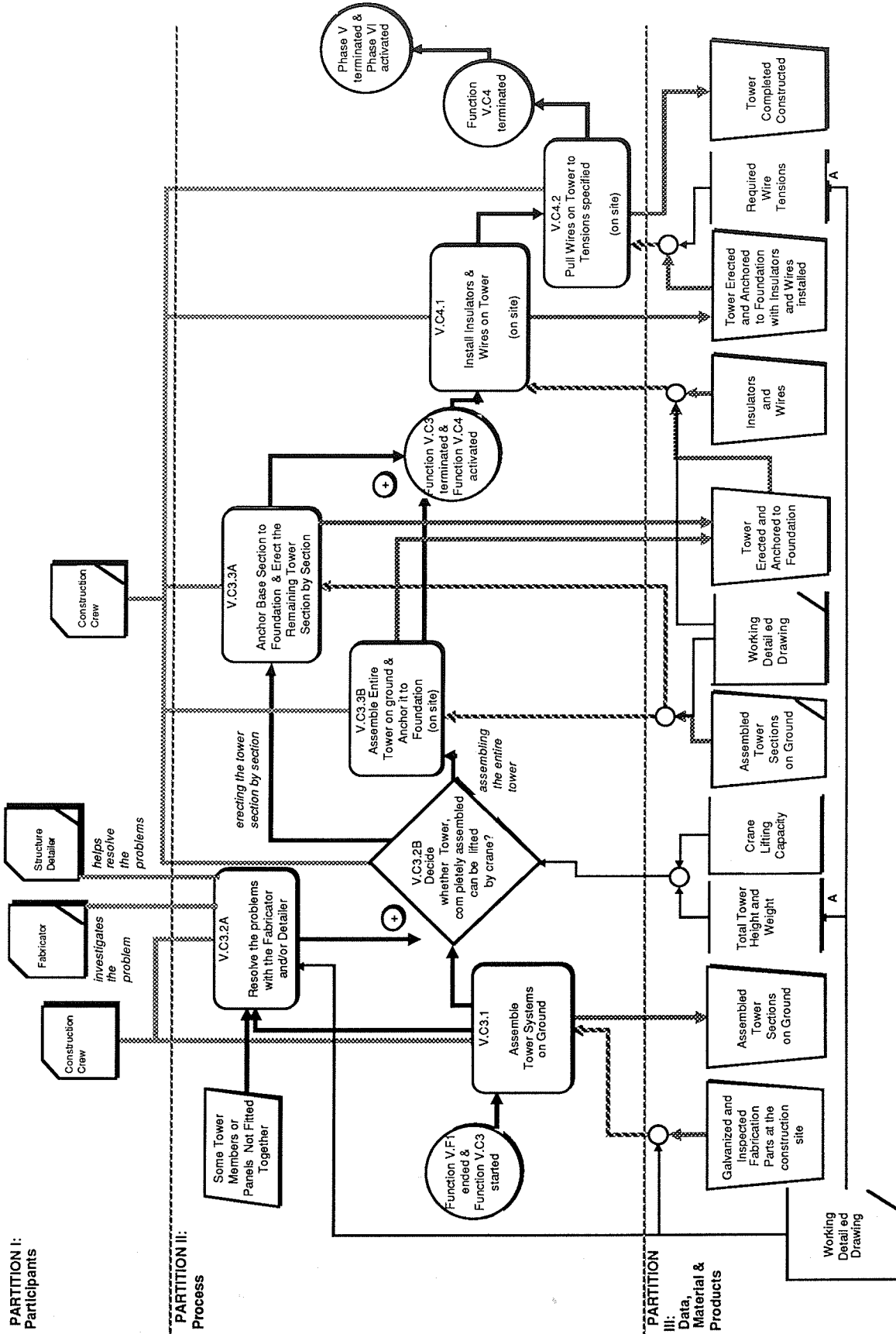


DIAGRAM 5.2: Function V.C3, Tower Assembly and Erection, of Phase V (Tower Construction Execution).

(See Diagram Note C)

# **SUMMARY & CONCLUSIONS**

To close the report, we summarize the preceding five chapters and present our conclusions. We also acknowledge the people that have made this research possible.

