A Paper submitted to the
2006 Command and Control Research and Technology Symposium
"The State of the Art and the State of the Practice"

Executable Architectures for Modeling Command and Control Processes

(C2 Modeling and Simulation Track)

Jason E. Lich* and Yun-Tung Lau, Ph.D
Science Applications International Corporation (SAIC)
5113 Leesburg Pike, Suite 200
McLean, VA 22041

* POC: Phone: (703) 575-4138
Fax: (703) 824-5836
E-mail: Jason.E.Lich@SAIC.COM

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
JUN 2006		**00-00-2006 to 00-00-2006**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Executable Architectures for Modeling Command and Control Processes	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Science Applications International Corporation,5113 Leesburg Pike Suite 200,McLean,VA,22041**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**		**18**	

Abstract

This paper presents a high-level modeling and simulation (M&S) approach that is intended to be used early in the systems engineering lifecycle. The approach leverages behavior modeling techniques for designing executable function flows (i.e. functional simulation). It is built on system architecture products (such as DoD Architecture Framework (DoDAF) products) and is referred to as Executable Architecture (EA).

EA is designed to provide a higher-level view or abstraction of certain definable processes. Besides being an effective way of disseminating architecture visions to a wide range of stakeholders, EA provides the following benefits:

- Validation of the system architecture based on operational processes
- Insight into the military usefulness of the system
- Generation of first order, end-to-end performance metrics
- Discovery of hidden or overlooked requirements
- A way to document, validate, refine and communicate strategic objectives
- An early construct for trade analyses in the systems engineering space

The design and implementation of EAs was a supplemental technique incorporated into the Net-Centric Enterprise Services (NCES) systems engineering effort. We have used NCES mission threads and early capability specifications to design 'As Is' and 'To Be' behavior models.

The primary intent of providing high-level 'To Be' models is to incorporate new system (functionality) into existing processes. Once this is done, the 'To Be' models can be compared to a validated 'As Is' models. This EA approach was applied to command and control mission threads including Joint Air Tasking Order (JATO) and Time Sensitive Targeting (TST). We present results from modeling the 'As Is' and 'To Be' Joint Close Air Support (JCAS) planning process within the JCAS mission thread. The results demonstrated the effectiveness of introducing new NCES capabilities (such as the Federated Search) to the 'To Be' model.

Table of Contents

List of Figures

1 Introduction

The Global Information Grid (GIG) is the globally interconnected, secured end-to-end set of information capabilities, associated processes, and personnel for collecting, processing, storing, disseminating, and managing information on demand to warfighters, policy makers, and support personnel [1].

The GIG as a transformational vision aims at achieving information superiority in a network-centric environment. It enables various systems to interoperate with each other. For the warfighters, it brings "power to the edge" through a Task, Post, Process, Use (TPPU) process [2]. For the business and intelligence communities, it provides the infrastructure for effective information gathering and collaborative operation.

Net-Centric Enterprise Services provides a set of Core Enterprise Services (CES) on the GIG to support operational missions conducted by various communities of interest (CoI) in the warfighting, business, and intelligence domains. NCES is built on a service-oriented architecture (SOA), which enables distributed, parallel information sharing, and dynamic collaboration on a ubiquitous network [3]. This architecture defines a broad set of loosely coupled services with well-defined interface definitions (such as those based on Web Service Definition Language, or WSDL). These services provide end-user functionality over the network.

As part of the NCES architecture development, a number of operational mission threads were analyzed. The analyses show the end-to-end traceability from operational activities, to system functions and CES. They ensure that the NCES architecture does fulfill the needs of the domain and CoI users and provide value to them.

This paper presents a modeling approach for extending the NCES architecture through the use of executable architectures (EAs). The approach uses detailed operational documentation such as Tactics, Techniques and Procedure (TTP) publications, mission thread documents and a systems engineering tool to collect, refine and execute process diagrams.

In the following sections, we first give an overview of steps in our behavior modeling methodology, which uses Enhanced Function Flow Block Diagrams (EFFBDs) to capture operational processes. We then give a step-by-step description of the approach, along with an operational example (Joint Close Air Support), showing how the approach is applied.

2 Defining and Refining Operational Process

It can be argued that the accurate capture of operational activities and processes is the most important step in the systems development process. Practically all requirements and system functionality are gleaned from them.

Prior to the refinement of operational concepts into requirements and clearly definable system functionality, organizations and the operational boundary must be defined. These two characteristics provide the scope for the overall effort and offer the context for activity analysis and architectural synthesis.

Once the scope of the operational architecture has been defined, the task to discover and document more discrete, detailed operational activities can begin. This effort will provide the raw materials needed for designing 'As Is' and 'To Be' processes which, in turn, make up the operational architecture. Executable architectures – the focus of this paper – are operational architecture models designed with executable simulation constructs.

2.1 Executable Architectures

Activity diagrams defined by methodologies such as the Unified Modeling Language (UML) or activity models in DoDAF v.1 [4] are not executable. That is, they cannot be run in a simulation. However, these methodologies can be extended through behavior modeling, which is a first-order form of modeling and simulation. In particular, executable architectures extend the system architecture using discrete event simulation techniques. Executable architectures provide an effective means for validating functional designs and generating simulated metrics for the system. The following sections describe our approach for developing such executable architectures.

