

Article

Systems Developmental Dependency Analysis for Scheduling Decision Support: The Lunar Gateway Case Study

Cesare Guariniello *  and Daniel DeLaurentis 

School of Aeronautics and Astronautics, Purdue University, 701 W Stadium Ave., West Lafayette, IN 47907, USA; ddelaure@purdue.edu

* Correspondence: cguarini@purdue.edu

Abstract: Project Managers face many difficulties when scheduling the development and production of multiple, largely independent systems required for a new capability, especially when there are multiple stakeholders, uncertainties in the expected development time, and developmental dependencies among the systems. The Systems Developmental Dependency Analysis methodology provides a systemic approach to address these challenges by offering decision support for such a ‘System-of-Systems’. The method, based on a parametric piece-wise linear model of dependencies between elements in the developmental domain, propagates the interactions between systems to estimate delays in the development of individual systems and to evaluate the impact of such delays on the expected schedule of completion for the establishment of the whole desired capability. The schedule can be automatically re-generated based on new system information, changed dependencies, and/or modified risk levels. As demonstrated in this paper using a complex space mission case, the method enhances decision-support by identifying criticalities, computing possible delay absorption strategies, and comparing different development strategies in terms of robustness to delays.

Keywords: system of systems; project management; scheduling; delays; lunar gateway



Academic Editors: Michael Henshaw and Siyuan Ji

Received: 29 December 2024

Revised: 26 February 2025

Accepted: 5 March 2025

Published: 9 March 2025

Citation: Guariniello, C.; DeLaurentis, D. Systems Developmental Dependency Analysis for Scheduling Decision Support: The Lunar Gateway Case Study. *Systems* **2025**, *13*, 191. <https://doi.org/10.3390/systems13030191>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

A System of Systems (SoS) is a collection of systems designed to achieve specific capabilities, distinguished by two features: component systems can and do operate independently of the whole SoS and are also managed, at least in part, for their own purpose [1]. Since the elements are designed and developed (mostly) independently, the overall schedule of the development of an SoS emerges through the interactions of the component systems under development. In such settings, it is useful to adopt a holistic viewpoint of the SoS schedule, accounting for partial dependencies, meaning that only a fraction of the development of a system is dependent on the development of other systems. Thus, a dependent system can be developed with a lead time, that is, a period of the development of this system occurs before the full development of a preceding system (Figure 1).

In traditional approaches, the developmental dependencies between systems are usually modeled as *absolute*, or “on-off”: for example, in the Project Evaluation and Review Technique (PERT) and Critical Path Method (CPM), a system must wait until the predecessor is fully developed before its development can begin [2]. Occasionally, seasoned managers plan the lead time according to their experience, but they rarely assess the need for changes to account for the ongoing reliability and delays of the current development schedule. Various authors have recognized that not many project management tools have

satisfactory effectiveness in complex programs [3–5]. Moreover, the nature of Systems of Systems, characterized by the presence of multiple, potentially independent stakeholders, introduces other challenges in program management and development scheduling. For example, participating stakeholders are likely to introduce their own biases, which impact the decision-making process and the design of the whole system [6]. This impact is further exacerbated if some power imbalance exists among stakeholders [7].

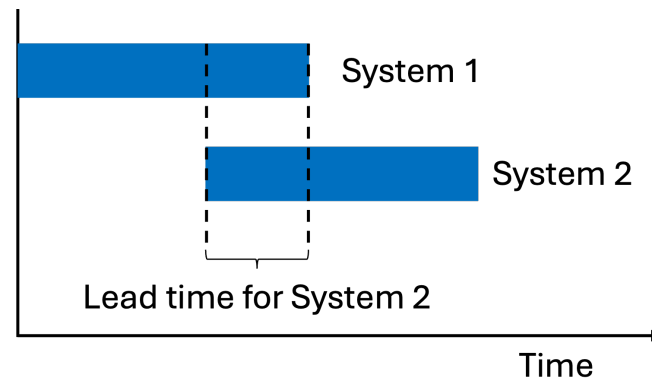


Figure 1. The development of System 2, depending on the development of Systems 1, can start with a lead time, during which both systems are being developed.

To address these limitations in the development schedule when dealing with complex systems, we propose Systems Developmental Dependency Analysis (SDDA). This method is based on a predecessor approach, used by Guariniello et al. [8] and by DeLaurentis et al. [9] to perform studies of the development of a Naval Defense System-of-Systems, encompassing the Littoral Combat Ship package [10], an analysis of criticalities in the development of satellite networks, and studies of alternative development schedules for space exploration systems.

The goal of SDDA is to model and analyze the developmental dependencies between systems, and to assess the impact of such dependencies via the cascading propagation of delays. SDDA computes the impact of delays and stakeholder decisions on the overall development based on the topology of a development network. The basic formulation of an SDDA model can be used to address various problems of interest in Project Management and Development Scheduling, providing capabilities that extend traditional PERT/CPM techniques. For example, by modeling and analyzing partial developmental dependencies, and accounting for the independence of stakeholders, SDDA can better identify critical systems and dependencies, quantify the robustness of the development network in terms of delay absorption, quickly analyze the effect of different stakeholder decisions, and evaluate the outcome of different developmental choices under various levels of reliability and risk acceptance. SDDA also allows for a trade-off between competing desired features, for example the time necessary to complete the overall development, risk, ability to absorb delays, and time spent on each system. The development schedule provided by SDDA can be automatically re-generated, based on new system information, modified dependencies, and/or modified risk levels. This feature provides the proposed method with the ability to account for the expected reliability of the participating stakeholders. SDDA also provides estimates of the “right time” to begin the development of dependent component systems in a System of Systems, exposing trade-offs between the exploitation of possible lead times for the early completion of development and the potential risks (for example, a higher cost or waste of resources) associated with an early start. This capability has a direct impact on the acquisition of complex systems and can also shed light on the viability of a concurrent approach [11–13] for the project under analysis and the optimization of limited budgets.

Finally, the basic formulation of SDDA is open for expansion to include stakeholder biases and potential demands, thus directly addressing typical System of Systems concerns.

2. Related Work and Prior Research

2.1. Dependencies, Risk Propagation, and Uncertainty in Development

Various authors proposed methods to deal with systems dependencies and the analysis of risk propagation among systems. Mane and DeLaurentis [14,15] and Mane et al. [16] analyzed the effect of dependencies in SoS development by means of a Markov network approach, meant to evaluate delay propagation before absorption. The method ranks systems based on the criticality of their impact on the development of the SoS; however, it does not consider partial dependencies, and it propagates delays to only one of the systems dependent on the delayed system. Other authors analyze risk propagation in a network of interdependent systems with simple models of the binary propagation of risk or delay [17], or with a qualitative description of the impact of delays [18]. Uncertainty in delay propagation, the amount of information available in a project or the lack thereof, and subsequent decisions in program management make the problem even harder. Yassine et al. note the churning effect caused by potential information-hiding, which results in oscillating delays in the development of a product [19]. It is therefore imperative to identify ways to evaluate performance and risks associated with complex projects, as well as appropriate responses [20]. Cheng and Ma [21] assess design dependencies in development based on an analysis of changes. These methods constitute a valid foundation upon which to assess the parameters of the dependencies in the SDDA model, which can then be used for a more thorough analysis of the development schedule and associated risks. Based on the needs highlighted in these publications, our work is meant to provide a systems perspective to the problem of decision support, which will be used in parallel with a managerial perspective [22]. A common framework used in systems engineering to deal with dependencies is the Design Structure Matrix (DSM). Analogous to the adjacency matrix in graph theory, DSM are used to model and analyze system structural features and dependencies [23,24]. DSM uses both metrics from graph theory and ad hoc algorithms for clustering systems and support organization in system development. DSM can support SDDA users in deciding on the adequate set of nodes required to model the system development and in identifying clusters in SDDA networks. SDDA can then identify criticalities inside and among clusters and suggest ways to shape the dependencies and the development schedule in order to improve developmental architectures and support policies to decrease cost and risk.

2.2. Decision Analysis, Stakeholder Bias, and Dynamic Risk Management

The decision-making process for the development of complex projects also includes aspects that go beyond the technical perspective. This is further exacerbated in the realm of System of Systems, characterized by the presence of multiple independent stakeholders that manage and operate component systems. White et al. [25] analyze dynamic preferences in Systems Engineering and highlight various difficulties: some approaches, for example Decision-Based Design (DBD), combine stakeholder preferences into a single static stakeholder [26,27], while other approaches delve into the social aspects of decision-making, including stakeholder biases. These aspects are hard to model due to the high level of uncertainty and unpredictability, yet it is important to give them appropriate consideration [28].

Various authors propose different ways to address the uncertainty stemming from stakeholder biases and the multiplicity of perspectives. For example, Mihret et al. suggest the use of appropriate policies to guide collaboration between multiple stakeholders to

facilitate the achievement of System of Systems goals [29]. Elfrey et al. reported on the usefulness of standards for international collaboration to facilitate both communication and fruitful cooperation among diverse groups [30].

In addition to identifying strategies to enhance collaboration, various authors highlight the importance of modeling residual risks and identifying ways to address and manage these risks. For example, Liu et al. [31] propose a combination of Failure Mode and Effects Analysis (FMEA) [32,33] and dynamic interaction models to manage risks.

Specializing in the risks associated with the development of Systems of Systems, the SDDA methodology is built in a way that makes it viable to model the risks stemming from stakeholder preferences and from the independent development of constituent systems. Compared to FMEA, SDDA can model multiple disruptions occurring at the same time while retaining the ability to identify the root causes of delays and critical systems related to their development times.

2.3. Classic Scheduling Methodology

Since 1959, industry has heavily relied on PERT/CPM for the scheduling and analysis of the time and cost of project development [34]. The concepts of expected times and latest time are still in use, with few changes from the original formulation, which assumes *absolute* dependencies for events and can treat the stochastic case only using a few selected Probability Density Functions (PDF). In the same year, Kelley and Walker added the coordination of activities, revision, and analysis of critical path to their CPM methodology [35]. Many studies have been performed and modifications have been added over time to PERT/CPM, beginning with an evaluation of the expected critical path and completion time of projects, based on the use of a simple combination of discrete random variables to model the completion time of individual tasks [36,37]. Cinicioglu and Shenoy proposed a way to solve stochastic PERT networks analytically using Bayesian networks [38], and Azaron and Tavakkoli-Moghaddam suggested a method to determine the time and cost trade-off in dynamic PERT networks [39]. Hameri and Heikkila [40] suggest increasing the emphasis on time usage along the critical chain of tasks, although the entire control of the schedule and of the time spent on single tasks is still the role of the program manager. In support of this kind of decision, a set of objective indices to monitor the performance of a project schedule is provided by Khamooshi and Golafshani [41]. Muriana and Vizzini [42] propose a deterministic technique for project risk assessment based on PERT.

2.4. Advanced Scheduling

PERT and CPM are powerful and well-established technologies. However, they still present some limitation when addressing specific challenges associated with complex systems, and particularly Systems of Systems. A literature survey in relevant areas in project management, decision support, intelligent and simulation-based scheduling, as well as delay propagation, shows that authors recognize the need for lead times accounting for possible partial dependencies [43,44] and have been advocating for intelligent scheduling and rescheduling for a long time [45–47]. The most common approach is to use static values for lead times [48] and reactive rescheduling, based on the current behavior of the systems under development, in terms of delays, rather than the expected behavior. Boehm et al. [49] surveyed various estimation methods for development schedules in software engineering. The common approach in this case is to compress the schedule by allocating an adequate number of workers to each activity. Some authors also underline the importance of modeling entire networks of projects when known developmental dependencies exist and when dealing with constituent systems in an SoS. The suggested approach in this case is to use traditional PERT/CPM methodology, using expert data for estimation, augmented

by on-demand scheduling and the improved visibility of work-in-progress and system status [50]. Some authors propose methods for simulation-based scheduling [51,52]. These methods have the advantage of optimizing the scheduling and can include the allocation of resources, at the cost of not being viable for the identification of good and bad scheduling patterns and sources of risk. When projects become too complex for simple simulations, intelligent scheduling methods, based on fuzzy logic, Machine Learning (ML), or other search algorithms, can be used [53,54]. Some of these works also highlight the problems involved with uncertainty and large systems and suggest methodologies to estimate development times [19,20]. Krishnan, Eppinger, and Whitney propose a framework to model overlapping in a basic scenario, although without accounting for delays [55]. The model is based on the required exchange of information to maximize parallel development, and can be used as one of the main sources for SDDA parameters. The impact of delays in complex projects is often evaluated only qualitatively. Other models of delay propagation are specific to fields of application, for example airline flight scheduling [56] and train scheduling [57], which are characterized by clear, non-overlapping sequences of vehicles and centralized control. This kind of approach cannot model the independence of stakeholders that characterizes SoS. Regarding risk assessment, Willumsen et al. [58] recognize the importance of the perceived value of risk assessment and mitigation, while Belvedere et al. [59] identify misperceptions related to the wastefulness that occurs in projects. Wang et al. [60] suggest using system dynamics to quantitatively evaluate the value of risk mitigation under uncertainties.