## **SUMMARY OF THE REPORT**

**CHAPTER 1 — INTRODUCTION** Functional analysis is the study of information flow among the activities of a process or processes in an enterprise. Its purpose is to understand what information is used in each activity and how that information is exchanged among the activities. Functional analysis is particularly important for representing data and developing databases in facility engineering. It helps in the understanding of the data needs of the engineering process, in improvements of the design requirements and design of the database, in verification and testing of the database design, and in support of data integration. Functional analysis is also highly relevant to our research. The goal of the research is to develop a general approach, the Primitive-Composite (or P-C) Approach, to the conceptual modeling of facility engineering data, and to the data integration support of many participants, life-cycle phases, and computer applications.

The main objective of the work described in this report was to develop a reference model for functional analysis that supports our research goal. The needed reference model should provide the concepts, rules and operations needed to do the job as well as a methodology and guidelines for using those concepts, rules and operations.

**CHAPTER 2 — PROBLEM DEFINITION** The problem was further defined by the capabilities that the reference model should provide and the properties it should have. The three required capabilities are: (1) representing the participants and their participation (i.e., the capacities in which they are involved), (2) representing complicated, non-linear processes of facility engineering, and (3) representing complex data flow, as well as physical flow, in the process and the types of operations performed to generate data. The required properties are: being more formal than natural languages, having a graphical representation, being capable to produce highly readable graphical functional schemata, and being as simple as possible and easy to use. These capabilities and properties were used to evaluate existing models and guide our development. They all have

valid justifications. The capabilities were itemized into features that we used in developing the reference model.

**CHAPTER 3 — BACKGROUND AND EVALUATION OF SELECTED MODELS** We briefly surveyed models that have been proposed for all purposes of modeling processes. These models stemmed from different areas of research and development. They differ vastly in terms of development objectives and emphasis, concepts, methodology, degree of formalization, etc. Less work has been done in the facility engineering domain. Moreover, most of what has been done emphasized the accurate depiction of the process itself rather than the development of a reference model for analyzing and representing the process. In other words, no “reference model” had been developed for facility engineering processes.

From this background study, we selected five candidate models for evaluation. The Data Flow model is one of them. The evaluation criteria used came directly from the required capabilities and properties in Chapter 1. The results were tabulated, and the performance of the models under each of the criteria was discussed. All five models offered advantages and shortcomings. However, we selected the Data Flow model because it provides the basic features required, has a graphical representation, and is simple to learn and use. We used it as the core of our model, extending it to meet our requirements. Our extension is the PARTitioned eNginEering DATA flow model (abbreviated as PANDA).

**CHAPTER 4 — THE EXTENDED MODEL: PANDA** This is an extended data flow model with a multi-leveled partitioned architecture. The analyst can use PANDA to functionally decompose a process into many hierarchical levels of description. At the detailed level, the data flow diagram has three major partitions: Participants, Process and Data-Material-Products. The diagram is also called the Partitioned Data Flow diagram, or P-diagram. This unique architecture serves two main purposes: It helps organize the analyst’s thinking about complicated processes and enhances both the conceptual readability and graphical readability of the process’ functional schema.

In our extension, we kept the concepts from the original Data Flow model: activity, data repository, data flow, and data source or sink. We added the concepts of participant, participation, precedence relationship, decision, alternative, interference, subprocess, boundary, data item, data generation, material or product, physical flow, mixed flow and flow network. We then structured all of these according to the major partitions. We gave graphical representations of the concepts. In addition, we defined syntactic and semantic rules that govern the use of the



concepts. Several basic schema transformation operations were also provided to enable the analyst to develop a functional schema incrementally.

**CHAPTER 5 — USING PANDA** In this chapter, we showed how to use PANDA to represent facility engineering processes and draw graphical functional schemata. We first emphasized the importance of defining the scope of the analysis before any work begins. Defining the scope involves stating which disciplines, participants, phases, data and data sources and sinks are to be included in and excluded from the analysis, and specifying as much as possible the extent to which the analyst should analyze these categories. Next, we presented a methodology for doing functional analysis using PANDA. This methodology uses a mixed two-pass strategy, which involves top-down functional decomposition and bottom-up functional refinement in two successive passes. It takes advantage of the partitioned architecture of the model by giving different priorities to the partitions at different stages of analysis. In addition, a number of detailed guidelines for using PANDA's concepts such as subprocess, boundary, and decision were provided. More guidelines were given for drawing Partitioned Data flow diagrams: labeling nodes, numbering activity and decision nodes, laying out subprocesses, and annotating diagrams. We also provided a check list for validating the functional schemata upon their completion.

