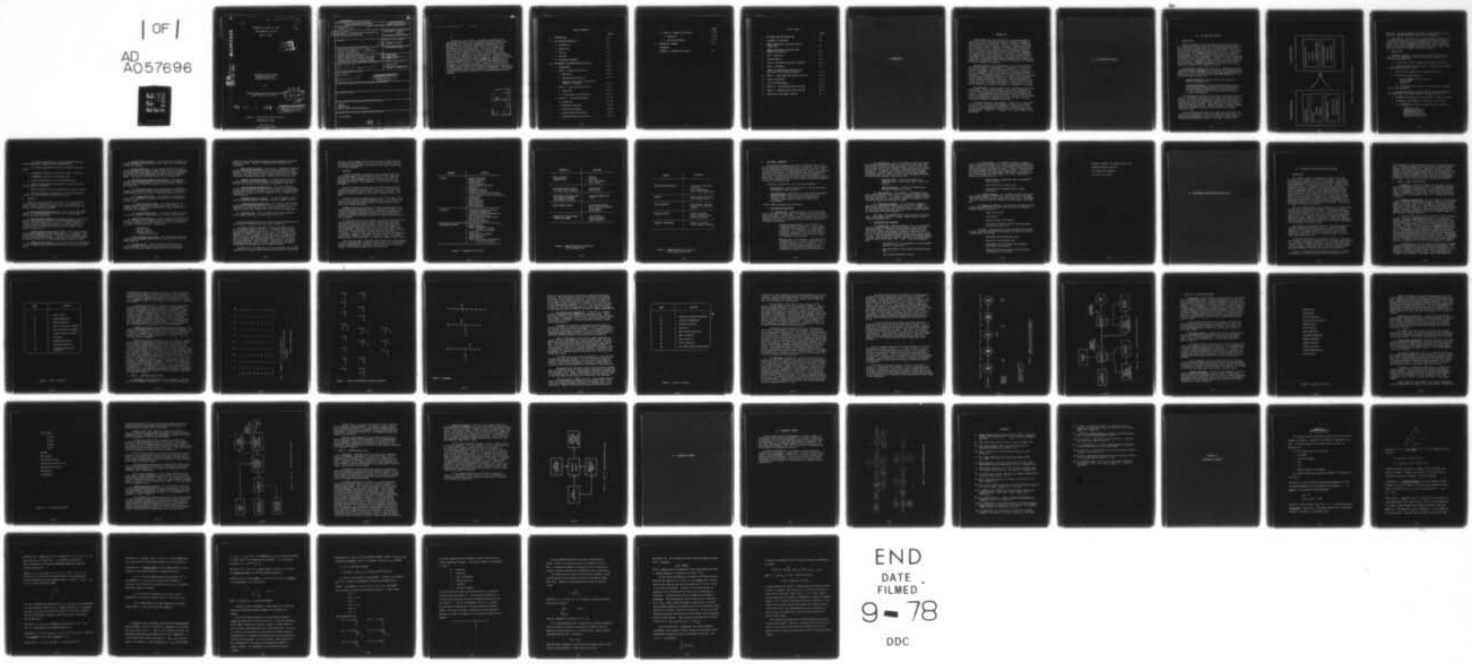
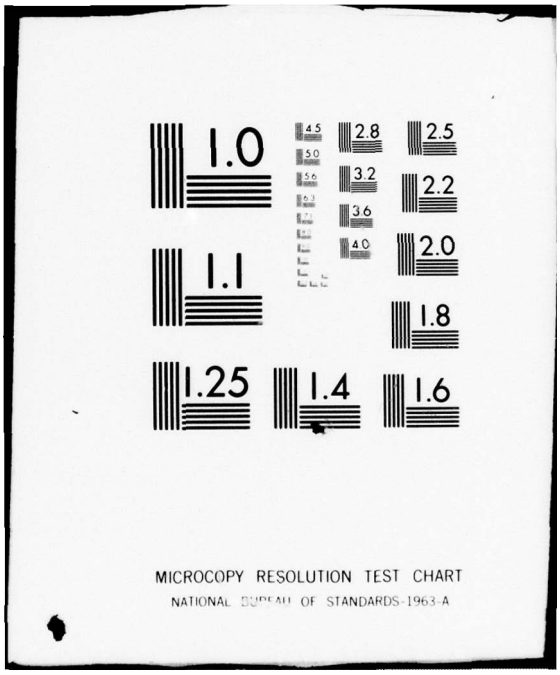
AD-A057 696 BOOZ-ALLEN AND HAMILTON INC BETHESDA MD F/G 17/2
STRUCTURE OF A METHODOLOGY FOR GENERATION OF IAS CANDIDATE ARCH--ETC(U)
MAY 78 DCA100-77-C-0057
UNCLASSIFIED SBIE-AD-E100 072 NL

| OF |
AD
A057696



END
DATE FILMED
9-78
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

CONTRACT NO. DCA100-77-C-0057

TASK ORDER NO. 10-77/1/1

May 10, 1978

STRUCTURE OF A METHODOLOGY FOR GENERATION OF IAS CANDIDATE ARCHITECTURES

for

Defense Communications Engineering Center
Defense Communications Agency

BOOZ · ALLEN & HAMILTON Inc.
Management Consultants

4330 EAST WEST HIGHWAY
BETHESDA, MARYLAND 20014
951-2200

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: DCA100-77-C-0057
2. GOVT ACCESSION NO.:
3. RECIPIENT'S CATALOG NUMBER:

4. (6) TITLE (and Subtitle): Structure of a Methodology for Generation of IAS Candidate Architectures.
5. TYPE OF REPORT & PERIOD COVERED: Final Report: 10/19/77 - 5/10/78
6. PERFORMING ORG. REPORT NUMBER:

7. AUTHOR(s):
8. (15) CONTRACT OR GRANT NUMBER(s): DCA100-77-C-0057

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Booz, Allen & Hamilton Inc.
4330 East West Highway
Bethesda, Maryland 20014
(12) 62p.
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:

11. CONTROLLING OFFICE NAME AND ADDRESS:
Defense Communications Engineering Center
Defense Communications Agency, Code R810
1860 Wiehle Ave., Reston, Virginia 22090
(11)
12. REPORT DATE: 10 May 1978
13. NUMBER OF PAGES: 61

14. (9) MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Final rept. 19 Oct 77 - 10 May 78.
15. SECURITY CLASS. (of this report): Unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)
(18) SBIE
(19) AD-E100 072

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

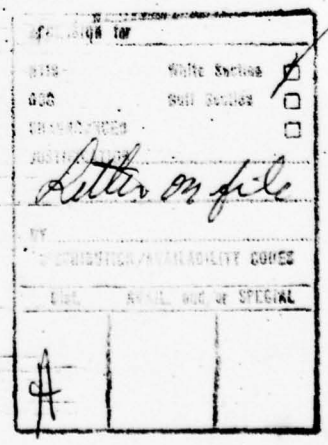
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
AUTODIN
Architecture
network system design methodology

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
See Attached.

78 07 20 022

DD FORM 1473 EDITION OF 1 NOV 65 IS OBSOLETE S/N 0102-014-6601 UNCLASSIFIED 408 597

The purpose of this study is to develop a structure for a methodology to produce a set of promising candidate architectures. This methodology incorporates subjective considerations with objective ones to control the generation of architectures from the complete space of alternatives. Four distinct phases are identified as the logical divisions in this development from a purely conceptual stage, based on users' requirements, to the stage where the components of a network have been defined within the context of a practical network architecture. Conceptual models, called logical architectures, are the plausible sequences of functions which are synthesized via a regular grammar (or, equivalently, an automation model) in Phase I. In Phase II, hierarchical structure is added to the logical architecture. Phase III is devoted to the development of the system architectures which are used in Phase IV to define network elements and to provide a framework for consideration of implementation issues, policy matters, and strategies. This framework guides the detailed definition of the candidate communications networks.



1473B

TABLE OF CONTENTS

		Page
I.	INTRODUCTION	I-1
II.	IAS FUNCTIONAL BASELINE	II-1
	1. Introduction	II-1
	2. Capabilities	II-3
	3. Services	II-4
	4. Functions	II-7
	5. Performance Parameters	II-11
III.	METHODOLOGY FOR DEVELOPING ARCHITECTURES	III-1
	1. Introduction	III-1
	2. Phase I - Logical Architectures	III-2
	a. Definition	III-2
	b. Generation and Selection	III-2
	c. Example of the Logical Architecture Generation Procedure	III-4
	3. Phase II - Functional Architectures	III-4
	a. Definition	III-4
	b. Stratification and Augmentation	III-8
	4. Phase III - System Architectures	III-14
	a. Definition	III-14
	b. Additional Functions	III-14
	c. Physical Environment	III-14
	d. System Element Assignments	III-16
	e. System Architecture Selection	III-18

		Page
	5. Phase IV - Network Architecture	III-20
	a. Definition	III-20
	b. Selection Procedures	III-21
IV.	METHODOLOGY SUMMARY	IV-1
	REFERENCES	
	APPENDIX A - MATHEMATICAL CONCEPTS	A-1

LIST OF FIGURES

		Page
1.	FUNCTIONAL BASELINE DERIVATION	II-2
2.	CATEGORIES OF FUNCTIONS	II-8
3.	SAMPLE DERIVATION OF FUNCTIONS FROM IAS CAPABILITIES	II-9
4.	SAMPLE DERIVATION OF FUNCTIONS FROM PROPOSED IAS SERVICES	II-10
5.	PHASE I FUNCTIONS	III-3
6.	RELATION MATRIX	III-5
7.	LOGICAL ARCHITECTURES AND THEIR GENERATORS	III-6
8.	PHASE II FUNCTIONS	III-9
9.	EXAMPLE OF HIERARCHICAL WEIGHTINGS FOR EACH FUNCTION IN A LOGICAL HIERARCHY	III-12
10.	PHASE II - FUNCTIONAL ARCHITECTURE SELECTION	III-13
11.	PHASE III FUNCTIONS	III-15
12.	LIST OF SYSTEM ELEMENTS	III-17
13.	PHASE III - SYSTEM ARCHITECTURE SELECTION	III-19
14.	PHASE IV - NETWORK ARCHITECTURE SELECTION	III-22
15.	ARCHITECTURE METHODOLOGY OVERVIEW	IV-2

I. INTRODUCTION

I. INTRODUCTION

The development of an architecture for the Integrated Autodin System (IAS) of the 1990s requires that many technical, engineering, functional and policy factors be included. The state of technology must be forecast years in advance. The type of system users and the services they will need in the next two decades must be considered. The constraints on cost, performance, and survivability must be defined. All these factors indicate that many projections, judgments, and nonquantifiable requirements will be involved in the development of a preferred or optimal architecture for the IAS along with those requirements which can be quantified. The purpose of this report is to describe an architecture development methodology which incorporates subjective considerations within a structure that permits the selection of the best candidate architecture.

The main objective of this methodology is to allow all possible IAS 1990 architectural alternatives to be considered in an effective and efficient manner. The methodology presented in this report is explicitly designed to permit the inclusion of technical and engineering value judgments as well as well-defined relationships and characteristics. The approach begins with the highest logical and functional considerations—the abstract aspects—and proceeds to more tangible aspects through a sequence of four phases. Each phase identifies decisions that have to be made, examines their inter-relationships, and incorporates them in a controlled synthesis of alternatives. In this way, the number of alternatives considered in each phase is minimized.