SDDA builds upon some of the ideas and methodologies in the areas of project management, systems engineering, and scheduling briefly described in this section to address some of the needs found in the literature, especially regarding the specific challenges presented by a System of Systems. The SDDA methodology provides a quantitative model for delay propagation, intelligent scheduling, and decision support, which also helps to address potential biases and includes considerations of the possible decisions taken by independent stakeholders and the impact of such decisions on risk and performance in the developmental domain.

3. Methodology

3.1. Overview

Systems Developmental Dependency Analysis (SDDA) is a methodology that includes a parametric model of the interactions between interdependent systems or technologies, where the dependencies occur in the developmental domain, meaning that the development of a system or a technology is at least partially dependent on the development of one or more other systems or technologies. Like in PERT/CPM, these developmental dependencies imply a temporal dependency during development. However, the dependencies in traditional models are *absolute*, meaning that the development of a predecessor (or execution of a task) must be fully completed before the following can begin. The development scheduling of System of Systems cannot always be modeled to a level of granularity that would result in this kind of *absolute* dependency and require a more complex model that can account for partial and possibly varying dependencies, which also take into consideration the performance and decisions associated with independent stakeholders. The SDDA model is based on previous work by Garvey and Pinto [61,62] on Functional Dependency Network Analysis (FDNA), a parametric model of the dependencies between system capabilities, which uses two parameters called the Strength of Dependency (SOD) and Criticality of Dependency (COD). SDDA adapts these concepts to identify two parameters characterized by an intuitive meaning associated with developmental dependencies. The use of such parameters facilitates the modeling process. SDDA is the counterpart—in the developmental domain—of a methodology proposed by Guariniello and DeLaurentis for

the analysis of dependencies in the operational domain, Systems Operational Dependency Analysis (SODA, [63]).

The raw outcome of SDDA modeling and analysis is a quantitative assessment of the expected beginning and completion time of activities (i.e., development of systems, technologies, or SoS capabilities) in a project. This assessment accounts for the combined effect of multiple developmental dependencies and for possible delays in the development of predecessors in two different ways: in accordance with the partial independence of stakeholders in a System of Systems, delays in the development of a predecessor are not simply added to the beginning time of a successor, but also impact the lead time in a way that models the reduced trust in and reliability of the stakeholder in charge of the predecessor. Therefore, the lead time, i.e., the amount of time in which a system can begin to be developed before a predecessor is fully developed, is calculated based both on the properties of the dependencies and on the expected and current performance of other stakeholders. The completion time of the development of each individual system or technology is then calculated based on the lead time and on the expected time of development of the individual system or technology. The SDDA methodology allows for deterministic or stochastic analysis based on the same SDDA model. In deterministic analysis, a certain amount of delay is assigned to each system, and SDDA evaluates the resulting schedule. In stochastic analysis, the amount of delay in each system follows a Probability Density Function, resulting in a stochastic beginning and completion time of each system or technology. The most direct outcomes of this analysis identify the most critical systems, technologies, and dependencies with respect to the overall development time, the delay absorption, and the delay propagation. This provides an important decision support tool for system managers and SoS architect(s). In addition, the availability of a quantitative and objective assessment of the development schedule and the risk associated with developmental decisions can help alleviate the challenges posed by stakeholder independence and potential biases. The SDDA model was developed in a way that can accommodate various types of procedures. For example, the results from the baseline analysis can be used to compare different architectures in terms of their development time, risk, and ability to absorb delays; different decisions by stakeholders regarding the priority to be given to the development of specific systems or capabilities can also be included. A few years ago, the methodology became part of a graduate-level collegiate course in System of Systems Modeling and analysis, which contributed to the development presented in this work. Part of the description of the formulation of the SDDA methodology is based on the textbook associated with the course [64].

3.2. Developmental Dependencies

SDDA uses a network representation of developmental dependencies, where nodes represent systems or technologies (or a mix of the two). The links, like in PERT networks, represent the developmental dependencies between systems (Figure 2). System j is developmentally dependent on system i if the development of system j needs some input (information, or other types of developmental dependencies) related to the development of system i . As mentioned above, differently from PERT, the dependencies are not *absolute* and account for the partial independence of development of each system, a feature that is especially important in the SoS context. In current practice, this partial independence is often modeled with a fixed lead time, usually evaluated solely based on expert judgment. SDDA instead provides a simple but more realistic model of the outcome of these partial dependencies. Each system or technology i in an SDDA network is associated with three pieces of input data.

The first two inputs (Figure 2) are the minimum independent development time t_{min}^i , and the maximum independent development time t_{max}^i . These are, similarly to PERT's optimistic time and pessimistic time, the minimum and the maximum expected time to complete the development of system i , not taking the dependencies into account. The inputs are analogous to the optimistic and pessimistic time in PERT. However, since SDDA also accounts for dependencies, the combination of a large lead time (optimistic expectations) and delays in the predecessors might result in an actual completion time that is longer than the maximum independent development time, consequently potentially wasting resources. Once again, this feature reflects the impact that the various stakeholders may have on each other, and SDDA provides means to identify potentially critical situations and to support decision-making that provides a trade-off between the advantages of early development and risk of delays and resource mismanagement.

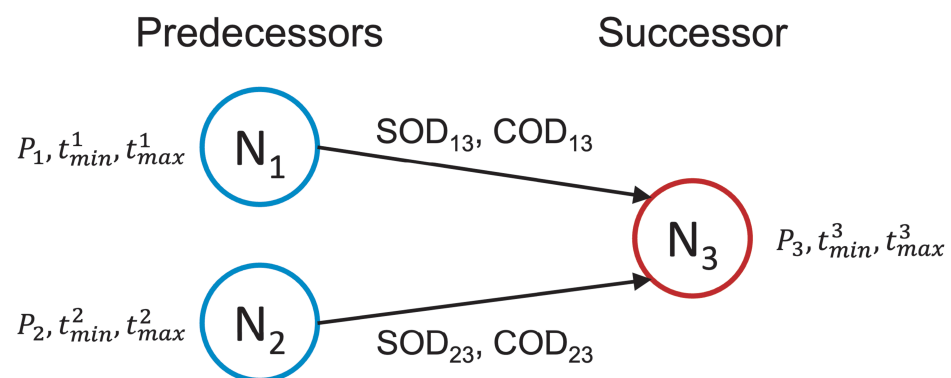


Figure 2. Example of a three-node developmental dependency network (from [64]). N: node; SOD: Strength of Dependency; COD: Criticality of Dependency; P : punctuality; t_{min} : minimum independent development time; t_{max} : maximum independent development time.

The third input associated with a system is the variable used to run the baseline SDDA analysis: this variable represents the timeliness of system development, or punctuality P_i . Punctuality, normalized between 0 and 100, is an assessment of the magnitude of development time with respect to the minimum development time. High punctuality corresponds to a short time ($P = 100$, meaning that the system is being developed in the minimum independent time) and vice versa ($P = 0$, meaning that the system is being developed in the maximum independent time). While the relation between punctuality and the time taken to develop a system is not required to be linear, in this work we used a simple linear correlation between punctuality and independent development time. When performing deterministic analysis, single numbers are assigned to the punctuality of each system, while for stochastic analysis, a Probability Density Function (PDF) is associated with each punctuality variable. When punctuality is not known *a priori*; for example, when estimating the punctuality of an independent stakeholder or the development of a technology in the near or far future, SDDA can be used to compare the outcome of different assumptions and different probability distributions of punctuality. Many existing studies and methodologies provide the initial and expected values of punctuality in specific fields, based, for example, on Technology Readiness Levels (TRL) [65,66] or Maturity Models [67].

Each developmental dependency, i.e., each link in an SDDA network, is associated with two parameters: Strength of Dependency (SOD) and Criticality of Dependency (COD). The small number of SDDA model parameters and their intuitive meaning, described below, make them suitable to be assessed by knowledgeable designers and managers. For larger problems, sets of predefined parameters can be identified that model typical system-to-system developmental dependencies (for example, *strong but unreliable dependency*). At the same time, these parameters overcome PERT/CPM's inability to manage

partial dependencies and dynamic lead time. Figure 3 shows the relation between the completion time of a predecessor system i (function of its punctuality P_i) and beginning time of a successor system j . In general, the parameters of a SDDA model can have three different sources:

- Expert judgment and evaluation: A survey conducted with various Program Managers indicated that, in most complex and large-scale projects, lead times and scheduling with overlapping activities are established by Subject Matter Experts (SME). For the SDDA model, a similar approach can be used. The use of SDDA, though, has the advantage that the SMEs do not have to directly evaluate the amount of overlap or the lead times based on various considerations regarding the stakeholders, but must only assess the strength of the one-to-one developmental dependencies based on the definitions provided below.
- Assessment based on historical data. This approach can be used to assess both the amount of information transfer that is necessary between two activities (and therefore the time overlap that can be assigned) and to evaluate the impact of stakeholder behavior and preferences.
- Evaluation based on existing models. This approach makes use of studies such as the framework used to overlap product development activities proposed by Krishnan, Eppinger, and Whitney [55], based on the required exchange of information for parallel development. Other methods are also available in the literature that can be used to quantify developmental dependencies [68,69]. Virtual developmental dependencies can also be added to model stakeholder decisions; for example, the choice to develop a specific system or technology ahead of some other system or technology, even when an actual developmental dependency does not exist. Other models that describe the effect of stakeholder bias and the implementation of policies can be used to quantify the SOD and COD.

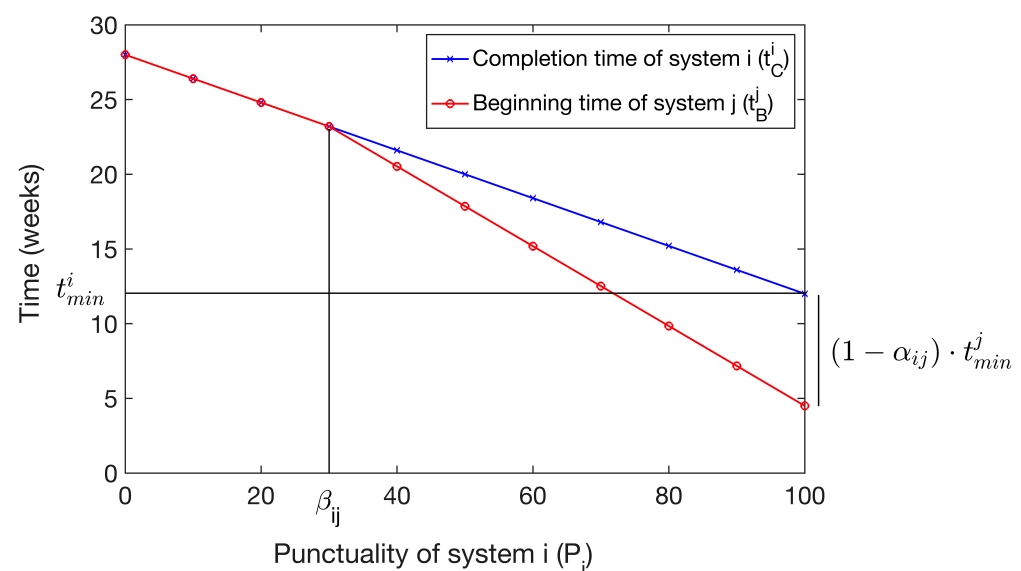


Figure 3. Completion time of system i and beginning time of system j as a function of the parameters of the developmental dependency between the two systems ($\alpha_{ij} = 0.25$, $\beta_{ij} = 30$). The red line shows how, due to the partial dependency modeled in SDDA, the successor system j can start its development before the predecessor system i is complete. The vertical gap between the red line (beginning time of system j) and the blue line (completion time of system i) is the lead time. This decreases with the decreasing punctuality of i until the critical threshold is reached, and the lead time decreases to zero.