**CHAPTER 6 — A CASE STUDY: ELECTRICAL TRANSMISSION TOWER FACILITY ENGINEERING** We presented a case study in which PANDA was used to model electrical utility transmission tower facility engineering. Electrical utility transmission towers are large lattice structures used for electrical power transmission. The complex tower engineering process involves several development stages, multiple participants from various disciplines, and computer applications that are used to automate structural analysis, member design, etc. In the case study, we examined the entire process, but put less emphasis on the initial electrical design and final facility management stages than other stages (e.g. tower structural engineering design and tower construction). In tower structural design, we focused on designing new tower structures rather than on retrofitting existing structures. The process was first decomposed into six phases: Transmission Line Analysis and Design, (2) Tower Structural Conceptual Design, (3) Tower Structural Detailed Design, (4) Tower Construction Planning, (5) Tower Construction Execution and (6) Tower Facility Management. Each phase was then divided into several functions, each of which is a group of coherent activities that help fulfill a distinct objective. The entire process was then described in detail. Each phase was explained in terms of its time of occurrence, key project participants, place, work involved and goal, new engineering terms and concepts introduced, functions to which the phase is decomposed, and end results. Finally, the graphical functional schemata of the process that result from using PANDA were presented.

**CONCLUSIONS** The following are the lessons that we learned from this development effort:

- *Functional analysis is crucial to the development of information systems in facility engineering.* To build an information system that can effectively support data communication among the principal participants of a facility engineering process, the designer must have a clear understanding of the process in general and the nature of cooperative work of those participants in particular. Functional analysis can provide the designer with that understanding. In addition, it could add to the development of the information system many other benefits, some of which were mentioned in the preface.
- *Having a proper model for functional analysis in facility engineering is important.* The model must be capable of representing multiple participants, non-linear subprocesses, design synthesis loops, decisions and alternatives, interferences, as well as complicated data, material and product flow networks. These are essential characteristics of facility engineering processes.
- *A useful model for functional analysis in facility engineering must have graphical representations and built-in features that would automatically produce highly readable graphical functional schemata.* Without these properties, the model might not be used.
- *A methodology and guidelines for using a suggested model improves the likelihood that the model will be used effectively.* Such a methodology and guidelines are very useful for the designer when considering use of the model, when learning the model or when actually using it.
- *Support software for functional analysis in facility engineering is definitely needed.* Functional analysis in any facility engineering domain can be time- and effort-consuming. Our case study took more than six months. Computer-Aided Software Engineering (CASE) tools that can assist the designer in doing functional analysis using a suggested model are necessary and can make possible the successful use of the model.

PANDA is the outcome of what we have learned and developed. Its objective was to meet the need for a reference model for functional analysis in support of the development of facility engineering information systems. With PANDA, the designer can perform functional analysis for a complicated facility engineering process. Being an extension of the Data Flow Model, which has been a popular choice for functional analysis, PANDA provides the concepts necessary to

analyze facility engineering processes while adhering as much as possible to the simplicity and ease of use of the original model. Using the graphical representations of PANDA, the designer can draw diagrams illustrating the three major aspects of the process: (1) participants, (2) subprocesses and activities, and (3) data, material and products. These diagrams provide a structured and concise means to describe and communicate about the process. With the three built-in partitions, each diagram is highly readable, both conceptually and graphically. Overall, the model's unique partitioned architecture helps the designer organize his or her thinking about a complicated engineering process by focusing on different aspects at various times during the functional analysis. Further, PANDA offers a customized methodology that benefits from the model's partitioned architecture and guides the designer in applying the model to his or her domain problem. PANDA also provides guidelines for using specific concepts of the model, validating the resulting functional schemata and drawing the partitioned data flow diagrams of those schemata. In the future, we plan to apply PANDA to other facility engineering domains. The experience gained will help us further enhance the model.

