JULY 1989 LIDS-P-1890

REQUIREMENTS SPECIFICATION USING THE CUBE TOOL METHODOLOGY*

Didier M. Perdu**

Alexander H. Levis***

ABSTRACT

The distributed nature of command and control requires the consideration of both processes and communications in the formulation of requirements. Cube Tool is a methodology used to derive the processing and communication needs for each system function. An approach is introduced for extending the applicability of Cube Tool to the determination of requirements for C3I systems. First, using Cube Tool, for each function, a Petri Net is derived that models all processes and communications for the correct execution of the function. Then, for a given scenario, these nets are interconnected and the steps of the methodology are applied again to derive the Petri Net that represents the mission-dependent requirements for the system.

* To appear in *Proc. 1989 Symposium on C2 Research,* National Defense University, Washington, DC, June 1989. This work was carried out at the MIT Laboratory for Information and Decision Systems with support provided in part by THOMSON-CSF, CIMSA-SINTRA Division, Boulogne-Billancourt, France and in part by the Basic Research Group of the Joint Directors of Laboratories through the Office of Naval Research under contract N00014-85-K-0782.

** The first author is with Thomson-CSF, CIMSA-SINTRA Division, Boulogne-Billancourt, France.

*** The second author is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139.

INTRODUCTION

The determination of the functional requirements of a system is usually done by representing the relationships among the different processes which have to take place for the execution of a mission. When the systems are distributed, the requirements must includenot only the processes, but also the communications among the different parts of the system. The Cube Tool has been developed as a methodology for deriving the processing and communication needs for each system function. In this paper, the methodology is extended to address the determination of system requirements and their representation in terms of Petri Nets.

Cube Tool (Tournes, 1988) is a methodology developed at THOMSON-CSF in France for the design and the analysis of C3I systems. The methodology allows for (1) the qualitative and quantitative design of the architecture of C3I systems; (2) the determination of the characteristics of the elements, which are known also as the attributes or parameters of the system; and (3) the definition of the general plan for realization. The Cube Tool covers the application domains which are common to all C3I systems: communication, information processing, information storage, supervision/management, and man/machine interface. The application of Cube Tool to the design and the analysis of a system is done in four steps, as shown in Figure 1.

- Identification of the system Functions and of the different resources (personnel and hardware/software) involved,
- Functional Analysis for the determination of the processing and information exchanges for each function,
- Quantitative Evaluation of Automated Data Processing (ADP) and communication loads in workstations,
- Consideration of different possible architectures through the allocation of the functions to different sites.

The first step consists of defining the system functions from the missions expected to be accomplished. Each function is divided in subfunctions. Simultaneously, the resources needed for the execution of these functions are defined. They consist of personnel and hardware/software entities such as databases or decision aids and are referred to as Actors. In a second stage, a functional analysis is performed for each function in a three dimensional space with axes coresponding to functions, actor and time. In this framework, subfunctions are defined as a collection of activities with their interrelated information exchanges. Each function and can be looked on three different planes in the 3-D space: Responsibilities (Actors-Functions), Actions (Actors-Time) and Sequences (Functions-Time). The main analysis is performed in the responsibility plane. Activities are differentiated according to the kind of processing they represent. For each function, the responsibility plane is constructed by allocating the activities to different actors.

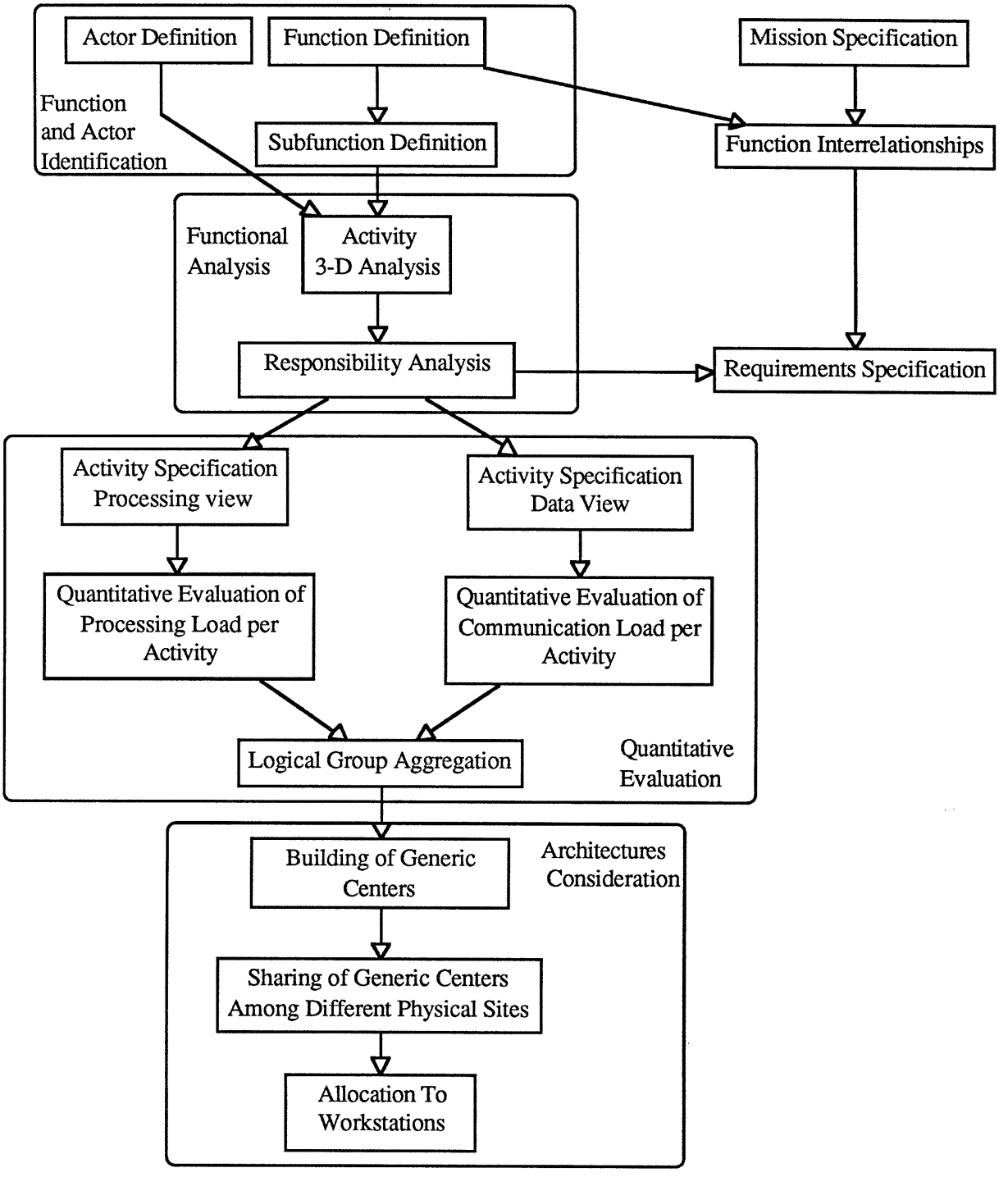


Figure 1 Methodology Flow Chart

The third step is the quantitative evaluation of Automated Data Processing and communication loads. To evaluate the processing load, each activity of the functional structure is defined using a pseudo-code formalism close to PASCAL or ADA. The number of queries to

databases, the kinds of display and the required computations are included by using a set of primitives gathered in a dictionary. To evaluate the communication load, the ways information is displayed and sent are analyzed for the incoming and outgoing data. A processing and communication load is assigned to each of these primitives. The processing and communication load quantification for each activity is made by summing the loads of the primitives used to describe the execution of this activity. Simultaneously, a quantification is made for the maximum response time to determine the minimum processing power threshold. By summing these estimates of each logical group (which is the set of activities related to a given system function and performed by a single actor,) the number and type of workstations, the processing requirements, the number of database updates and retrievals and the load associated with processing and related communication flows can be determined.