It must be noted that, for projects with a high number of tasks, the upfront modeling effort can become considerable, regardless of the option chosen to assess the modeling parameters. While this shapes some of the future research directions, it is fundamental that an appropriate amount of care is exercised when balancing and trading off the amount of modeling effort and the amount of information and decision support that the model can provide. In particular, we suggest the following potential strategies (which are beyond the scope of this publication) to overcome the curse of dimensionality in the setup phase of the SDDA methodology:

1. Establish and utilize databases of typical one-to-one development dependencies, especially when modeling multiple projects in the same field. This approach is also naturally suitable for use with Machine Learning techniques to extract information about developmental dependencies in past projects.
2. Use methodologies to cluster the tasks and organize them hierarchically, adding complexity only where more detail is necessary, as suggested by Simon [70] in the analysis of Nearly Decomposable Systems (NDS).
3. When data are non-existent, begin by using standard forms of low, medium, and high levels of interdependency parameters, and perform a sensitivity analysis to identify first-order criticalities in the development network.

3.2.1. Strength of Dependency

The Strength of Dependency (SOD), indicated as α_{ij} , ranges between 0 and 1 and evaluates the fraction of development time of system j that is dependent on inputs by a predecessor system i . As indicated in Figure 3, partial dependencies allow for a system to begin its development before the development of its predecessor is complete. If the input system i has punctuality P_i equal to 100, it is completed in its shortest development time. In this case, the amount of lead time of the successor j is calculated as the product of its own minimum development time and a factor equal to 1 minus the Strength of Dependency between i and j . This represents the assumption that, as the predecessor i progresses through its development, system j can simultaneously complete the portion of its development that does not rely on input from the predecessor. The value of the SOD parameter balances the risk of initiating development early against the advantage of mitigating delays through this lead time. If the predecessor experiences delays in development, this will impact the beginning time of the successor system in two ways. Firstly, the delay is directly added to the beginning time of the successor. Secondly, the lead time determined by SDDA will decrease in proportion to the reduction in the punctuality of the predecessor, reflecting its reduced reliability, or that of the associated stakeholder. Once the punctuality drops below a critical threshold, the lead time is reduced to zero and the dependency becomes *absolute*, analogous to a PERT dependency. SOD is used not only to model the actual amount of information that a following activity needs from preceding activities, but also to model dynamics between the stakeholder, stakeholder biases, and preferences.

3.2.2. Criticality of Dependency

The Criticality of Dependency (COD), indicated as β_{ij} , ranges between 0 and 100 and indicates the normalized level of punctuality P_i of the predecessor i , below which the successor j is not allowed any lead time. As mentioned before, the choice of appropriate modeling parameters of the partial dependencies is very important: in the SDDA model (as well as in projects where equivalent decisions are taken by experts), the decision to have a lead time must be taken while the predecessor is still under development and thus entails uncertainty and risk. From this point of view, the choice of Criticality of Dependency

models the amount of risk that a manager is willing to take on each dependency, and therefore directly addresses the trust that stakeholders put into one another. Independently of the Strength of Dependency, a high value of Criticality means that even a small delay in the development of the predecessor will highly decrease the amount of lead time or wipe it out completely. When the COD β_{ij} is 100, all dependencies are *absolute* and the model becomes equivalent to PERT.

3.3. Formulation of the SDDA Model

Based on the parameters described in the previous section and associated with each developmental dependency, on the minimum and maximum independent development time associated with each system, and on the punctuality variables, as well as being associated with each system, the raw output of the model contains two sets of information. These are the beginning time of the development of each system i , t_B^i , and the completion time of the development of each system i , t_C^i .

For a root node (that is, a node without any predecessor, which can therefore be developed without delays) i , the beginning time is defined as 0:

$$t_B^i = 0 \quad \text{for root nodes.} \quad (1)$$

The completion time of a root node depends only (and linearly) on its punctuality, and is calculated as follows:

$$t_C^i = t_{\min}^i + \left(1 - \frac{P_i}{100}\right) (t_{\max}^i - t_{\min}^i) \quad \text{for root nodes.} \quad (2)$$

If a system j has at least one predecessor, SDDA first computes the time required for its development t_D^j , based only on its punctuality, without accounting for the dependencies:

$$t_D^j = t_{\min}^j + \left(1 - \frac{P_j}{100}\right) (t_{\max}^j - t_{\min}^j). \quad (3)$$

SDDA then calculates the beginning time of the development of j based on each of its dependencies from an input systems i . We indicate these values to be t_B^j . If j depends only on one system i , this is the actual beginning time of the development of the successor system j .

If $P_i < \beta_{ij}$, system i has accumulated, or is expected to accumulate, a critical amount of delay (threshold shown in Figure 3). In this case, the beginning time of system j , solely based on its dependency on one system i , becomes equal to the time of completion of the predecessor system i :

$$t_B^j = t_C^i \quad \text{for } P_i < \beta_{ij}. \quad (4)$$

Otherwise, the beginning of the development time of system j solely based on its dependency on one system i is computed as a function of the SOD and COD, and of the punctuality of system i :

$$t_B^j = t_C^i - t_{\min}^j (1 - \alpha_{ij}) \frac{P_i - \beta_{ij}}{100 - \beta_{ij}} \quad \text{for } P_i \geq \beta_{ij}. \quad (5)$$

In Equation (5), the term on the right, which is subtracted from the completion time of system i , is the lead time of system j .

In this formulation of SDDA, which is the baseline SDDA, the actual beginning time t_B^j of a system j that is subject to more than one dependency on predecessor systems is calculated as the average of the beginning development times due to each depen-

dency. This formulation prevents a single input system from critically impacting the beginning time.

$$t_B^j = \frac{1}{n} \sum_{k=1}^n k t_B^j, \quad (6)$$

where n is the number of systems on which j is developmentally dependent.

At this point, analogous to the calculation of the beginning times of the development based on each of the multiple dependencies, SDDA calculates the completion times of the development of j based on each dependency from a system i . We denote these values as t_C^j and they are calculated as follows:

$$t_C^j = \max(t_B^j + t_D^j, t_C^i + \alpha_{ij} t_{\min}^j), \quad (7)$$

where $t_B^j + t_D^j$ is the sum of the beginning time and development time, which is the completion time that system j would have without accounting for the presence of multiple dependencies. However, using only this term to calculate the completion time might result in cases where a system or capability j has an earlier completion time than some system or capability i on which j is dependent. To avoid this violation, we introduce the second term in Equation (7). This term uses the SOD parameter to ensure that the completion time of each system reflects the presence of developmental dependencies: the term $t_C^i + \alpha_{ij} t_{\min}^j$ accounts for this dependency by ensuring that the completion of the development of system j cannot occur before the completion of system i in addition to an amount proportional to the amount of required input from i .

The actual completion time of system j is calculated as the maximum of the completion times given by each dependency (to avoid any violation of the constraints imposed by developmental dependencies):

$$t_C^j = \max_n^n t_C^j, \quad (8)$$

where n is the number of systems on which j is developmentally dependent.

Baseline SDDA analysis calculates the beginning and completion times for each system, thus producing a comprehensive schedule for the development of the SoS. This illustrates how partial development dependencies impact the overall development.

3.4. Conservative Formulation of SDDA

In the initial formulation of SDDA, Equation (6) shows that the beginning time of the development of a successor system j is the average of the times resulting from each dependency of this system on input systems. With this formulation, if there is a large spread among the completion times of the predecessors of j , this choice could cause too large a lead time, with a potential consequent waste of resources and/or increase in cost. To avoid this effect, a more conservative model can be used. In this formulation, called *SDDAmax*, the beginning time is the maximum (instead than the average) of the beginning times due to each dependency. In general, this results in less ability to absorb delay, in return for a reduced risk of “wrong” decisions being made in terms of too large lead times. In the conservative model, Equation (9) is used instead of Equation (6):

$$t_B^j = \max_n^n t_B^j. \quad (9)$$

3.5. Deterministic Analysis

To execute this type of analysis, SDDA evaluates a single instance of the developmental dependencies. Given a single value for the punctuality of each system and based on SOD, COD, and on the minimum and maximum independent development times, the

SDDA model computes the resulting deterministic beginning and completion times for the development of each system.

This kind of analysis is useful to quantify the impact of delays in a specific system on the overall development of the complex system or SoS. By varying the value of the punctuality of a system, the user can use SDDA to perform an analysis of the whole developmental network's sensitivity to delays in the chosen system. This is executed by assigning appropriate values to the punctuality of the system under consideration and assessing their effect on the development schedule by analyzing the corresponding beginning and completion times, which are listed in tables or shown in graphic format. Other deterministic studies can be run by adding delays in multiple systems in order to evaluate the effect of several combined delays. Using deterministic analysis, SDDA can identify which systems and technologies are the most critical under specified conditions, i.e. which nodes most strongly impact the schedule with their delays. The user can compare different architectures and different choices for the prioritization of systems and technologies (including decisions imposed by stakeholders) based on their performance in terms of development schedule and response to delays. It is important to note that, when running this kind of analysis, the results also depend on the specific output of interest: for example, a project manager might be interested in intermediate deadlines (for example, the completion time of the development of specific systems or technologies), or in assessing the minimum time required to begin the development of certain systems, or in measuring delay absorption in the case of the low reliability of one or more systems.

Figure 4 shows a Gantt chart [71] for the simple network from Figure 2, comparing results from SDDA, the conservative formulation SDDAmax, and traditional PERT when all nodes have a punctuality P_i equal to 100.

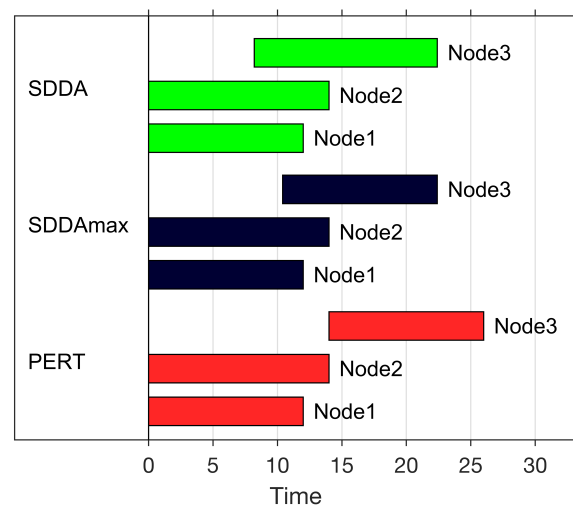


Figure 4. Gantt chart showing the schedule of the development of the simple three-node network shown in Figure 2 according to the basic SDDA, conservative SDDAmax, and PERT models.

