A Framework to Assess the Impacts of Jointness

Morgan M. Dwyer
Engineering Systems Division
Massachusetts Institute of Technology
Cambridge, MA
mdwyer@mit.edu

Prof. Zoe Szajnfarber
Engineering Management and Systems Engineering
The George Washington University
Washington, D.C.
zszajnfa@gwu.edu

Abstract—**This paper defines a framework to represent organizational and technical architectures and to quantitatively assess the complexity mechanisms within them. The framework is then applied to the case of the NPOESS program and used to illustrate the relationship between complexity, cost growth, and the concept of jointness.**

I. Introduction

The *Department of Defense Dictionary of Military and Associated Terms* defines the concept of *jointness* as "activities, operations, organizations, etc. in which elements of two or more Military Departments participate" [1]. In practice, jointness is often defined more broadly and includes activities, operations, and organizations that involve more than one service department *or* more than one government agency. This paper adopts the more expansive definition of jointness but specifically focuses on one type of activity—system acquisition—and presents a framework that can be used to study the technical and organizational challenges that are often encountered when government agencies acquire systems jointly.

Jointness has numerous benefits: it enables government agencies to design for interoperability, to leverage a particular agency's unique technical capabilities, and to benefit from mission and technical synergies. In addition to these benefits, one of the most common motivations for joint system acquisition is cost savings: theoretically, when agencies acquire systems jointly, both the cost-per-agency and the overall cost to the government are reduced, since agencies can share development, production, and operations costs. However, despite their cost savings potential, recent studies suggest that joint programs experience greater cost growth than non-joint programs [2]-[5] and that in some cases it may be more cost-effective for agencies to acquire systems independently rather than jointly [2].

To understand *why* joint programs incur greater rates of cost growth or *how* jointness itself may contribute to a program's costs, the acquisition community requires an improved understanding of the organizational and technical *mechanisms* that have induced cost growth in the past. Our paper responds to this need by presenting a semi-quantitative framework that can be used to study the evolution of cost growth on joint programs and we demonstrate the utility of our approach by applying it to a case study of the National Polar-orbiting Operational Satellite System (NPOESS).

II. Problem Statement

In our proposed framework, we define jointness in terms of a program's organizational and technical *architecture*. Crawley et al. define architecture as "an abstract description of the entities of a system and the relationships between those entities" [6]: essentially, a system's architecture is defined by the system's components and by the relationships between them. In our framework, we distinguish between two types of jointness: organizational and technical. A joint technical architecture is one that meets a diverse set of requirements from distinct and separate user groups. A joint organizational architecture is one that accommodates participation from more than one government agency. Given this definition, programs can be classified as either technically joint, organizationally joint, or as exhibiting both types of jointness.

Importantly, joint architectures can also be defined by their ability to be *disaggregated*. Specifically, a joint system executes an aggregated set of requirements that could be alternatively executed by multiple distinct systems. Similarly, joint organizational architectures are also aggregated and can be *disaggregated* if government agencies develop systems independently instead of collaboratively.

A current movement in the space acquisition community, which supports the disaggregation of previously joint programs, suggests two hypotheses that connect jointness to cost growth and motivate our focus on joint program architectures. The first hypothesis suggests that aggregated technical architectures are more complex than disaggregated ones and that when this complexity is not identified, budgeted for, and actively managed—it induces cost growth on joint programs [7]-[11]. The second hypothesis suggests that aggregated organizational architectures are more complex than disaggregated ones and that this complexity induces and enables cost growth on joint programs [3],[5],[12]-[14]. Given these hypotheses, we suggest that in order to understand the relationship between jointness and cost growth, one must characterize the complexity mechanisms inherent to joint programs' architectures and observe complexity's evolution over time. As a result, our proposed framework suggests a methodology to represent a joint program's organizational and technical architectures and to observe the evolution of architectural complexity and its relationship to cost growth.

III. Proposed Architectural Framework

The proposed framework contains five steps wherein the joint program's organizational and technical architectures are represented and metrics that assess their complexity are calculated. In the final step, the evolution of the joint program's complexity and cost is observed by plotting the complexity metrics and cost over time.

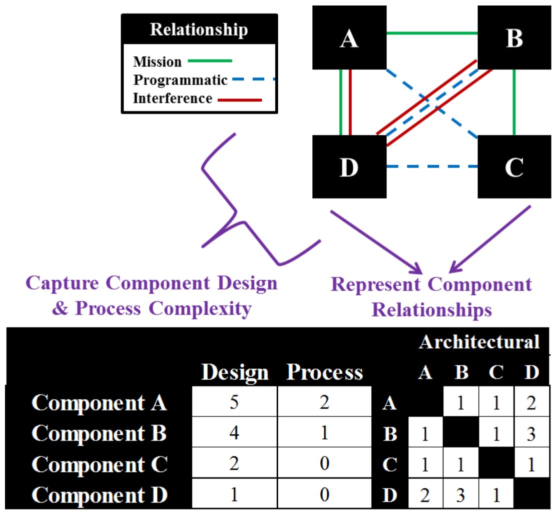
To represent a program's architectures, we use design structure matrices (DSMs). DSMs are typically NxN matrices that are used to represent product, organizational, or process architectures or some combination of all three [15]. Our framework defines two separate DSMs to represent a program's organizational and technical architectures and the complexity mechanisms within them. Previous studies [16]-[17] have demonstrated the utility of using DSM-based metrics to study the evolution of architectures; therefore, in order to study cost growth that was induced or enabled by complexity, we calculate metrics using our proposed DSMs.

A. Step 1: Represent the Technical Architecture

First, all major technical components are represented in the technical architecture DSM (*DSM_T*). Three types of complexity mechanisms—which emerge as a function of the individual components or the relationships between them—are also represented. The three complexity types are defined as:

- ***Design complexity*** is a function of the technical maturity of each component.
- ***Process complexity*** is a function of the constraints or conflicting requirements that are imposed during the component development process.
- ***Architectural complexity*** is a function of the interactions and relationships between components.

As shown in Fig. 1, architectural complexity mechanisms are represented using traditional DSM notation where +1 is added to indicate the presence of *any* relationship between two components. Three relationship types are possible—mission, programmatic, and interference—and the presence of each relationship adds +1 to the corresponding *DSM_T* entry; components can share more than one relationship and each relationship type adds +1 to the corresponding entry in *DSM_T*.