The last stage is the investigation of different architectures through the allocation of logical groups to different sites. Generic sites are first defined by gathering logical groups meant to operate together and sufficient to constitute an independent node. This is done to check data coherency. Then, the logical groups with their associated loads are assigned onto different sites according to the areas of responsibilities and interests specific to each logical group and to the different modes of operation (normal and backups). Within these new system sites, the load is reallocated to the different workstations according to the type of processing (scientific vs. expert system) and the security requirements. Different architectures can be obtained and the selection is made according to criteria such as cost or ease of implementation.

The first two stages, which are essential for the specification of the requirements of a system, are described in detail in the next section.

FUNCTION IDENTIFICATION AND RESPONSIBILITY ANALYSIS

System Functions and Actors Identification

The first stage of Cube Tool consists of identifying the functions of the system to be designed. At this stage, the designer must find out the user needs, the type of missions the system will have to accomplish, and the personnel and types of hardware and software which will be used. This process requires intensive interviews with the user to determine exactly what the range of operations of the system will be. The missions that the system is expected to carry out are determined and are used as the basis for the identification of the global tasks that must be executed for the fulfillment of a mission. These global tasks are the system Functions. For example, a system for planning an air interdiction mission will have as functions the determination of the status of allied forces, weather projection, threat assessment, strike assessment, target prioritization and development, weapon system availability, etc.

Then, each system function can be decomposed in subfunctions. The processing tasks are differentiated from the transmission tasks. A processing task only involves the processing of data received by an actor in charge of creating or inferring new information. Transmission tasks only involve the communication of information between two different actors without any alteration in the content (for example : digital communication, reading of a display, typing, or voice transmission). A function can be considered to be an interleaved sequence of processing and communication tasks, a subfunction can be defined as a single pair of a process task and a communication task. The execution of a function will require the sequential execution of its subfunctions.

Functional Analysis

The initial specification of system elements, activities, and information exchange is done through functional analysis in the three dimensions of the Cube Tool, as shown on Figure 2. The three axes of interest are :

- Functions: These are the processes which have to be executed for the fulfillment of the mission.
- Actors or Hierarchical Levels: These are the personnel and the hardware and software nodes responsible for executing the different tasks. Personnel are layered in hierarchical levels and are most of the time specialized per functional domain
- Time: This axis allows to define on the same time scale the execution time of the functions, their frequency and their sequence.

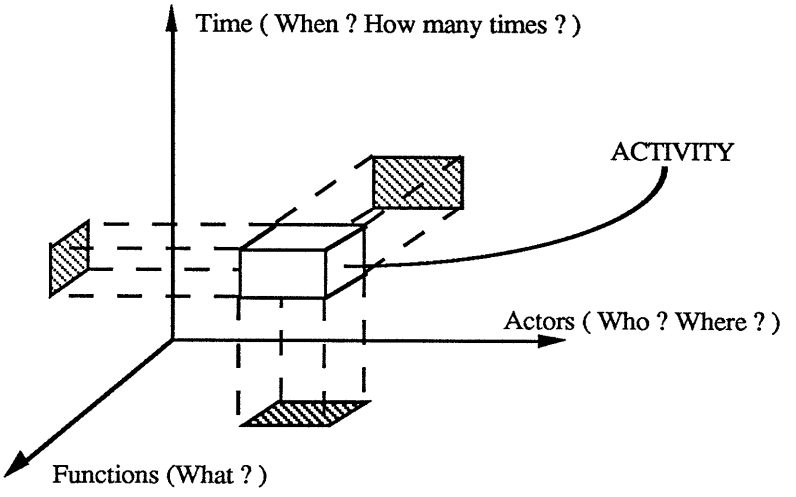


Figure 2 Three Dimensional Functional Analysis

In this analytical framework, a subfunction is composed of activities. An activity is defined as a process which supports a given system function and which is performed by a single actor or hierarchical level without major interruption. Therefore, activities can be part of a processing task, a communication task, or contain elements of both. Activities are differentiated according to the type of processing they represent and which are called roles. The roles considered by the method are:

- *Elaborate* (**E**): transform or generate information.
- *Acknowledge* (**A**): receive an order important enough to warrant the generation of an acknowledgement.
- *Check* (**C**): receive a report in response to an order previously generated.
- *Warn* (**W**): receive an information which does not require taking any measures in the current mode of operation.
- *Monitor* (**M**): receive an information on system operation allowing to accomplish command control and communication resources management.
- *Monitor Locally* (**L**): same as M but on a local basis
- *Secure* (**H**): exchange of secured data such as encryption keys, access keys and certification mechanisms of users trustworthiness.

These activities can be looked at from three different perspectives represented by the analysis planes defined by the three axes, as shown in Figure 3. These are:

- Responsibilities Plane (Functions / Actors): This plane shows which actor is in charge of a set of specific activities.
- Sequences Plane (Functions / Time): This plane shows when (and how many times) an activity will be executed.
- Actions Plane (Time / Actors): The plane of actions shows when actors are busy performing some activity.

The main analysis is performed in the responsibility plane. The roles which are used most and are the only ones considered for the requirements specification are E, A, C and W. The responsibility plane is constructed by allocating the roles for each subfunction to the different actors. This allocation must verify the following rules:

- There is one and only one role E per subfunction.
- Except for the first subfunction which starts the execution of a function, a role E can only be triggered by a role A or C.
- The presence of a role A requires the presence of a role C in the column of the actor which has generated the order. More generally the exchange which take place from a higher hierarchical level to a lower one is done by the presence of roles A, W. Exchanges which take place from a lower hierarchical level to a higher one are done by the presence of role C. The pairs E-A, E-W and E-C correspond to exchange of

information from the actor performing the role E to the actor performing the other role (A, W or C).

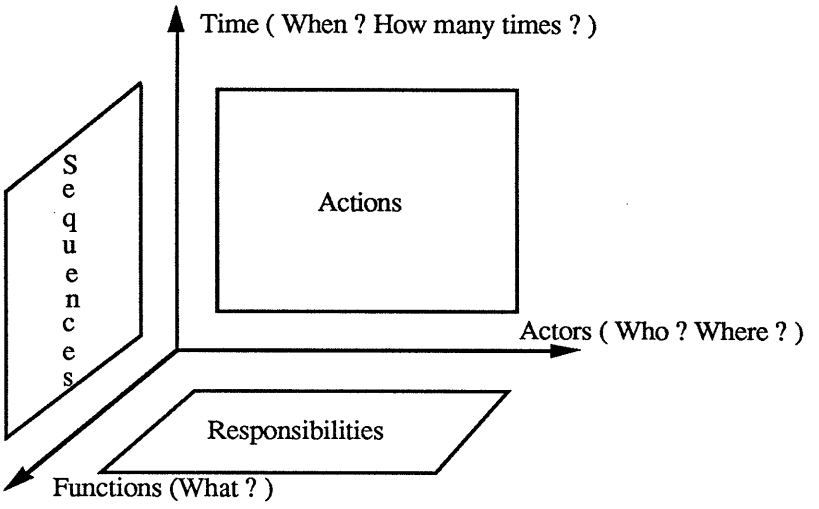


Figure 3 The three Analysis planes

This is illustrated in the example shown on Table 1.

Table 1: Responsibilities for a Function with six subfunctions performed by four actors

	actor 1	actor 2	actor 3	actor 4
subfunction 1	E	A	W	
subfunction 2	C	E	A	W
subfunction 3		C	E	A
subfunction 4			C	E
subfunction 5		C	E	
subfunction 6	C	E	W	

Explicit exchanges take place across columns, between activities contributing to the execution of the same subfunction (i.e., on same row). Implicit exchanges occur from row to row between activities performed by a single actor. The interesting aspect of this methodology is that several configurations, differing as to the resources used or reflecting variations in operational needs, can be represented in a consistent manner. This allows to define different thresholds of responsibilities in different modes (normal mode or emergency modes) and to point out how the reallocation of the tasks has to be made among the available actors when the system switches from one mode to another.