**ACKNOWLEDGMENTS** This research was supported by a grant from the Center for Integrated Facility Engineering (CIFE) at Stanford University. We also would like to extend our special thanks to the domain experts that we interviewed for their full support and cooperation, Dr. Martin Fischer for his review of this report, and Dr. William Rasdorf for his encouragement and input to this work.

## REFERENCES

- [AISC 89] American Institute of Steel Construction (AISC), Inc., Manual of Steel Construction — Allowable Stress Design, Ninth edition, Chicago, IL, 1989.
- [Albano 83] Albano, A., Cardelli, L. and Orsini, R., “Galileo: A Strongly Typed, Interactive Conceptual Language,” Report 83-11271-2, Bell Laboratories, Murray Hill, N. J., July 1983. (Also appears in [Borgida 85])
- [ANSI 82] American National Standards Institute (ANSI), Inc., *Minimum Design Loads for Buildings and Other Structures*, ANSI, New York, NY, 1982.
- [ASCE 71] American Society of Civil Engineers (ASCE), Task Committee on Tower Design of the Committee on Analysis and Design of Structures of the Structural Division, *Guide for Design of Steel Transmission Towers*, Report No. 52, New York, NY, 1971.
- [Batini 92] Batini, C., Ceri, S., and Navathe, S., *Conceptual Database Design — An Entity-Relationship Approach*, The Benjamin/Cummings Publishing Company, Inc., Redwood City, CA, 1992.
- [Billmers 92] Billmers, M. and Adler, M. (AI Technology Center, Digital Equipment Corporation), “Micromodeling As A Tool for Enterprise Integration,” *Workshop Notes of the 1192 Workshop Program on AI in Enterprise Integration*, sponsored by the American Association for Artificial Intelligence, San Jose, July, 1992.
- [Borgida 85] Borgida, A., “Features of Languages for the Development of Information Systems at the Conceptual Level,” *IEEE Software*, Vol. 2, No. 1, January, 1985.
- [Bradshaw 92] Bradshaw, J. M, et al. (Research & Technology, Boeing Computer Services), “*eQuality: A Knowledge Acquisition Approach to Enterprise Integration*,” *Workshop Notes of the 1192 Workshop Program on AI in*

*Enterprise Integration*, sponsored by the American Association for Artificial Intelligence, San Jose, July, 1992.

- [Brandt 83] Brandt, I., "A Comparative Study of Information Systems Design Methodologies," In [Olle 83].
- [Bravoco 85a] Bravoco, R. R. and Yadav, Surya, B., "Requirement Definition Architecture—An Overview," *Computer in Industry*, Vol. 6., pp. 237-251, 1985. (Also appears in [Chadha 91])
- [Bravoco 85b] Bravoco, R. R. and Yadav, Surya, B., "A Methodology to Model the Information Structure of an Organization," *Computer in Industry*, Vol. 6., pp. 345-361, 1985. (Also appears in [Chadha 91])
- [Brodie 82] Brodie, M. L. and Silva, E., "Active and Passive Component Modeling: ACM/PCM," In T.W. Olle, H. G. Sol and A. A. Verrijn-Stuart (editors), *Information Systems Design Methodologies: A Comparative Review*, Proceedings of the CRIS-1 Conference, North-Holland, pp. 93-142, 1982.
- [Ceri 86] Ceri, S., "Requirements Collection and Analysis in Information Systems Design," *Proceedings of the IFIP Conference*, edited by H. J. Kugler, North-Holland, 1986.
- [Chadha 91] Chadha, B. et. al., "An Appraisal of Modeling Tools and Methodologies for Integrated Manufacturing Information Systems," *Proceedings of the Fifth Symposium on Engineering Databases: An Engineering Resource*, 1991 ASME International Computers in Engineering Conference, American Society of Mechanical Engineers, Santa Clara, CA, August, 1991.
- [Chandrasekaran 88] Chandrasekaran, B., "Design: An Information Processing Level Analysis," In D. Brown and B. Chandrasekaran, *Design Problem Solving: Knowledge Structures and Control Strategies*, Pitman, London, UK, 1988.
- [Chen 76] Chen, P. P. S., "The Entity-Relationship Model - Toward a Unified View of Data", *ACM Transactions on Database Systems*, Vol. 1, No. 1, pp. 9-36, March 1976.
- [De Marco 82] De Marco, T., *Structured Analysis and System Specification*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1982.