Relationship
Mission
Programmatic
Interference
A
B
C
D
Capture Component Design & Process Complexity
Represent Component Relationships

	Design	Process	Architectural	A	B	C	D
Component A	5	2	A	■	1	1	2
Component B	4	1	B	1	■	1	3
Component C	2	0	C	1	1	■	1
Component D	1	0	D	2	3	1	■

Fig. 1. Example Technical Architecture DSM

Mission relationships between components include physical, data, or design interfaces as defined in [18]. Physical interfaces mean two components are physically attached and often also share other relationships (such as data or power). Two components have a data (but not a physical) relationship when they communicate at a distance and two components have a design relationship when they are designed to enable parts sharing (e.g. they are designed to maximize commonality).

Programmatic relationships indicate that components share management resources like budget, schedule, and staff. Although this relationship is not purely technical, we include it here because it can induce non-recurring cost growth: specifically, even though two components may not share a mission interface, they may still interface programmatically because the budget, schedule, and staff assigned to one component can impact the resources that are allocated to the other. For example, if a component's costs grow but the program's budget is fixed, management may decide to prioritize its development at the expense of others, whose budgets will be reduced and schedules lengthened; this decision will ultimately increase the total non-recurring cost of the lower priority components [19]. To capture this behavior, when two components do not have a mission relationship but share a programmatic relationship, +1 is added to *DSM_T*.

Finally, +1 is added to *DSM_T* entries to account for *interferences* between components. As noted by [20]-[22], components can interfere electromagnetically, mechanically, optically, and through their system's reliability budget. Interferences induce complexity because they must be actively managed and compensated for during the system development process.

As also shown in Fig. 1, *DSM_T* includes two extra columns that contain a design and process complexity score for each component. The design complexity score captures the degree of cost risk associated with the component's development, since as its technical maturity increases, so does the certainty with which a program can estimate its development cost [23]. Brady & Nightingale previously demonstrated the utility of including technical maturity in a DSM when they developed the technology risk DSM to assess development and operational risk in NASA systems [24]. Although their approach used a standard Technology Readiness Level (TRL) system to categorize component maturity [24], other rating schemes can also be applied. For example, AIAA categorizes a component's design maturity in according to a component's location in the traditional development lifecycle [25].

While there is no formal scheme to categorize a component's process complexity, we suggest that +1 can added for each process complexity mechanism. The key process complexity mechanism that can affect joint programs is *conflicting requirements*. This complexity mechanism captures the costs that emerge when a program has unclear lines of authority [26], misaligned stakeholder objectives [27], or conflicting requirements which hinder its ability to function efficiently.

A secondary process complexity mechanism is a function of a program's oversight model. If two oversight models are available but the component is developed according to the one with more stringent requirements, +1 should also be added to the component's process complexity score. For example, if a component was developed under a Systems Engineering and Technical Assistance (SETA) oversight model instead of a more cost-effective Total System Performance Responsibility (TSPR) oversight model, +1 should also be added to its process complexity score. Although process and product architectures have shared DSMs differently in the past [15], because process ultimately affects the cost of the technical system, our framework represents both complexities using a shared DSM.

B. *Step 2: Calculate Technical Complexity Metric*

After the DSM_T is defined, we use it to calculate our proposed technical complexity metric; in this framework, we define technical complexity to be a function of the number of components in a system, the complexity of the individual components, and the interactions between them. Importantly, we also suggest that as a system's complexity increases, so do its non-recurring costs; as a result, our technical complexity metric is actually an estimate of the technical architecture's lifecycle cost, with penalties applied to its non-recurring costs to account for each design, process, and architectural complexity mechanism.

Equation (1) shows the general form of our complexity-corrected lifecycle cost metric (L_{cc}), which includes the non-recurring costs with complexity penalties applied (N_{cc}), recurring costs (R), and other costs (O) which can include launch or operations costs. As shown, L_{cc} is normalized by the cost of a reference disaggregated system.

$$L_{cc} = \frac{N_{cc} + R + O}{Baseline\ L_{cc}} \tag{1}$$

The formula for N_{cc} is derived from a complexity metric that was used to study the disaggregation of spacecraft architectures for planetary exploration [21]; while the form of our metric is similar, several of the complexity mechanisms that we identify are unique. More importantly, by classifying mechanisms in terms of three complexity types—design, process, and architectural—we intend for our approach to be generalizable to other systems as well.

$$Ncc = \sum_{i=1}^{N}\{(\sum_{j=1}^{N} DSM_{T(i,j)} - 1) * W_A + PC_i + 1\} * C_i(DC_i) \tag{2}$$

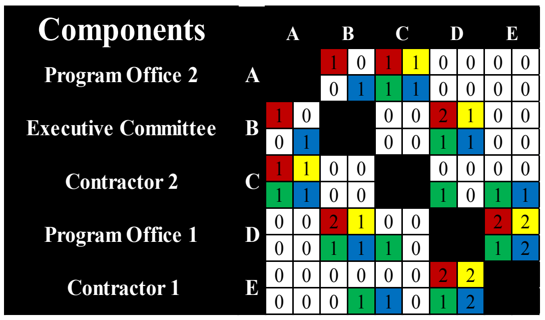
As shown, N_{cc} is calculated using each component's design complexity (DC_i), process complexity (PC_i), DSM_T, and cost penalty weighting for each complexity mechanism (W_A). Component cost (C_i) can be estimated using system-specific cost-estimating relationships and corrected for design complexity either by adding a penalty to component mass prior to estimating its cost (as in [20]-[21]) or by adding a penalty after its costs have been calculated (as in [28]). The weightings that are applied to correct the cost estimate for process and architectural complexity mechanisms are also system-specific and should be determined on a case-by-case basis.

C. *Step 3: Represent the Organizational Architecture*

The organizational DSM (DSM_O) contains the major components of a joint program's organizational architecture; components are distinct sub-units within an organization that include government agencies, user communities, program offices, and contractors. As shown in Fig. 2, the DSM_O actually contains four distinct DSMs that map four different relationship types between organizational components. DSM_O also indicates relationship strength; a score of +2 is used when components' relationship is weak and +1 is used when the relationship is strong. The four relationship types are defined as follows:

- When a component has ***technical capability*** (*TC*), it has the knowledge and experience to make decisions effectively. DSM_{TC} is shown in blue.
- When a component has ***mission responsibility*** (*MR*), it is responsible for delivering a technical system that executes its specified mission. DSM_{MR} is shown in green.
- Alternatively, when a component has ***financial responsibility*** (*FR*), it is responsible for funding the technical system. DSM_{FR} is shown in yellow.
- Finally, when a component has ***decision authority*** *(DA)*, it is able to make and sustain effective decisions. DSM_{DA} is shown in red.