The matrices A (whose element (i, j) corresponds to the strength of dependency α_{ij}) and B (whose element (i, j) corresponds to the criticality of dependency β_{ij}) are as follows:

$$A = \begin{bmatrix} 0 & 0 & 0.5 \\ 0 & 0 & 0.7 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 25 \\ 0 & 0 & 40 \\ 0 & 0 & 0 \end{bmatrix}.$$

The arrays of minimum and maximum independent development times (whose element i corresponds, respectively, to $t_{\min}^i$ and $t_{\max}^i$) are, in weeks:

$$t_{\min} = \begin{bmatrix} 12 \\ 14 \\ 12 \end{bmatrix} \quad t_{\max} = \begin{bmatrix} 19 \\ 18 \\ 17 \end{bmatrix}.$$

Figure 4 shows that, in PERT analysis, the third system (Node3), which is dependent on the first two (Node1 and Node2), must wait until its two predecessors are fully developed. SDDA and SDDAmax exhibit a lead time for the development of Node3. Due to the high strength and criticality of the dependency of node3 on Node2, this lead time is small. We can also notice how SDDAmax is more conservative than the basic SDDA model; therefore, the lead time is even smaller. SDDA and SDDAmax models result in the same completion time for the whole network (which is shorter than that of PERT, thanks to the partial dependencies). However, the earlier beginning time for the development of Node3 allows for better delay absorption if Node1 and Node2 exhibit delays.

3.6. Stochastic Analysis

Differently from CPM, SDDA models the partial developmental dependencies between systems, which allows for the partial absorption of delays even on the critical path. A more accurate, holistic understanding of the impact of dependencies on the development of a whole project in the presence of uncertainty, can be achieved by running a stochastic SDDA analysis. In this kind of analysis, a PDF, rather than a single value, is assigned to the punctuality of each system, and the corresponding output is a PDF for the beginning and completion times of each system, which accounts for the dependencies and the network topology. The expected value of the beginning and completion times can be used as an initial guideline for decisions, while the variance in these distributions provides insight into the reliability of the expected development times, which is tied to the risks associated with the schedule. The outputs of stochastic analysis can also be used to identify architectural patterns and features related to the whole architecture. For example, the user can utilize the expected distribution of punctuality to calculate the probability that deadlines will be met and then compare alternative architectures and their critical systems, and identify the dependencies and decisions that cause the observed criticality. Due to the low computational cost of SDDA (Table 1), Monte Carlo simulation is the best choice for performing this type of analysis, rather than calculating analytical expressions for the mixture of probability distributions. To evaluate these distributions, we generate instances of punctuality based on the given input distributions and compute the beginning and completion times with SDDA.

Table 1. Computational cost of SDDA analysis (time per single instance of developmental network, random developmental dependencies, with a total of 1000 runs. Processor Apple M1 Pro, 16 GB RAM).

Number of Nodes	Average Number of Edges	Average Time	Maximum Time
100	2474	2.248×10^{-4} s	5.408×10^{-4} s
500	62,373	0.0053 s	0.0112 s
1000	249,744	0.0210 s	0.0252 s
2000	999,492	0.0838 s	0.0877 s

To simplify the implementation of a case study, in this paper, we use the following model of uncertainty:

- Similarly to SDDA deterministic analysis, the user is required to input the expected punctuality of each system.

- The input level of punctuality will be the mode of a symmetric Beta PDF. The initial input level is chosen as the mode of the PDF, rather than the mean or median, because the algorithm might need to cut off the tails of the distribution to respect the range of feasible punctuality.
- There are three available levels of uncertainty (low, medium, or high), and the user must select one level for each system in the SoS. The Beta PDF is multiplied by a spreading factor, which models different levels of uncertainty. High uncertainty means that the assumption made on the expected punctuality is less reliable. This lower reliability results in a higher variance in the PDF in the model.
- SDDA analysis can also be run while the SoS is already being developed. The user can input the time (on the development schedule) at which to run the SDDA model and compute the expected developmental performance. As this time becomes closer to the completion time of a system, the uncertainty regarding the punctuality of this systems will decrease (that is, the variance in the punctuality PDF will be lower). When the selected time of the SDDA analysis is equal to or greater than the completion time of a system, the uncertainty regarding the punctuality of this system becomes zero.
- If the PDF that results from the user's choice of punctuality, the reliability factor, and the modified variance due to the chosen time of analysis falls partially outside the allowed range of punctuality, the function is cut to fit within the range (punctuality ranging between 0 and 100) and normalized to an area of 1.

Figure 5 shows some simple results of a stochastic SDDA analysis of the development schedule, executed on 10,000 samples. The Beta PDF of the punctuality of each system was chosen based on the model described above and used to generate the stochastic samples of punctuality. Then, these samples were used to calculate the beginning and completion times of each system according to the SDDA model. The same stochastic approach was also applied to the conservative SDDAmax and to the PERT analysis. The model parameters used for this sample problem are the same that were used in the deterministic problem of Figure 4. Regarding the variables, the punctuality of systems 1 and 3 is 100; the punctuality of system 2 is 60. System 1 has a medium level of uncertainty, system 2 has high level of uncertainty, and system 3 has low level of uncertainty.

Some of the results provide a simple validation of the method: for example, the uncertainty at the beginning and completion times is the largest at time 0, decreases at 8 weeks, and disappears at 20 weeks on the two systems that are already fully developed. Furthermore, according to the user inputs, system 2 exhibits higher uncertainty (larger spread of completion time) than system 1. The results from the original SDDA model show that early completion of development is allowed, but the Gantt chart also shows that at time 8 there is some residual risk due to uncertainty. This type of SDDA methodology results can be used to support informed decisions on the most appropriate beginning time for the development of each system based on the observed results and on the amount of accepted risk. For example, the expected value of the PDF of the beginning time can be used as the initially scheduled value. The resulting probability density functions can also be used to compute the probability that systems of interest will be fully developed by the given deadlines. The plot also shows how the more conservative SDDAmax results in a longer development time, due its lower capability of absorbing delays.

In the sample problem shown in Figure 5, the expected punctuality does not change over time. Of course, both these values and the levels of uncertainty can vary over time. The SDDA stochastic analysis can be repeated multiple times during the development of the SoS when further decisions are required. In this case, the initial results can be used to identify possible times at which it would be valuable to perform the analysis again, using updated information on the variables.

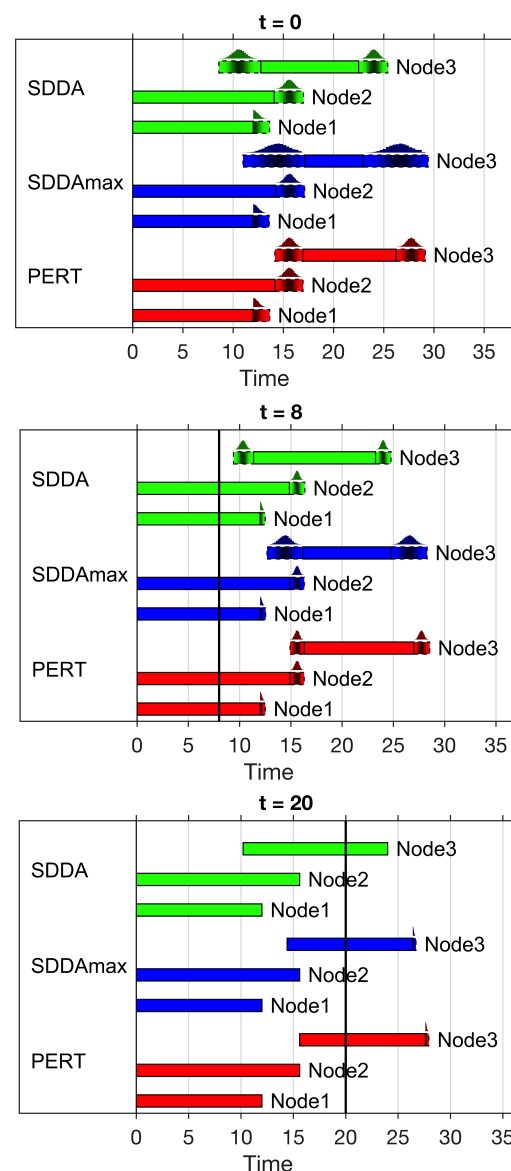


Figure 5. Gantt chart showing the schedule of development of the simple three-node network shown in Figure 2, according to the SDDA, SDDAmax, and PERT stochastic models. The resulting PDF of the beginning and completion times is shown above the corresponding bar of the Gantt chart. The darker shadows on the bars indicate zones of higher probability. **(Top)** analysis of expected schedule at time 0. **(Center)** analysis at 8 weeks. **(Bottom)** analysis at time 20 weeks.

3.7. Further Considerations for the SDDA Methodology

3.7.1. Uncertainty Models and Sensitivity Analysis

The SDDA methodology was developed with the threefold objective of directly addressing SoS characteristics, providing a simple and intuitive model of developmental interdependencies and their impact on risk and delay propagation, and having enough flexibility to be able to model stakeholder preferences, risk acceptance, and a stochastic analysis of delays. There are many sources of uncertainty in System of Systems modeling, including both stochastic uncertainty and epistemic uncertainty. Since, even when data on the past behavior of the stakeholders are known, a large amount of uncertainty can still be present, we preferred to provide a simple model of uncertainty, but one which has enough properties to be able to model different conditions. This uncertainty model can provide an adequate first-order analysis of the largest sources of risk and delay propagation and the probability of meeting the desired deadlines. The methodology was built in such a way

that more complex uncertainty models than the symmetric Beta distributions demonstrated in this work can be used. However, it is up to the user to determine the trade-off between the effort necessary to implement a more complex uncertainty method and the increased value of more detailed results. In general, we suggest performing a sensitivity analysis to identify the parameters and variables that most impact the results, and to increase the modeling efforts and evaluations of the sources of uncertainty for these parameters and variables.

3.7.2. Use of SDDA in Combination with Other Methodologies and the Scalability of SDDA

Another characteristic of the baseline SDDA model is that it can be used in combination with other methodologies. The most important caveat in this respect concerns the scalability of the model. While the computational cost of SDDA is extremely low (Table 1), the upfront modeling effort can be quite heavy. One of the best strategies in this case is to use libraries of simple and standard one-to-one developmental dependencies based on how tight the interconnection (SOD) is and the reliability of the development of the feeder, as well as the acceptability of its delays (COD). An alternative strategy is to use Machine Learning methodologies to extract information from existing data concerning the development of large-scale complex systems.

Section 4 presents the basic SDDA analyses, both deterministic and stochastic, as well as the use of SDDA for decision-making with various preferences regarding risk acceptance and preferred lead time. However, the SDDA methodology could be expanded to further types of analysis and use with other existing methodologies (as mentioned in Section 2). In particular, the analysis of resource allocation and the optimization of a schedule under budget constraints could be easily added to the initial SDDA analysis. Furthermore, SDDA can be used for analyses of the flexibility of different development schedules when facing program changes and the potential insertion of new technologies. These studies are within the scope of further publications.

4. Case Study: Lunar Gateway, a Lunar-Exploration System of Systems

4.1. Overview

This SoS case study is based on NASA's Lunar Gateway concept [72] and its related missions, as re-defined in 2022 [73] and updated in 2024 [74]. Figure 6 shows the modules that are currently planned for NASA's Lunar Gateway, as well as the contractors for some of the modules and missions. In the current plan, the initial modules of the Gateway will be the Power and Propulsion Element (PPE) and the Habitation and Logistics Outpost (HALO), which will be launched in a co-manifested launch on a SpaceX's Falcon Heavy rocket (a *Commercial Heavy Lift*) and placed in the Gateway Near-Rectilinear Halo Orbit (NRHO) with a robotic mission. The remaining modules will be carried on NASA's Space Launch System (SLS, a *Government Super Heavy Lift*), together with an Orion spacecraft. In these crewed missions, named Artemis IV–VII, astronauts will install the modules and perform lunar landings with the Starship Human Landing System (HLS) and the Blue Moon lander [75,76]. In this study, an alternative architecture is also assessed, which utilizes a larger number of commercial vehicles to assemble the Gateway before crewed missions begin.

In order to present a full SDDA study, the problem includes tasks that have already been performed or that are currently under development. The resulting nodes in the SDDA development network include Research and Development (R&D) of new technologies, the design and production of systems, crew training for specific missions, and the logistics of various planned missions. Development dependencies are evaluated based on considerations from the literature: R&D and system design must occur prior to the correspondent

system production; the production of systems, training of crews, and definition of missions follow the sequence established for the development of the Gateway.