The methodology is used to produce a set of candidate architecture alternatives. These can be evaluated in more detail to determine a final selection on the basis of forecasted cost and performance capabilities. The methodology permits the determination of alternatives without the enumeration of all possibilities. The tools used to develop and select architectural alternatives include the theory of regular structures and dynamic programming techniques.

The report consists of four chapters. Chapter II is a discussion of the services, functions and performance considerations which are critical to the IAS architecture development. Chapter III is a description of the methodology in general and each of the phases within the methodology. Chapter IV is a summary of the major features of the methodology.

II. IAS FUNCTIONAL BASELINE

II. IAS FUNCTIONAL BASELINE

1. INTRODUCTION

Before proceeding with the IAS architecture development process, it is important to identify the services and capabilities required of the system, and translate these into a functional baseline which will serve to guide the generation of feasible architectures. The IAS must possess certain basic capabilities in order to satisfy its objectives. In broad terms, the system must provide end-to-end record and data communications, as well as teleprocessing and resource sharing services, to a wide variety of DOD subscribers. In addition to these fundamental capabilities, the requirements of the IAS users are met by a variety of service features. Services are offered in response to established or projected needs for increased performance and efficiency in specific operations carried out by a community of users.

The aggregate of capabilities and services, together with detailed information transfer requirements constitute the IAS requirements baseline. This in turn translates into the functional baseline (Figure 1), an important input to the architecture development methodology. The functional baseline consists of:

- Necessary Functions - those functions which support the services and capabilities.

- Performance Parameters - those performance measures, goals, and constraints, derived from user and system requirements, which characterize system behavior, provide evaluation criteria, and form the basis for the rating and selection of viable architectures.

The information on services and functional requirements presented later in this chapter is not meant to be exhaustive and definitive. Instead, it tries to list those services and functions which are likely to be present in any IAS architecture, together with a few which are somewhat speculative in nature. This effort has relied on a careful study of existing systems (e.g., ARPANET, AUTODIN I, TELENET), and the information provided by several requirements definition studies (Refs. [1] - [8]).

For the 1978-1983 time frame, the services and requirements are rather firmly established. For the post-1983 time frame, however, the forecasting and validation process is affected by the rapid pace of technological developments, as well as the impact of major policy

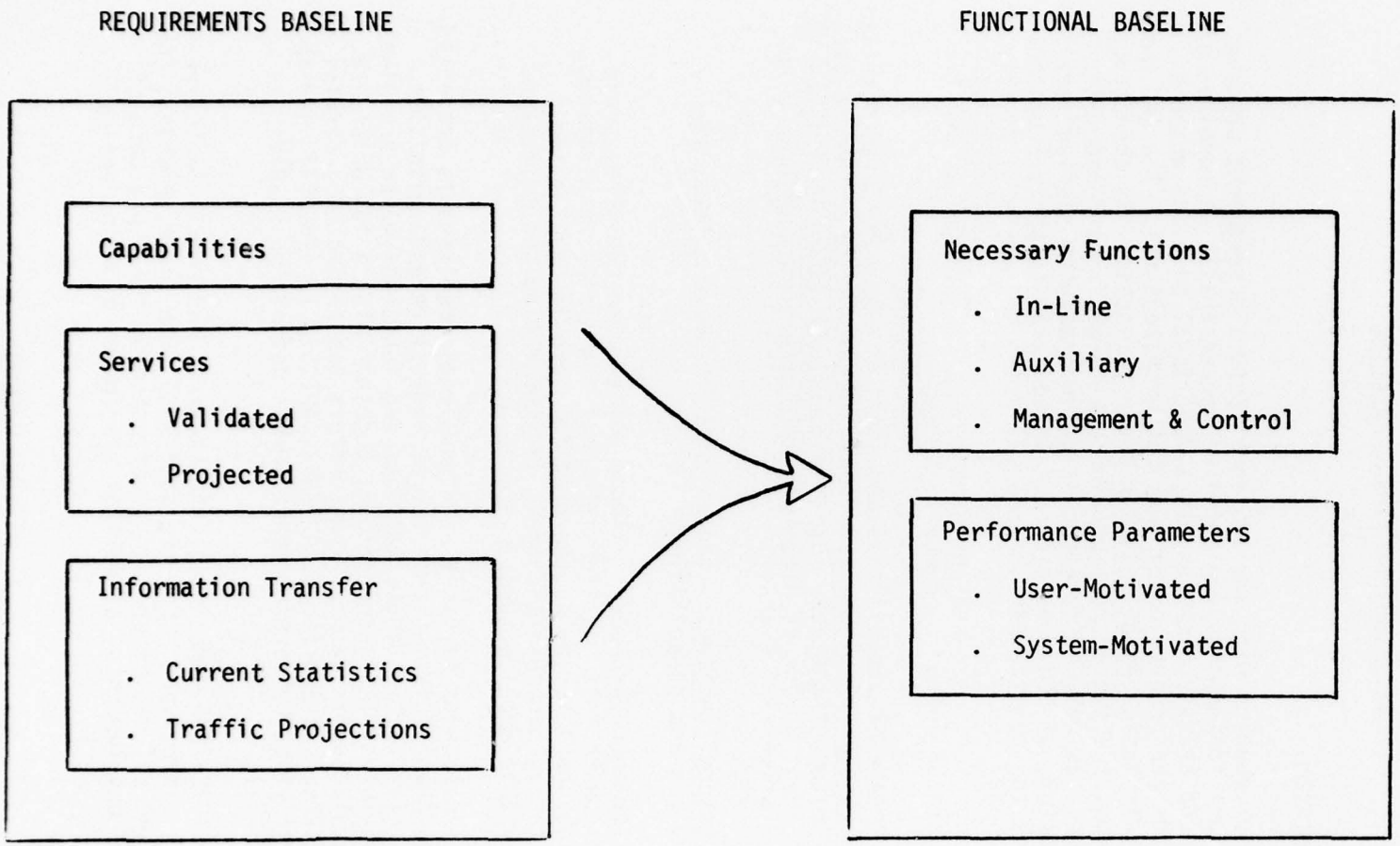


Figure 1. Functional Baseline Derivation

decisions. The IAS development effort must be geared towards a robust design to accommodate these uncertainties and limitations.

The parameters identified in this chapter should be sufficient for the purpose of applying the IAS development methodology in the generation of viable long-term candidate architectures. The requirements analysis will continue throughout the development process, with refinements to the current information performed as necessary. These refinements will update the candidate architecture designs.

2. CAPABILITIES

In order to accomplish its objectives and meet the user's basic needs, the IAS network must possess several general capabilities. As a minimum, the system must:

a. Provide rapid, reliable communications between subscribers.

b. Provide cost-effective utilization of communications resources to handle projected traffic loads.

c. Provide secure communications throughout the system.

d. Interface with other networks:

- Military (WWMCCS, NTS, TRI-TAC, etc.)
- Allied (NATO)
- Commercial.

This interoperability can range from interfacing on a gateway level to full integration.

e. Provide cost-effective and efficient utilization of hardware and software facilities through resource management and sharing.

f. Accommodate subscribers with diverse codes, formats and speeds.

g. Incorporate a high degree of flexibility; in particular:

(1) Accommodate a variety of transaction types, including:

- Interactive (I/A)
- Narrative/Record (N/R)
- Query/Response (Q/R)
- Bulk Data Transfer (BDT)
- System Control (Overhead).

(2) Provide adaptability to future requirements and technology developments through incremental modernization.

(3) Provide expandability through incremental and modular growth.

h. Accommodate transactions with various levels of priority.

i. Accommodate a variety of input/output media.

j. Provide accountability of all system transactions.

k. Provide a high degree of transparency in user-to-user and user-to-computer transactions.

l. Meet the necessary privacy requirements of the user community.

m. Provide the necessary maintainability and reconfigurability features to ensure continuity and quality of service.

3. SERVICES

Beyond the required capabilities, the IAS will offer additional service features to meet specific user needs. Listed below are several services which are candidates for inclusion in the IAS architecture in the 1978-1990 time frame.

a. Narrative/Record Message Service. This is the present JANAP 128/ACP 127 formatted messages service, meeting the unique needs of official message communications.

b. Narrative/Record Message File Retrieval Service. This service would provide additional features to the Narrative/Record Message Service, including the ability for the user to request copies of previously received or transmitted messages. The service would be restricted to official communications and would not attempt to be a general filing service.

c. AUTODIN Limited Privacy Service (ALPS). This service provides the user with a measure of privacy above that which is available in the Narrative/Record Message Service. It ensures that no permanent record of the text content of the message is available at the switching center, and restricts access to it as the message passes through the system.

d. Commercial Refile Service. This service allows the interchange of record communications between AUTODIN and public data and record networks.

e. Base Distribution Service. This service will be aimed at satisfying the needs of all services for base, camp, and station message and data distribution.

f. Digital FAX Service. It will provide facsimile service in applications where the standard communications services are inadequate to the task being performed by the user, and/or where the economics of using facsimile clearly demonstrates savings over more conventional means. This service constitutes one version of electronic mail.

g. ADP Teleprocessing Communication Services. These are the services which are made possible by the implementation of AUTODIN II. The basic types of transactions available in this packet switched environment are:

(1) Interactive (I/A). The rapid exchange of information between subscribers in a conversational mode whereby the continuity of the information transfer process is maintained during the "session."

(2) Query/Response (Q/R). The exchange of a question and answer with no attempt to sustain the continuity of the information transfer process.

(3) Bulk Data Transfer (BDT). The transmission of files, programs, processing results, or data bases from one computer to another in bulk or block form with transaction lengths ranging from 10 Kilobits to 100 Megabits.

(4) Narrative/Record (N/R). The transfer between subscribers of narrative messages in record form in a character-oriented format.

h. Resource Sharing Services. These go beyond the teleprocessing services (category g) to include those which increase the availability of network resources to the subscriber and enhance the cost-effective utilization of these resources. Included are:

- Load Sharing
- Hardware Sharing
- Data Base Sharing
- Program/Software Sharing.

i. Data Teleconferencing Service. This would include data conferencing using keyboards and CRT displays, and possibly a slow scan video capability.

j. Mailbox Service. In this version of an electronic mail service, messages destined for action officers are delivered to a holding file on a host computer, where modification, redirection,

electronic filing for future reference and other operations may be performed on the data. The file is accessed by the addressee at his convenience.

k. Word Processing Service. This service is composed of automated editing features and functions which aid in the preparation of typed material. If document distribution and interconnection between distant subscribers are incorporated into this service, the result is a Communicating Word Processor Service (CWP), another form of electronic mail.

l. Computer Internetting Service. By combining the individual features present in the various networks associated with the IAS, desired sets of services can be attained and presented as "new" ones.

m. Informal Message Exchange Service. This service features a simple format, interactive connection for the passage of correspondence. Message accountability is the user's responsibility. The service would be capable of interfacing with the companion service of "Mailbox" and "Word Processing."

n. Management/Financial Services. This would encompass a broad range of accounting, payroll, inventory, and funds transfer services.

o. Limited Digitized Voice Service. This service, which supports voice passage through the network in a digital, encrypted form, is intended for use in applications where the furnishing of voice as an alternate to data is an integral part of the overall user requirement.

p. Paging Services. This is a modification of the commercial paging concept, using the IAS communications facilities.