2.2 Approach

Here we describe an overall approach for activity analysis and architectural synthesis that can be used to derive operational behavior and develop executable architectures. The goal of this approach is to extend the architectural process in order to promote the discovery of key activities for fulfilling an operational mission. It is within these key activities that process improvement will likely take place. Executable architectures allow us to perform a high level evaluation of such improvements.

The steps used in this paper to develop executable architectures for command and control systems are as follows:

1. Capture and refine the 'As Is' operational activities and nodes
2. Validate the 'As Is' operational architecture
3. Engineer the 'To Be' design

4. Model the desired behavior
5. Refine the executable architecture
6. Evaluate and analyze the results

2.3 Example Summary

This paper incorporates an example to demonstrate how our approach is applied. The example is derived from the overall Joint Close Air Support (JCAS) mission thread and will specifically focus on the planning phase of that mission thread. The intent of the example is to identify a JCAS operational process area suitable for applying NCES capabilities. The following sections describe each step of the approach, along with the JCAS example.

3 Approach and Applied Example

Our methodology adheres to two overarching concepts: functional decomposition (top-down evaluation) and comparative analysis ('As Is' vs. 'To Be'). Although this is by no means the only approach to capturing, designing, and evaluating operational activities, it does provide a high level context for the steps described below.

3.1 Capturing 'As Is' Operational Activities

Operational activities identified in documents such as mission threads and Tactics, Techniques and Procedures (TTPs) provide a rich source of information for identifying discrete operational activities. The activities captured in these documents are typically composite in nature and usually require some level of derivation and/or refinement. Operational activities are performed by operational nodes, which also need to be identified and refined.

It is important that each level of detail be captured during refinement so that top-down or bottom-up traceability is preserved. From the architectural standpoint, this traceability is important in order to ensure that requirements are satisfied, gaps are identified, and alternative solutions can be analyzed at the earliest possible point in the systems engineering lifecycle. By adhering to this goal, project risks can be mitigated, costs minimized and expectations managed.

3.1.1 JCAS Example: Capturing the 'As Is'

The example below is representative of a top-down approach to identifying and documenting JCAS operational activities. Using mission-centric documents such as TTPs and mission threads, we develop several decompositions of the JCAS process [5, 6, 7, 8, 9 and 10]. Shown in **Figure 3.a** is the highest level process diagram (Level 0). This diagram is the root 'As Is' JCAS process documented and decomposed into more specific operational activities.

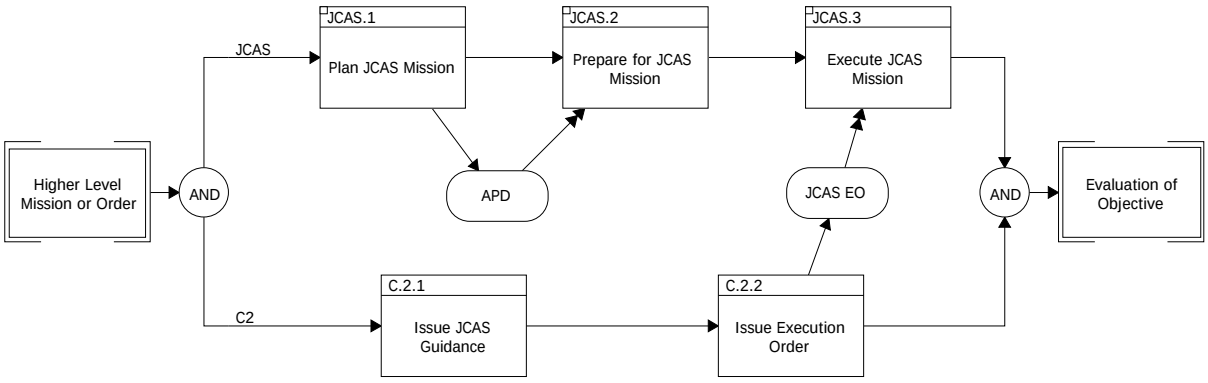

Figure 3.a - JCAS Process (Level 0)

Note: The above process notation uses that of an Enhanced Function Flow Block (EFFBD) diagram. This diagram captures functions and processes such as operational activities. Unlike many other forms of process diagrams, EFFBDs provide an executable construct for simulation. Here boxes denote activities or functions, whereas ovals represent data. Double arrows denote triggers; an activity or process cannot start until a trigger has been received. These diagrams are the building blocks for executable architectures.

3.1.2 JCAS Example: Process Decomposition

The following diagrams are decompositions of the JCAS.1 activity block (Plan JCAS Mission) in **Figure 3.a**. It should be stated that in documenting 'As Is' operational construct and nodes, various levels of granularity can be attained, depending on the needs, the project constraints and/or the level of effort. For the example here, the level of granularity is somewhat coarse. However, our methodology and the EFFBDs can model granularity at any detailed level.