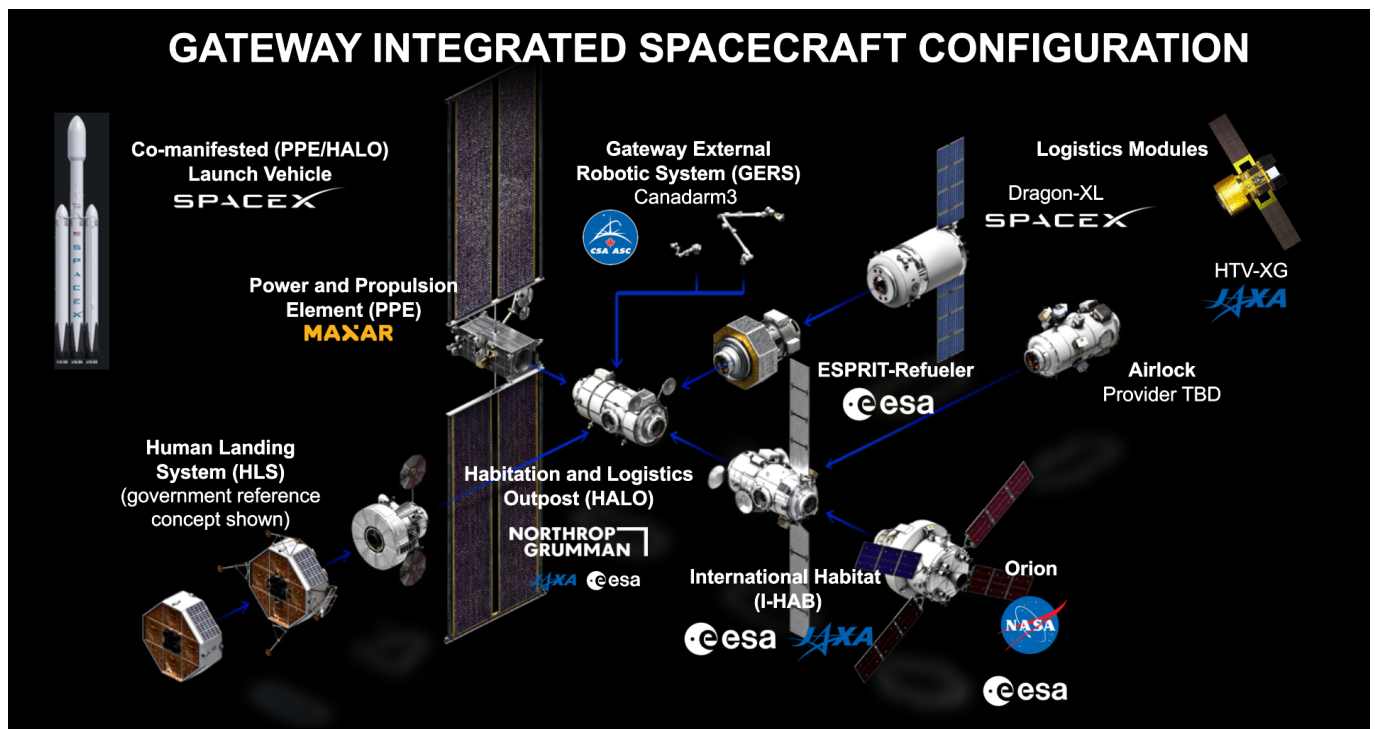


Figure 6. Infographic for the currently planned version of NASA's Lunar Gateway (credit: NASA, November 2022).

Table 2 lists the systems included in the primary architecture of the Lunar Gateway SoS case study, and Figure 7 shows the complexity of their developmental dependencies.

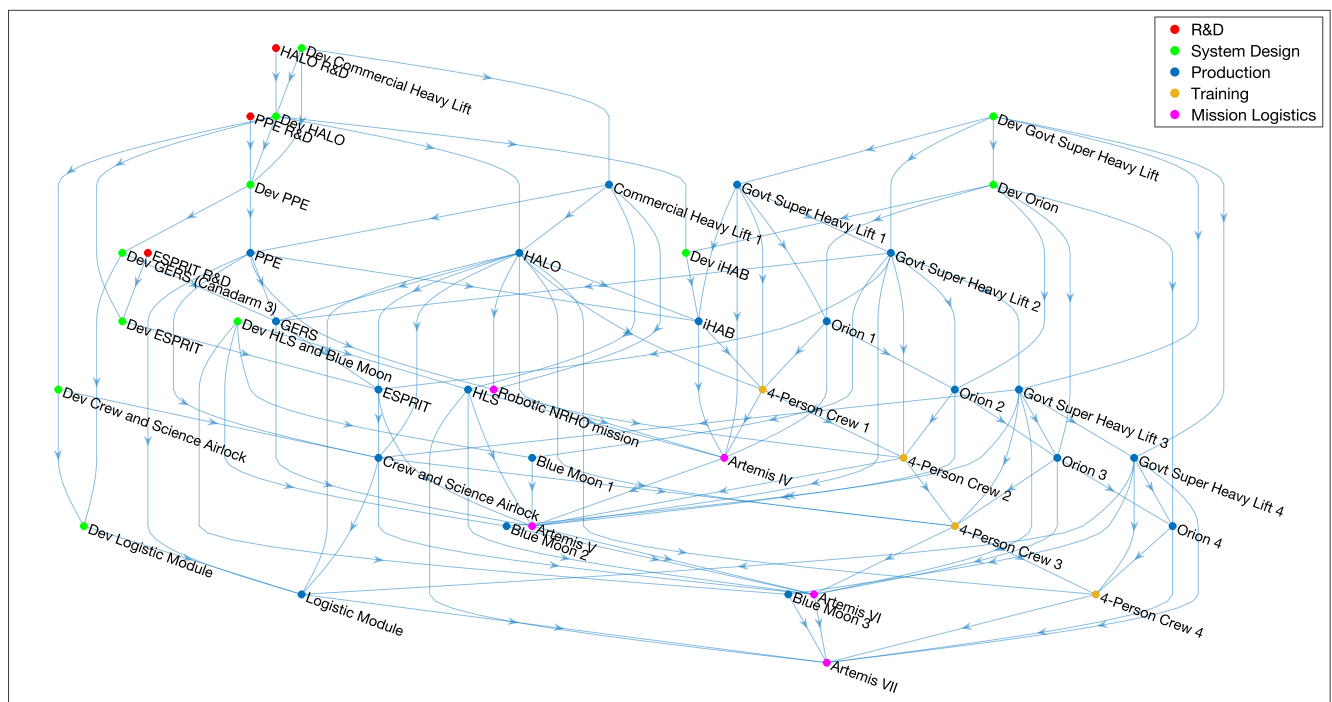


Figure 7. Developmental dependencies (SDDA network) between the systems and technologies in the primary architecture for the Lunar Gateway System of Systems.

Table 2. Systems and technologies in the Lunar Gateway System of Systems. The nodes of the development network represent research and development, system design, system production, crew training, and mission logistics.

Node No.	System/Technology	Category	Node No.	System/Technology	Category
1	PPE R&D	R&D	23	4-Person Crew 1	Training
2	HALO R&D	R&D	24	Artemis IV	Logistics
3	ESPRIT R&D	R&D	25	Govt Super Heavy Lift 2	Production
4	Dev Crew and Science Airlock	Sys design	26	Orion 2	Production
5	Dev PPE	Sys design	27	Blue Moon 1	Production
6	Dev Logistic Module	Sys design	28	GERS	Production
7	Dev Govt Super Heavy Lift	Sys design	29	ESPRIT	Production
8	Dev Commercial Heavy Lift	Sys design	30	4-Person Crew 2	Training
9	Dev Orion	Sys design	31	Artemis V	Logistics
10	Dev HALO	Sys design	32	Govt Super Heavy Lift 3	Production
11	Dev I-HAB	Sys design	33	Orion 3	Production
12	Dev ESPRIT	Sys design	34	Blue Moon 2	Production
13	Dev GERS (Canadarm 3)	Sys design	35	Crew and Science Airlock	Production
14	Dev HLS and Blue Moon	Sys design	36	4-Person Crew 3	Training
15	Commercial Heavy Lift 1	Production	37	Artemis VI	Logistics
16	PPE	Production	38	Govt Super Heavy Lift 4	Production
17	HALO	Production	39	Orion 4	Production
18	HLS	Production	40	Blue Moon 3	Production
19	Robotic NRHO mission	Logistics	41	Logistic Module	Production
20	Govt Super Heavy Lift 1	Production	42	4-Person Crew 4	Training
21	Orion 1	Production	43	Artemis VII	Logistics
22	I-HAB	Production			

The objective of this case study is to illustrate both deterministic and stochastic SDDA analyses. The case study provides examples of how to interpret the SDDA results to obtain insight into the impact of delays and the key elements that impact the propagation of schedule risks from the perspective of both technical system developers and SoS architects. Uncertainty is assigned to the various systems considering the Technology Readiness Level (TRL) and the amount of control over the stakeholder (for example, commercial partners are generally assigned more uncertainty in R&D).

4.2. Deterministic Analysis

We first performed a deterministic analysis to compute the shortest time necessary to develop the overall SoS architecture and to identify the most critical systems and dependencies in the development network when certain delays occur.

4.2.1. Nominal Scenario

Figure 8a shows a comparison of the schedule of the development of the Lunar Gateway exploration SoS, computed with the SDDA model with PERT-based dependencies. This is the nominal case, i.e., all the systems and tasks have the highest punctuality, which means that their development takes the shortest time. As expected, the SDDA model, exploiting partial dependencies, results in a schedule that allows for the early completion of the development of the SoS. However, due to the lead times, the SDDA model also shows a longer individual development time for many of the component systems. This approach

can result in an increased use of resources and can cause more risk (for example, early development reduces the flexibility of the SoS architecture should changes occur later); however, it also increases the partial recovery ability when delays occur. The nominal schedule also provides information related to the constraints of the problem: for example, in this case study, the user can determine when each of the robotic and crewed missions will be available, assuming the nominal scenario, and when each technological capability will be available. These considerations are weighed together with the criticalities and delay absorption to provide support to the project managers. The model can also be used in combination with the methods for resource allocation, which are not considered in this paper.