* * * *

More detailed descriptions and information about the above services can be found in References [3], [4], and [6] through [9].

At present, services a through f have been implemented in the existing AUTODIN system to some extent. In addition, the Teleprocessing and Resource Sharing services are in the early stages of implementation, as AUTODIN II begins to take shape. The inclusion of the remaining services (i through p) in the IAS remains to be determined. Given the current technological trends toward office procedure automation, electronic mail, and teleconferencing, services i, j, and k are likely offerings in the IAS network. Additionally, the Computer Internetting Service could be a likely outgrowth of the IAS goals of interoperability and effective use of network resources.

The above list of sixteen services is not exhaustive, but includes those which are currently under consideration for the IAS. While other

services, such as telemedicine and training, may be added to the IAS architecture in the future, it is not envisioned that these services will require additional major functions for their support beyond those in the current baseline.

4. FUNCTIONS

In order to support the capabilities and services discussed in previous paragraphs, the IAS architecture must include a wide range of functions. These functions can be categorized into three basic areas for discussion purposes; these are: in-line, auxiliary, and management and control. The categorized list of functions is given in Figure 2.

The in-line category includes those functions which are usually in the direct sequence of events when a basic unit of information is transferred through a network from one subscriber to another. These functions are most essential to the flow of information within a network.

Auxiliary functions are those which are required to support either the in-line functions or certain network services. Unlike the in-line functions, the auxiliaries are not directly encountered by a transaction flow through the network.

Management and control functions are those functions which are required to maintain and control order in a communication system. They include the functions required to control the flow of information, to allocate the hardware and software resources in the network and to perform the monitoring and data processing required for system performance assessment.

As indicated earlier in this chapter, the services and capabilities offered by the IAS architecture translate into necessary functions. This translation process involves careful study of each service and capability to determine which functions are required to support it. The relationship is not one-to-one. That is, a service might require several functions; conversely, a single function might support a variety of services. The derivation of functions from IAS capabilities is illustrated in Figure 3. Only those functions which are most immediately derived from a capability are shown. Similarly, Figure 4 illustrates the derivation of functions from a representative sample of IAS services.

Given the wide range of functions which support the IAS architecture, as well as the functional similarities among services, it is very likely that the implementation of additional services will require only minor changes in the functional make-up of the architecture.

CATEGORY	FUNCTION
In-Line	Address Generation Address Conversion Access Control Local Distribution External Network Interface Switching Red/Black Conversion Black/Red Conversion Transaction Preparation Assistance (e.g., prompting, editing, formatting) Concentration Multiplexing Speed Conversion Code Conversion Format Conversion Mode Conversion
Auxiliary	Transaction Storage & Retrieval Transaction Readdressal Transaction Editing & Reformatting Data Storage & Retrieval User Identification/Verification Multiple Delivery Data Base Maintenance
Management & Control	Network Control Network Management (e.g., statistics, billing, resource allocation) Journaling Traffic Control & Routing Status Monitoring System Security (e.g., key variable distribution, security processing)

Figure 2. Categories of Functions

CAPABILITY	FUNCTIONS
Rapid, reliable communications	. Routing . Switching . Local distribution . Error control
Cost-effective utilization of communications resources	. Concentration . Multiplexing
Cost-effective and efficient utilization of hardware and software facilities	. Network management and control
Secure communications	. Encryption/decryption . Key variable distribution . Access control . Verification
Acceptance of diverse codes, formats and speeds	. Code conversion . Format conversion . Speed conversion

Figure 3. Sample Derivation of Functions From IAS Capabilities

SERVICE	FUNCTIONS
Narrative/Record Message	. Transaction Preparation Assistance . Local Distribution . Plain Language Addressing
Mailbox	. Data Storage & Retrieval . Edit Messages on File
Word Processing	. Data Storage & Retrieval . Standard Forms Storage & Recall
Resource Sharing	. Network Management . Resource Access Control . Precedence/Preempt
Computer Internetting	. Protocol Conversion . Traffic Control & Routing

Figure 4. Sample Derivation of Functions From Proposed IAS Services

5. PERFORMANCE PARAMETERS

The performance parameters are useful to determine how well the system supports the required capabilities and services. They characterize the system behavior under anticipated operating conditions. Additionally, they are used to evaluate systems with respect to predetermined performance requirements. At the detailed design and implementation level, the use of quantitative and explicit performance measures is appropriate. These measures are derived from the performance parameters. At the system concept and logical level, however, performance parameters are used in a qualitative way.

Performance parameters fall into two major categories:

- User-Motivated - those derived directly from the operational requirements of the subscribers.
- System-Motivated - those which arise from system requirements. System requirements are relatively transparent to the individual users. They are derived from the need to provide an organizational framework to support user requirements.

The most common parameters are listed below.

a. User-Motivated Parameters

(1) Response Time. Broadly defined as the time the system takes to respond to a given input, this parameter must be described in statistical terms. In the absence of a precise distribution function, maximum tolerable values, as well as mean and variance, may be specified. Depending on the type of transaction being considered, response time can take on two meanings:

- Interactive Response Time - also referred to as "turnaround time," this measure of responsiveness is applicable when operating in a "conversational" environment. It is defined as the time interval between the completion of the transaction entry at a terminal and the point when the system response first appears at this terminal.
- Delivery Time - this parameter is used when the information transfer is primarily unidirectional, as in record transmission or file transfers. It is defined as the time interval between the start of transmission at the sending location and the completion of reception at the receiving station.

(2) Availability. This is the probability that the network will provide the service required by a user at any given time. The wide variety of resources in the IAS network, as well as the distinct nature of each user's service demands, suggest descriptions of partial availability. In general, availability is closely tied to system reliability. For this reason, availability calculations frequently incorporate reliability measures (such as mean time between failures). The two principal aspects of availability are:

- Connection Time - involves the probability of obtaining access to the system within a given time interval.
- Service Continuity - involves the probability of obtaining uninterrupted service.

(3) Data Integrity. This parameter is concerned with the errors which can occur in the transfer of information between sending and receiving locations. It is usually specified in terms of distributions and rates which are categorized according to the nature of the information transfer and the source of errors.

(4) Subscriber Throughput. This performance parameter is concerned with the data rates the system can provide in support of user-oriented transactions. The characterization of throughput should be statistical and provide peak and average values, as well as daily and "busy hour" traffic distributions.

(5) Cost. This parameter deals with the various subscriber costs. These include hardware, software, communications, services, growth and expansion, etc.

b. System-Motivated Parameters

(1) Reliability. Generally speaking, this is the probability that the system will perform its function properly under specified operating conditions. Reliability analysis must take into account failures of various types (hardware, software, transmission, etc.) as well as the resulting system degradation and the system recovery capabilities. With the knowledge of network topology and expected component lifetime, a wide variety of reliability measures can be formulated and specified statistically (Refs. [10] - [13]). Three typical measures are:

- The probability of a disconnection occurring between any pair of subscribers.
- Mean time between failure (MTBF) and expected repair time.
- Fault detection/diagnostic delays.

(2) Survivability. This parameter involves the degree to which services can be retained or rapidly restored after disruption by natural calamity or by enemy action. Survivability analysis must consider system threat operational procedures, as well as vulnerabilities of network elements and resources to various threats. Survivability and reliability deal with many common issues and rely on similar analysis and performance statistics (Refs. [14], [15]). Some typical measures of survivability are:

- Average number of subscribers able to communicate after a failure.
- Reconfiguration and recovery time.
- Expected network throughput under a threat environment.

(3) System Throughput. This parameter describes the overall data transfer capacity of the network, which takes into account both user-oriented transactions and computer-generated traffic. Throughput statistics should be specified according to traffic type, time of day, etc.

(4) Resource Utilization. This describes the degree to which an architecture makes efficient and effective use of its resources. Attention should be given to:

- Channel utilization
- Load leveling
- Potential network "bottlenecks"
- Utilization of shared resources (ADP, data bases, service facilities, etc.).

(5) Cost. A wide variety of issues are dealt with in terms of cost. Some of the engineering management considerations likely to be addressed in the IAS are:

- Development and implementation costs
- Operations and maintenance costs
- Development risks associated with technology forecasting and assessment
- Difficulty and cost associated with evolutionary implementation of the system

- Hardware, software, and communications costs
- Hardware/software trade-offs
- Cost/complexity trade-offs
- Life cycle cost studies.

III. METHODOLOGY FOR DEVELOPING ARCHITECTURES

III. METHODOLOGY FOR DEVELOPING ARCHITECTURES

1. INTRODUCTION

This chapter outlines a methodology for identification and definition of the feasible set of IAS architectures. The functions and performance parameters discussed in the previous chapter drive the methodology which synthesizes a complete space of possibilities. These possibilities are the feasible architectures. Moreover, this methodology provides a framework for subjective judgments and it produces a manageable number of promising architecture candidates. After consideration of the architecture development process, four distinct phases were identified as the logical divisions in this development from a purely conceptual stage, based on users' requirements, to the stage where the components of a network have been defined within the context of a practical network architecture. The performance parameters are used qualitatively and quantitatively to guide the synthesis of the alternatives in each phase. This controls the number of alternatives produced.

The four distinct phases are the development of (1) logical architectures, (2) functional architectures, (3) system architectures, and (4) network architectures. In Phase I, the synthesis of logical architectures, several of the functions in Figure 2 are used to configure a set of architectures which represent plausible sequences of required functions. They are called logical architectures because they do not include references to physical or system level functions, but rather model the flow of information and the functions encountered by the transaction flow between users.

The functional architectures consist of stratified logical architectures augmented by additional functions. The stratification consists of the hierarchical grouping of functions in the logical architecture. The added functions are selected from the list in Figure 2 and they serve, in part, to support the Phase I functions. These steps comprise Phase II.

The system architectures are developed in Phase III. They are the functional architectures integrated with the system level functions (e.g., security, management and control, etc.) and with physical environment factors. System elements are postulated from this integration as groupings of functions; they are defined in terms of their functional responsibilities and their logical interconnection.

Phase IV is the definition of network architectures. A network architecture is a system architecture which has been defined in

sufficient detail to form the basic structure of a communications network. The network elements are defined on the basis of the system elements identified in Phase III. Objectives of Phase IV include selection of an optimal interconnection policy and identification of protocol structures and routing strategies for each alternative. The optimal architectural candidates (with respect to the performance parameters) can be determined from these alternatives by application of selection criteria based on analysis, modeling and simulation.

2. PHASE I - LOGICAL ARCHITECTURES

a. Definition. Logical architectures are defined as organizations and combinations of functions which are independent of network hierarchy or physical implementation. A logical architecture represents an arrangement of functions which are necessary for the basic process of communication to take place. Most of these functions are listed as the in-line functions in Figure 2. A typical set is given in Figure 5.

b. Generation and Selection. The total number of possible organizations is not practical to enumerate even in the case of a moderate number of functions. Moreover, this enumeration would contain many inadmissible architectures. The inadmissible architectures would be those which violate constraints. These constraints include engineering judgment, practices, technological realities, and predictions. For example, the "address generation" function must precede "external network interface" (gateway functions).