Figure 3.b shows activities in the JCAS planning phase. At this level the activities are not detailed enough to begin our incorporation of NCES capabilities. Hence we continue to decompose them, as shown in **Figure 3.c** and **Figure 3.d**.

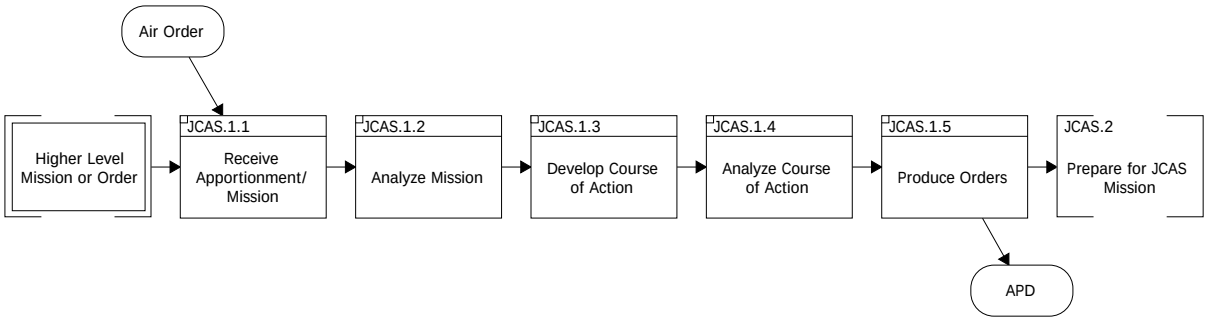

Figure 3.b - JCAS Process (Level 1), showing the details of activity JCAS.1 in Level 0

Figure 3.c documents the typical steps involved in developing courses of action (JCAS.1.3 in **Figure 3.b**) for a generalized JCAS mission. One observation here is the linear/serial characteristic of these process diagrams. This is somewhat typical with higher level views that define doctrinal philosophy and/or legacy operations. As process decomposition continues, many more discrete activities and dynamic interactions are discovered.

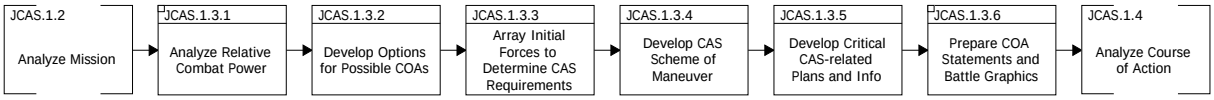

Figure 3.c - JCAS Process (Level 2), showing the details of activity JCAS.1.3 in Level 1

Figure 3.d depicts the decomposed activity "Prepare COA Statements and Battle Graphics" (JCAS.1.3.6 in **Figure 3.c**). With activities at this level of detail, we will start to consider a 'To Be' design. Prior to constructing the 'To Be' notional model, validation of the 'As Is' should take place. Optimally, this effort runs in parallel with the development of each level of details. The next section describes the objectives and benefits of validation.