Although the four relationship types are depicted separately, the two relationships that contribute most significantly to our organizational complexity metric (defined in the next step) are *mission responsibility* and *decision authority*. An example mission responsibility relationship between two component contractors is illustrated in Fig. 3. These contractors share a mission responsibility relationship because the components that they produce share a technical interface: in order to execute their mission, both components need to function. In this way, mission responsibility relationships between contractors *mirror* [29]-[30] the program's technical architecture. However, in addition to this mirroring, mission responsibility relationships on joint programs extend throughout the organizational hierarchy and ultimately connect agency leaders, who Congress holds responsible for mission execution, to the contractors that agencies hold responsible for the system's development.



Components		A		B		C		D		E	
Program Office 2	A			1	0	1	1	0	0	0	0
				0	1	1	1	0	0	0	0
Executive Committee	B	1	0			0	0	2	1	0	0
		0	1			0	0	1	1	0	0
Contractor 2	C	1	1	0	0			0	0	0	0
		1	1	0	0			1	0	1	1
Program Office 1	D	0	0	2	1	0	0			2	2
		0	0	1	1	1	0			1	2
Contractor 1	E	0	0	0	0	0	0	2	2		
		0	0	0	1	1	0	1	2		

Fig. 2. Example Organizational Architecture DSM

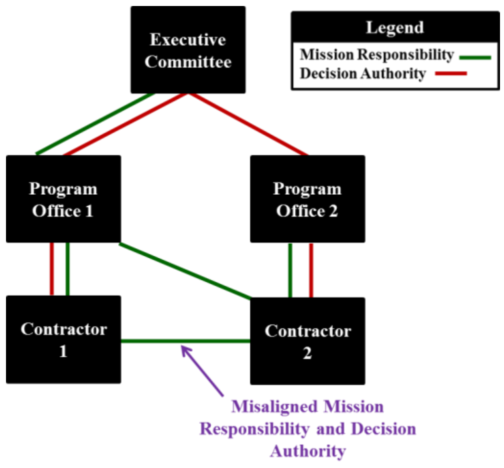


Fig. 3. Misalignment of Mission Responsibility & Decision Authority

DSM$_{DA}$ represents the organization's authority structure. As shown in Figure 3, Program Office 1 holds a contract for Component 1 and Program Office 2 holds a contract for Component 2; additional decision authority relationships between components are also illustrated. Fig. 3 also notes that although there is a mission responsibility link between component contractors, there is no decision authority relationship. This misalignment of mission responsibility and decision authority is critical to our organizational complexity metric.

In addition to the relationship between mission responsibility and decision authority, two other relationships—financial responsibility and technical capability—can also affect the organization by eroding decision authority. Although these relationships can be conceptualized in terms of their impact on decision authority, they are represented separately because both relate to the concept of jointness; as noted previously, the ability to share financial responsibility between organizations or to capitalize on one organization's unique technical capabilities are common motivations for jointness.

D. Step 4: Calculate Organizational Complexity Metric

To assess organizational architectures, we use a separate metric and define organizational complexity (*OC*) to be a function of the number of components in an organization, the interfaces between each component, and components' mission responsibility and decision authority. We suggest that as an organization's complexity increases, it becomes more difficult for the organization to make effective and efficient decisions; as a result, complex organizations are more likely to enable, sustain, and induce cost growth.

Although recent studies of *OC* in government-funded engineering projects have demonstrated that complexity correlates with cost growth [26],[31]-[33], within this set of research, authors use different metrics to assess *OC*. Despite the quantitative differences between their metrics, authors generally agree that organizational complexity is a function of the number of components and the interfaces between them [32]-[33]. Additionally, previous studies have also suggested that convoluted authority structures have contributed to organizational complexity on past joint programs [5],[34]-[36]. The metric that we propose to assess organizational complexity accounts both for the number of interfaces, the number of components, and for the authority structure in a joint program's organizational architecture.

The proposed organizational complexity metric is given in (3), which is derived from a structural complexity metric proposed and validated by Sinha [33]. Sinha defined structural complexity to be a function of the number of elements in a system and the connectivity structure between those elements [33]; unlike other metrics that have been used to assess organizational complexity, Sinha's metric uniquely captures not only the number of interfaces between components but also the connectivity of those interfaces [33]. This provides a richer measure of the complexity that can be inherent to an architecture but is not captured by a simple interface count [33]. Although Sinha's metric has only been applied to assess the complexity of technical architectures, by calculating complexity using *DSM*$_{MR}$ (which mirrors much of the technical architecture), our proposed metric is only a minor extension of previous work. The most significant difference between our proposed metric and Sinha's is the weighting scheme that we developed specifically so that the metric could be applied to assess the complexity of organizations. The final organizational metric is shown in (3) where W_A corresponds to the weighting scheme, N corresponds to the number of components and $E(DSM_{MR})$ corresponds to singular values of the *DSM*$_{MR}$ matrix. Finally, like L_{cc}, *OC* is normalized by a reference disaggregated organization.

$$OC = \frac{N + \left\{\sum_{i=1}^{N}\sum_{j=1}^{N} W_{ij} * DSM_{MR}(i,j)\right\} * \left[\frac{1}{N}\right] * E(DSM_{MR}(i,j))}{Baseline\ OC} \tag{3}$$

The process for calculating (3)'s weights begins by adjusting *DSM*$_{DA}$, to account for factors that eroded decision authority. For each factor, a score of +1 was added to each affected decision authority link. Two authority erosion factors are related to common motivations for forming joint programs. First, *multiple sources of technical capability* weaken decision authority when technical capability is not aligned with decision authority. This misalignment erodes decision authority because the additional sources of technical capability can question another component's decisions or offer additional opinions that delay decision-making and induce technical "swirl."

Mission and financial responsibility misalignment can also weaken decision authority by hindering an organizational component's ability to appropriately weigh cost and risk. Specifically, when an organizational component is responsible for executing multiple distinct missions for different collaborating partners but only one partner is responsible for funding those missions, the partner without financial responsibility is more likely to make technically conservative decisions or decisions that enable or sustain cost growth by minimizing risk. Alternatively, when mission and financial responsibility are aligned, an organization is able to make decisions that more effectively balance cost and risk.