The next section shows how to convert the allocation of roles into Petri Nets and how the detailed requirements of a system for a particular mission can be generated.

PETRI NET REPRESENTATION OF REQUIREMENTS

The requirements of a system are the set of processes which have to take place for the correct execution of a mission. These requirements are scenario-dependent and are most often defined by the set of functions with their sequences and interrelationships. In Valraud (1989), the requirements are described by a Petri Net in which system functions are represented with transitions and the data produced by these functions, necessary for the execution of subsequent functions, with places. These nodes are connected together to model the relationships among functions and to show what should be their order of execution. This section describes how to develop more detailed requirements of a system which take into account not only the different processes which have to take place, but also the communication exchanges between the different parts of the system.

The Cube Tool can be used to define, for each function, the processes and the communication exchanges among the different actors involved in the execution of that function. The Petri Nets depict graphically these processes and communication exchanges for each function. When these representations are linked together to construct the requirements, a global and consistent graphical representation can be defined that lets the designer or the analyst take advantage of the mathematical framework which underlies Petri Nets.

Representing the Responsibilities for a Function with Petri Nets

As we have seen in the previous section, the first two steps of Cube Tool result in the definition of the different system functions, their subfunctions, and how the activities constituting these subfunctions are allocated to the different actors of the system. For each system function, the responsibility analysis plane defines the activities performed by the different actors. From this representation, the generation of the equivalent Petri Nets representation of the responsibilities for each function is done in three steps.

In the first step, each activity is depicted by a transition. The transitions representing the activities performed by the same actor are aligned horizontally, while the ones representing the activities belonging to the same subfunction are aligned vertically. In other words, the transpose of the array of responsibilities is obtained and the non-null elements of this array are transformed into transitions, as shown in the Figure 4.

A label is attached to each transition identifying (1) the function, (2) the subfunction to which the represented activity belongs, (3) the type of activity (E, A, C or W) and (4) the actor performing this activity. For example, in Figure 4, the label 1.3E3 means that the activity represented by the transition belongs to subfunction 3 of function 1, is of type E, and is performed by actor 3. In the application described in this paper, the subfunctions are identified by the identification number of the processing they represent throughout the system.

function 1

	subfunction 1	subfunction 2	subfunction 3	subfunction 4	subfunction 5	subfunction 6
actor 1	E	C				C
actor 2	A	E	C		C	E
actor 3	W	A	E	C	E	W
actor 4		W	A	E		

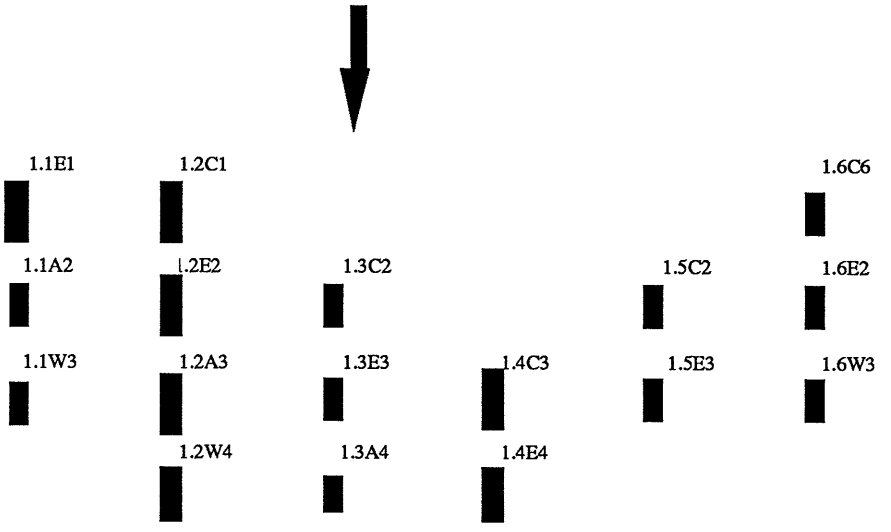


Figure 4 Drawing the transitions grid

The second step is to add places between the transitions representing the activities performed by a single actor and to connect them. In this way, the implicit information exchanges which take place between the successive activities performed by each actor are modeled. Figure 5 shows the net obtained for the example.

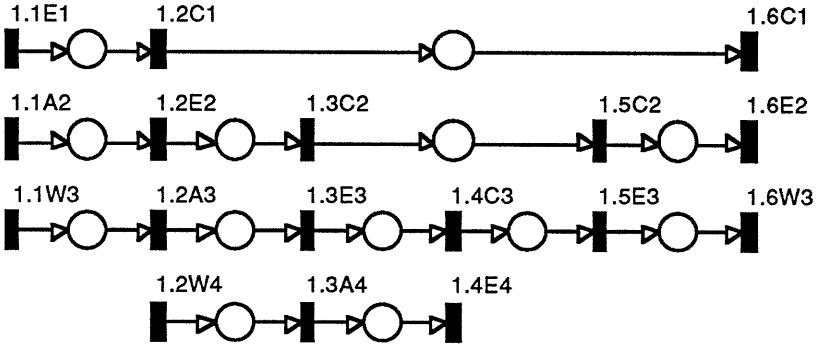


Figure 5 Adding Implicit Information Exchanges

The third step consists of adding the information exchanges which take place among the actors for each subfunction. Let us recall that in the Cube Tool methodology, an exchange originates from a role E and ends at a role A, W or C and that there is one and only one role E for

each subfunction. Therefore, for each column of the Petri Net representation obtained after the two first steps, the transition representing the role E is identified and is connected to the other transitions of the columns with a connector-place-connector set. Figure 6 shows the net obtained by adding these explicit information exchanges.

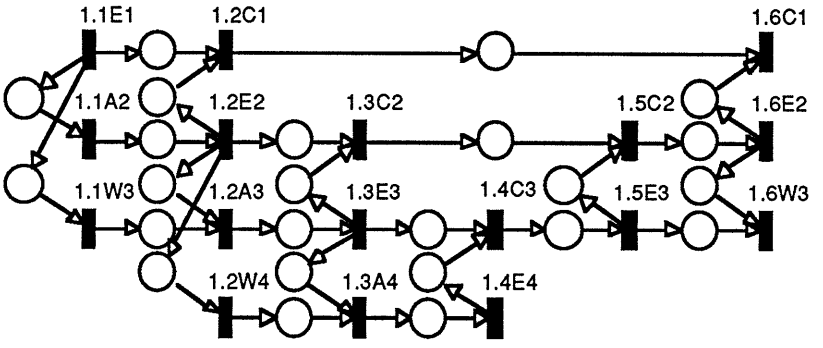


Figure 6 Add Explicit Information Exchanges

Modeling the requirements for a scenario

The procedures for modeling the detailed requirements for a given scenario is shown on Figure 7.

The definition of a scenario, that is a mission to be carried out, leads to the specification of the relationships and sequences of system functions. For the fulfillment of a mission, one can identify the system functions which can be executed concurrently as well as the functions which will have to be executed first to trigger the execution of a sequence of functions. These interrelationships among functionsvary from one scenario to another. Petri Nets are used to represent the sequencing and concurrency of functions so that the global requirements of a system can be derived. The procedure for determining the detailed requirements starts with the definition of the responsibilities for the chosen scenario. To list the functions on the Functions axis, the slices (Hillion,1986) of the Petri Nets representing the global requirements are computed. These slices represent the functions which can be executed concurrently. The functions are listed on this axis in the order of appearance in the slices list. Then, for each function, the actor which triggers the execution and gets the final report is identified. This actor is designated as the main one responsible for the execution of this functions. Once the main actors are listed on the Actors axis, the responsibility plane for the scenario can be constructed. For each function:

- A role E is placed on the cell defined by the function and by the main actor.
- Roles W are placed on the cells defined by the functions and by the main actors who are responsible for the execution of the subsequent functions as determined by the Petri Nets of the global requirements.