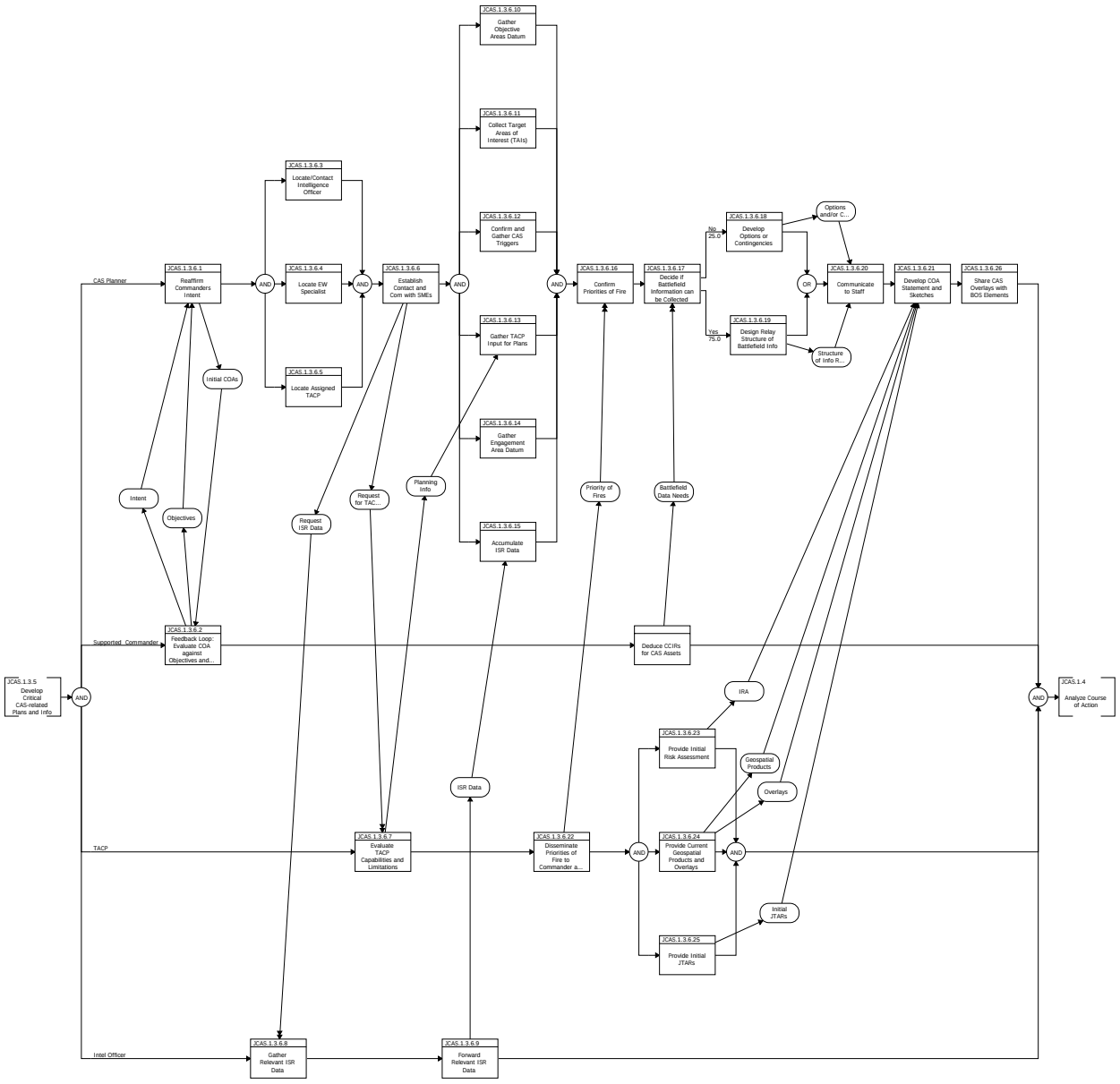

Figure 3.d - JCAS Process (Level 3), showing the details of activity JCAS.1.3.6 in Level 2

3.2 Validate the 'As Is' Architecture

One of the most overlooked activities in developing any type of process diagram, behavior model and/or process representation such as **Figure 3.d** is that of validation. Validation of events, activities, timing and other key characteristics is crucial for an accurate depiction of the operational architecture. It also provides other benefits because it:

- Facilitates stakeholder buy-in

- Legitimizes the effort through Subject Matter Expert (SME) involvement
- Promotes stakeholder involvement, information flow and project goal dissemination
- Incorporates necessary detail that enhances high level design and decision making

The process of validation can be very time intensive but its importance should not be underestimated.

3.3 Engineering the 'To Be' Design

Once the 'As Is' operational activities have been designed and validated, the elements of a 'To Be' operational architecture can be incorporated. The goal here is to evaluate different and/or new means of performing an operational activity.

3.3.1 Architecting New or Modified Activities

Executable architectures provide a visually powerful method for incorporating change and demonstrating utility. Not only do EAs provide a visual depiction of new or modified operational activities, they also provide functional validation and raw data that can be captured and analyzed. Tools that allow for this type of architectural development provide a means for disseminating objectives, reflecting changes, managing expectations and involving stakeholders at varying levels of expertise or authority. All of these benefits are extremely valuable in the overall engineering of any system.

3.3.2 JCAS Example: Designing the 'To Be'

Our example shows possible enhancements to the 'As Is' process in **Figure 3.d** with new conceptual operational activities. These new activities improve the JCAS Planners' ability to perform his/her duties and expedite the planning process.

Figure 3.e shows the 'To Be' Process for "Prepare COA Statements and Battle Graphics", whereas **Figure 3.f** is an enlarged view of part of the 'To Be' process with new operational activities. The new activities appear inside the dotted ellipses.