Additional factors that erode decision authority can also be included in DSM_{DA} by adding +1 to each interface where decision authority is eroded; although authority erosion factors may be case-specific or not directly related to jointness, they should also be represented in DSM_{DA}. Once DSM_{DA} has been adjusted to account for all authority erosion factors, DSM_{DA} and DSM_{MR} are compared and misalignments between the two are identified. If decision authority and mission responsibility between two components are misaligned, an additional penalty is calculated and added according to the following process: First, the adjusted DSM_{DA} is transformed into a graph, where components are represented by nodes in the graph and decision authority links are represented by edges. Edge lengths correspond to the values in the adjusted DSM_{DA}. Next, when a mission responsibility link between two components exists, the weighting W_A, between those components is calculated by determining the shortest path length between components i and j, in the decision authority graph. The length of the decision authority path between two components with shared mission responsibility is intended to simulate the efficiency and effectiveness of organizational decision-making.

E. Step 5: Observe Evolution of Complexity Over Time

The utility of calculating a single complexity metric to represent a joint program's organizational and technical architectures is that it enables changes in those architectures and their relationship to the program's reported cost growth to be observed over time. Specifically, we recommend studying joint programs in terms of *epochs*, or periods of time when the program's organizational and technical architectures are unique and stable. One organizational and technical architecture should be defined per epoch, a complexity metric for each should be calculated, and the complexity should be plotted as a function of time and compared to the program's cost estimate during each epoch. Plotting the complexity metrics in this way enables the researcher to identify epoch shifts that induced complexity and to compare the program's own cost estimates to the complexity that was inherent to its architectures.

IV. Case Study: Jointness and the NPOESS Program

In this section, we demonstrate the above five step framework using the NPOESS program as an illustrative case study. NPOESS was a collaboration between the Department of Defense (DoD), the National Oceanic and Atmospheric Administration (NOAA), and the National Aeronautics and Space Administration (NASA) that was intended to develop a constellation of environmental monitoring satellites for low-Earth orbit; NPOESS was established in 1993 and cancelled—due to cost growth and schedule delays—in 2010. The NPOESS system met the requirements of multiple user groups and was developed collaboratively by all three agencies; as such, NPOESS is an important example of both organizational and technical jointness.

In order to apply our framework to study the impact of jointness on NPOESS, we collected a mix of qualitative and quantitative data. In total, we interviewed 57 representatives from the program and collected over 70 hours of semi-structured qualitative interview data [37]-[38]. As recommended by [39], we sampled interviewees from multiple levels in the program's organizational hierarchy and across numerous functional specialties and we triangulated [37] our interview data using over 150 primary and secondary source documents from the program. These documents provided quantitative technical data or official maps of the program's organization. Additional information on our data set or our qualitative research methods is contained in [40]-[41].

A. Step 1: Represent NPOESS Technical Architecture

The NPOESS DSM_T contains three types of components: spacecraft, instruments, and ground processing systems. The operational NPOESS constellation used three common spacecraft that were assigned to different longitude of ascending node crossing times and one non-common spacecraft for NASA's NPOESS Preparatory Project (NPP); NPP was a joint risk reduction-climate science mission that was developed collaboratively by NASA and the NPOESS program office. The DSM_T also contained 12 different instruments which were assigned to one or more spacecraft and components to represent system's algorithms and ground system. Although the ground system ultimately executed the algorithms, they were represented separately because the interface between these components was observed to have an effect on the system's cost.

Instrument design complexity was categorized using a ranking system similar to TRL but specifically focused on the instruments' relationship to heritage designs. Spacecraft designs were assigned a complexity rating of 0, 1, or 2 depending on how many complexity mechanisms were present; complexity mechanisms that were identified included bus and software design maturity. Process complexity mechanisms were identified similarly and the mechanisms noted in this case study are the same as those discussed in Section III. Finally, the architectural complexity mechanisms on the NPOESS program included design relationships, physical interfaces, and electromagnetic, optical, mechanical, and reliability budget interactions between components.

Step 2: Calculate NPOESS Technical Complexity

The process for calculating N_{cc} began by applying design complexity penalties (taken from [37]-[38]) to the instruments' mass. Instrument cost was then calculated using parametric equations given in [42]. Next, again using parametric equations from [42], complexity-corrected instrument mass was used to estimate spacecraft bus mass and bus non-recurring cost. Finally, the remaining architectural, design, and process complexity penalties were applied to the non-recurring cost estimates according to (3). The penalties, W_A, applied for bus design complexity, conflicting requirements process complexity, and architectural complexity were all taken from [39].

Recurring system costs were also estimated using parametric equations from [42] and the launch cost for each spacecraft was also included in L_{cc}. The NPOESS L_{cc} does not include ground system and operations costs because there is little quantitative understanding of disaggregation's impact on these costs [8] and because the NPOESS ground system architecture remained relatively constant throughout the program. Instead, changes to the ground system were modeled as complexity penalties on the components of the space segment with which they interfaced. Finally, each NPOESS L_{cc} was normalized by the analogous non-recurring, recurring, and launch costs for the NOAA-NASA Polar-Orbiting Environmental Satellite (POES) program and the DoD's Defense Meteorological Satellite Program (DMSP). POES and DMSP were the disaggregated programs that existed prior to NPOESS's formation in 1993.

B. *Step 3: Represent NPOESS Organizational Architecture*

The NPOESS DSM_o contained all three government agencies, the program's executive committee, both the NPOESS and NPP program offices, several councils of user groups, and the system's prime and sub-contractors for the spacecraft, instruments, algorithms, and ground system. While most instruments and algorithms were represented separately, we grouped leveraged sensors and their algorithms together. Leveraged sensors were those that did not require a significant amount of technology development or their development was not directly managed by the NPOESS program office; we made this decision because leveraged sensor contractors were observed to have a minimal impact on organizational behavior throughout the NPOESS program.

C. *Step 4: Calculate NPOESS Organizational Complexity*

NPOESS organizational complexity was calculated using the process described in Section III. In addition to the authority erosion factors noted previously, several case-specific authority erosion factors—including limited contractual authority, infrequent decision-making, weak financial responsibility, and weak technical capability—were included in the NPOESS DSM_o *Limited contractual authority* refers to the limitations that contracts can place on the government's authority over its contractors. On the NPOESS program, the TSPR-like contracts significantly limited the government's authority over contractors' design and development activities. *Infrequent decision-making* refers to organizational components—like the NPOESS executive committee—that failed to make timely decisions. Decision authority on the NPOESS program was weakened by infrequent decision-making because components were unable to effectively manage complexity or to efficiently respond to programmatic issues. *Weak financial responsibility***,** which was induced by cost-plus contracts and extended competition periods, also weakened decision authority on the NPOESS program because it prevented decisions from being made with a full appreciation of their costs. Finally, *weak technical capability* weakened decision authority because it hindered organizational components' ability to recognize and to manage technical complexity; this factor primarily affected the NPOESS program office.