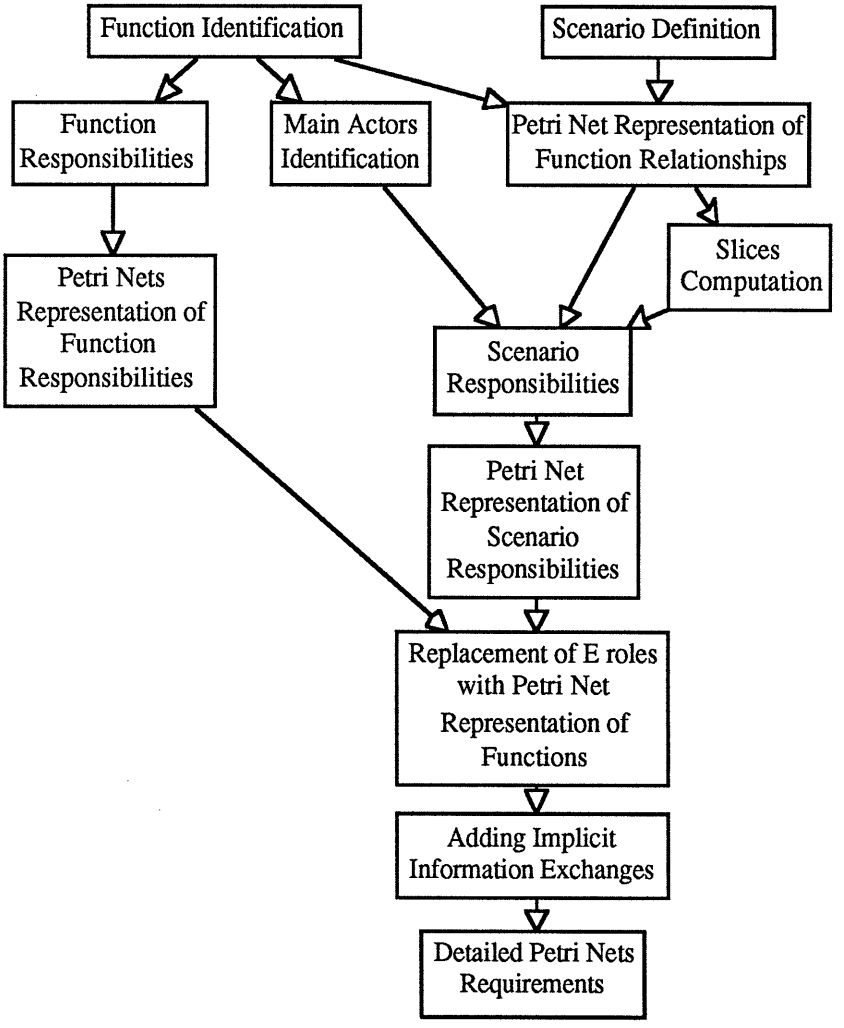


Figure 7 Procedures to Model the Detailed Requirements of a System

Let us consider an example where there are three functions: f1, f2 and f3, and three actors (A1, A2 and A3). The scenario specification has determined that f1 and f2 have to be executed before f3. The Petri Net is shown on Figure 8 and the slices are:

Slice 1: f1, f2

Slice 2: f3

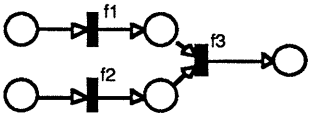


Figure 8 An example of Global Requirements represented with Petri Nets

The responsibilities specification of each function show that A1 is the main actor for f1, A2 for f2, and A3 for f3. The scenario responsibility plane is constructed by placing a role E in the cells (f1, A1), (f2, A2) and (f3,A3) and a role W in the cells (f1, A3) and (f2, A3), as shown on Table 2.

Table 2 Example of Scenario Requirements

	A1	A2	A3
f1	E		W
f2		E	W
f3			E

From the information in the scenario responsibilities plane, the equivalent Petri Net can be constructed following the same procedure that was used for the functions. The next step is to replace each transition representing a function with the equivalent representation of the responsibilities of this functions. By adding the implicit exchanges among actions for each actor, the Petri Net of the detailed requirements is constructed.

In the next section, an application of the methodology to a system for planning air interdiction missions is presented.

A PLANNING SYSTEM FOR AIR INTERDICTION MISSION

The system used to illustrate the methodology is a fictitious one called MESACC, which stands for Modular, Endurable, Survivable, Austere, Command Center and which has been studied by Valraud (1989).

The objective of an air interdiction mission system is to plan operations against the enemy's military potential before it can be effectively used against friendly forces. These operations restrict the combat capability of the enemy by:

- delaying, disrupting, or destroying his lines of communications
- destroying enemy supplies
- attacking fixed, moving and movable point and area targets
- destroying unengaged or uncommitted enemy attack formations before they can be brought into the battle.

The result of these operations is to disrupt enemy plans and time schedules. The integration of air interdiction operations with the fire and maneuver plans of surface forces is not required. However, these offensive air operations are planned and conducted as part of the unified effort of all friendly forces. Therefore, air interdiction demands precise coordination in timing.

Function Identification

The identification of the functions of the system requires the examination of the context and the environment in which the system operates. The context consists of the geographical characteristics of the battle area. It is assumed that the system is operating in Europe, and more specifically in the central region (CENTAG). The environment consists of the friendly forces, their assets, strength, current plans and orders, the enemy forces, their assets, strength, current plans and orders. Also, the current weather is part of the environment as it is a particularly important factor in air interdiction mission planning. The functions needed to plan an Air Interdiction Mission are listed in Table 3.

Table 3 System Functions of MESACC

Function #	Description
1	Weather Projection
2	Format Messages/Information Fusion/Update Database
3	Status of Allied Forces
4	Strike Assesment
5	Threat Assesment
6	Current Intelligence
7	Target Development/Prioritization
8	Aimpoint Construction/Weaponeering
9	Penetration/Attrition Analysis
10	Mission Planning
11	Weapon System Availabilty

Let us describe briefly each of these functions:

- *weather projection*: This function forecasts the weather from the current weather reports.
- *format messages / information fusion / update databases* This function transforms the format of the various data inputs into a common format. The function performs also decoding. Then, this function updates the current information as new messages come in the system. For example, if the database contains the position of a particular enemy battalion, and later an intelligence report confirms that the battalion has moved to another position, then the function is used to update the position of that battalion, its strength, and current plans.
- *status of allied forces*: This function is used to assess the current state of the allied forces, number of troops, equipment, and of available aircraft for missions.
- *strike assessment*: This function updates the situation on the battlefield, as a result of previous air interdiction missions.
- *threat assessment*: This function evaluates the threat of the enemy forces in the different subareas of the battlefield.

- *intelligence report*: under certain circumstances, reports from intelligence may be requested when the uncertainty about some parameters of the problem is deemed too high.
- *target prioritization/target development*: This function is needed to prioritize the most important objective to be destroyed, given the situation. The resources that can be used for the next mission may be scarce so that it may not be possible to allocate assets to all objectives.
- *aimpoint construction/weaponeering*: This function provides the coordinates of the target, and allocates certain classes of friendly assets according to the objective and its intrinsic characteristics.
- *penetration/attrition analysis*: This function forecasts the degree of redundancy that is adequate for each objective. Different platforms may be assigned the same objectiveto protect friendly assets. Therefore, redundancy insures a greater degree of certainty over the outcome of the mission.
- *mission planning*: this function delivers the final output of the system to the environment. It consists of a set of missions with the objectives, the type of aircraft to be used, its armament, the number of aircraft to be used for each objective, the route to be followed, and the time to perform the mission.
- *weapon system availability*: This function describes what weapons are available for the mission at the time it is planned. This function tells what is available according to the weather forecasts (some aircraft cannot fly under certain circumstances), and the status of the allied forces (losses, use of reserves).