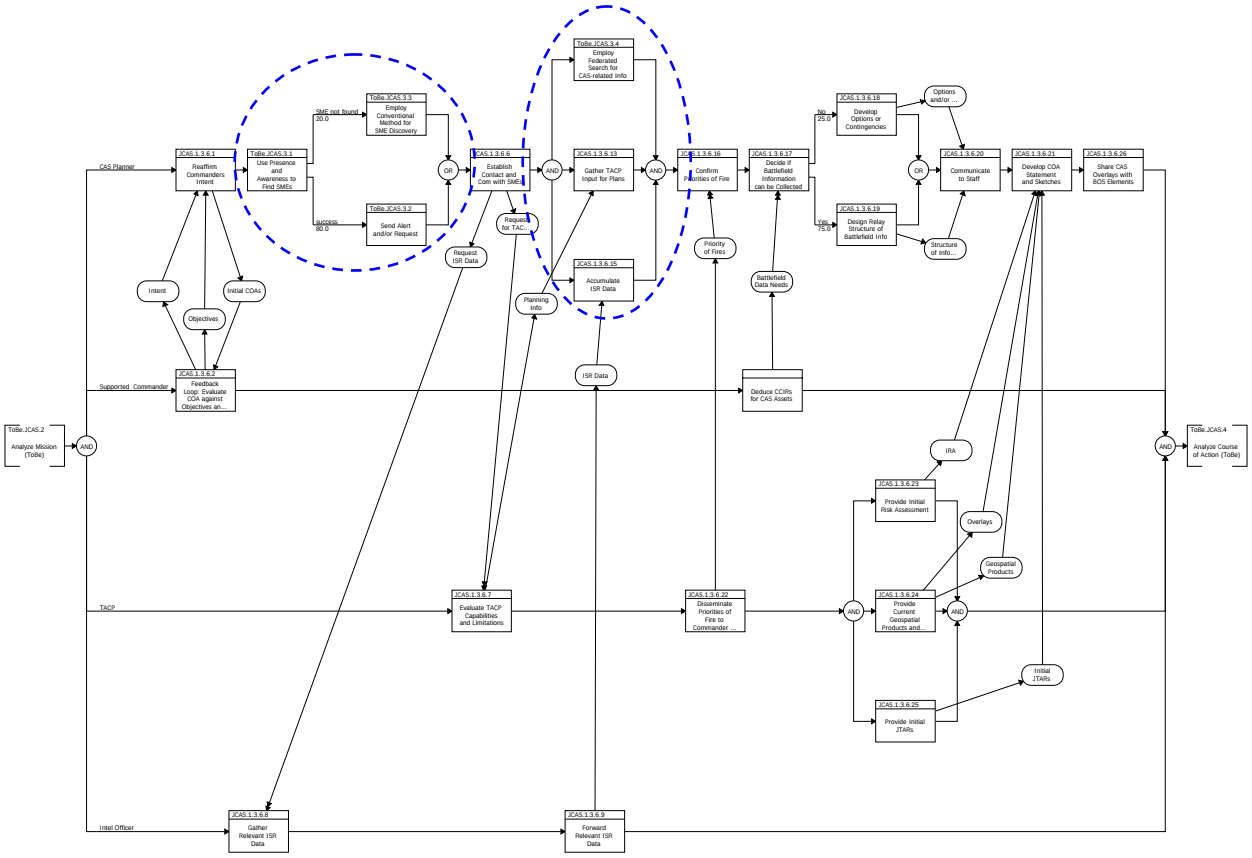

Figure 3.e -'To Be' Process for "Prepare COA Statements and Battle Graphics"

Figure 3.e incorporates three new operational activities, which utilize the following NCES capabilities:

- **Presence and Awareness** – this capability provides online access to status information for users in a global directory.
- **Alert/Notification** – this capability provides an alert service (messaging capability) that can send alerts/notifications to a variety of applications such as email, instant messaging, web browser, etc.
- **Federated Search** – this capability provides a way to search enterprise contents across various search-enabled data sources and aggregate the results for the users.

Our goal is to determine if the 'To Be' process is meaningful and the NCES solution feasible. This is where executable architectures become a valuable analytical and demonstrative tool.