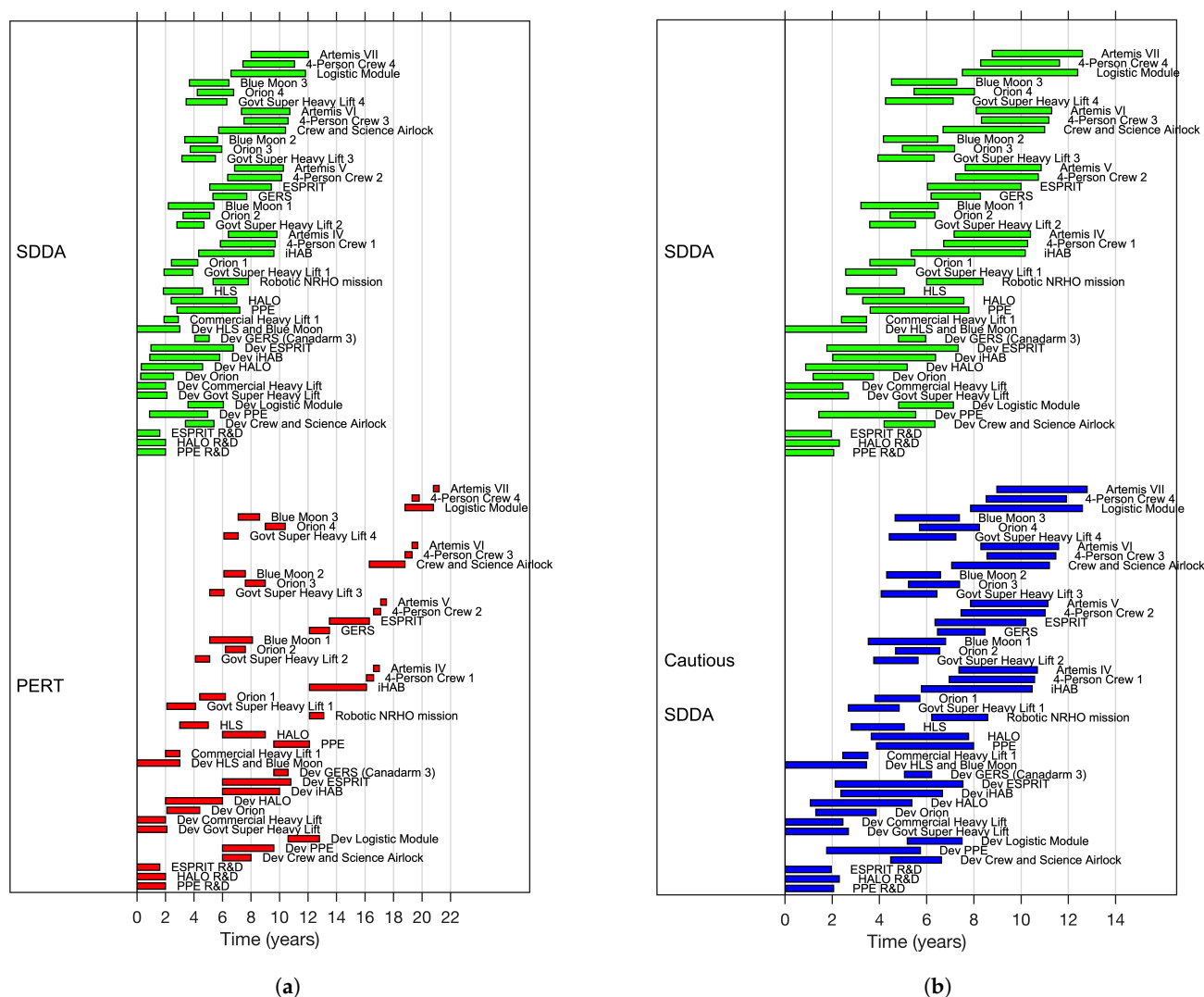


Figure 8. (a) Gantt chart of the development schedule of the Lunar Gateway primary architecture, in the nominal case (no delays). (Top) SDDA model, with partial overlap. (Bottom) PERT-based schedule. (b) Gantt chart of the development schedule of the Lunar Gateway primary architecture, when each system has Punctuality P_i equal to 80. (Top) SDDA model. (Bottom) Cautious SDDA, with higher COD (less trust when there are delays).

4.2.2. Cautious Approach

As described in Section 3.2, the COD parameter in the SDDA model is related to the amount of risk that a project manager is willing to take. If a dependency has a high criticality, a later start, with less overlap with preceding tasks, might be preferred. This conservative approach reduces the possible waste of resources, but also removes some of the advantages

of the partial parallel development of systems. Project managers can evaluate various levels of risk acceptance by varying the value of COD, and the corresponding outcome. Figure 8b presents a comparison of the schedule of the Lunar Gateway primary architecture modeled with the same parameters shown in Figure 8a (COD values between 20 and 65) and the schedule of the same architecture modeled with a more cautious approach (COD values between 40 and 90), assuming that all the systems have punctuality $P_i = 85$ (a small amount of delay). The delay is added to show the effect of the different levels of risk acceptance, which would not have any impact in the nominal case. The results show that the completion time when using the cautious model is expected to be slightly longer than the baseline case: the logistics of the Artemis VII mission are complete after 12 years and 7 months in the baseline case, and in 12 years and 10 months in the cautious approach. However, the total time allocated to the development of all the systems is 128.7 years of activity (that is, sum of the duration of each task) in the baseline case, and 127.5 years of activity in the cautious case, which resulted in less waste of resources. In general, a cautious approach will result in longer completion times and a reduced delay absorption ability, but will also reduce total activity time. This effect is even larger when there are longer delays and a large number of systems with multiple developmental dependencies.

Due to the complexity of the problem, stochastic analysis can provide more information in support of the project manager's decision to use a more or less risk-prone approach.

4.2.3. Delays and Criticalities

In the last example of deterministic analysis described in this paper, we simulate instances of decreased punctuality in selected systems to evaluate the delay recovery ability through an exploitation of partial overlap and lead times. In this way, we can also identify the most critical nodes in terms of their impact on development delays. In this type of analysis, we add a deterministic delay to one node at a time, with a punctuality equal to 50. We then compute the final delay in the development of the whole SoS and compare it to the initial delay in the affected system to obtain a measurement of the delay recovery.

Figure 9 show the amount of initial and final delay due to reduced punctuality in individual systems for both architectures using a schedule based on SDDA, conservative SDDA, and PERT. The rectangular frames indicate the initial delay accumulated in the affected system. The colored rectangles indicate the final delay in the whole schedule: an empty frame means that the delay has been fully absorbed thanks to partial development overlap and lead times; otherwise, a delay can be partially absorbed, not absorbed, or even propagate through the SDDA network to produce an even larger delay in the overall development of the SoS. Since the minimum and maximum independent development times for each system are the same in the three models, the initial delay in the affected system is also the same. However, the final delay in the whole schedule is different. SDDA and cautious SDDA approaches can absorb delays in more systems than PERT, due to the partial parallel development of interdependent systems. These models, however, reduce the lead time, and therefore the partial overlap of the development of systems when delays occur. If the delays affect systems that have critical developmental dependencies, the entire schedule can be heavily affected (even more so in the cautious SDDA approach, due to the lower initial lead times). However, the development time of the whole architecture according to SDDA models is still shorter than the PERT-based schedule. In the primary architecture, the most critical systems are the HALO R&D and system design (due to its centrality in the Gateway design and its relative novelty), the development of the Commercial Heavy Lift, which is the first launcher to be used in this Lunar Gateway architecture, and the R&D of the innovative European System Providing Refueling, Infrastructure, and Telecommunications

(ESPRIT) module. The delay on the HALO R&D fully impacts the development of the whole architecture; the remaining three of these four initial delays are partially absorbed. Delays in every other system can be fully absorbed. The cautious SDDA model shows a lower partial delay absorption ability, and cannot fully absorb an initial delay in the design of the Government Super Heavy Lift. In the PERT-based schedule, the ESPRIT R&D results are not critical, but 10 systems and technologies out of the 43 are on the critical path, meaning that initial delays in any of these systems fully impact the final development time. Tasks that were not critical in the SDDA model but are critical in PERT are the production of the PPE, Gateway External Robotic System (GERS, Canadarm3), ESPRIT, Crew and Science Airlock, and Logistic Module and the Artemis VII mission logistics.

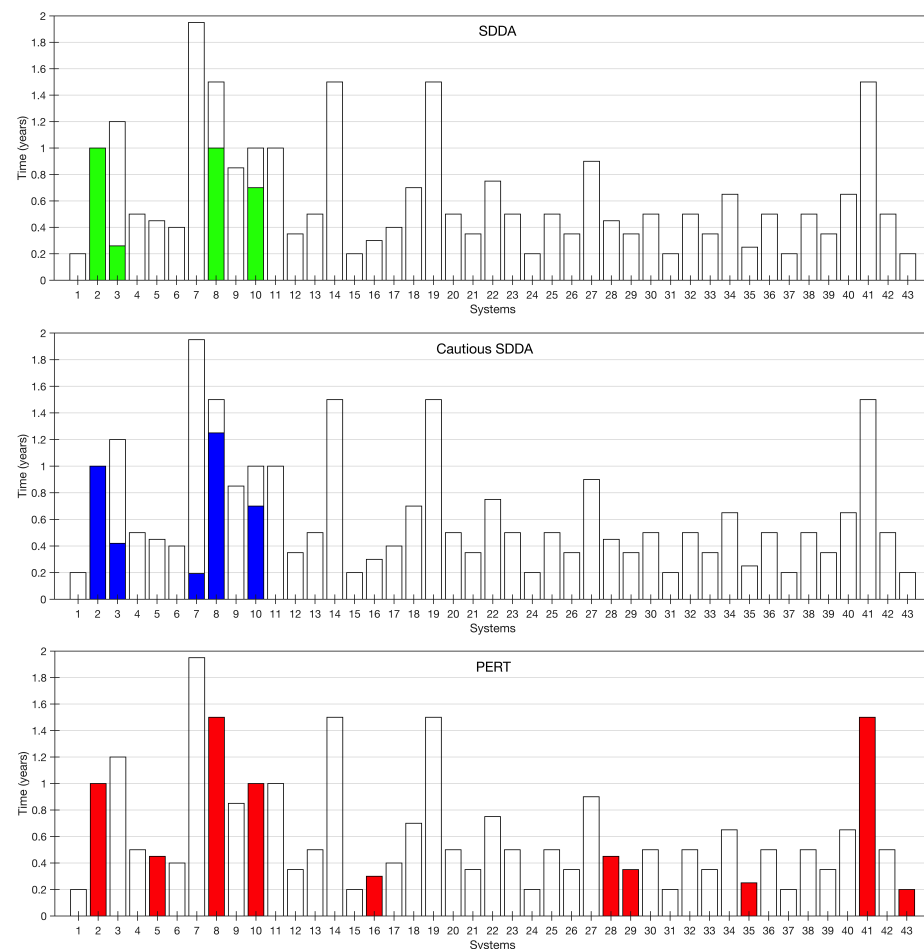


Figure 9. Impact of developmental delay in a single system on the whole schedule of the primary Lunar Gateway architecture. System numbering from Table 2. Empty bars represent the initial delay in the development of the affected system. Colored bars represent the final delay in the entire schedule with respect to the nominal case. If the frame is empty, the initial delay is completely absorbed. SDDA and conservative SDDA approaches exhibit a higher amount of partial and total delay recovery with respect to PERT, due to the partial parallel development of interdependent systems. However, for the same reason, delays in critical systems have a large impact on the whole schedule. In PERT model, more systems show unabsorbed delays.

Besides comparing more and less cautious approaches, we also show how to evaluate different architectures. In this study, we assess an alternative architecture, which is characterized by the increased use of commercial launchers. In particular, three robotic assembly missions with commercial vehicles are planned in order to assemble the Lunar Gateway modules before three crewed Artemis missions with lunar landings. SDDA and cautious SDDA models for the alternative architecture (Figure 10) show partial delays in

more systems than the primary architecture, with the Government Super Heavy Lift also causing partial delays in the SDDA model and the systems design of Orion having a small impact on the alternative architecture. However, this architecture is expected to be less expensive due to the use of more commercial heavy lift launchers. These SDDA analysis results provide insight into the schedule and delays in support of the trade-off between competing variables in program manager decisions; for example, when SDDA is used in combination with methodologies to evaluate cost and performance.