D. *Step 5: Observe Evolution of Complexity Over Time*

To observe the evolution of complexity over time, we defined six epochs and represented the NPOESS program's organizational and technical architectures for each; the DSMs created for each epoch are available upon request. Fig. 4 identifies each epoch and the lifecycle cost that the program reported during that time; the costs shown reflect then-year dollars that were collected from [43]-[48]. Key events during each epoch include:

- **Epoch A** (1994-1996): NPOESS was established by converging the POES and DMSP programs and was motivated by the desire to save \$1.3 billion in lifecycle costs [49]. These cost estimates assumed that the NPOESS technical architecture would be composed to of a constellation of two operational spacecraft and would offer modest performance improvements compared to POES and DMSP.
- **Epoch B** (1996-1999): The NPOESS system requirements were defined and the technical architecture gained greater functionality and performance. The program office managed risk reduction contracts for key instruments but delayed selecting a prime contractor.
- **Epoch C** (1999-2002): The NASA-managed NPP program was established to provide additional risk reduction for key sensors and to provide data continuity for several climate science variables. To meet the needs of the new climate science users, requirements were updated to enhance instrument performance.
- **Epoch D** (2002-2005): An additional third operational orbit was added and the prime contractor was selected. Instrument and algorithm contracts previously managed by the government were transferred to the prime. Cost estimates began to grow.
- **Epoch E** (2005-2007): Costs grew so significantly that the program breached the Nunn-McCurdy threshold and had to undergo re-certification.[1] As a result, several instruments and spacecraft were cancelled and changes to the program's authority structure were implemented.
- **Epoch F** (2007-2010): Several instruments were added back to the program. Despite the new authority structure, management challenges persisted and costs continued to grow until the program's cancellation in 2010.

Fig. 4 suggests that the program's costs did not begin to increase until after Epoch D. This is consistent with previous studies of the program's history; specifically, [49] suggests that prior to Epoch D, the government significantly under-estimated the system's cost and in particular, the design complexity of its instruments. Other reports suggest that after Epoch D, the prime contractor's poor management contributed to further cost growth [46],[49]. By focusing on the complexity of program's organizational and technical architectures instead of its reported costs, we are able observe when complexity was injected into the program's organizational and technical architectures, which our metrics

[1] The Nunn-McCurdy Act requires defense programs to be re-certified by Congress it their cost growth exceeds a specified level.

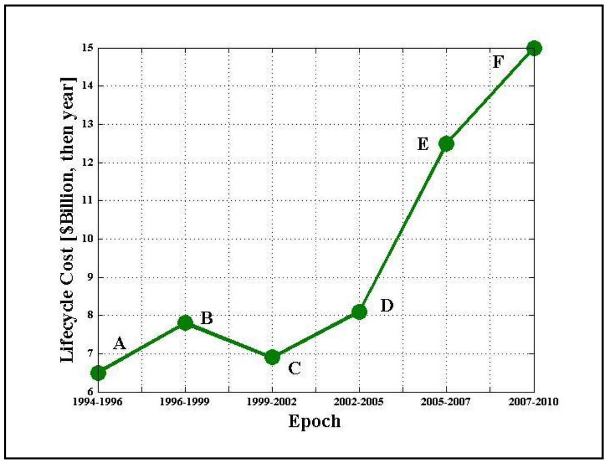


Fig. 4. Evolution of NPOESS Lifecycle Cost Estimates

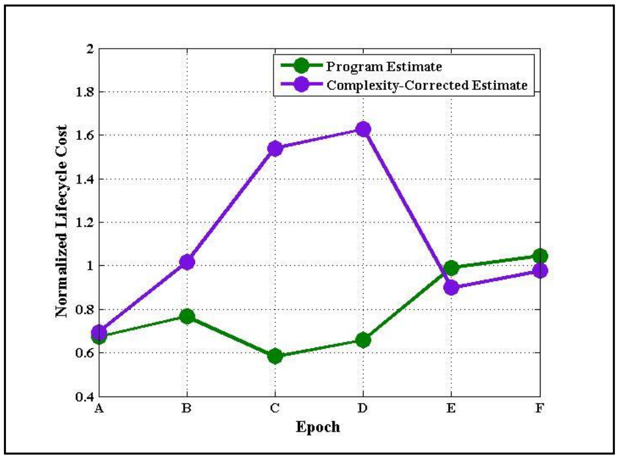


Fig. 5. Complexity Corrected Lifecycle Costs Compared to Program's Lifecycle Cost Estimate

suggest was actually well before the program's reported costs began to grow.

V. Discussion: Complexity, cost, and the NPOESS prorgam

Figures 6 and 7 illustrate how our framework was applied to observe the evolution of complexity and cost on the NPOESS program. First, Fig. 6 depicts a miniature version of each epoch's DSM_T and its corresponding $L_{cc.}$ This figure suggests that even according to our complexity-corrected lifecycle cost estimate, the program's initial aggregated technical architecture was less costly than the disaggregated POES and DMSP programs. However, as shown in Fig. 5, which compares the program's lifecycle cost estimates (corrected for inflation) to our technical complexity metric, L_{cc} increased significantly after Epoch A, while the program's cost estimates continued to remain low. This suggests that the changes to the technical architecture between Epochs A and C added a significant amount of design, process, and architectural complexity that was under-estimated and under-managed by the program.

The increase in the program's lifecycle cost estimates after Epoch D suggests that it was not until later epochs that the program began to recognize and to budget for this complexity. While our assessment agrees with previous studies that suggested that NPOESS under-estimated its costs, our complexity metric also suggests that the program's early under-estimation and under-management of technical complexity was most significant after Epoch B and had a much greater impact on its later cost growth than did any mismanagement by the program's prime contractor. Furthermore, our calculations suggest that the program's earliest cost estimate (Epoch A) was reasonable and that it was not until the technical architecture changed significantly between Epochs A and C, that the program's estimates failed to account for complexity. Additionally, our technical complexity metric identified two types of complexity mechanisms—process and architectural—that impacted the NPOESS program but have not been the focus of previous studies which typically emphasize instrument design complexity [49].

Fig. 7 depicts a miniature version of each epoch's DSM_O and its corresponding OC. Like technical complexity, during the program's early epochs, the aggregated NPOESS organizational architecture was less complex than the disaggregated POES and DMSP programs. However, like technical complexity, organizational complexity increased significantly after Epoch A and remained high throughout the remainder of the program. The increase in organizational complexity is attributed to the delayed selection of the prime contractor and addition of the NPP program since both of these changes misaligned authority and mission responsibility and induced many of the authority erosion factors identified above. Several epoch shifts, from Epoch C to D (after the prime contractor was selected) and D to F (after a Nunn-McCurdy mandated reorganization), reduced organizational complexity; however, because the NPP program was maintained, minor changes to the organizational architecture were unable to reduce the complexity that NPP initially induced.