For the identification of the actors, Valraud (1989) considers eleven workstations, located in two shelters, and seven databases. In addition, the intelligence center is considered as an actor providing the latest information about the situation. In this example, databases are considered to be actors because they are distributed and exchanges have to take place on the network to access them. These seven databases are:

- DB-wt: contains weather forecasts produced by f1.
- DB-wp: contains data about weapons availability, given the status of the allied equipment, and the weather forecasts.
- DB-en: contains data about the enemy .
- DB-ba: contains data about the situation on the battle field.
- DB-st: contains data about strike assessment, i.e., the result of f4.
- DB-th: contains data about threat assessment, i.e., the results of f5.
- DB-al: contains data about the allied forces.

There are, therefore, a total of nineteen actors as listed in Table 4.

Table 4 The Nineteen Actors of MESACC

Actor #	Description	Notation
1	Workstation 1	WS1
2	Workstation 2	WS2
3	Workstation 3	WS3
4	Workstation 4	WS4
5	Workstation 5	WS5
6	Workstation 6	WS6
7	Workstation 7	WS7
8	Workstation 8	WS8
9	Workstation 9	WS9
10	Workstation 10	WS10
11	Workstation 11	WS11
12	Intelligence Center	INC
13	Data Base Weather	DB-wt
14	Data Base Weapons	DB-wp
15	Data Base Enemy Forces	DB-en
16	Data Base Battlefield	DB-ba
17	Data Base Strike Assesment	DB-st
18	Data Base Threats	DB-th
19	Data Base Allied Forces	DB-al

For each function, subfunctions have been defined. Some of them are used in different functions and, for clarity, a unique identification number has been assigned to each one, which is used consistently in the figures and tables. The list is given in Table 5.

Table 5 Subfunctions used in MESACC

Subfunction #	Description	Subfunction #	Description
1	Acknowledge	27	Request Strike Assessment Data
2	Request Weather Data	28	Get Strike Assessment Data
3	Get Weather Data	29	Assess Strike
4	Deduce Weather Projection	30	Update Strike Assessment DataBase
5	Update Weather Data Base	31	Request Strike Report
6	Request Weather Projection	32	Generate Strike Report
7	Get Weather Projection	33	Request Threat Assessment Data
8	Request Allied Data	34	Get Threat Assessment Data
9	Get Allied Forces Data	35	Update Threat Assessment DataBase
10	Fuse Allied Forces Information	36	Modify Threat Assessment
11	Update Allied Forces DataBase	37	Rank Threats
12	Determine Allied Status	38	Request Targets List
13	Request Allied Report	39	Get Targets list
14	Get Allied Report	40	Request Available Weapons
15	Request Enemy Forces Data	41	Get Available weapons
16	Get Enemy Forces Data	42	Request Weapon Data
17	Fuse Enemy Forces Information	43	Get Weapon Data
18	Update Enemy Forces DataBase	44	Generate new Weapon status
19	Request Enemy Report	45	Generate List of Available Weapons
20	Generate Enemy Report	46	Request Current Intelligence
21	Request Battlefield Data	47	Get Current Intelligence
22	Get Battlefield Data	48	Aimpoint Construction Weaponeering
23	Fuse Battlefield Information	49	Perform Analysis
24	Update Battlefield DataBase	50	Request Penetration Analysis
25	Request Battlefield Report	51	Get Penetration Analysis
26	Generate Battlefield Report	52	Plan Mission

Responsibility Specifications for each Function

The allocation of roles is illustrated for function f3, Status of Allied Forces. To determine the status of allied forces, Workstation WS7 needs to get information from the battlefield and to deduce from the last state of the allied forces the new status. Therefore, WS8 is queried to obtain a battlefield report (subfunction "Request Battlefield Report"). To do this, WS8 has to access the database Battlefield (subfunctions "Request Battlefield Data" and "Get Battlefield Data") to make the report that it sends to WS7 (subfunction "Generate Battlefield Report"). According to the data that WS7 has just received, WS7 accesses the allied forces data base (subfunction "Request Allied Data" and "Get Allied Data") to determine the new status of the allied forces (subfunction "Determine Allied Status"). Once this is performed, the allied forces database has to be updated (subfunctions "Update Allied Forces" and "Acknowledge"). The responsibility analysis plane is shown on Table 6.

Table 6 Responsibilities for Function f3

f3	Status of Allied Forces	Actor#	7	8	16	19
Subf#	Subfunction	Actor	WS7	WS8	DB-ba	DB-al
25	Request Battlefield Report		E	A		
21	Request Battlefield Data			E	A	
22	Get Battlefield Data			C	E	
26	Generate Battlefield Report		C	E		
8	Request Allied Data		E			A
9	Get Allied Forces Data		C			E
12	Determine Allied Status		E			
11	Update Allied Forces DataBase		E			A
1	Acknowledge		C			E

Figure 9 displays the Petri Net deduced from this responsibility analysis plane.

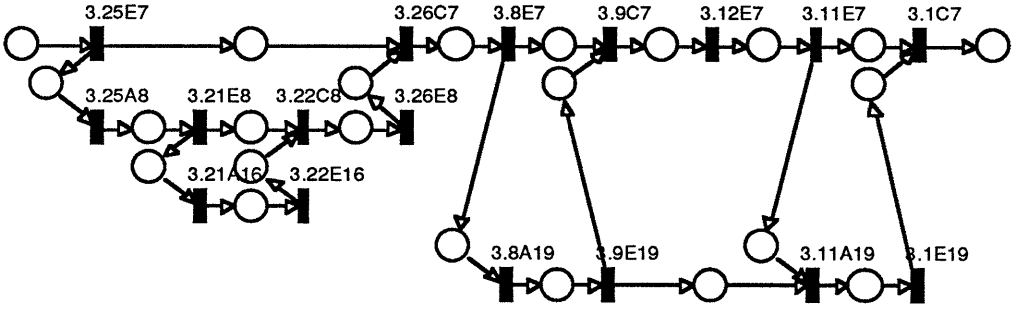


Figure 9 Petri Net for Function f3

By performing the same analysis for the other functions, the responsibilities planes described on Tables 7 to 15 and the Petri Nets displayed on Figures 10 to 19 are obtained.

Function 1: Weather Projection

Table 7 Responsibilities for Function f1

f1	Weather Projection	Actor#	1	13
Subf#	Subfunction	Actor	Ws1	Db-wt
2	Request Weather Data		E	A
3	Get Weather Data		C	E
4	Deduce Weather Projection		E	
5	Update Weather Data Base		E	A
1	Acknowledge		C	E

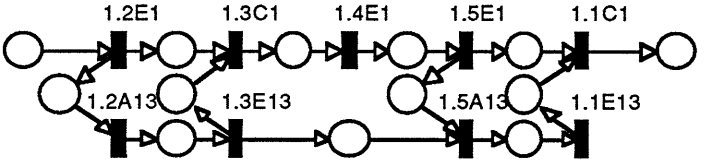


Figure 10 Petri Net for Function f1

Function 2: Format Messages/Fusion of Information/Update Database

Table 8 Responsibilities for Function f2

f2	Form. Mess./IF/Update DB	Actor#	7	15	8	16	11	19
Subf#	Subfunction	Actor	WS7	DB-en	WS8	DB-ba	WS11	DB-al
8	Request Allied Data						E	A
9	Get Allied Forces Data						C	E
10	Fuse Allied Forces Information						E	
11	Update Allied Forces DataBase						E	A
1	Acknowledge						C	E
15	Request Enemy Forces Data		E	A				
16	Get Enemy Forces Data		C	E				
17	Fuse Enemy Forces Information		E					
18	Update Enemy Forces DataBase		E	A				
1	Acknowledge		C	E				
21	Request Battlefield Data				E	A		
22	Get Battlefield Data				C	E		
23	Fuse Battlefield Information				E			
24	Update Battlefield DataBase				E	A		
1	Acknowledge				C	E		