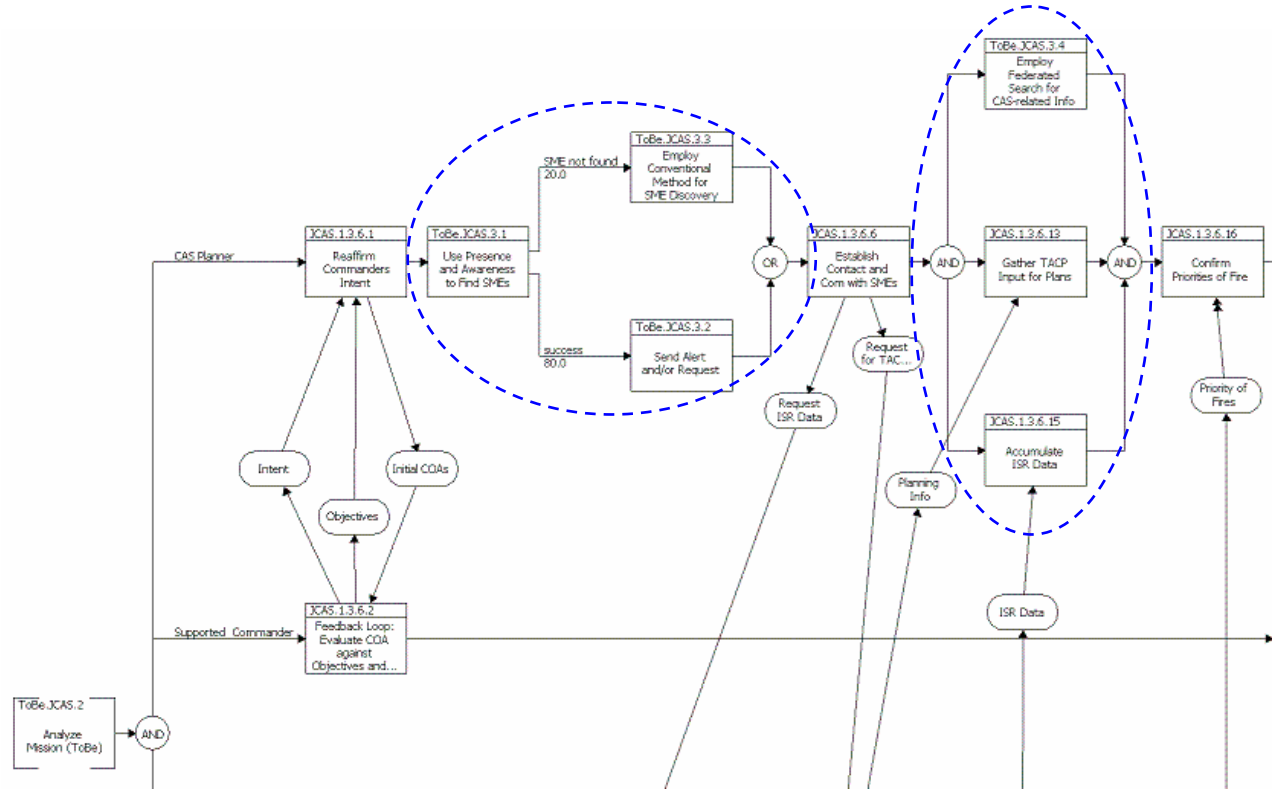

Figure 3.f - Enlarged View of Part of the 'To Be' Process

3.3.3 JCAS Example: Abstraction of Model

Common to all modeling techniques is a layer of abstraction that hides details from the users. The executable architecture relies on abstraction to simplify models and identify the key issues (e.g. utility and feasibility) that need to be resolved early in the systems engineering process. Resolving such key issues related to operational concepts will help meet or exceed user expectations.

In our example, our goal is to determine the utility and feasibility of the new 'To Be' functions (Presence and Awareness, Federated Search, and Alert/Notification) within the planning phase of JCAS. For this purpose we model those three functions as 'black boxes', which are abstractions that hide system specifics.

3.4 Refine the Executable Architecture

After the EFFBDs have been designed and validated, we can refine them before performing simulation on the 'As Is' and 'To Be' models. This refinement process enhances the executable architectures by incorporating:

- Timing information
- Network characteristics, if applicable
- Data characterizations

- Assigned probabilities
- Additional activity decomposition
- Scripts for control and dynamic behavior

The above attributes provide an additional level of detail to the behavior models and allows for a more robust extension of the overall operational architecture. For simplicity, the example in this paper will focus only on accurate process decomposition, timing and assigned probabilities.

3.5 Analysis and Evaluation

Eventually, behavior models are combined and refined to form executable architectures. They not only show military utility at high levels, but also allow for a first look at characteristics such as performance and trades.

Also, the impact of new functionalities/activities or the redesign of an operational node can impact the overall operational architecture. It is not enough to simply identify this architectural change. The change itself must be analyzed and evaluated in order to evaluate its impact on other activities, manage risks, and evaluate trade offs and alternatives.

3.5.1 JCAS Example: Simulation and Data

The following example within the JCAS mission analyzes and evaluates the high level impact on planning process timing as a result of adding new functionality to an existing process. This is a very basic form of comparative analysis that can be done at the early stages of system design.

Figure 3.g graphically depicts the simulation of the JCAS 'Prepare COA Statements and Battle Graphics' To Be activity, whereas **Figure 3.h** shows a sample output from the simulation. In **Figure 3.g** the horizontal bars represent the time spans of active functions on the left. The horizontal axis is time in minutes.

In our example (as in other modeling and simulation tools), discrete data items such as activity start time, time of completion, wait time and data flow can be captured for analysis. Multiple runs or executions can be performed in order to obtain statistical results.