- [Ellis 79] Ellis, C. A., "Information Control Nets: A Mathematical Model of Office Information Flow," *Proceedings of ACM Conference on Simulation Modeling and Measurement of Computer Systems*, 1979.
- [EPRI 87] Electric Power Research Institute (EPRI), "Guidelines for Specifying Integrated Computer-Aided Engineering Applications for Electric Power Plants," *Final Report No. EPRI NP-5259M*, Project 2514-3, May, 1987.
- [Fox 92] Fox, M. (Department of Industrial Engineering, University of Toronto), "The TOVE Project: Toward A Common-Sense Model of the Enterprise," *Workshop Notes of the 1192 Workshop Program on AI in Enterprise Integration*, sponsored by the American Association for Artificial Intelligence, San Jose, July, 1992.
- [Gane 79] Gane, C. and Sarson, T., *Structured System Analysis*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1979.
- [Greenspan 86] Greenspan, S. J., Borgida, A., Mylopoulos, J., "A Requirements Modeling Language and its Logic," *Information Systems*, Vol. 11, No. 1, pp. 9-23, 1986.
- [Grosf 92] Grosf, B. and Morgenstern, L. (T. J. Watson Research Center, IBM), "Applications of Logicist Knowledge Representation to Enterprise Modelling," *Workshop Notes of the 1192 Workshop Program on AI in Enterprise Integration*, sponsored by the American Association for Artificial Intelligence, San Jose, July, 1992.
- [Gruber 92] Gruber, T., Tenenbaum, J. and Weber, J., (Stanford Knowledge Systems Laboratory, EIT, Inc. and Lockheed AI Center), "Toward a Knowledge Medium for Collaborative Product Development," *Workshop Notes of the 1192 Workshop Program on AI in Enterprise Integration*, sponsored by the American Association for Artificial Intelligence, San Jose, July, 1992.
- [Gustavsson 82] Gustavsson, M. R., Karlsson, T., Bubenko, J. A., A Declarative Approach to Conceptual Information Modeling, In T.W. Olle, H. G. SOl and A. A. Verriijn-Stuart (editors), *Information Systems Design Methodologies: A Comparative Review*, Proceedings of the CRIS-1 Conference, North-Holland, pp. 93-142, 1982.

- [Inmon 86] Inmon, W. H., *Information Systems Architecture — A System Developer's Primer*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1986.
- [Jagannathan 92] Jagannathan, V. et al. (Concurrent Engineering Research Center, West Virginia University), "Information Sharing System," *Workshop Notes of the 1192 Workshop Program on AI in Enterprise Integration*, sponsored by the American Association for Artificial Intelligence, San Jose, July, 1992.
- [Lockemann 86] Lockemann, P. C., *Information System Design: Techniques and Software Support*," *Proceedings of the IFIP Conference*, edited by H. J. Kugler, North-Holland, 1986.
- [Lundeberg 82] Lundeberg, M., "The ISAC Approach to Specification of Information Systems and its Application to the Organization of an IFIP Working Conference," In T.W. Olle, H. G. SOI and A. A. Verrijn-Stuart (editors), *Information Systems Design Methodologies: A Comparative Review*, Proceedings of the CRIS-1 Conference, North-Holland, 1982.
- [Luth 91] Luth, G. P., *Representation and Reasoning for Integrated Structural Design*, Ph.D. Thesis, Department of Civil Engineering, Stanford University, Stanford, CA, June, 1991.
- [Mayer 92] Mayer, R. J. et al., "Information Integration For Concurrent Engineering (IICE) IDEF3 Process Description Capture Method Report," Technical Report AL-TR-1992-0057, Armstrong Laboratory, May 1992.
- [Mylopoulos 80] Mylopoulos, J., Bernstein, P. A., and Wong, H. K. T., "A Language Facility for Designing Database-Intensive Applications," *ACM Transactions on Database Systems*, Vol. 5, No.2, pp. 185-207, June 1980.
- [Olle 83] T.W. Olle, H. G. SOI and C. J. Tully (editors), "Information Systems Design Methodologies: A Feature Analysis," *Proceedings of the IFIP WG 8.1 Working Conference on Feature Analysis of Information Systems Design Methodologies*, York, United Kingdom, July 5-7, North-Holland, 1983.
- [Phan 92] Phan, D. H., Appendix D: Application Domain — Electrical Utility Transmission Towers, Department of Civil Engineering, Stanford University, Stanford, CA, 1992.