The description of the logical architectures should proceed in two ways. One way, a restrictive production, is to generate all possible configurations of functions and then single out those which satisfy constraints. Construction of an efficient algorithm to sift through this enumeration of all possible organizations appears to be difficult. For this reason the second way, a generative description, is the approach taken. The constraints are translated into structural rules, called generators, which generate the logical architectures. That is, the generators specify admissible combinations of functions. This provides the organizational means to deal with the necessary combinations which make up the set of viable architectures. The benefits from this approach include a checklist feature, rigor, and completeness. That is, all functions can be accounted for in this formal system to generate as complete a set of logical architectures as is possible.

A generation and selection procedure which reduces the number of possible organizations to be considered would be desirable. The selection is based on the rank order of logical architectures. It is possible to introduce a system of weights on the generators which is consistent with a preference rank order. Once these weights are assigned to the generators, the optimal viable architectures are

LABEL	FUNCTION
F_1	Access Control
F_2	Address Generation
F_3	Address Conversion
F_4	Local Distribution, Incoming
F_5	Local Distribution, Outgoing
F_6	External Network Interface
F_7	Switching
F_8	Red/Black Conversion
F_9	Black/Red Conversion
F_{10}	Transaction Preparation Assistance

Figure 5. Phase I Functions

produced by application of dynamic programming. Dynamic programming also provides the means to a sensitivity analysis. This uses the parameters and the computational work that defined the original solution to produce new solutions for ranges in the parameters. In this way, a manageable number of promising logical architectures is generated efficiently.

The generation process is developed and uses concepts from the mathematical theory of regular structures. (The terminology, concepts, and notation are described briefly in Appendix A. A more complete development can be found in Reference [16].) This formal scheme is a rigorous framework which specifies the constrained combinatory system as a weighted grammar. This rigor provides a means to analyze the sensitivity of logical architectures to changing (modified, added, or deleted) requirements. This scheme and the dynamic programming techniques are automatable. This is important to the methodology because of the number of functions and the potential for a combinatorially large number of possible architectures.

c. Example of the Logical Architecture Generation Procedure. An example is presented in this section to clarify the formal notions. The presentation is informal so that the technical aspects of algebra and the detail of networks do not interfere with the illustration. The functions used in this example are presented in Figure 5. They are labeled for ease of reference.

Before the generators are identified, it is useful to identify any precedence and concurrence relations among the functions. A precedence relation indicates which function must precede another one. The concurrency relation indicates which functions can be logically concurrent with respect to a transaction flow. These relations are conveniently represented in matrix form and, for the sample functions, this matrix appears in Figure 6. These relations are based on the constraints. The relations are checked for consistency and then analyzed for the structural inter-relationship among the functions. This analysis produces the stratification for each function; it gives the number of successors. For example, functions f_3, f_7, and f_8 have one or two. The generators are derived from this analysis. They consist of terminals (functions) and non-terminals (denoted by 1,2,...). The process begins by application of a generator for the "start" symbol, or axiom, which is denoted by "1." It continues by rewriting (replacement of) the non-terminals until a tree containing only terminals remains. A representative set of generators and the three logical architectures they define are presented in Figure 7.

3. PHASE II - FUNCTIONAL ARCHITECTURES

a. Definition. The second phase of the methodology is the construction of functional architectures. The functional architectures

	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
F_1	0	0	0	1	1	1	1	0	1	0
F_2	0	0	1	1	1	1	1	0	1	0
F_3	0	0	0	1	0	0	1	0	1	0
F_4	0	0	0	0	0	2	0	0	0	0
F_5	0	0	0	1	0	2	2	0	1	0
F_6	0	0	0	2	2	0	0	2	2	0
F_7	0	0	0	1	2	0	0	0	2	0
F_8	0	0	0	0	0	2	0	0	1	0
F_9	0	0	0	0	0	2	2	0	0	0
F_{10}	0	0	0	1	1	1	0	0	1	0

The i, j element is: 1 if f_i must precede f_j;
2 if f_i and f_j can be concurrent;
0 otherwise.

Figure 6. Relation Matrix

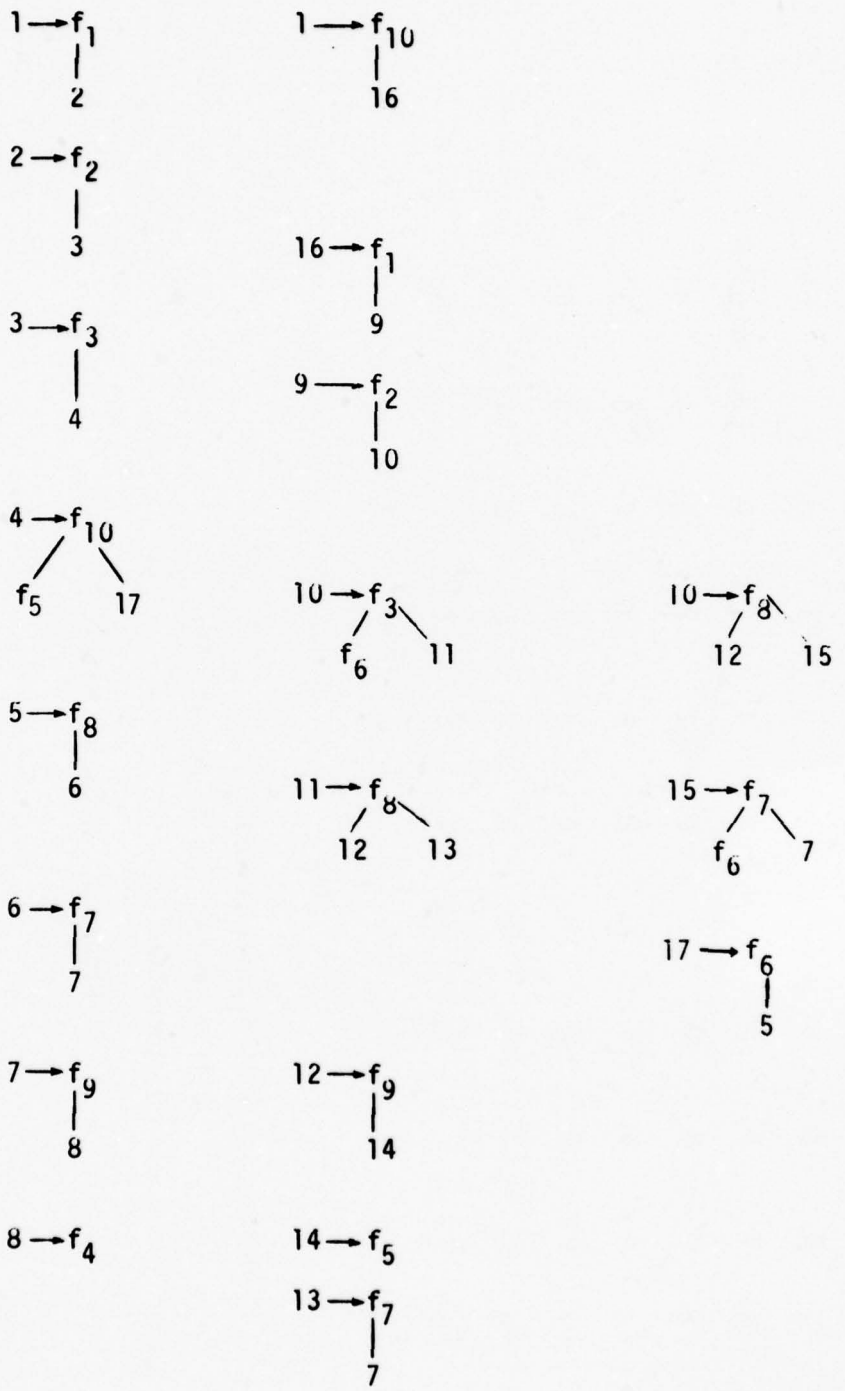
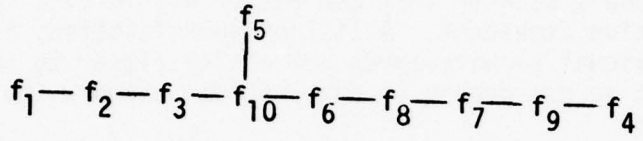
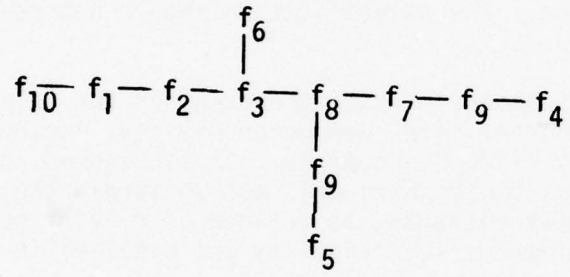


Figure 7. Logical Architectures and their Generators





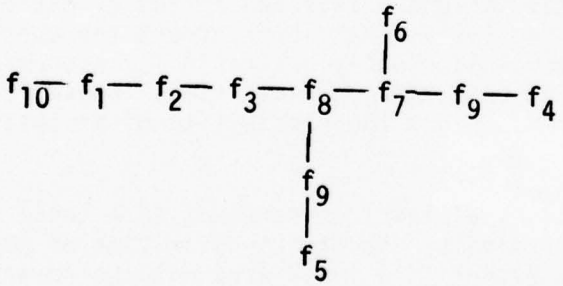


Figure 7. Continued

consist of stratified logical architectures augmented by additional functions. The stratification consists of the hierarchical grouping of functions in the logical architecture. In addition to these functions, there will be other functions included in Phase II which are not included in Phase I because they are either not in-line functions or more implementation dependent. A list of such functions is given in Figure 8. Functional architectures are a step closer to the physical environment but do not depend on any specific network implementation.

b. Stratification and Augmentation. A hierarchy is a ranking, ordering or grouping into a sequence of subordinate levels. There are several factors on which the grouping of functions into a hierarchy can be based. A fundamental factor is the logical relationship that exists among the functions. The stratification should not re-order a logical architecture.

The number of functions in each grouping and the level to which the grouping is assigned might depend on physical implementation and technical feasibility of such groupings. It is assumed, of course, that the use of functional hierarchies will not constrain the interconnections between network elements, as defined in a later phase, for achieving required performance, survivability and availability.

Where a large number of users share a common communications network, at least two hierarchical levels naturally exist: the level which contains the individual users and the shared portion of the network. The shared level may include a switched system or may consist of only the transmission facilities that interconnect the users. In large networks, a third level is usually desirable for economic sharing of resources and conserving transmission facilities. Phase II of the methodology will therefore consider the possibility of at least three hierarchical levels.

The lowest hierarchical level corresponds to a local area and includes users and user terminals. No strict definition of what constitutes the geographical extent of a local area will be advanced in Phase II. Such a definition depends on transmission media, which is a physical factor not considered until later phases in the methodology.

The second level corresponds to what could be called a regional area. It would be broader in scope, extent or functional responsibility than the lowest level. Again, no geographical description or definition of a second level will be made in this phase but will be held off until a later phase when the more physical characteristics are addressed.