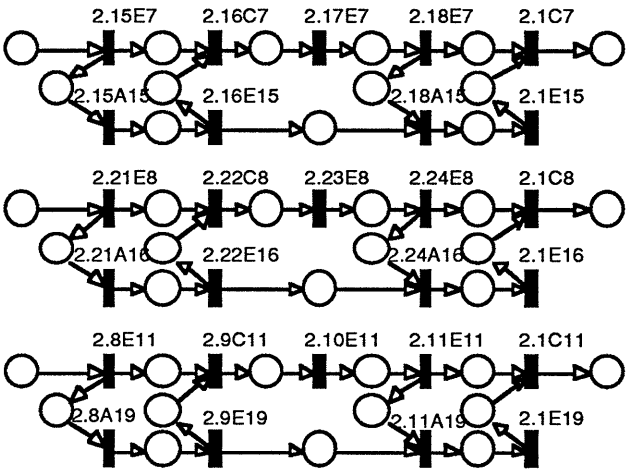


Figure 11 Petri Net for Function f2

Function 4: Strike Assessment

Table 9 Responsibilities for Function f4

f4	Strike Assesment	Actor#	7	15	8	17	18
Subf#	Subfunction	Actor	WS7	DB-en	WS8	DB-st	DB-th
27	Request Strike Assessment Data				E	A	
28	Get Strike Assessment Data				C	E	
19	Request Enemy Report		A		E		
15	Request Enemy Forces Data		E	A			
16	Get Enemy Forces Data		C	E			
20	Generate Enemy Report		E		C		
33	Request Threat Assessment Data				E		A
34	Get Threat Assessment Data				C		E
29	Assess Strike				E		
30	Update Strike Assessment DataBase				E	A	
1	Acknowledge				C	E	

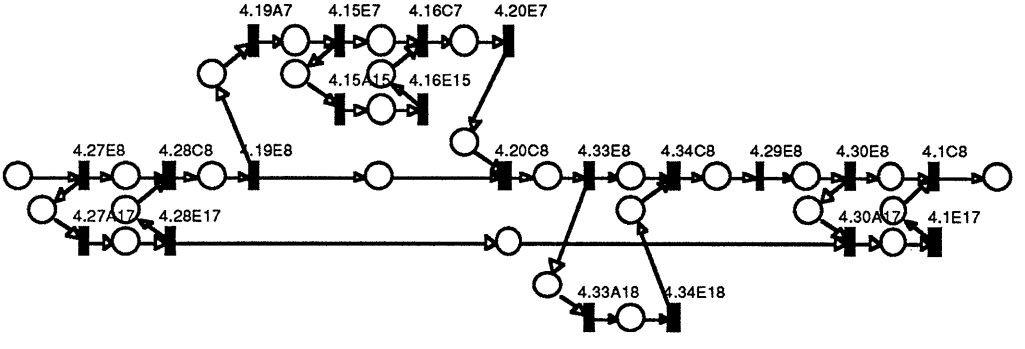


Figure 12 Petri Net for Function f4

Function 5: Threat Assessment

Table 10 Responsibilities for Function f5

f5	Threat Assesment	Actor#	7	15	8	16	9	18
Subf#	Subfunction	Actor	WS7	DB-en	WS8	DB-ba	A9	DB-th
33	Request Threat Assessment Data		W		W		E	A
34	Get Threat Assessment Data						C	E
19	Request Enemy Report		A				E	
15	Request Enemy Forces Data		E	A			C	
16	Get Enemy Forces Data		C	E				
20	Generate Enemy Report		E				C	
25	Request Battlefield Report		E		A			
21	Request Battlefield Data				E	A		
22	Get Battlefield Data				C	E		
26	Generate Battlefield Report				E		C	
35	Update Threat Assessment DataBase						E	A
1	Acknowledge						C	E

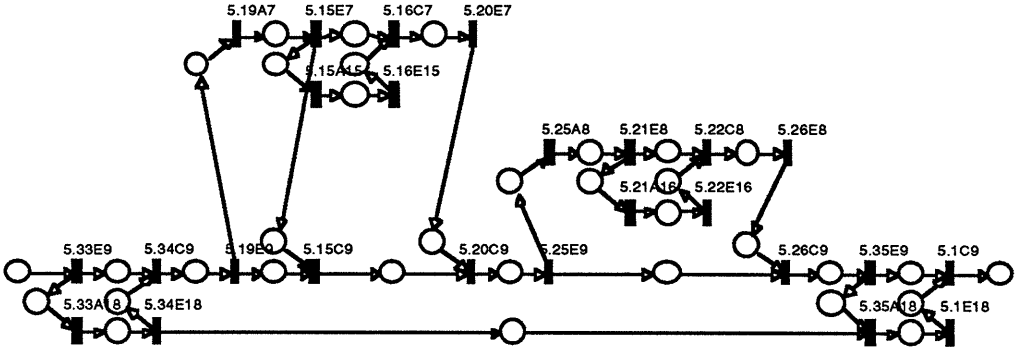


Figure 13 Petri Net for Function f5

Function 6: Current Intelligence

Table 11 Responsibilities for Function f6

f6	Current Intelligence	Actor#	9	12
Subf#	Subfunction	Actor	WS9	INC
46	Request Current Intelligence		E	A
47	Get Current Intelligence		C	E
36	Modify Threat Assessment		E	

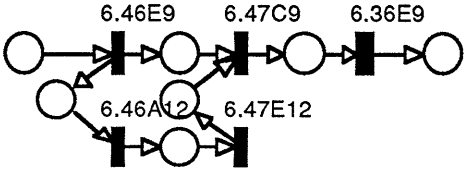


Figure 14 Petri Net for Function f6

Function 7: Target Prioritization/Target Development

Table 12 Responsibilities for Function f7

f7	Target Develop./Priorit.	Actor#	2	6	16	18
Subf#	Subfunction	Actor	WS2	WS6	DB-ba	DB-th
33	Request Threat Assessment Data		E			A
34	Get Threat Assessment Data		C			E
25	Request Battlefield Report		E	A		
21	Request Battlefield Data		C	E	A	
22	Get Battlefield Data			C	E	
26	Generate Battlefield Report		C	E		
37	Rank Threats		E			

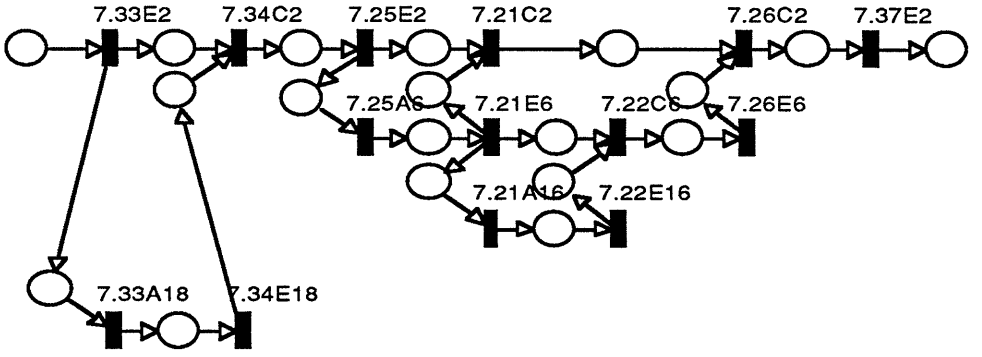


Figure 15 Petri Net for Function f7

Function 8: Aimpoint Construction/ Weaponeering

Table 13 Responsibilities for Function f8

f8	Aimpoint Construct./Weapon	Actor#	4
Subf#	Subfunction	Actor	WS4
48	Aimpoint Construction Weaponeering		E

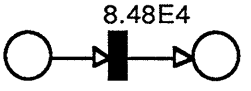


Figure 16 Petri Net for Function f8

Function 9: Penetration/Attrition Analysis

Table 14 Responsibilities for Function f9

f9	Penetration/Attrition Anal.	Actor#	3	6	15	17	10	19
Subf#	Subfunction	Actor	WS3	WS6	DB-en	DB-st	WS10	DB-al
19	Request Enemy Report		E				A	
15	Request Enemy Forces Data				A		E	
16	Get Enemy Forces Data				E		C	
20	Generate Enemy Report		C				E	
31	Request Strike Report		E	A				
27	Request Strike Assessment Data		C	E		A		
28	Get Strike Assessment Data			C		E		
32	Generate Strike Report		C	E				
8	Request Allied Data		E					A
9	Get Allied Forces Data		C					E
49	Perform Analysis		E					