In fact, we suggest that the misalignment of decision authority and mission responsibility induced additional non-technical cost growth during the program's later years. Specifically, when decision authority and mission responsibility are aligned, we suggest that decision-makers can more effectively respond to the needs of the organization. When they are misaligned, decisions are delayed, since information has to traverse multiple components until it reaches one with the authority to make a decision. On NPOESS, this misalignment was observed to induce standing-army costs since the organization's progress was delayed as it waited for a decision. It also enabled cost growth by separating decision-makers from sources of requirements creep or from critical interfaces that needed to be carefully specified and managed throughout the systems' development.

Finally, although the proposed metrics provide a means to observe the evolution of complexity on the NPOESS program, we have yet to directly link this observed complexity to the concept of jointness. Table I identifies which of the complexity mechanisms can be attributed to the NPOESS program's aggregated organizational and technical

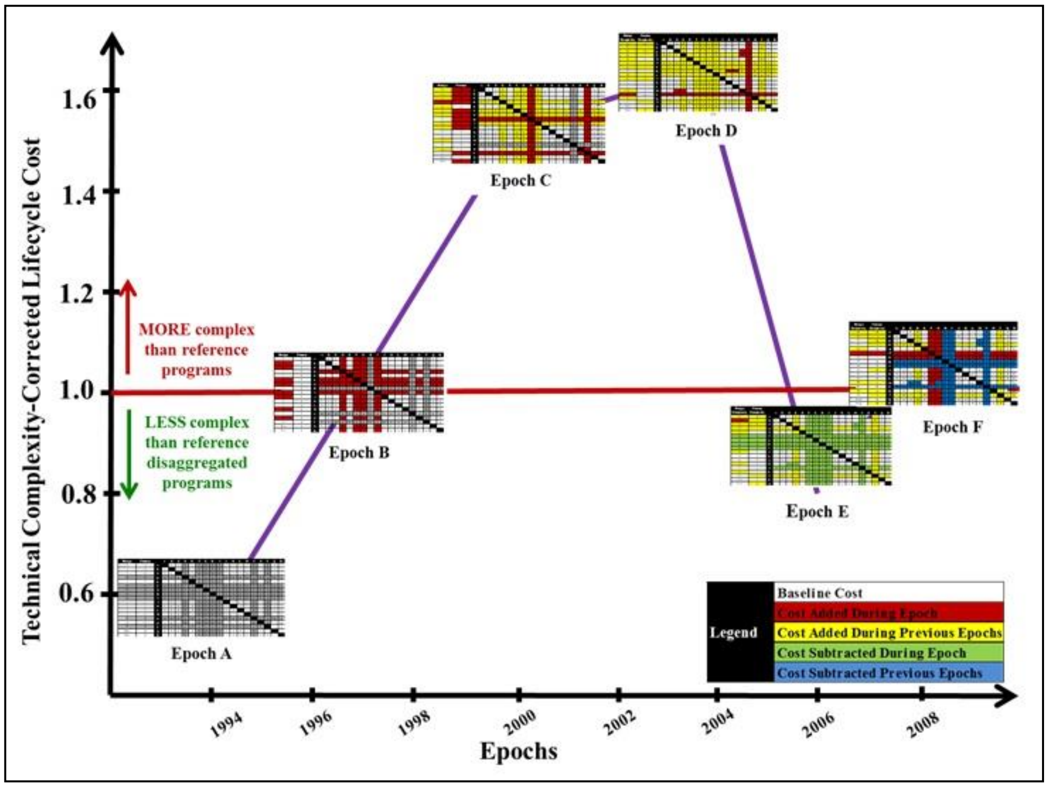


Fig. 6. Technical Complexity Metric Plotted for the Six Epochs of the NPOESS Program

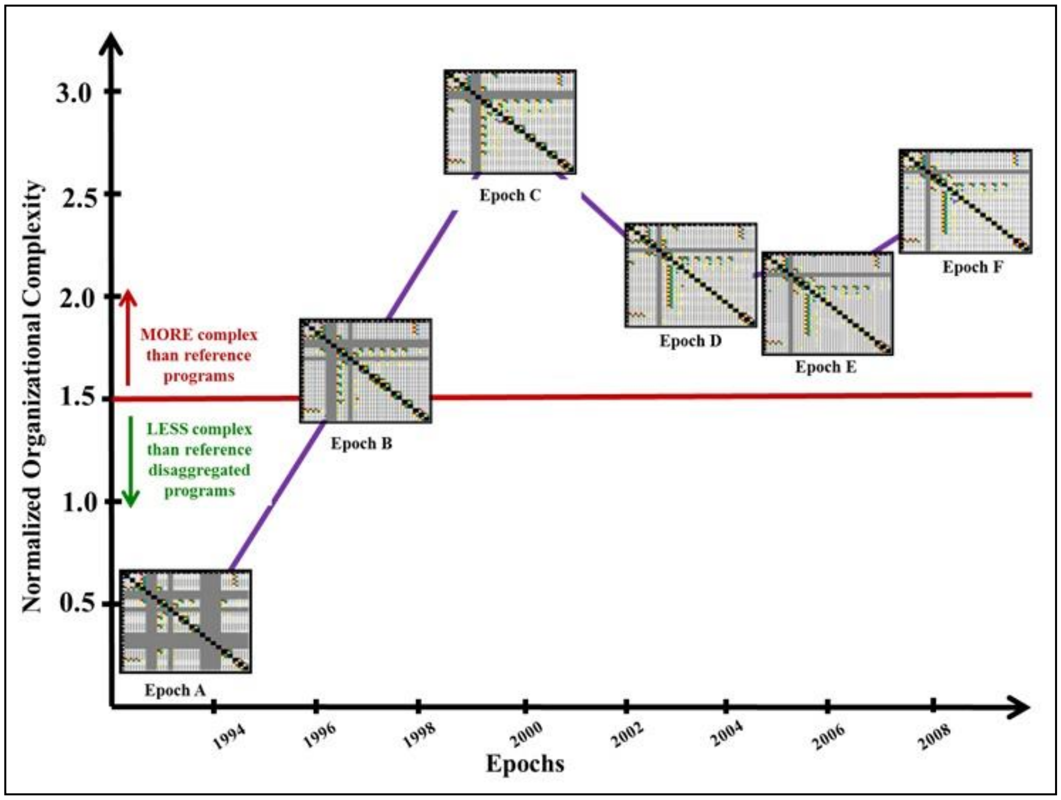


Fig. 7. Organizational Complexity Metric Plotted for the Six Epochs of the NPOESS Program

architectures and which can affect both joint and non-joint programs. This list will be refined by future research that compares the complexity mechanisms observed on NPOESS to those on other joint programs. Importantly, although not all of the mechanisms in the complexity metrics can be attributed to jointness, we included them in our analysis because we assessed their impact to be significant and suggest that organizational and technical complexity—regardless of its origin—can have emergent and coupled effects on a program's cost that should be captured by any holistic measure of a program's complexity.