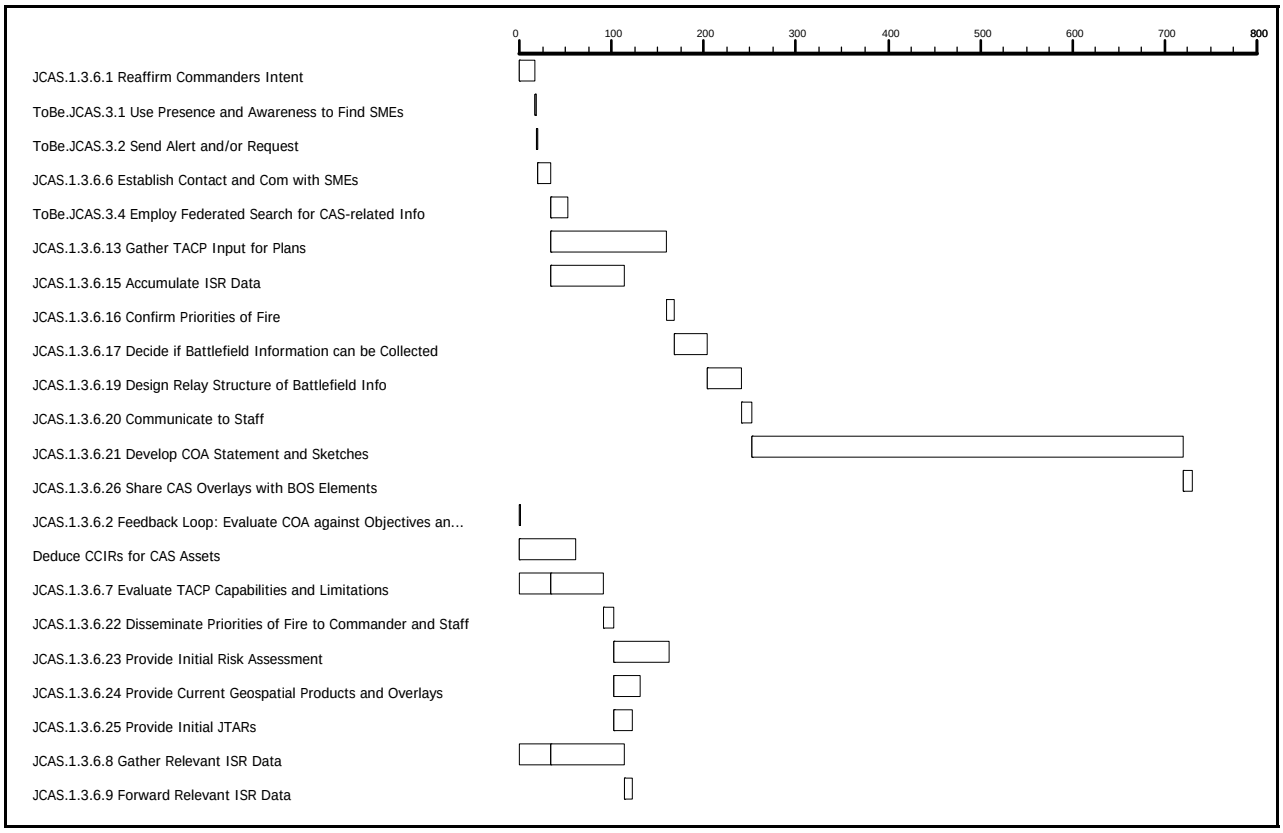

Figure 3.g - Graphical Representation of Simulation

Time	Event ID	Process ID	Event Name	Construct ID	Structure	Number	Name	Event Execution Data		
19.376225	ID(14)	PROC(1.3)	start	1.1.2.3.1		JCAS.1.3.6.5	Locate Assigned TACP			
26.5081064	ID(18)	PROC(1.3)	finish	1.1.2.3.1		JCAS.1.3.6.5	Locate Assigned TACP			
30.4437245	ID(16)	PROC(1.1)	finish	1.1.2.1.1		JCAS.1.3.6.3	Locate/Contact Intelligence Officer			
30.7133388	ID(17)	PROC(1.2)	finish	1.1.2.2.1		JCAS.1.3.6.4	Locate EW Specialist			
30.7133388	ID(15)	PROC(1.3)	finish	1.1.2	Parallel					
30.7133388	(aux)	PROC(1)	enabled	1.1.3		JCAS.1.3.6.6	Establish Contact and Com with SMEs			
30.7133388	(aux)	PROC(1)	waitingForResources	1.1.3		JCAS.1.3.6.6	Establish Contact and Com with SMEs			
30.7133388	ID(19)	PROC(1)	start	1.1.3		JCAS.1.3.6.6	Establish Contact and Com with SMEs			
45.2811661	(aux)	PROC(1)	queued	1.1.3		JCAS.1.3.6.6	Establish Contact and Com with SMEs	Request ISR Data	to	Request ISR Data
45.2811661	(aux)	PROC(1)	queued	1.1.3		JCAS.1.3.6.6	Establish Contact and Com with SMEs	Request for TACP Input	to	Request for TACP Input
45.2811661	ID(20)	PROC(1)	finish	1.1.3		JCAS.1.3.6.6	Establish Contact and Com with SMEs			
45.2811661	(aux)	PROC(3)	waitingForResources	1.3.1		JCAS.1.3.6.7	Evaluate TACP Capabilities and Limitations			
45.2811661	(aux)	PROC(3)	dequeued	1.3.1		JCAS.1.3.6.7	Evaluate TACP Capabilities and Limitations	Request for TACP Input	from	Request for TACP Input
45.2811661	ID(4)	PROC(3)	start	1.3.1		JCAS.1.3.6.7	Evaluate TACP Capabilities and Limitations			
45.2811661	(aux)	PROC(4)	waitingForResources	1.4.1		JCAS.1.3.6.8	Gather Relevant ISR Data			
45.2811661	(aux)	PROC(4)	dequeued	1.4.1		JCAS.1.3.6.8	Gather Relevant ISR Data	Request ISR Data	from	Request ISR Data
45.2811661	ID(5)	PROC(4)	start	1.4.1		JCAS.1.3.6.8	Gather Relevant ISR Data			
45.2811661	ID(21)	PROC(1)	start	1.1.4	Parallel					
45.2811661	(aux)	PROC(1.4)	enabled	1.1.4.1.1		JCAS.1.3.6.10	Gather Objective Areas Datum			
45.2811661	(aux)	PROC(1.4)	waitingForResources	1.1.4.1.1		JCAS.1.3.6.10	Gather Objective Areas Datum			
45.2811661	ID(24)	PROC(1.4)	start	1.1.4.1.1		JCAS.1.3.6.10	Gather Objective Areas Datum			

Figure 3.h - Simulation Output (sample)

3.5.2 JCAS Example: Evaluation

Once a large enough set of sample data is collected, a more intensive evaluation of the new functionality can begin. In our example, the completion times of the 'Prepare COA Statements and Battle Graphics' function (JCAS.1.3.6 from **Figure 3.c**) are examined.