- [Peterson 77] Peterson, J. L., "Petri Nets," *Computing Surveys*, Vol. 9, No. 3, pp. 223-252, September, 1977.
- [Richter 82] Richter, G. and Durchholz, R., "IML-Inscribed High-Level Petri Nets," In T.W. Olle, H. G. SOI and A. A. Verrijn-Stuart (editors), *Information Systems Design Methodologies: A Comparative Review*, Proceedings of the CRIS-1 Conference, North-Holland, 1982.
- [Ross 77a] Ross, D. and Shoman, K., "Structured Analysis for Requirements Definition," *IEEE Transactions on Software Engineering*, Vol. SE-3, No. 1, pp. 6-15, January, 1977.
- [Ross 77b] Ross, D., "Structured Analysis (SA): A Language for Communicating Ideas," *IEEE Transactions on Software Engineering*, Vol. SE-3, No. 1, pp. 16-34, January, 1977.
- [Sanvido 84] Sanvido, V. E., *Designing Productivity Management and Control Systems for Construction Projects*, Ph.D. Thesis, Department of Civil Engineering, Stanford University, Stanford, CA, June 1984.
- [Sanvido 90] Sanvido, V. E., "An Integrated Building Process Model," *Technical Report No. 1*, Computer Integrated Construction Research Program, Department of Architectural Engineering, Pennsylvania State University, University Park, PA, January 1990.
- [Sause 89] Sause, R., *A Model of the Design Process for Computer Integrated Structural Engineering*, Ph. D. Dissertation, University of California, Berkeley, CA, 1989.
- [SEA 88] Structural Engineers Association of California (SAE), Seismology Committee, *Recommended Lateral Force Requirements and Tentative Commentary*, San Francisco, CA, 1988.
- [Srinivasan 92] Srinivasan, K. and Jayaraman, S. (School of Textile & Fiber Engineering, Georgia Institute of Technology), "Design and Development of an Enterprise Modeling Methodology," *Workshop Notes of the 1192 Workshop Program on AI in Enterprise Integration*, sponsored by the American Association for Artificial Intelligence, San Jose, July, 1992.
- [Tsichritzis 82] Tsichritzis, D. C. and Lochovsky, F. H., *Data Models*, Prentice-Hall, Englewood Cliffs, NJ, 1982.



- [Vanegas 87] Vanegas, J. A. P., *A Model for Design/Construction Integration During the Initial Phases of Design for Building Construction Projects*, Ph.D. Thesis, Department of Civil Engineering, Stanford University, Stanford, CA, 1987.
- [Verheijen 82] Verheijen, G. M.A., and Bekkum, J. v., "NIAM: An Information Analysis Method," In T.W. Olle, H. G. SOl and A. A. Verrijn-Stuart (editors), *Information Systems Design Methodologies: A Comparative Review*, Proceedings of the CRIS-1 Conference, North-Holland, 1982.
- [Yourdon 79] Yourdon, E. and Constantine, L., *Structured Design*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1979.
- [Webster 86] Merriam-Webster Inc., Publishers, *Webster's Third New International Dictionary of the English Language Unabridged*, Editor in Chief P.B. Grove and the Merriam-Webster Editorial Staff, Springfield, MA, 1986.
- [Wiederhold 86] Wiederhold, G., "Knowledge versus Data," *On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies*, pp. 77-82, edited by M. L. Brodie, and J. Mylopoulos, Springer-Verlag, NY, 1986.