VI. CONCLUSION

This paper presented a framework that can be used to understand the evolution of complexity and cost on joint programs and demonstrated this framework using a case study of the NPOESS program. Although we focused only one case study, our DSM-based complexity metrics were defined generally so that they could be applied to aid future research that explores complex system development in large government organizations, including but not limited to, organizations that acquire systems jointly. Specifically, we suggest that with an understanding of a particular technical system's design, process, and architectural complexity mechanisms, future researchers can represent system architectures using our proposed DSM framework and quantify complexity using a process similar to our technical complexity metric. Similarly, by mapping decision authority, mission responsibility, technical capability, and financial responsibility relationships between organizational components and noting instances where authority is eroded in an organization, future researchers can also apply our DSM framework to represent organizational architectures and to assess the complexity inherent within them.

TABLE I. COMPLEXITY MECHANISMS ATTRIBUTABLE TO JOINTNESS

Architecture Type	Complexity Type	Complexity Mechanism	Joint?
Technical	Design	Instrument Design Maturity	X
Technical	Design	Bus Design Maturity	X
Technical	Design	Software Design Maturity	X
Technical	Process	SETA-Like Governance Structure	
Technical	Process	Conflicting Requirements	X
Technical	Architectural	EMI Interactions	X
Technical	Architectural	Mechanical Interactions	X
Technical	Architectural	Optical Interactions	X
Technical	Architectural	Programmatic Interactions	X
Technical	Architectural	Reliability Interactions	X
Technical	Architectural	Physical Interfaces	X
Organizational	Misalignment	Misalignment of Mission Responsibility & Decision Authority	X
Organizational	Authority Erosion	Limited Contractual Authority	
Organizational	Authority Erosion	Infrequent Decision Making	X
Organizational	Authority Erosion	Weak Financial Responsibility	
Organizational	Authority Erosion	Weak Technical Capability	
Organizational	Authority Erosion	Multiple Sources of Technical Capability	X
Organizational	Authority Erosion	Mission & Financial Responsibility Misalignment	X

ACKNOWLEDGMENT

This work was sponsored by the Sandia National Laboratories – MIT Excellence in Engineering Graduate Student Fellowship.

REFERENCES

[1] *Department of Defense Dictionary of Military and Associated Terms*. Washington, D.C.: Defense Technical Information Center, 2010.

[2] Lorell, M.A., Kennedy, M., Leonard, R.S. , Munson, K., Abramzon, S., An, D.L. et al. *Do joint fighter programs save money?*, Santa Monica, CA: RAND Corporation, 2013.

[3] Brown, M.M., Flowe, R.M. and Hamel, S.P.. "The Acquisition of Joint Programs: The Implications of Interdependencies," *CROSSTALK: The Journal of Defense Software Engineering*, 2007.

[4] Cameron, B. *Costing Commonality: Evaluating the Impact of Platform Divergence on Internal Investment Returns* (Doctoral Dissertation). Available on MIT DSpace, http://hdl.handle.net/1721.1/68511, 2011.

[5] Committee on the Assessment of Impediments to Interagency Cooperation on Space and Earth Science Missions. *Assessment of Impediments to Interagency Collaboration on Space and Earth Science Missions*. Washington, D.C.: The National Academies Press, 2011.

[6] Crawley, E.F., DeWeck, O., Eppinger, S., Magee, C., Moses, J., Seering, W. et al. *The Influence of Architecture in Engineering Systems – Engineering Systems Monograph*. Cambridge, MA: MIT Engineering Systems Division, 2004.

[7] Resiliency and Disaggregated Space Architectures: White Paper. Air Force Space Command, 2013

[8] Rendleman, J.D. "Why SmallSats?" Proceedings of the AIAA Space 2009 Conference & Exposition. Pasadena, CA, 2009.

[9] Burch, R. "OPS: Disaggregation & Diversification of U.S. MILSATCOM." *Milsat Magazine*. April 2012. Online: http://www.milsatmagazine.com/story.php?number=510061234, Accessed 1/13/2014.

[10] Pawlikowski, E., Loverro, D., Cristler, T. "Space: Disruptive Challenges, New Opportunities, and New Strategies." *Strategic Studies Quarterly*, 2013.

[11] Taverney, T. "Resilient, Disaggregated, and Mixed Constellations." *The Space Review*, August 2011. Online: http://www.thespacereview.com/article/1918/1, Accessed 1/13/2014

[12] Bogdanos, M.F. *Joint Interagency Cooperation: The First Step*. Washington, D.C.: National Defense University, 2005.

[13] Johnson, D.J., Hilgenberg, G.H., Sarsfield, L.P. *Policy Issues and Challenges for Interagency Space Systems Acquisition*. Santa Monica, CA: RAND, National Security Division, 2001.

[14] Moore, A.P., Novak, W.E., Cohen, J.B., Marchetti, J.D., Collins, M.L. "The Joint Program Dilemma: Analyzing the Pervasive Role that Social Dilemmas Play in Undermining Acquisition Success." *Proceedings of the 10th Annual Acquisition Research Symposium Acquisition Management*. Monterey, CA, 2013.

[15] Eppinger, S., Browning, T. *Design Structure Matrix Methods and Applications*. Cambridge, MA: The MIT Press, 2012.

[16] Suh, E.S., Furst, M.R., Mihalyov, K.J., deWeck, O. "Technology Infusion for Complex systems: A Framework and Case Study." *Systems Engineering* (2009) Vol. 13, No 2, pp.186-203.

[17] Sosa, M.E., Browning, T., Mihm, J. "Studying the Dynamics of the Architecture of Aoftware Products." Proceedings of the AMSE 2007 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Las Vegas, NV, 2007.