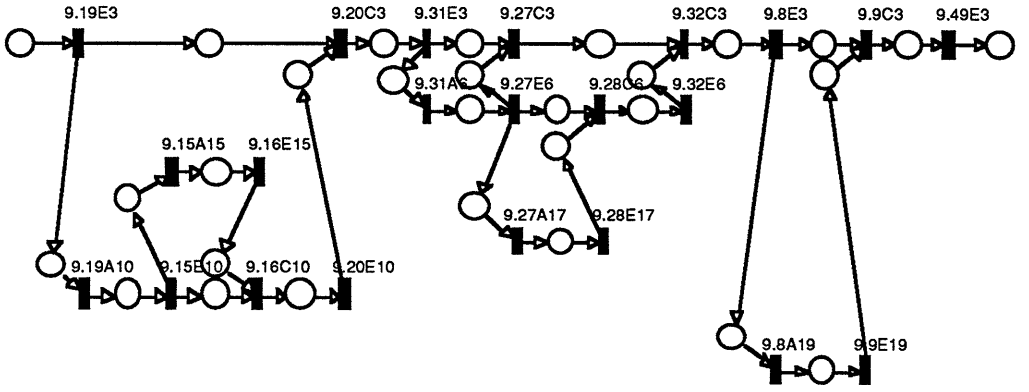


Figure 17 Petri Net for Function f9

Function 10: Mission Planning

Table 15 Responsibilities for Function f10

f10	Mission Planning	Actor#	1	2	3	4	5
Subf#	Subfunction	Actor	WS1	WS2	WS3	WS4	WS5
38	Request Targets List			A		E	
39	Get Targets list			E		C	
40	Request Available Weapons					E	A
41	Get Available weapons					C	E
6	Request Weather Projection		A			E	
7	Get Weather Projection		E			C	
50	Request Penetration Analysis				A	E	
51	Get Penetration Analysis				E	C	
52	Plan Mission					E	

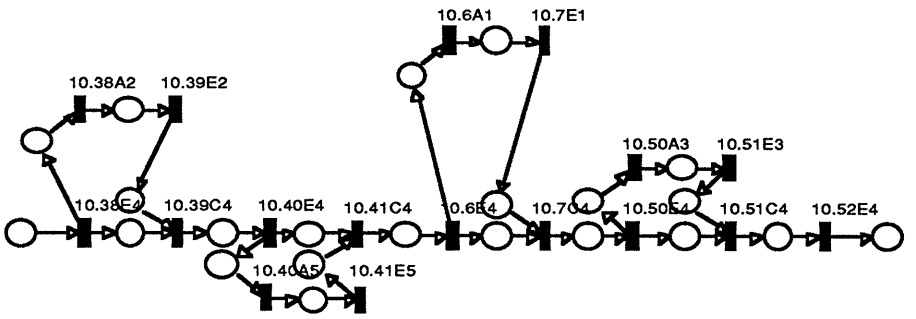


Figure 18 Petri Net for Function f10

Function 11: Weapon System Availability

Table 16 Responsibilities for Function f11

f11	Weapon System Availability	Actor#	13	5	14	10	19
Subf#	Subfunction	Actor	DB-wt	WS5	DB-wp	WS10	DB-al
42	Request Weapon Data			E	A		
43	Get Weapon Data			C	E		
13	Request Allied Report			E		A	
8	Request Allied Data			C		E	A
9	Get Allied Forces Data					C	E
14	Get Allied Report			C		E	
44	Generate new Weapon status			E	A		
1	Acknowledge			C	E		
2	Request Weather Data		A	E			
3	Get Weather Data		E	C			
45	Generate List of Available Weapons			E			

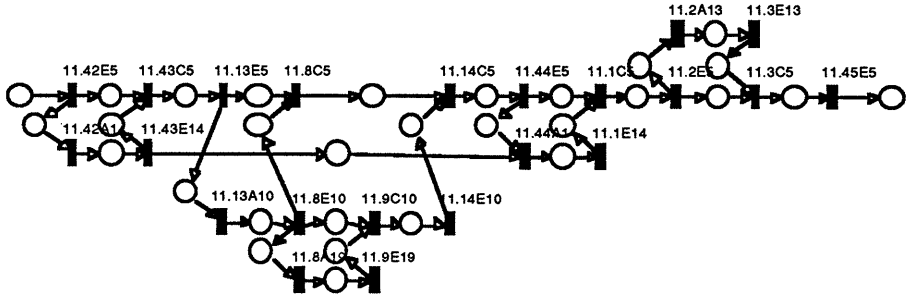


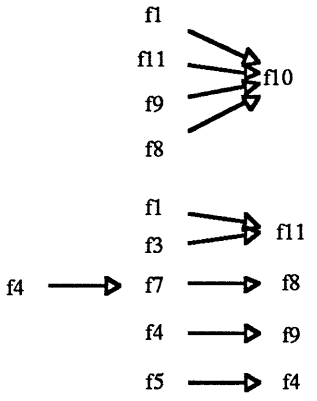
Figure 19 Petri Net for Function f11

The Global Functional Requirements

A specific scenario is considered next. It is assumed that the hostilities started two days ago. Although the enemy has gained ground on the battlefield, the friendly forces resist the pressure, and major assets in reserve have not been committed on either side. The conflict is a conventional one. The friendly forces and the enemy forces have both fairly accurate information about the situation on the other side. Each side knows what the resources are on the opposing side, as well

as the location of these assets, although some uncertainty remains. In certain areas, the battle line is difficult to assess. Therefore, there is a need to use MESACC to plan long distance, high altitude interdiction missions. Since the conflict started two days ago, the database already exists. All the other functions described earlier are in use.

The various data inputs from the sensors are weather reports, reports on friendly and enemy forces (strength, position, status), combat reports, request from the local Command Center for assistance, mission reports, and current and future operations plans. The output is unique and consists of air interdiction mission plans. The interrelationship between the various functions is as follows:



It should be understood that this description of the interrelationship between functions is purely functional. If a function is derived from another, it does not mean that the input of that function is sufficient. Indeed, data from the context may be necessary (terrain information for example). The global functional requirements of MESACC are described by the Petri Net of Figure 20. This Petri Net contains only one switch, s1, which represents the optional use of the "Current Intelligence" function, f6.

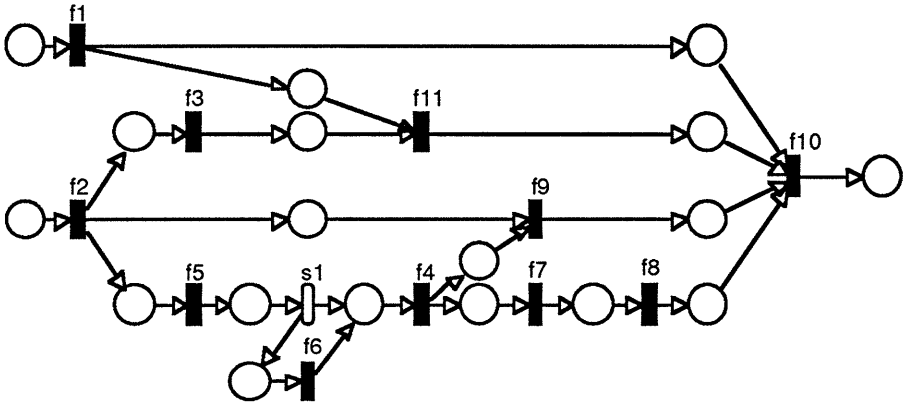


Figure 20 MESACC: The Global Functional Requirements

Detailed Requirements

The Petri Net obtained from the global functional requirements is used to determine the slices of the Net. The slices are:

Slice 1: f1, f2.
Slice 2: f3, f5.
Slice 3: f6.
Slice 4: f4,f11.
Slice 5: f7, f9.
Slice 6: f8.
Slice 7: f10.