# **GLOSSARY**

## **ACTIVITY**

An organizational unit of a process for performing a specific task [Webster 86]. This concept is included in the original Data Flow model [Gane 79, Yourdon 79, De Marco 82, Batini 92]. In our model, PANDA, this concept is graphically represented by the activity node.

## **ALTERNATIVE**

One possible method of achieving a particular objective. Making a decision typically involves considering one or more alternatives. In our model, PANDA, alternatives are represented as annotations to the precedence links coming out of the decision nodes.

## **ANALYST**

A person or team that carries out the functional analysis of a process.

## **BOUNDARY**

A delimiter that marks the beginning or end of a subprocess, or the borderline between a subprocess and another activity or subprocess with which it is in contact. In our model, PANDA, boundaries are graphically represented as boundary nodes.

## **CONCEPTUAL MODELING**

The process of modeling that leads toward a conceptual schema. (Also see MODELING.)

## **CONCEPTUAL SCHEMA**

A high-level description of the database structure that is independent of any particular data model [Batini 92]. The database structure includes entities, properties, constraints and relationships. This term should be distinguished from “data model,” which is the set of conceptual modeling tools used to define the conceptual database schema for a particular application.

## **DATA**

Specific instances of recorded information. Data has the following properties: It includes great detail, changes rapidly over time, is voluminous, and is commonly stored in database systems [Wiederhold 86].

## **DATABASE**

A formal collection of related data organized according to an a priori-defined logical schema [Tsichritzis 82].

## **DATA FLOW**

This concept represents the fact that a data item or data repository flows into or out of an activity. This concept is included in the original Data Flow model [Gane 79, Yourdon 79, De Marco 82, Batini 92]. In our model, PANDA, this concept is graphically represented with the directed data flow link between a data item node or data repository node and an activity node.

## **DATA GENERATION**

This concept represents the special relationships among data as the process evolves and the types of operations (e.g., abstracting, deriving, versioning, storing, combining and presenting data) performed on the data. This concept is included in our model, PANDA, and is graphically represented as the directed data generation link. This link exists only among data items and data repositories. The link is also annotated with an abbreviation showing the type of data generation relationship involved.

## **DATA ITEM**

An individual datum or collection of data that is created by activities of the process and used as input by other activities. This concept is included in our model, PANDA, and is graphically represented by the data item node.

## **DATA MODEL**

A collection of modeling tools for describing data, data relationships, data semantics and constraints upon data items [Tsichritzis 82].

### **DATA REPOSITORY**

A permanent storage of data in paper or electronic format. Examples include files, permanent records, look-up tables, paper or electronic forms, electronic databases, vendors' standard parts catalogs, standard design codes, companies' design manuals, engineering drawings, and program input and output printouts. This concept is included in the original Data Flow model [Gane 79, Yourdon 79, De Marco 82, Batini 92]. In our model, PANDA, this concept is graphically represented by the data repository node.

### **DATA SOURCE OR SINK**

A person or thing that is the prime originator or receiver of data repositories or data items. This concept is included in the original Data Flow model [Gane 79, Yourdon 79, De Marco 82, Batini 92]. In our model, PANDA, this concept is graphically represented by the data source or sink node.

### **DATUM**

A unit of information that describes a real life phenomenon or an abstract idea that people formulate and record [Tsiehrizis 82].

### **DECISION**

A special activity that involves answering a preponderant question by considering one or more options (or alternatives) and choosing among them. This concept is included in our model, PANDA, and is graphically represented by the decision node.

### **ENTITY**

Representation of a distinguishable real-world object that can be concrete or abstract [Chen 76].

### **FLOW NETWORK**

The concept that represents an aggregated way of showing complex data flow among activities: Several different data items and data repositories can be used in a single activity in order to generate more data. This concept applies to physical flow as well as mixed flow. It is included in our model, PANDA, and can be shown graphically using flow merge nodes.

## **FUNCTIONAL ANALYSIS**

The study of information flow among the activities of a process or processes in an enterprise. The purpose is to understand what information is used and how it is exchanged among the activities. It is an important part of developing databases and also applications that operate on databases [Batini 92].

## **FUNCTIONAL DECOMPOSITION**

The hierarchical breakdown of a process into smaller functional components such as subprocesses and activities. This breakdown allows the analyst to gradually uncover the details of the process, and thus provides a powerful mechanism for analyzing complicated processes.