The third level or highest level will include those functions which can be performed on the broadest or most general scale. This level is an obvious analogue to the familiar "backbone" level in network design. However, the term backbone connotes switching and will

LABEL	FUNCTION
H_1	Transaction Storage & Retrieval
H_2	Transaction Readdressal
H_3	Transaction Editing
H_4	Journaling
H_5	Data Storage & Retrieval
H_6	Speed Conversion
H_7	Code Conversion
H_8	Format Conversion
H_9	User Identification/Verification

Figure 8. Phase II Functions

not be used in the following discussion about methodology to avoid any confusion. While the switching function is certainly a notable and practical function to be included in the highest level, it does not have to be included in all architectures.

From the above, a three level hierarchy appears to be a natural structure from which to start the Phase II process. However, the process described below can be extended to include hierarchies with more than three levels if necessary. Reasons to consider more than three levels would be motivated by potential savings in cost or improvement in performance. Such potential should become obvious after analyses of the three level structure have been completed. The tools that will be used in Phase II to assign hierarchies to logical architectures will build upon and take advantage of the relationships of functions within the logical architectures. The goal of Phase II is not to enumerate all possible groupings in a combinatorial sense but to identify those groupings which are promising.

A procedure by which logical architectures are assigned to hierarchies is given below. It involves two steps. The first step is to examine each of the functions in the logical architecture individually and assess its applicability to and feasibility for each of the hierarchical levels. During this assessment, weights could be assigned to indicate the relative applicability of each function to each of the hierarchical levels. The second step is the identification of the most desirable or most feasible hierarchical grouping or groupings for each logical architecture. A dynamic programming approach can be used to select the most highly weighted hierarchical grouping for each architecture. The qualitative factors of engineering judgment, design guidance, and technical feasibility will determine the relative ordering or weighting of hierarchical levels for each function. The grouping of functions into a hierarchy can then be done for each logical architecture on the basis of these weightings.

The development of weights for each hierarchical level must be done on a function-by-function basis because of the many qualitative factors involved. For example, the switching function could certainly be allocated to each of the hierarchical levels on the basis of technical feasibility alone. However, the assignment of switching to the lowest level could have advantages or disadvantages in terms of engineering design, user acceptance, or total system cost which are outweighed by assigning switching to a higher level. Thus, instead of assigning equal weights of 0.33, 0.33, and 0.33, which might be indicated by technical feasibility alone, a different unbalanced weighting of 0.2, 0.5, 0.3 might be assigned due to the other qualitative factors. On the other hand, a function such as addressing could be considered to be realistically assigned to only the two lower levels which could indicate a weighting of 0.6, 0.4, 0.0.

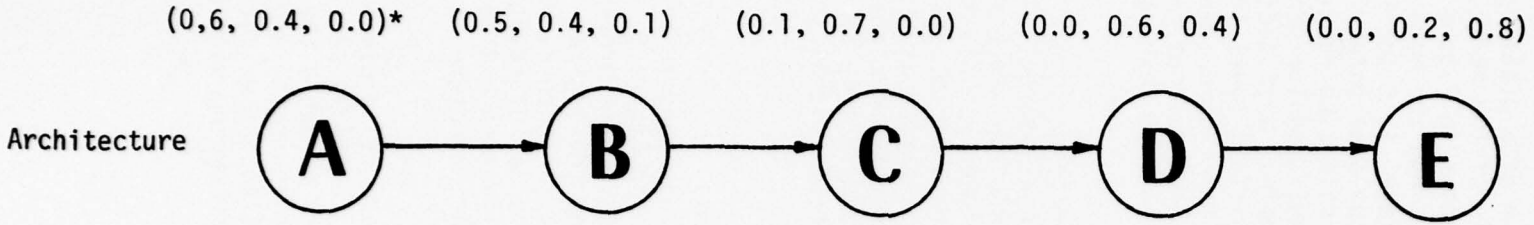
The second step involves the selection of a viable grouping of functions for each architecture on the basis of the relative weighting. A simple hypothetical example is shown in Figure 9. In this example, the logical hierarchy is a string and the maximum total weight exists for the grouping of A and B in level I, C and D in level II, and E in level III. This grouping could be determined by computing the weights of all combinations and selecting the largest. This process can also be treated as a multistage decision process where each stage corresponds to a function in the logical architecture, and a decision must be made as to which hierarchical level it should be assigned. In this form, the grouping decision process can be treated with a dynamic programming algorithm which selects the most heavily weighted grouping with fewer computations than exhaustive search.

It is possible that the selected hierarchical group will include only two or one of the levels rather than all three. If the weightings of the functions are such that the maximum group weighting occurs when all functions are included in the lowest level, then the selected hierarchy has only one level. Similarly, if the selected grouping has no third level, then a two-level hierarchy is the appropriate hierarchy. Should there be a fourth level or additional levels identified during the weighting development process, they can be handled in the same decision structure with the additional levels included.

* * * *

After the logical architectures have been assigned to hierarchies, the functions included in Phase II, i.e., those in Figure 8, can be assigned to the stratified logical architectures. This procedure is similar to the two-step procedure used to assign the logical architectures to hierarchies. The set of function allocation alternatives for the Phase II functions must be defined and a set of relative weights developed. The difference here is that the allocation and weighting procedure depends on which stratified logical architecture is under consideration.

A flow chart of the entire process for Phase II is given in Figure 10. A feature of this approach is that it allows sensitivity analyses to be performed. This provides a means to gauge the effect of changes, such as new technology, qualitative insights, and the like. Also, the sensitivity of the selected groupings to the weights assigned to each function should be determined before proceeding with other phases of the methodology.



Level:

Heaviest Weighted Grouping	I (AB)	II (CD)	III (E)

* Relative Weighting for Function A for Levels: (I, II, III)

Figure 9. Example of Hierarchical Weightings For Each Function in a Logical Hierarchy

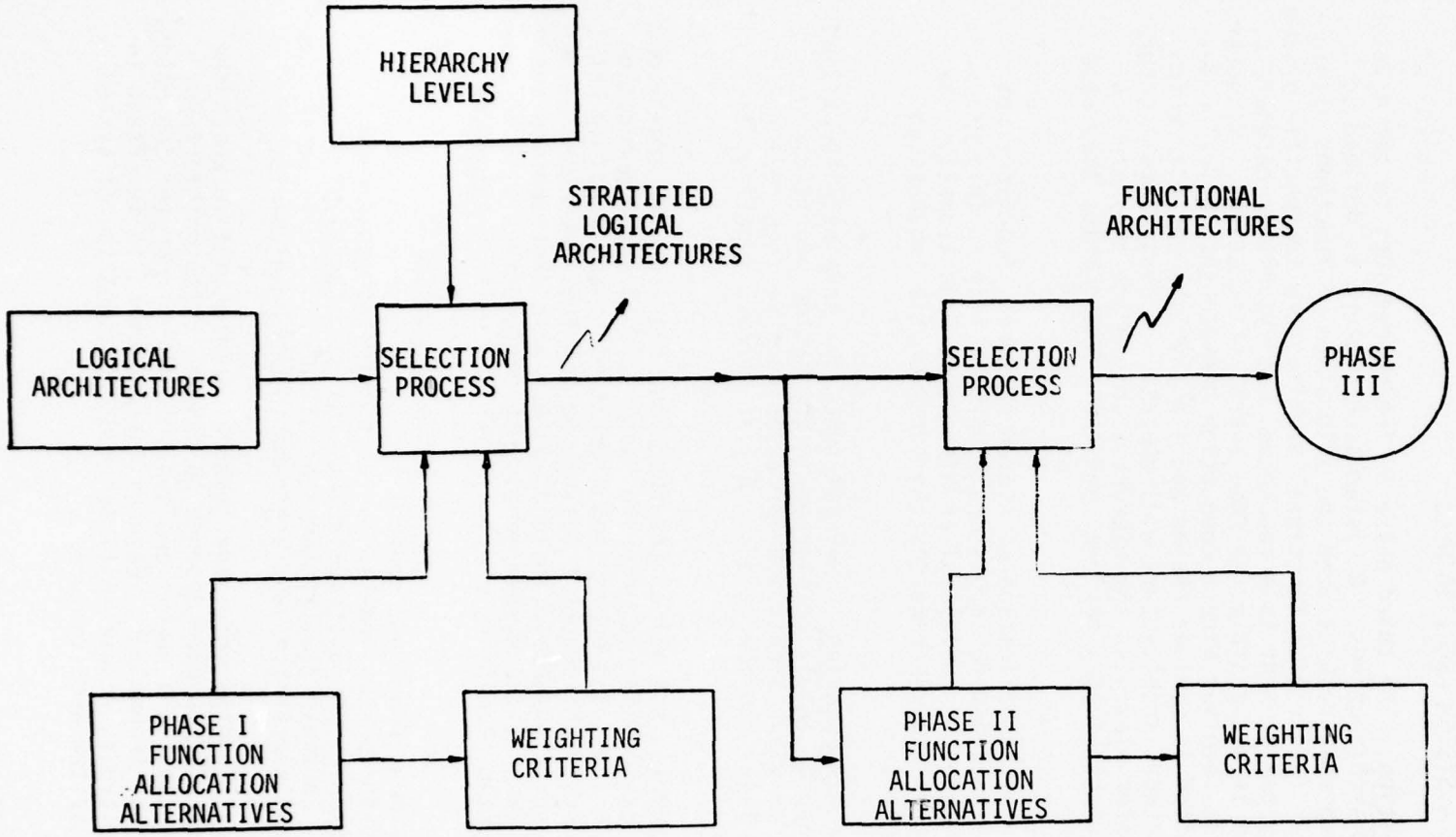


Figure 10. Phase II - Functional Architecture Selection

4. PHASE III - SYSTEM ARCHITECTURES

a. Definition. The third phase in the methodology is the development of system architectures. A system architecture is defined to be the architecture that results from the allocation of functions within the hierarchical levels of a functional architecture to specific system elements. A system element is a conceptual entity, either hardware or software, which is responsible for some subset of functions. Moreover, a system element involves direct connection between the logical relationships examined in earlier phases and the physical world in which the eventual network architecture will operate. A system architecture consists of system elements, the hierarchy into which they are configured, the functions they perform, and the relationships they have with each other.

Prior to the definition of system elements, functions not considered earlier and which apply most appropriately in this phase are integrated into the functional architecture. Additionally, the system development process requires the consideration of physical environment factors.

b. Additional Functions. Several functions have been identified as being more appropriately considered in the system architecture rather than in previous architectural stages because they depend more on system level considerations. A list of such functions is given in Figure 11.

For example, not all of the functions required for end-to-end security have been considered prior to Phase III. The recommendation of security techniques and encryption algorithms is not possible until a preferred security philosophy is developed for IAS. However, such functions as encryption/decryption and key variable distribution are extremely important at the system level. These functions will be considered when system element responsibilities and relationships are examined.

System management and control functions have also not been considered in prior phases because system control concepts depend more on system hierarchies and system elements than on logical or functional architectures. In Phase III, system control functions will be included in the functional hierarchies and assigned to system elements.

c. Physical Environment. The physical environment in which the eventual network will operate depends on several different factors. These factors include the number and type of users, computer technology and communications technology. Each of these factors is important in determining the types, responsibilities and relationships of system elements for a system architecture.