[18] Smaling, R., deWeck, O. "Assessing Risks and Opportunities of Technology Infusion in System Design." *Systems Engineering* (2007), Vol. 10, No. 1.

[19] Committee on Cost Growth in NASA Earth and Space Science Missions. *Controlling Cost Growth of NASA Earth and Space Science Missions.* Washington, D.C.: The National Research Council, 2010.

[20] Selva, D. *Rule-Based System Architecting of Earth Observation Satellite Systems* (Doctoral dissertation) Available from MIT DSpace, http://hdl.handle.net/1721.1/76089, 2012.

[21] Alibay, F., Strange, N.J. "Trade Space Evaluation of Multi-Mission Architectures for the Exploration of Europa." *Proceedings of the 2013 IEEE Aerospace Conference*. Big Sky, MT, 2013.

[22] Rasmussen, A., Tsugawa, R. "Cost effective applications of constellation architectures of large, medium, and small satellites." *American Institute of Aeronautics & Astronautics*, AIAA Paper 97-3950. 1997.

[23] Dubos, G.F., Saleh, J.H. "Spacecraft technology portfolio: probabilistic modeling and implications for responsiveness and schedule slippage." *Acta Astronautica* (2011), Vol. 68, pp. 1126-1146.

[24] Brady ,T., Nightingale, D. "NASA Mars Pathfinder technology readiness." *in* Eppinger, S., Browning T. *Design Structure Matrix Methods and Applications*. Cambridge, MA: The MIT Press, 2012.

[25] American National Standards Institute / American Institute of Aeronautics and Astronautics. *Mass Properties Control for Space Vehicles, Missiles, and Launch Vehicles (ANSI/AIAA R-020A-1999).* Washington, DC: AIAA, 1999.

[26] Leising, C.J., Wessen, R., Ellyin, R., Rosenberg, L., Leising, A. "Spacecraft complexity subfactors and implications for future cost growth." *Proceedings of the 2013 IEEE Aerospace Conference*; Big Sky, MT. ; 2013.

[27] Sheard, S.A., Mostashari, A. "A complexity typology for systems engineering. *Proceedings of 2010 International INCOSE Symposium*. Philadephia, PA, 2010. [Internet]

[28] Leising, C.J., Sherwood, B., Adler, M., Wessen, R., Naderi, F.M. "Recent Improvements in JPL's Mission Formulation Process." *Proceedings of the 2010 IEEE Aerospace Conference*, Big Sky, MT, 2010.

[29] Henderson, R.M., Clark, K.B. "Architectural innovation: the reconfiguration of existing product technologies and the failure of established firms." *Administrative Science Quarterly* (1990), Vol. 35, Issue 1, pp. 9-30.

[30] Baldwin, C., Clark, K.B. *Design Rules: Volume 1, The Power of Modularity*. Cambridge, MA: The MIT Press, 2000.

[31] Sheard, S.A. *Assessing the Impact of Complexity Attributes on System Development Project Outcomes*. (Doctoral Dissertation). Available on Proquest, http://dl.acm.org/citation.cfm?id=2518902.

[32] Cameron, B. *Costing Commonality: Evaluating the Impact of Platform Divergence on Internal Investment Returns* (Doctoral Dissertation). Available on MIT DSpace, http://hdl.handle.net/1721.1/68511, 2011.

[33] Malone P., Wolfarth, L. "Measuring System Complexity to Support Development Cost Estimates." *Proceedings of the 2013 IEEE Aerospace Conference,* Big Sky, MT, 2013.

[34] Sinha, K., de Weck, O. "Structural Complexity Quantification for Engineered Complex Systems and Implications on System Architecture and Design. *Proceedings of the International Design Engineering Technical Conference;* Portland, OR, . 2013.

[35] Manthorpe, W.H.J. "The Emerging Joint System of Systems: A Systems Engineering Challenge and Opportunity for APL." *Johns Hopkins APL Technical Digest* (1996), Vol. 17, No. 3.

[36] *Joint Program Management Handbook*. 3rd edition. Fort Belvoir, VA: Defense Acquisition University Press, 2004

[37] Harris H., Lewis, M. "Proposed Leadership Structure for Joint Acquisition Programs." *Acquisitions Research Journal* (2012), Vol. 19, No. 1, pp. 33-52.

[38] Eisenhardt, K.M., Graebner, M.A. "Theory-Building from Cases: Opportunities and Challenges." *Academy of Management Journal* (2007), Vol. 19, No. 1, pp. 33-52.

[39] Yin, R.K. *Case Study Research: Design and Methods,* 4th ed. Los Angeles: SAGE, 2009.

[40] Larson. W., Wertz, J.R., eds. *Space Mission Engineering: The New SMAD.* 1st ed. Hawthorne, CA: Microcosm, 2011.

[41] Dwyer, M., Szajnfarber, Z., Cameron, B. Bradford, M., Crawley, E. "The Cost of Jointness: Insights from the NPOESS Program." *Proceedings of the 2014 Acqusitions Research Conference,* Monterey, CA, 2014.

[42] Dwyer, M. *The Cost of Jointness: Insights from Environmental Monitoring Satellites in Low-Earth Orbit* (Doctoral Dissertation, expected fall 2014)

[43] *Agencies Must Act Quickly to Address Risks that Jeopardize the Continuity of Weather and Climate Data*. Washington, D.C.: The Government Accountability Office, 2010. Report No.: GAO-10-558.

[44] *Polar-orbiting Operational Environmental Satellites: Restructuring is Under Way, but Technical Challenges and Risks Remain*. Washington, D.C.: United States Government Accountability Office, 2007. Report No.: GAO-07-498.

[45] *Technical Problems, Cost increases, and Schedule Delays Trigger Need for Difficult Trade-Off Decisions*. Washington, D.C.: The Government Accountability Office, 2005. Report No.: GAO-06-249T.

[46] *Polar-Orbiting Environmental Satellites: Information on Program Cost and Schedule Changes*. Washington, D.C.: The Government Accountability Office; 2004. Report No.: GAO-04-1054.

[47] "Final Phase 0 Cost and Operational Benefits Requirements Analysis Report (COBRA)." *National Polar Orbiting Operational Environmental Satellite System Integrated Program Office*, 1996.

[48] Polar-Orbiting Environmental Satellites: Status, Plans, and Future Data Management Challenges. Washington, D.C.: The Government Accountability Office, 2002. Report No.: GAO-02-684T.

[49] *NPOESS Lessons Evaluation: Executive Summary*. El Segundo, CA: The Aerospace Corporation, Space Systems Division, 2010. Report No.: ATR-2011(5558)-1.