To determine how data are transmitted among functions, Cube Tool has to be applied once more. The purpose is to define the global responsibility for the scenario. In the definition of the responsibilities for each function, only one actor triggers the execution and gets the final report. By looking at the global functional requirements, and at the different responsibility planes, one can identify where the output of each function has to be sent in order to generate the Scenario Responsibilities Plane shown on Table 17. The derived Petri Net is shown on Figure 21.

Table 17 Scenario Responsibilities

	WS1	WS2	WS3	WS4	WS5	WS7	WS8	WS9	WS10
f1	E			W	W				
f2			W			W	W	W	E
f3					W	E			
f5								E	
f6							W	E	
f11				W	E				
f4		W	W				E		
f7		E		W					
f8				E					
f9			E	W					
f10				E					

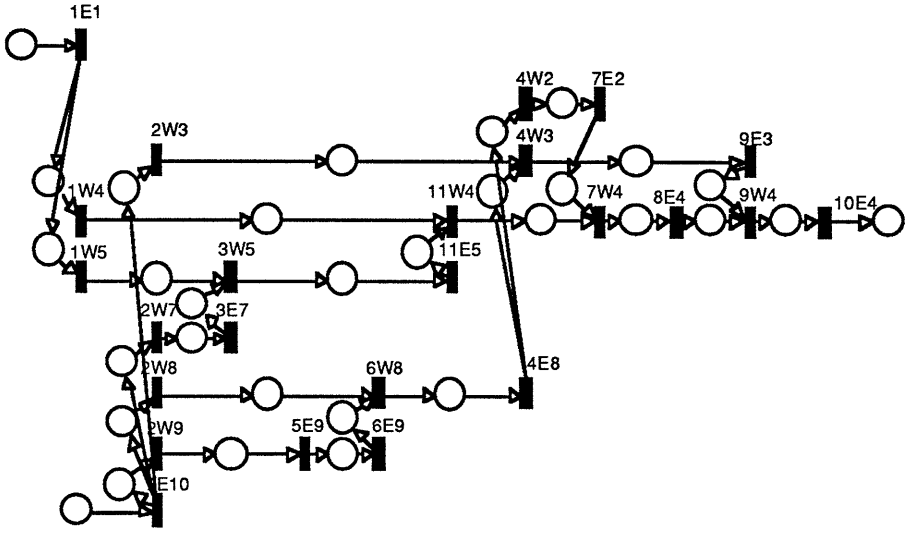


Figure 21 Petri Net of the Scenario

The representation of the detailed requirements is obtained (1) by replacing each transition containing the letter E with the Petri Net representation of the responsibilities of the functions this role E models and (2) by adding the implicit information exchanges between the functions performed by a single actor. Figure 22 displays the detailed requirements of MESACC.

CONCLUSION

Cube Tool has been extended from functions to systems. A methodology for deriving structural requirements has been proposed. It is used to represent with the Petri Net formalism the processes and communications which take place for the correct execution of a mission. This methodology fills a gap between the description of requirements and the quantitative models needed for the analysis and evaluation of C3I systems designs.

REFERENCES

Hillion, H. P. and Levis A. H. (1987). Performance Evaluation of Decisionmaking Organizations, *Proc. 1987 Symposium on C2 Research*, National Defense University, Fort Mac Nair, Washington, DC.

Peterson, J. L. (1980). *Petri Net Theory and the Modeling of Systems*. Prentice Hall, Inc., Englewood Cliffs, NJ.

Reisig, W. (1985). *Petri Nets, An Introduction.* Springer Verlag, Berlin.

Tournes, C. (1988). Cube Tool a C3I Specification Oriented Tool, *Proceedings of the 9th AFCEA European Symposium and Exposition*, Brussels, Oct. 1988.

Valraud F. R. H. and Levis A. H. (1989).On the Quantitative Evaluation of Functionality in C3 Systems. *Proc. 1989 Symposium on C2 Research,* to appear.

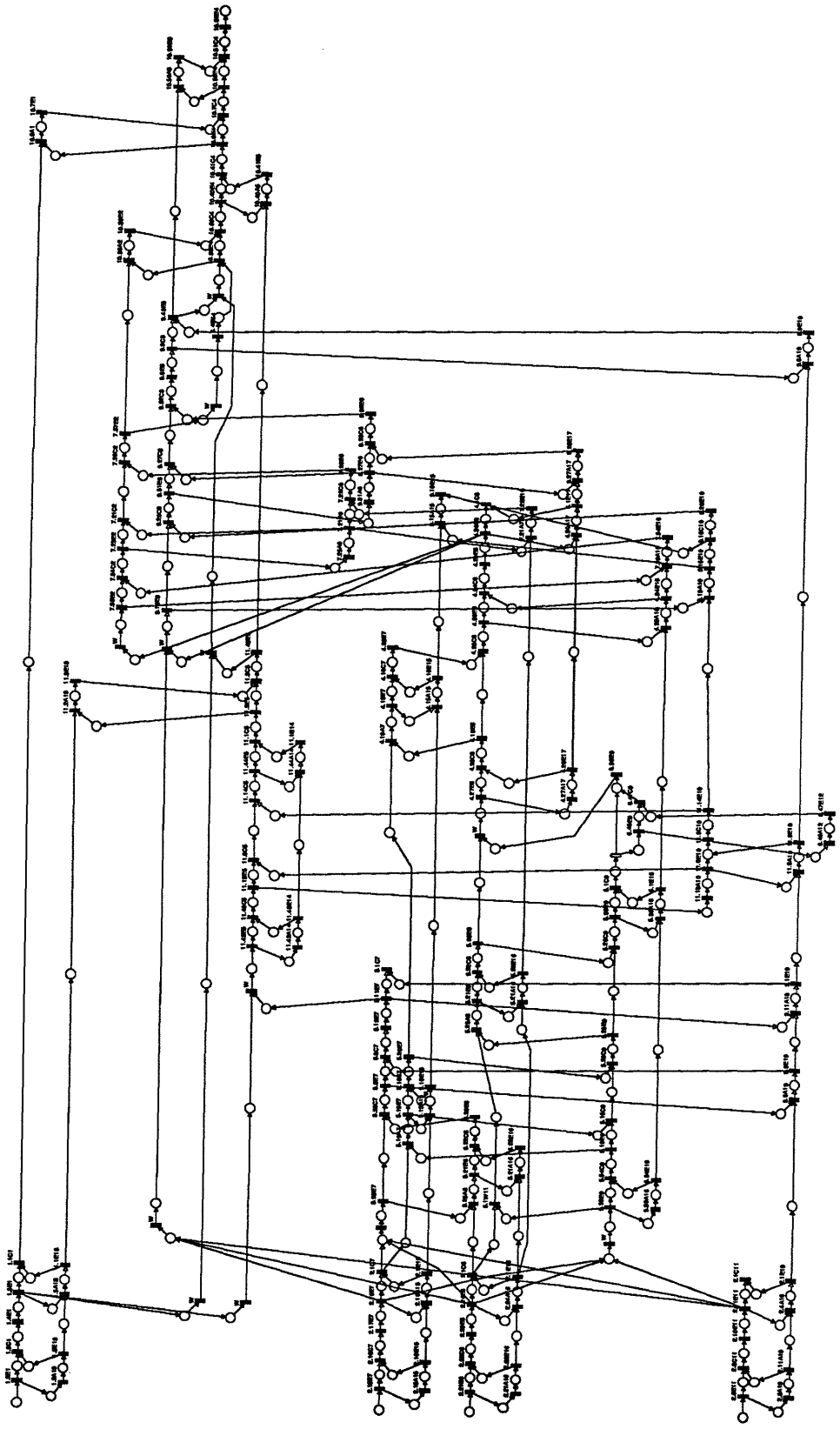


Figure 22 The Detailed Requirements Of MESACC