Concentration

Multiplexing

Security Processing

Key Variable Distribution

Network Control

Traffic Control

Status Monitoring

Data Base Maintenance

Message Switching

Packet Switching

Circuit Switching

Encryption/Decryption

Error Control

Figure 11. Phase III Functions

Computer technology is a key factor because many, if not all, of the functions will require information processing and computational power. Computer miniaturization and packaging are contributing to the more frequent application of modular hardware equipments. Higher order languages and structured programming techniques are making the development of modular software more efficient. With these developments the packaging of functions into system elements on an incremental and modular basis is feasible. Thus the alternatives in terms of the different combinations of functions per system element can be varied and numerous.

Communications technology is a contributor to the possible system element configurations because of modern techniques in switching, multiplexing, transmission and signaling. Each of these areas will affect the decisions made in Phase III and in Phase IV. The major effect in Phase III will be on the relationships of system elements and on the processing functions required to support the various communications techniques.

The numbers and types of users will affect system element selection in several ways. The need for multiplexing or concentration, the flexibility required in message processing, the access techniques required are examples of areas where system elements will be affected.

d. System Element Assignments. After the additional system level functions have been included in the functional hierarchies, the assignment of functions to system elements is the next step in the system architecture process. The function allocation will be made against a list of system elements with generic names. A list of such elements is given in Figure 12.

The elements suggested in Figure 12 are derived from several basic groupings of functions which have naturally or logically evolved. For example, switching nodes are easily identifiable because of their major role in network topology and because of the significant amount of resources required to perform the switching function for a network with thousands of users. The type of switching which is required determines the nature of the system element for switching.

A gateway is an important element because it represents the functions required for one network to interoperate with other dissimilar networks. It can be considered a distinct system element because there are geographical as well as functional considerations involved. The location and sizing of a gateway as well as the hierarchical level in which it resides are issues in final network design.

Access nodes are system elements which provide intermediate services between users and higher levels of the network. The existence

Switch Nodes

- Circuit
- Packet
- Message
- Hybrid

Gateways

Access Nodes

Network Control Nodes

Centralized Service Facility

Key Distribution Center

Multiplexers

Concentrators

Figure 12. List of System Elements

and responsibilities of an access system element depend on the hierarchical structure of the architecture as well as the functions required for access to other parts of the network.

Network control nodes generically represent the element required to control the network when implemented. The type, functional responsibility, and hierarchical location depends as much on the system control philosophy as on the system architecture.

A centralized service facility represents an element that performs the functions required by many of the services the system is to provide. Many of the message processing functions could reside in this element as well as those functions required to support sophisticated services such as word processing and electronic mail. This facility will be heavily dependent on computer processing capabilities.

Key distribution centers are required primarily by the security functions. For a system-wide, end-to-end security capability, one or several key distribution centers might be required. The exact nature of the centers and the degree of automation required will be determined by the security philosophy adopted.

Multiplexers and concentrators are elements which have evolved from the need to maximize the efficiency of the usage of transmission resources. These elements could incorporate more functions in their responsibilities in some cases.

The a priori existence of a list of system elements does not imply that all system elements must be used or that two or more of the system elements cannot be merged. Rather, the list represents the baseline set of elements which will be used in the system element assignment process. Analysis of the architecture might suggest a new system element which would amend the list.

e. System Architecture Selection. The procedure for identifying and selecting system architectures includes the assignment of functions to system elements, determination of system element relationships, and evaluation of system architecture feasibility. A flow diagram of this procedure is given in Figure 13.

The fundamental architectures from Phase II have to be modified with the inclusion of system level functions as described in Section b. For each functional architecture, the system level functions must be grouped into the appropriate hierarchical levels. As the system functions are applied, there may be more than one modified hierarchy possible for each functional hierarchy. If this is the case, tradeoffs might be established and a preferred modified hierarchy selected.

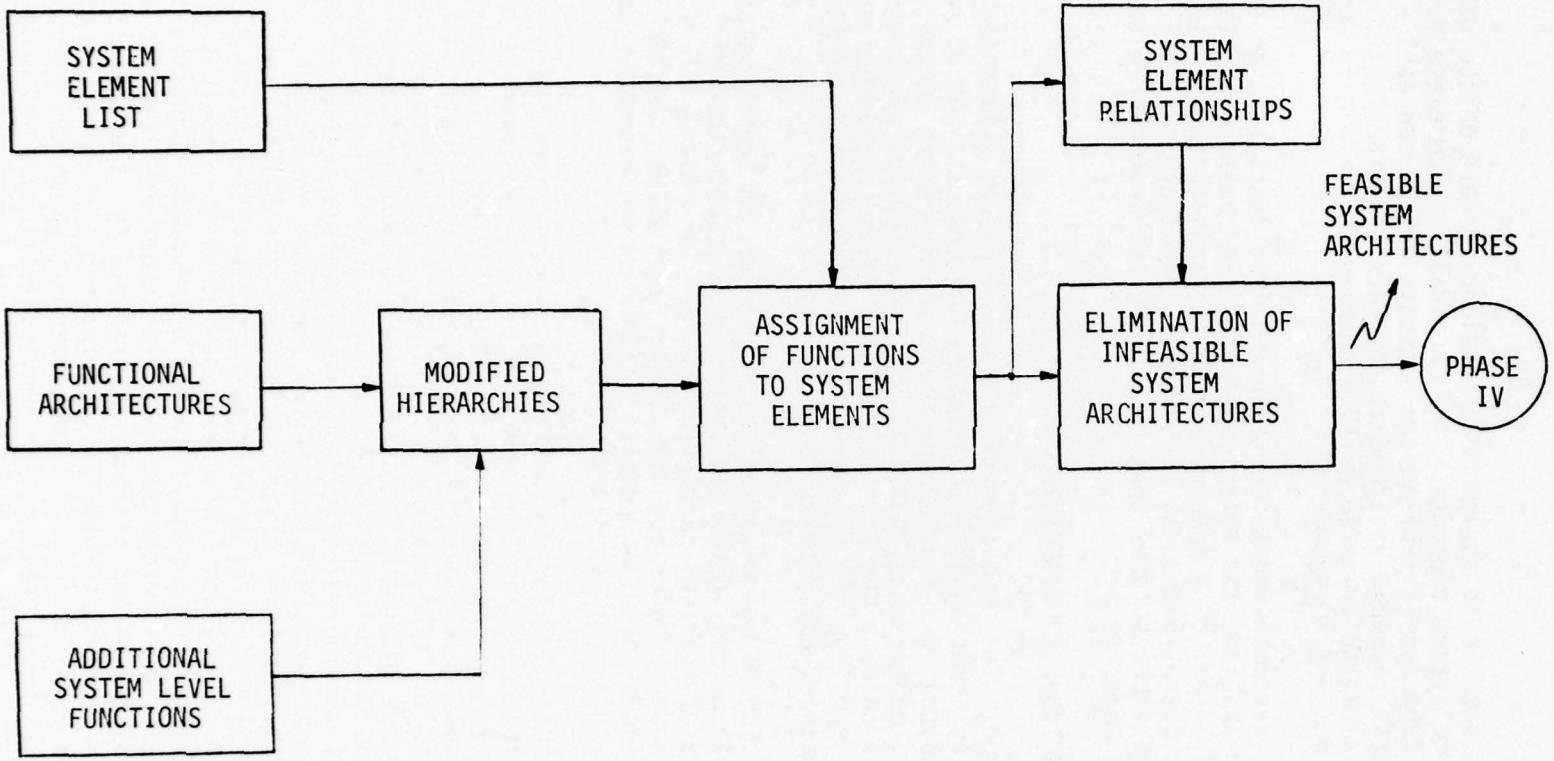


Figure 13. Phase III - System Architecture Selection

The next step is to assign the functions in each of the modified hierarchies to system elements. The candidate system elements will be selected from the list of system elements introduced earlier. There may be several possible assignments for each modified functional architecture. Evaluation criteria will be developed to evaluate and compare the alternatives.

When the system elements for each modified functional architecture have been applied, the relationships of the system elements can be determined. From this determination, the architectures which are functionally, operationally or technically infeasible will be identified. Those architectures that remain are the system architectures to be examined in Phase IV.

5. PHASE IV - NETWORK ARCHITECTURE

a. Definition. The objective of this architecture methodology study is to develop a methodology to identify and select a set of candidate IAS architectures that can be analyzed in greater detail in terms of the cost, performance, survivability, and reliability tradeoffs. The definition of a network architecture, whose components are network elements, provides the framework for consideration of subjective issues, policy matters, and strategies.

The question of which connectivity policies or transmission media models should be considered for each system architecture will be partially answered by analyses carried out in earlier phases. As the various types of architectures are selected at each stage, the implications concerning issues such as connectivity policy and transmission media will be more clearly determined.

A network architecture is defined as a system architecture described in sufficient detail to serve as a guide to implement a communications network. This involves the connectivity policy among network elements taking into account the geographic distribution of the system elements and transmission media types. Thus a network architecture includes topological considerations of which elements are interconnected, transmission media considerations of technology and capacity, cost considerations of number of elements, channels and trunks, traffic considerations of number of users and throughput, and performance considerations of reliability, survivability, and availability. To determine the best network architecture for each system architecture on the basis of these considerations requires an evaluation and selection procedure which can accommodate connectivity and media policies. It is recognized that the number and location of users and network elements and traffic requirements for the IAS will not be known at the time that Phase IV is performed. Projections and assumptions can be made for these factors, however, since the purpose of the evaluation is to select transmission facility types and general connectivity policies and not to provide a network design or optimization.

b. Selection Procedures. When the set of network elements has been determined, the connectivity can be studied through the application of one of several techniques. A number of heuristic algorithms are available for this purpose. Models of the projected subscriber, element and traffic distributions and transmission facilities will be developed and one of the algorithms will be applied to the models to the extent required to select the general connectivity policies. Some of the applicable algorithms are Branch Exchange, Concave Branch Elimination and Cut-Saturation (Refs. [19] - [21]).

Each of these algorithms requires an initial topological condition which generally is far from optimum. Also the initial condition will depend on the type of transmission media being modeled. A satellite-based network would probably look very different from a terrestrial-based network. In order for any of the search algorithms to be appropriately applied, models for the appropriate transmission media for each system architecture must be defined. After the initial models are defined, rules for substituting branches must be defined to reflect the physical nature of the branches. For example, if branches are being added, eliminated or exchanged, the range, bandwidth and vulnerability of the branches depends on whether it is long line terrestrial, satellite, microwave or high frequency (HF). When these models and rules are defined, the analysis of the best network architecture for each system architecture and, therefore, the best network architectures overall can proceed.

A flow diagram of the process for Phase IV is given in Figure 14. The result of Phase IV will be a set of candidates network architectures that can be evaluated in more detail to determine the preferred IAS architecture.