The results are shown in **Figure 3.i**. Here based on the average completion time computed from a set of 15 simulations, the 'To Be' operational process (which utilizes NCES capabilities) on

average achieves a nearly four hour improvement over the 12-hour 'As Is' process. Note that individual completion time varies because of the inherent randomness in the simulation model. Thus a specific run (such as Run #1) of the 'To Be' model should not be compared with another specific run (e.g. Run #3) of the'As Is' model.

Joint Close Air Support
Functional Simulation: *Prepare COA Statements and Battle Graphics*

RUN	'As Is' Totals (sample size = 15) total time (minutes)	'To Be' Totals (sample size = 15) total time (minutes)
1	673.92	605.94
2	791.74	563.81
3	463.30	575.62
4	933.79	298.66
5	364.93	434.66
6	715.84	628.30
7	660.93	385.47
8	414.01	270.56
9	1068.69	640.92
10	680.23	830.90
11	843.52	486.15
12	827.18	317.18
13	685.76	765.22
14	882.17	363.89
15	1085.49	483.10
	Averaged Total Elapsed Time for Function (minutes): 739.43	Averaged Total Elapsed Time for Function (minutes): 510.02

Figure 3.i - Simulation Results of the completion times for the 'Prepare COA Statements and Battle Graphics' function

The improvement in completion clearly shows the effectiveness of utilizing NCES capabilities in these operational activities.

4 Summary

We have presented an end-to-end approach for developing executable architectures. The approach involves functional decomposition and enables quantitative and comparative analysis of the 'As Is' and 'To Be' operational processes.

We have applied executable architectures to the JCAS mission planning phase to show the feasibility and military utility of new functionality provided by Net-Centric Enterprise Services.

For the specific example of the 'Prepare COA Statements and Battle Graphics' activity, we have shown that the 'As Is' process can be shortened by 33% on average.

As part of a requirement definition process, executable architectures help solidify an accurate and complete representation of operational needs and show viability and utility of new functionality. Executable architectures and the associated high-level evaluation is useful in identifying potential issues, disseminating visions and project goals, capturing first-order data, involving subject matter experts and stakeholders, and collecting vital information on mission operations.

Because of these benefits, we view executable architectures as a useful effort especially in the early phase of a systems engineering process. The sooner such efforts occur in the system engineering lifecycle, the greater the chance a solution is developed that fully satisfies the needs of the user community.

5 References and Acronyms

1. Global Information Grid Capstone Requirements Document, JROCM 134-01, August 2001 <https://jdl.jwfc.jfcom.mil>.
2. Net-Centric Operations and Warfare (NCOW) Reference Model, Draft version 1.1, October, 2004.
3. Capability Development Document (CDD) for Net-Centric Enterprise Services (NCES), Draft version 0.9.2, November 2005.
4. DoD Architecture Framework (DoDAF) Version 1.0 < http://www.defenselink.mil/nii/doc/>.
5. Joint Close Air Support Joint Mission Thread (Draft), OASD(NII), October 2004 (FOUO).
6. Joint Publication 3-30, Command and Control for Joint Air Operations, 5 June 2003.
7. Joint Publication 3-09.3, Joint Tactics, Techniques and Procedures for Close Air Support, 3 September 2003.
8. Capstone Requirements Document for Close Air Support, JROCM 067-02, 6 May 2002.
9. Beyond Close Air Support, Forging a New Air-Ground Partnership, RAND Research Brief, March 2005, <www.rand.org/publications/MG/MG301>.
10. Joint Staff Officers Guide AFSC Pub 1, Crisis Action Planning, January 1997, <http://www.fas.org/man/dod-101/dod/docs/pub1_97/Chap7.html>

The acronyms used in this paper are listed below.

APD	Air Planning Document
BOS	Battlefield Operating Systems
C2	Command and Control
CCIR	Commander's Critical Information Requirements
CES	Core Enterprise Services
COA	Course of Action
CoI	Community of Interest

DoD	Department of Defense
DoDAF	DoD Architecture Framework
EA	Executable Architecture
EAM	Executable Architecture Model
EFFBD	Enhanced Function Flow Block Diagram
GIG	Global Information Grid
GIG ES	GIG Enterprise Service
IRA	Initial Risk Assessment
ISR	Intelligence, Surveillance and Reconnaissance
JATO	Joint Air Tasking Order
JCAS	Joint Close Air Support
JFC	Joint Force Command
JTAR	Joint Tactical Air strike Request
M&S	Modeling and Simulation
NCES	Net-Centric Enterprise Services
NCOW	Net-Centric Operations and Warfare
SOA	Service-Oriented Architecture
TACP	Tactical Air Control Party
TAI	Target Areas of Interest
TPPU	Task, Post, Process, Use
TST	Time-Sensitive Targeting
TTP	Tactics, Techniques and Procedures
UML	Unified Modeling Language