## **FUNCTIONAL SCHEMA**

A representation of a process that shows the activities of the process and the way in which data is used and exchanged among those activities. This representation shows the database designer how the database being developed will be used.

## **GRAPHICAL FUNCTIONAL SCHEMA**

A formatted drawing showing a functional schema of a process. Such a drawing results from using a model that has a predefined graphical representation for doing functional analysis. For example, one can use the Data Flow model to produce a graphical function schema of a process, also called a "Data Flow diagram." Similar, using PANDA, one can produce a "Partitioned Data Flow diagram," or P-diagram.

## **INTERFERENCE**

A special occurrence that interrupts the successful execution of the process. This concept is included in our model, PANDA, and is graphically represented by the interference node.

## **LOGICAL DATABASE SCHEMA**

A computer-processable representation of the database structure as described by the conceptual database schema. This representation uses a particular data model [Batini 92].

## **MATERIAL**

This concept represents the resources used in the process. In our model, PANDA, the graphical representation of this concept (and also of the concept of product) is the physical node.

## **MIXED FLOW**

This concept represents the fact that a data item or repository and a material or product flow together into or out of an activity. Our model, PANDA, graphically represents this concept with the directed mixed flow link. The term “mixed flow” was introduced in the Information Systems Work and Change Analysis (ISAC) model [Lundeberg 82].

## **MODELING**

The act of observing, and abstracting the objects and properties of interest in the real world, and structuring them in ways that can be processed by computers.

## **PANDA (abbreviation of PARTitioned eNginneering DAta flow model)**

An extension of the Data Flow Model [Gane 79, Batini 92] used for functional analysis of facility engineering processes. It adds the concepts needed to analyze complex engineering processes. In addition, it has a unique architecture that includes three partitions: (1) Participants, (2) Process and (3) Data-Material-Products. This architecture helps organize thinking about complicated processes. It also enables the analyst to produce functional schemata that are highly readable, both conceptually and graphically.

## **PARTICIPANT**

A class of personnel that takes part in activities of a process. Each participant can be involved in more than one activity in the process. This concept is included in our model, PANDA, and is graphically represented by the participant node.

## **PARTICIPATION**

The capacity in which a participant is involved in an activity. Our model, PANDA, represents each participation graphically with a participation link from a participant node to an activity node in the next partition.

## **PHYSICAL FLOW**

This concept represents the fact that a material or product flows into or out of an activity. Our model, PANDA, graphically represents this concept with the directed physical flow link between a physical node and an activity node. The term “physical flow” was introduced in the Information Systems Work and Change Analysis (ISAC) model [Lundeberg 82].

### **PROCESS NON-LINEARITY**

A characteristic of any process that does not involve only linear sequences of consecutive actions. Non-linear processes can have simultaneous actions or iterations over a same set of actions.

### **PRODUCT**

This concept represents the intermediate or final results of the process. In our model, PANDA, the graphical representation of this concept (and also of the concept of material) is the physical node.

### **PRECEDENCE RELATIONSHIP**

The way in which activities of a process are related to one another. This particular relationship expresses explicit temporal constraints placed on the execution of the activities. In our model, PANDA, the graphical representation of this concept is the directed precedence link.

### **PROCESS**

A collection of related actions that serves a long-term goal and brings about certain results.

### **REFERENCE MODEL (FOR FUNCTIONAL ANALYSIS)**

A set of concepts, rules and operations needed to do a job (in this case, functional analysis) as well as a methodology and the guidelines for using those concepts, rules and operations.

### **RELATIONSHIP**

An association among entities in the data model; or formally a tuple of entities  $[e_1, e_2, \dots, e_n]$  from a mathematical relation  $R_i$  among  $n$  entities, each of which belongs to an entity set  $E_i$  [Chen 76].

### **REPRESENTATION**

A simplified description of reality. A representation defined using some form of representation language is a schema.

## **SUBPROCESS**

A fixed set of activities, decisions, interferences and delimiting boundaries, which have precedence relationships to each other. In our model, PANDA, the graphical representation of a subprocess is of the subprocess' components: activities, decisions, interferences and boundaries.