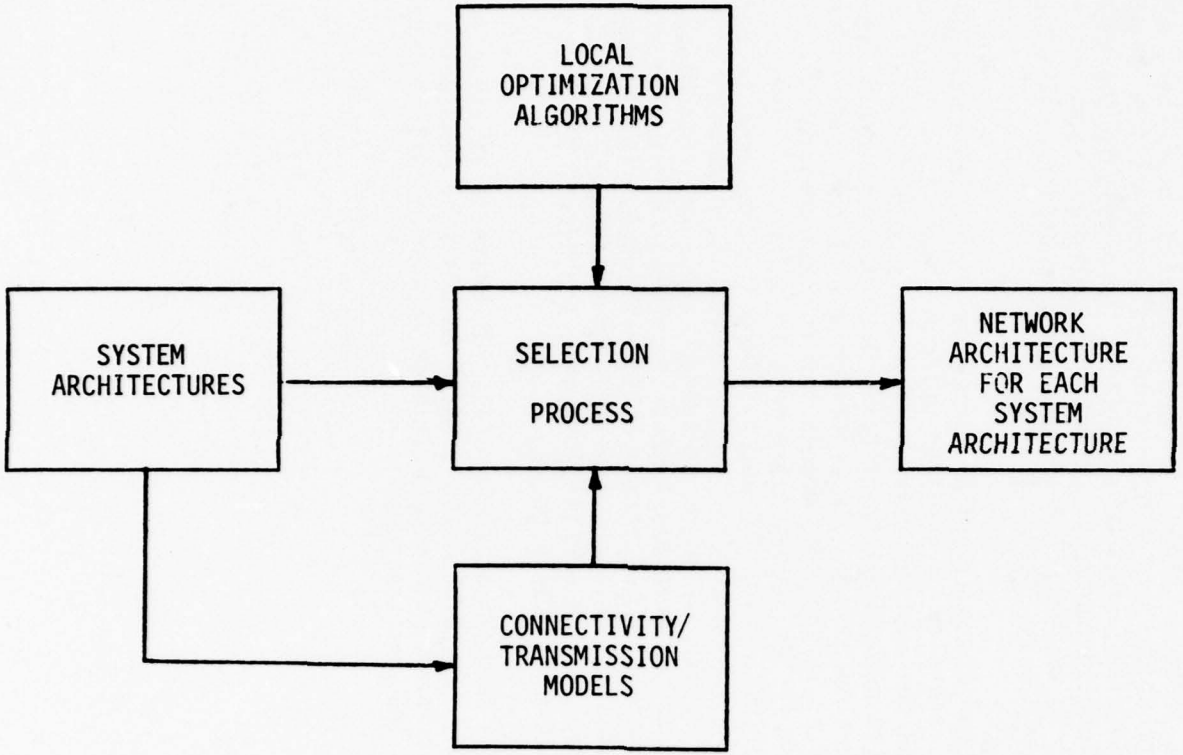


Figure 14. Phase IV - Network Architecture Selection

IV. METHODOLOGY SUMMARY

IV. METHODOLOGY SUMMARY

A graphic description of the methodology is given in Figure 15. Two types of effort pervade the methodology. One type is based on a well-defined structure for the definition, generation and optimization of alternatives. It includes algorithms which are automatable. The other type involves less tangible elements such as issues, policies, assessments and judgments. This type is not directly automatable, but has an impact on the automatable portions.

The 1990 IAS architecture is a response to a need defined by system and user requirements. Capabilities, services, and information transfer—the requirements—translate into a functional baseline which drives alternative generation. The four-phase approach incorporates technological factors, physical characteristics, and subjective considerations into a structure for identification of a complete space of possibilities.

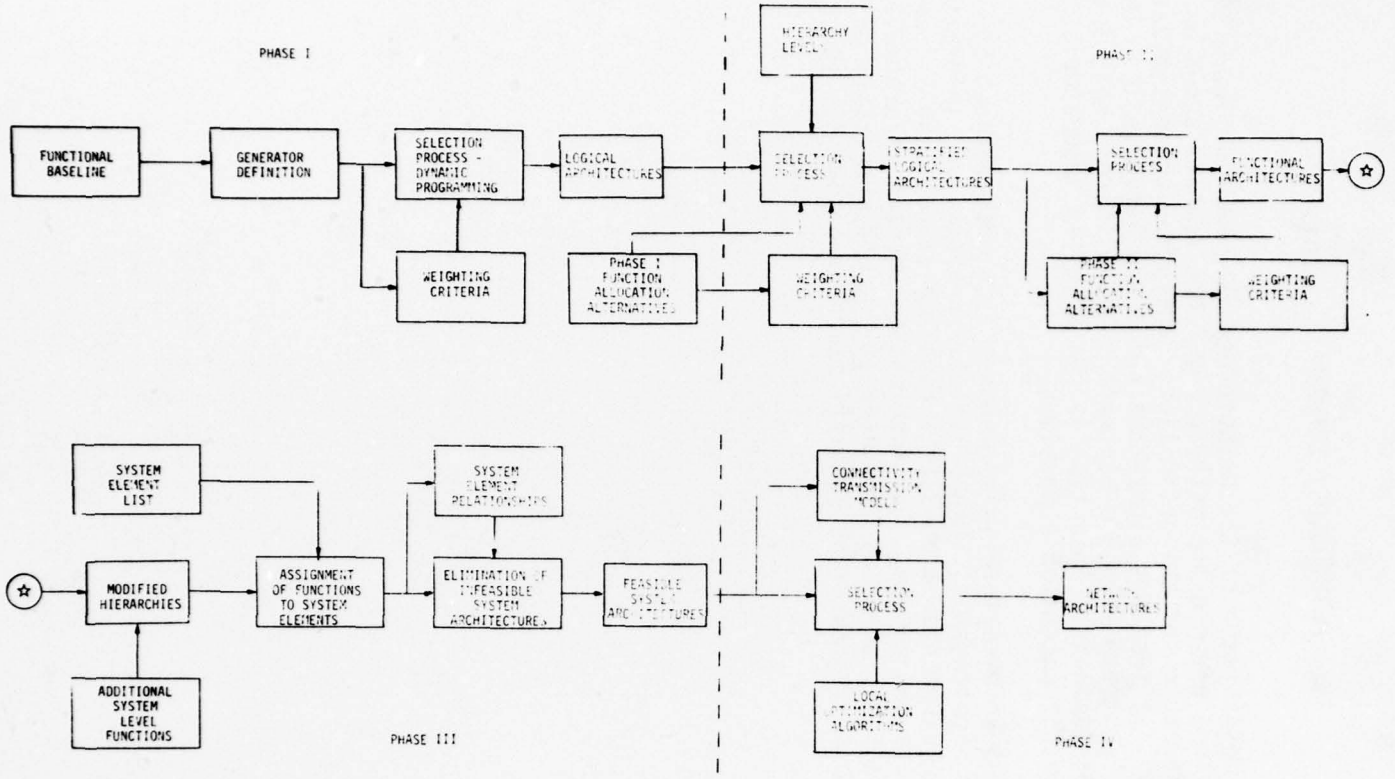


Figure 15. Architecture Methodology Overview

REFERENCES

(1) Defense Communications Engineering Center (DCEC), "Integrated AUTODIN System Architecture: A Systems Engineering Approach", June 1976.

(2) "DOD Data Internet Study", Phase II Report, December 1974.

(3) DCEC Technical Paper, "Architectural Directions for an Integrated AUTODIN System", October 1977.

(4) DCEC, "A Preliminary IAS Requirements Definition", Draft, October 1977.

(5) A. P. Sage, Methodology for Large-Scale Systems, McGraw-Hill (1977).

(6) Naval Electronics Laboratory Center (NELC), TN 3234, "IASA Requirements Generation Methodology and Plan", 2 November 1976.

(7) DCEC Technical Note No. 24-77, "An Evaluation of Data Information Transfer Requirements for the Future DCS", November 1977.

(8) V. G. Cerf and A. Curran, "The Future of Computer Communications", Datamation (May 1977) pp. 105-114.

(9) DCEC, "Preliminary Report on the Impact of Facsimile on the DCS", November 1977.

(10) R. S. Wilkov, "Analysis and Design of Reliable Computer Networks", IEEE Trans. Commun., COM-20, No. 3 (June 1972).

(11) E. Hänsler et al, "Optimizing the Reliability in Centralized Computer Networks", IEEE Trans. Commun., COM-20, No. 3 (June 1972).

(12) D. W. Davies and D. L. A. Barber, Communication Networks for Computers, Wiley (1973), Chapter 12.

(13) D. E. Morgan et al, "A Survey of Methods for Improving Computer Network Reliability and Availability", Computer, IEEE Computer Society, 10, No. 11, (Nov 1977) pp. 42-50.

(14) K. Steiglitz et al, "The Design of Minimum-Cost Survivable Networks", IEEE Trans. Circ. Theory, CT-16, No. 4 (Nov 1969).

(15) H. Frank, "Survivability Analysis of Command and Control Communications Networks - Parts I and II", IEEE Trans. Commun., COM-22, No. 5 (May 1974).

(16) U. Grenander, Pattern Synthesis: Lectures in Pattern Theory - Vol. 1, Springer-Verlag, New York (1976).

(17) W. S. Brainerd, "Tree Generating Regular Systems", Information and Control, 14 (1969) pp. 217-231.

(18) R. E. Bellman and S. E. Dreyfus, Applied Dynamic Programming, Princeton University Press, Princeton (1962).

(19) H. Frank et al, "Optimal Design of Centralized Computer Networks", Networks, 1 (1971) pp. 43-57.

(20) M. Gerla, "Approximations and Bounds for the Topological Design of Distributed Computer Networks", (1973).

(21) R. R. Boorstyn and H. Frank, "Large-Scale Network Topological Optimization", IEEE Trans. Commun., COM-25, No. 1 (Jan 1977) pp. 29-47.

APPENDIX A

MATHEMATICAL CONCEPTS

APPENDIX A
MATHEMATICAL CONCEPTS

This appendix contains terminology, concepts, and definitions for grammars and automata. Examples are presented to illuminate the concepts. For further details on the formal aspects of trees see Reference [17].

In a tree system, the basic notions are:

- Tree domain
- Stratified alphabet
- Tree
- Term
- Regular system or tree grammar.

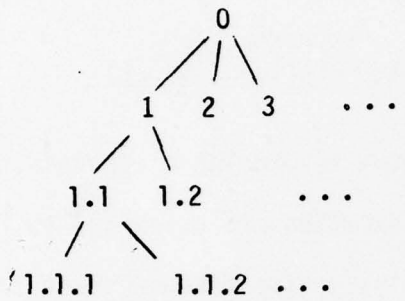
The following definitions present the theoretical foundation for these terms:

Definition 3.1. Let N^+ be the set of positive integers and U be the free monoid generated by N^+ with identity 0 and operation "$\cdot$". The depth of $a \in U$ is denoted d(a) and defined by:

$$d(0) = 0$$

$$d(a \cdot i) = d(a) + 1, \; i \in N^+$$

Write $a \leq b$ if there exists $x \in U$ such that $a \cdot x = b$. Elements a and b are incomparable if $a \not\leq b$ and $b \not\leq a$. This partial ordering on U, called the universal tree domain, is illustrated below:



Definition 3.2. D is a <u>tree domain</u> iff D is a finite subset of U which satisfies

(1) $b \in D$ and $a < b$ implies $a \in D$, and

(2) $a \cdot j \in D$ and $i < j$ in N^+ implies $a \cdot i \in D$.