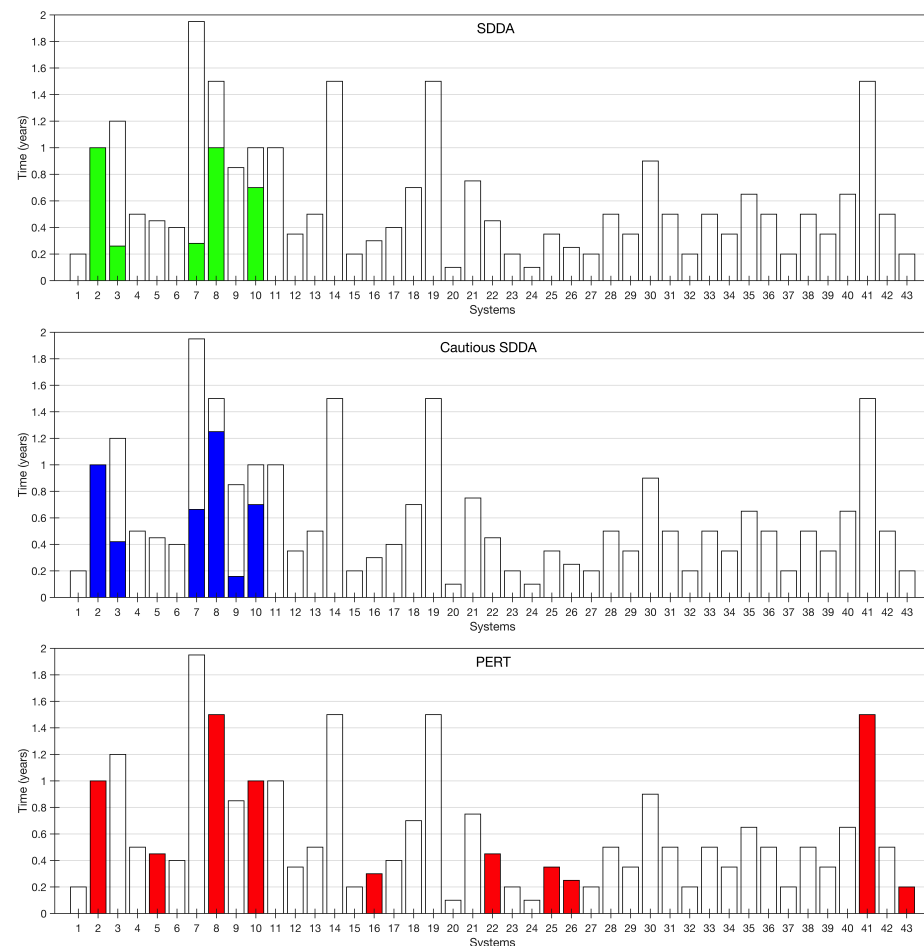


Figure 10. Impact of a developmental delay in a single system on the whole schedule of the alternative Lunar Gateway architecture. System numbering from Table A1. Empty bars represent the initial delay in the development of the affected system. Colored bars represent the final delay in the entire schedule with respect to the nominal case. If the frame is empty, the initial delay is completely absorbed. This architecture shows a reduced delay absorption ability compared to that of the primary architecture when the punctuality P_i of individual systems is equal to 50.

4.3. Stochastic Analysis

In this section, we show two examples of the use of a stochastic SDDA model for analysis and for decision support. The uncertainty levels, according to the model described in Section 3.6, are based on the reliability of the stakeholders and on existing previous work in the development of specific systems and technologies.

4.3.1. Development Schedule in Presence of Uncertainty

Figure 11 shows the GANTT charts resulting from a stochastic analysis of the primary architecture when all the systems have a baseline punctuality P_i equal to 80. The probability distributions on top of each bar show the PDF of the beginning and completion time of each task. This type of analysis provides a complete overview of the cascading impact

of delays under uncertainty and can be used in different ways. For example, in the case of space systems, the distribution of all the tasks necessary to launch a mission can be used to quantify the probability that the deadlines for launch windows will be met. The stochastic SDDA model can be used to evaluate uncertainty in the development schedule based on the information provided at different time steps. As expected, running a stochastic SDDA analysis based on information acquired later reduces the amount of uncertainty in the schedule. Figure 11a shows the uncertainty in the development of the primary architecture according to the baseline SDDA model at time $t = 0$. Besides the different levels of uncertainty in the development time of individual systems and tasks, there is also a cascading effect due to the dependencies. Later systems and systems with multiple dependencies exhibit even more uncertainty. This kind of assessment is usually left to individual expertise, rather than being treated quantitatively. Figure 11b illustrates the importance of regularly re-running the stochastic analysis. In this case, although the expected (and actual, for systems that had already completed their development) punctuality did not change, the reduced amount of uncertainty due to some systems having already been partially or fully developed is clearly visible in the much narrower PDF for the beginning and completion times of future systems. This constituted the basis for the broader decision support, which includes considerations of the initial decisions made by project managers in terms of whether there was more or less risk acceptance, and the possibility of revising the initial decisions during the development of the whole architecture.

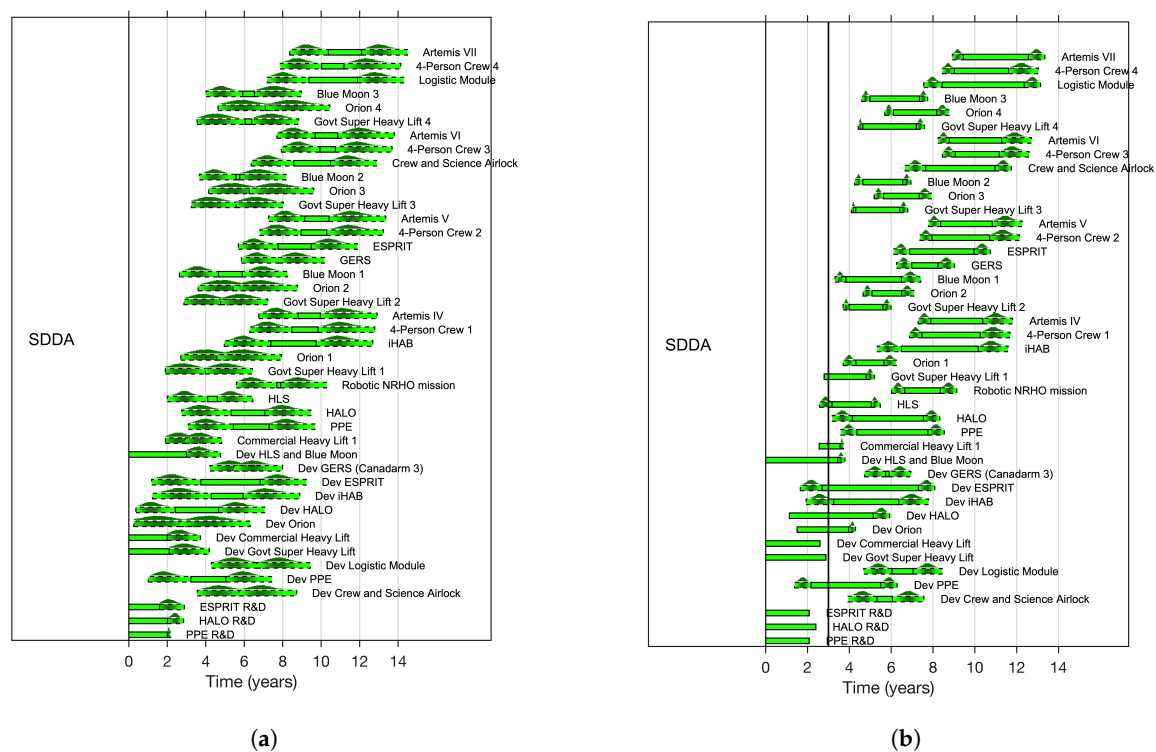


Figure 11. Stochastic GANTT chart representing the development schedule of the primary Lunar Gateway SoS architecture when each system has punctuality P_i equal to 80. (a) uncertainty based on values at time $t = 0$. (b) uncertainty based on values at time $t = 3$ years. While decisions have already been taken during the first three years, the uncertainty during some of the start time means that the decision is still affected by future events (interdependent tasks not yet completed). Dash line areas represent the areas of uncertainty, with the corresponding PDF shown above each area.

4.3.2. Decision Support

Table 3 demonstrates a possible use of SDDA stochastic analysis to support project management decisions. Using the primary architecture, we make the following assumptions:

Table 3. Outcome of scheduling decisions by program managers on the primary architecture with an SDDA model. The first three columns define 18 different cases, based on the initial schedule decision, punctuality of the systems, and whether it was decided to review the schedule after 3 years or not. The fourth column shows the expected completion time of the whole architecture. The fifth column shows the actual average completion time in each case, accounting for program manager decisions and punctuality. Columns six to nine list the percentage of instances which have a longer completion time and instances which have a shorter completion time with respect to a model with exact knowledge, as well as the average delay or gain with respect to the model with exact knowledge. The tenth column shows the minimum activity time required to complete all the tasks with the actual punctuality (sum of the minimum development time of each system). The eleventh column shows the actual cumulative time required to complete all the tasks.

Manager Decision for t_B^i	Actual P_i	Schedule Review at $t = 3$ Years	$E(T_C^{43})$ with $P_i = 80$	Actual $avg(t_C^{43})$	% Instances Later Than $E(t_C^{43})$ with Actual P_i	Avg. Delay	% Instances Earlier Than $E(t_C^{43})$ with Actual P_i	Avg. Gain	Min $\sum_{i=1}^{43} t_D^i$	Actual $\sum_{i=1}^{43} t_D^i$
Expected value	80	No	12.99	13.03	48.3	0.227	51.7	0.277	90.28	131.86
		Yes	12.99	13.03	46.9	0.236	53.1	0.287	90.28	131.42
	70	No	12.99	13.87	2.1	0.104	97.9	0.71	95.72	139.73
		Yes	12.99	13.87	2.2	0.11	97.8	0.734	95.72	132.4
	90	No	12.99	12.49	91.4	0.406	8.6	0.147	84.84	125.56
		Yes	12.99	12.49	90.9	0.398	9.1	0.144	84.84	130.78
10th percentile (late start)	80	No	13.92	13.03	97.7	0.77	2.3	0.124	90.28	129.98
		Yes	13.92	13.02	95.4	0.593	4.6	0.139	90.28	134.03
	70	No	13.92	13.88	60.7	0.373	39.3	0.286	95.72	136.86
		Yes	13.92	13.87	59.6	0.301	40.5	0.254	95.72	137.08
	90	No	13.92	12.49	100	1.116	0	-	84.84	124.17
		Yes	13.92	12.49	100	0.951	0	-	84.84	132.39
90th percentile (early start)	80	No	12.25	13.03	0.9	0.06	95.5	0.544	90.28	141.06
		Yes	12.25	13.03	1	0.051	94.9	0.551	90.28	131.13
	70	No	12.25	13.86	0	-	100	1.15	95.72	150.86
		Yes	12.25	13.87	0	-	99.9	1.166	95.72	133.39
	90	No	12.25	12.49	17.8	0.086	63.8	0.264	84.84	132.92
		Yes	12.25	12.49	17.0	0.087	65.1	0.273	84.84	129.96

- The initial estimate of punctuality was 80 for each system. With this estimate and the systems uncertainty, the user can generate probability distributions of the beginning and completion times of each system.
- We tested three different initial stakeholder decisions for the beginning time of each dependent system: the expected value of the beginning time, resulting from the initial estimate (meaning that the project manager trusts the estimate), the 10th percentile (meaning that the project manager prefers a less risky choice; that is, a late start), or the 90th percentile (meaning that the project manager prefers a riskier choice; that is, an early start).
- The manager can also decide if the programming policies will keep the beginning times defined initially, even when more information about the actual punctuality of the systems becomes available, or if the schedule will be reviewed after 3 years and the beginning times will follow the new estimate.
- The actual punctuality of the system can be 80 (as estimated), 70, or 90. If the schedule is reviewed, information about the actual punctuality will be available at the time of the review.

- The minimum sum of the development time of each system is what would result from a PERT model without overlaps. The SDDA model accounts for partial dependencies, using a lead time that might cause some tasks to take a longer time due to delays in other tasks. The actual sum of the development time of each system is used as an indicator of the use of resources, which, in turn, will impact cost.

Based on each combination of scheduling policies and the actual punctuality of the systems, we compute the expected completion time resulting from the initial estimate and the actual average completion time resulting from the different choices made in terms of scheduling policies, simulating ten thousand instances of each combination. We then compare the actual completion time in each case to the completion time that would result if the full information were obtained in order to determine the percentage of cases in which a policy results in a longer completion time or a shorter completion time. We also compute the average delay in the cases where the actual completion time is higher and the average gain in the cases where the actual completion time is shorter. Finally, we compute the minimum and actual sum of the development times (activity time), which suggests whether a policy is consuming too many resources in terms of time. While, in general, it is known *a priori* that risk-averse policies will likely accumulate more delay but result in less waste of resources, and that riskier policies will better absorb delays at the cost of a longer activity time, this sensitivity analysis of the specific problem is useful to identify potentially bad decisions, especially in cases where the initial estimate is incorrect.