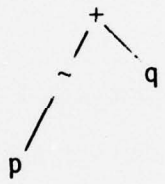
A labeled tree will be defined as a mapping from a tree domain into some set of symbols. However, only trees of a certain type will be considered; to this end, introduce a stratified alphabet.

Definition 3.3. A <u>stratified alphabet</u> (also called ranked or graded alphabet) is a pair (V, r) where V is a finite set of symbols and $r: V \to N$, where $N = N^+ \cup \{0\}$. For $x \in V$, $r(x)$ is called stratification of x. Let $V_n = r^{-1}(n)$.

Definition 3.4. A <u>tree</u> over (V,r) is a function $\alpha: D \to V$ such that D is a tree domain and $r[\alpha(a)] = \max\{i \mid a \cdot i \in D\}$. That is, the stratification of a label at a must be equal to the number of branches in the tree domain at a. The domain of a tree is denoted $D\alpha$. Let T_V denote the set of trees over V. Note that trees are generalizations of strings.

Definition 3.5. A <u>term</u> t over (V,r) is defined as $t = x \in V_0$, or $t = xt_1 \ldots t_n$, where $x \in V_k$ and t_i is a term $1 \leq i \leq n$. T_V is the set of terms over V. There is obviously a one-to-one correspondence between the terms over V and the trees over V.

Example: Let $V = \{+, \sim, p, q\}$, $r(+) = 2$, $r(\sim) = 1$, $r(p) = r(q) = 0$. In this case, T_V is the set of well-formed formulas of the propositional calculus involving one or two statement letters. The tree $\alpha = \{(0,+), (1,\sim), (1.1,p), (2,q)\}$, is usually written



The tree α represents the expression $(\sim p) + q$. This can also be represented in a Polish postfix notation as $p\sim q+$. Another possibility is to represent the tree α by a string $+\sim pq$ which corresponds to Polish prefix notation. The set of trees over V, written in Polish prefix form, is called the set of terms over V.

Definition 3.6. Let a,b,b' be members of U such that $b = a \cdot b'$. Then $b/a = b'$. Note that b/a is not defined unless $a \leq b$.

Definition 3.7. Let $\alpha \in T_V$ and $a \in D\alpha$. Let $\alpha/a = \{(b,x) | (a \cdot b, x) \in \alpha\}$. Then α/a is the <u>subtree</u> of α at a and α/a <u>occurs</u> at a in α.

Definition 3.8. Let $\alpha \in T_V$, $a \in U$. Then $a \cdot \alpha = \{(b,x) | (b/a, x) \in \alpha\}$.

Definition 3.9. Let $a \in D\alpha$, $\alpha, \beta \in T_V$. Let $\alpha(a \leftarrow \beta) = \{(b,x) \in \alpha \mid b \not\geq a\} \cup a \cdot \beta$. This is the result of replacing the subtree α/a at a by the tree β.

Definition 3.10. A <u>regular system</u> R or tree grammar over (V_T, r) is a system (V, r', G, S) which satisfies the following conditions:

(V, r') is a finite-ranked alphabet with $V_T \subseteq V$ and $r' | V_T = r$. The elements of $V - V_T$ are called the non-terminal symbols or variables. The variables will be denoted by $X_0, X_1, X_2, \ldots, X_{n_v}$ where n_v denotes the number of variables.

G is a finite set of generators (also called rules or productions) of the form $\Phi \rightarrow \Psi$ where Φ and Ψ are trees over (V, r').

S is a finite subset of T_v whose elements are called the axioms, where T_v is the set of trees over alphabet V.

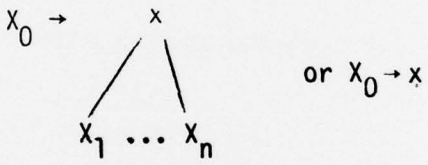
* * * *

In the general case, the regular system R defined above generates trees as follows: $\alpha \xrightarrow{a} \beta$ is in R if there is a generator $\Phi \rightarrow \Psi$ in G such that $\alpha/a = \Phi$ and $\beta = \alpha(a \leftarrow \Psi)$. That is, Φ is a subtree of α at a; β is obtained by replacing the occurrence of Φ at a by Ψ. Denote by $\alpha \rightarrow \beta$ in R iff there exists $a \in D(\alpha)$ such that $\alpha \xrightarrow{a} \beta$. Also $\alpha \Rightarrow \beta$ is in R iff either $\alpha = \beta$ or there is a finite sequence $\alpha_0, \alpha_1, \ldots, \alpha_m$, $m > 0$ such that

$\alpha = \alpha_0 \rightarrow \alpha_1 \rightarrow \cdots \rightarrow \alpha_m = \beta$ in R. The sequence $\alpha_0, \ldots, \alpha_m$ is called a derivation of β from α and m is the length of the derivation. α is a derivation of length 0 if $\alpha \Rightarrow \alpha$ for all $\alpha \in T_V$.

Definition 3.11. $L(R) = \{\alpha \in T_{V_T} \mid$ there exists $Y \in S$ and $Y \Rightarrow \alpha\}$ is called the language generated by R over the terminal elements V_T.

Definition 3.12. A tree grammar $R = (V, r', G, S)$ over (V_T, r) is expansive iff each generator in G is of the form



or $X_0 \rightarrow x$

where $x \in V_T$ and $X_0, X_1, \ldots, X_n$ are non-terminals.

A theorem is given in Reference 17 which shows how to effectively construct an equivalent expansive grammar for any regular tree grammar.

It was noted above that there is a one-to-one correspondence between the terms over V and the trees over V. If the trees generated by the regular system R are expressed as terms, in either postfix or prefix notation, then these terms form a context-free set. It is also of interest to note that the sets produced by the regular system are recognizable (i.e., accepted) by machines which are generalizations of the usual finite automata. This also provides a useful notation for the implementation of the regular system generating scheme on a digital computer. For completeness, the following definition is included:

Definition 3.13. Let (V,r) be a stratified alphabet, where $V = \{x_1, x_2, \ldots x_k\}$. A finite tree automaton over V is a system $M = (Q, t_1, t_2, \ldots, t_k, F)$ where

a. Q is a finite set of states

b. For each i, $1 \leq i \leq k$, t_i is a relation on $Q^{r(x_i)} \times Q$, and

c. $F \subseteq Q$ is a set of final or accept states. If each t_i is a function $t_i : Q^{r(x_i)} \rightarrow Q$, then M is deterministic (otherwise non-deterministic).

Example: The elements $V = \{A,E,D,L,B,I,1,2,3\}$ in the tree grammar (V,r,G,S) where $V_T = \{A,E,I,D,L,B\}$ and the function r is specified by

r(A) = 1

r(E) = 1 or 2

r(D) = 0

r(B) = 1 or 2

r(L) = 0

r(I) = 0

and the generators are

(a) S → A
|
1

(b) 1 → E
|\
L 2

(c) 1 → E
|
2

(d) 2 → B
|
3

(e) 2 → B
|\
I 3

(f) 3 → D

This grammar generates trees representing logical architectures for a simple communication network. The terminal symbols are interpreted as follows:

A - addressing

E - encryption

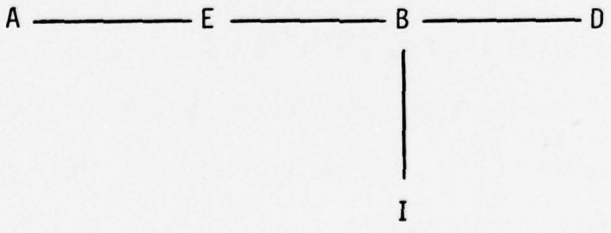
L - local distribution

B - backbone switching

D - decryption

I - interface (gateway)

The generation process begins with the application of a generator containing the start symbol "S". This generator, labeled (a), is followed by the application of one of the generators which rewrites the variable "l". That is, the generator (b) or (c) is applied. The generators are applied until the tree contains only terminal symbols. Note that the choice of generators to apply represents a decision to be made. An example of an architecture produced by this grammar is



* * * *

In the subsequent discussion, the trees are simplified to strings. That is, the stratification of each element in V is ≤ 1. This is a reasonable assumption for modeling logical architectures, in which concurrent functions are represented by serial organizations.

The following section contains the definitions required to define a weight function on the syntactic structure of the regular system (Ref. [16]). Denote by G_X the generators which rewrite variable X so that

$$G = \bigcup_{X \in V - V_T} G_X$$

Definition 3.14. For each $X \in V - V_T$, introduce a weight distribution over G_X which satisfies

$$w_g \geq 0 \qquad g \in G_X$$

$$\sum w_g = 1$$

where the summation is carried for all g in G_X.

For a given generation tree T, denote by G(T) the set of generators used at the nodes of T taking into account the number of times the generator has been applied (i.e., its multiplicity). Then the weight associated with the tree T is given by

$$W(T) = \prod w_g$$

where the factor w_g appears as many times as the index g occurs in the set G(T), and the product is carried over all g in G(T).

Definition 3.15. For an arbitrary string $\sigma \in L(R)$ the weight associated with σ is given by

$$W(\sigma) = \sum W(T)$$

where T_σ denotes the set of generation trees which produce the strings σ, and the summation is carried over all trees T in T_σ.

The best logical architecture is defined as that string σ which maximizes the weight for all strings σ in the language L(R). The computation which might be required in the generation of all the strings in L(R) might be formidable. In place of this direct approach of generation of all architectures for ranking, the following may be considered. To simplify the discussion, assume that the grammar is unambiguous. Then the generation tree for each string is unique and $W(\sigma) = W(T_\sigma) = \Pi w_g$, where the product is taken over all g in G(T). Then the weight measure can be denoted w(X,X') which corresponds to the generator for variable X rewritten as variable X'. In the case of the terminating generator the weight vector w(X) contains the corresponding weight for each variable. Then the weight associated with a string is given by $W(\sigma) = w(X_{i_1}, X_{i_2}) \cdot w(X_{i_2}, X_{i_3}) \ldots\ldots w(X_{i_{m+1}})$.

Since the derivation is unambiguous, the variable sequence (equivalently, the sequence of states) replaces the variables and the maximization problem for the best architecture has the form: find $i_2, i_3 \ldots\ldots i_m$ to maximize

$$\prod_{n=1}^{m} w(i_n, i_{n+1})$$

To reduce the computational work to solve this problem, reformulate it as follows:

$$\text{Let } L(j;k) = \max \prod_{n=j}^{m} w(i_n, i_{n+1}) \text{ over } i_j, i_{j+1}, \ldots, i_{m+1},$$

where $i_j = k$ and i_{m+1} is fixed. Then the recursion

$$L(j=1;1) = \max_k \{w(1,k) \cdot L(j,k)\}$$

is used to produce the solution. Denote by $k(j,1)$ the index for which $L(j=1;1)$ is attained. Then using $L(m;1) = w(1,m+1)$ the recursion is solved for the successive values $j=m,m-1,\ldots,2$. For this special case outlined above, the solution is developed as a dynamic programming problem. This is based on the Bellman principle of optimality: "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision" (Ref. [18]).

This formulation provides efficient determination of the "best" logical architecture(s). Moreover, it permits efficient sensitivity analysis to be carried out by varying the weights over a critical range of values to observe how the optimal policy is affected by these changes.