The results yield many interesting points to support management decision-making. Some of them include the following:

- The decision to use the expected value for the beginning times results in a completion time close to that resulting from the model with exact information.
- The decision to use a late start causes delays with respect to the model with exact information. However, a late start without review also has the shortest sum total in terms of development time. This choice could be appropriate when a review of the schedule is not possible and the most important objective is to reduce the use of time resources.
- The decision to make an early start yields the shortest completion time and the shortest delays. However, this comes at cost of longer waits during task execution. An early start with a schedule review offers a good trade-off between the completion time and use of resources.
- Reviewing the schedule when the actual punctuality is higher than the initial estimate reduces the final average delay, but at cost of a much longer total development time. This policy is suggested if the completion time is more important than the use of resources.
- Reviewing the schedule when the actual punctuality is lower than the initial estimate reduces the final gain or the percentage of early finishes, but also reduces the total development time. This policy is suggested if the use of resources is more important than the completion time.
- Reviewing the schedule when the actual punctuality follows the initial estimate produces varied results, depending on the initial policy. The revision reduces the total development time (but increases the percentage of late instances) if the initial decision was an early start. It increases the total development time (but also decreases the percentage of late instances) if the initial decision was a late start.

Table 4 lists the results of the same analysis for the alternative architecture. This architecture shows trends similar to those observed for the primary architecture. However, the decisions have a larger impact in the alternative architecture: a late start can cause delays of almost two years compared to the case with exact knowledge, as well as a 10% decrease

in the use of time resources. A policy of early start can be very effective to counteract unexpected reductions in the punctuality of the tasks; however, even if the schedule is reviewed, this policy can cause a large increase in the use of resources. This example of the use of the SDDA for the analysis of project management decisions provides insight into the ability of the tool to provide quantitative information about the expected outcome of different policies regarding the execution of tasks in the developmental domain. It also suggests its potential use in combination with other methodologies; for example, resource allocation tools, analyses of TRL and associated risks, and the probability of delays.

Table 4. Outcome of scheduling decisions by program managers on the secondary architecture with the SDDA model. Columns defined as in Table 2.

Manager Decision for t_B^i	Actual P_i	Schedule Review at $t = 3$ Years	$E(T_C^{43})$ with $P_i = 80$	Actual $avg(t_C^{43})$	% Instances Later Than $E(t_C^{43})$ with Actual P_i	Avg. Delay	% Instances Earlier Than $E(t_C^{43})$ with Actual P_i	Avg. Gain	Min $\sum_{i=1}^{43} t_D^i$	Actual $\sum_{i=1}^{43} t_D^i$
Expected value	80	No	13.15	13.34	39.6	0.262	60.4	0.358	89.80	130.73
		Yes	13.15	13.34	40.5	0.275	59.5	0.375	89.80	130.41
	70	No	13.15	14.52	0.8	0.139	99.2	1.104	95.05	138.31
		Yes	13.15	14.52	1.6	0.1590	98.4	1.096	95.05	131.09
	90	No	13.15	12.59	93.1	0.511	6.9	0.150	84.55	124.77
		Yes	13.15	12.59	92.7	0.514	7.3	0.158	84.55	130.09
10th percentile (late start)	80	No	14.73	13.34	99.7	1.222	0.3	0.089	89.80	129.87
		Yes	14.73	13.35	96.7	0.668	3.3	0.139	89.80	133.89
	70	No	14.73	14.53	67.0	0.516	33.0	0.350	95.05	136.50
		Yes	14.73	14.52	66.6	0.379	33.4	0.257	95.05	136.85
	90	No	14.73	12.59	100	1.779	0	-	84.55	124.29
		Yes	14.73	12.58	100	1.088	0	-	84.55	132.92
90th percentile (early start)	80	No	12.25	13.34	0.4	0.06	99.2	0.787	89.80	139.86
		Yes	12.25	13.35	0.4	0.710	99.3	0.806	89.80	130.45
	70	No	12.25	14.52	0	-	100	1.771	95.05	149.29
		Yes	12.25	14.52	0	-	100	1.741	95.05	132.63
	90	No	12.25	12.59	21.5	0.110	71.1	0.307	84.55	132.27
		Yes	12.25	12.59	19.3	0.102	73.2	0.317	84.55	129.58

4.4. Sensitivity Analysis

Given the high level of uncertainty embedded in the System of Systems due to the complexity of the interdependencies, the presence of multiple stakeholders, and the difficulty of modeling future trends, and in the absence of established libraries of developmental dependency models, it is appropriate to perform a sensitivity analysis. This will provide information on the variability of the models in response to uncertainty in their parameters and variables. This is particularly important when approaches are taken to deal with the complexity in large models (as indicated at the end of Section 3.2), which might reduce the accuracy of the model in favor of a reduced setup effort. A sensitivity analysis could drive the decision to improve the level of detail and the accuracy of parts of the model.

4.4.1. Setup

The case study of the Lunar Gateway primary architecture has 43 systems and technologies and 115 developmental dependencies, as shown in Figure 7. This results in 230 model parameters (115 values of SOD and 115 values of COD), and 43 variables (the Punctuality P_i of each system). Due to the large number of parameters and variables, we

implemented a Monte Carlo analysis of correlation with a sample size of 100,000. Although a single SDDA dependency is a piecewise linear model, the combination of many dependencies can be far from linear. Therefore, we calculated both Pearson's linear correlation coefficients and Kendall's τ . Since Pearson's coefficients are more sensitive to outliers, we present the results of rgw sensitivity analysis with Pearson's coefficient. We implemented various types of sensitivity analysis, one for each combination illustrated in Table 5.

Table 5. Combinations of uncertain model parameters/variables and outputs tested in the sensitivity analysis.

Test No.	Uncertain Parameters/Variables	Model Outputs
1	SOD ($\pm 20\%$) and COD ($\pm 20\%$)	max completion time $\max_i(t_C^i)$
2	SOD ($\pm 20\%$), COD ($\pm 20\%$), and P_i ($\pm 30\%$)	max completion time $\max_i(t_C^i)$
3	SOD ($\pm 20\%$) and COD ($\pm 20\%$)	total development time $\sum_{i=1}^n (t_C^i - t_B^i)$
4	SOD ($\pm 20\%$), COD ($\pm 20\%$), and P_i ($\pm 30\%$)	total development time $\sum_{i=1}^n (t_C^i - t_B^i)$

4.4.2. Sensitivity of Maximum Completion Time

We ran two scenarios to evaluate the sensitivity of the maximum completion time $\max_i t_C^i$. The longest time to run the sensitivity analysis (second scenario) was 2.11 s. In the first scenario, the 100,000 samples for Monte Carlo simulation used a fixed punctuality $P_i = 70$ for all systems and technologies, while the 115 SOD parameters and 115 COD parameters of the Lunar Gateway primary architecture had a 20% uncertainty around the baseline value.

Figure 12 shows the results of the Monte Carlo simulation and study of Pearson's correlation coefficients for the first scenario. The histogram in Figure 12 shows the distribution of the resulting maximum completion time $\max_i t_C^i$ due to the uncertainty in all 230 SDDA model parameters. The model appears robust to uncertainty, with one standard deviation around the expected value (which is $\max_i t_C^i = 13.987$ years) equal to 0.466 years. Figure 12b shows a tornado diagram of the Pearson's correlation coefficients for the 20 most impactful parameters. From this diagram, we can draw the following conclusions regarding the sensitivity of the maximum completion time to the uncertainty of the model parameters:

- As expected, all correlation coefficients were positive: in general, larger values of SOD and COD (stronger developmental dependencies, more similar to PERT/CPM) will result in a longer completion time.
- The results show a relatively low correlation with individual parameters, with the largest correlation coefficient being 0.51 and the second largest being 0.326. This is due to both the large number of parameters and the same topological properties that provide this developmental network with the ability to absorb delay. In this case, the effect of potential mistakes on the evaluation of model parameters is attenuated by the topology. Nonetheless, it is important to gather as much information as possible to guarantee that the most impactful parameters are assessed correctly.
- Most of the sensitivity concerns parameters related to the interdependent R&D and development of systems. An advantage of this feature is that these activities are completed early in the development of the whole architecture, so the schedule can be reassessed and updated as necessary, without having a large impact on the completion time.

In the second scenario, the 100,000 samples used for Monte Carlo simulation are generated using a 20% uncertainty around the baseline values of the 115 SOD parameters and the

115 COD parameters, and a 30% uncertainty around the baseline value of punctuality, for all systems and technologies.

Figure 13 shows the results of the Monte Carlo simulation and the study of Pearson's correlation coefficients for the second scenario. As expected, the histogram of the resulting maximum completion time $\max_i t_C^i$ shown in Figure 13a shows larger uncertainty, with one standard deviation of 0.858 years. The tornado diagram of the Pearson's correlation coefficients for the 20 most impactful parameters and variables, shown in Figure 13b provides some similar conclusions to scenario 1 and a few differences regarding the sensitivity of the maximum completion time on the uncertainty of the model parameters and punctuality variables.

- This time, the correlation coefficients relative to uncertainty in the punctuality variables are negative. This is compatible with the intuitive statement that a higher punctuality will result in a shorter completion time.
- The absolute value of the correlation coefficients is even lower than in scenario 1. In general, the sensitivity to variations in punctuality is stronger than the sensitivity to model parameters. This is due to the fact that punctuality has a direct effect on the delays in systems development, while the model parameters have only an indirect effect, due to the propagation of delay along the developmental network.
- Even in this case, most of the sensitivity concerns parameters and levels of punctuality related to interdependent R&D and the early development of systems, resulting in the same considerations described for scenario 1.

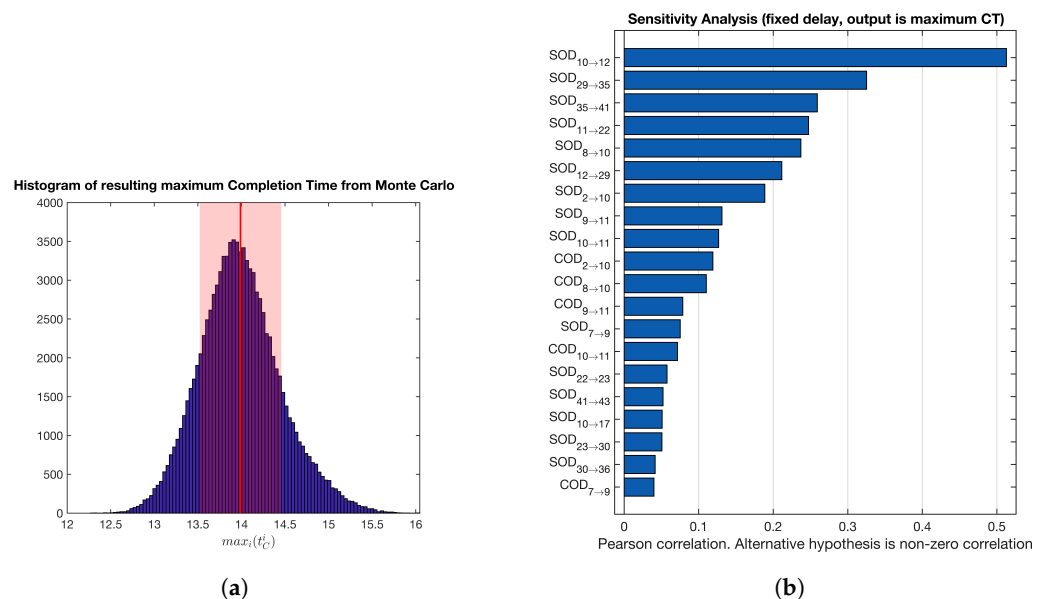


Figure 12. Sensitivity of maximum completion time to the model parameters. **(a)** Histogram of the resulting maximum completion time from Monte Carlo simulation. The shaded area is the 1σ standard deviation from the expected value. **(b)** Tornado diagram of Pearson's correlation coefficients for the 20 parameters with which the maximum completion time exhibited the highest correlation. System numbers from Table 2.

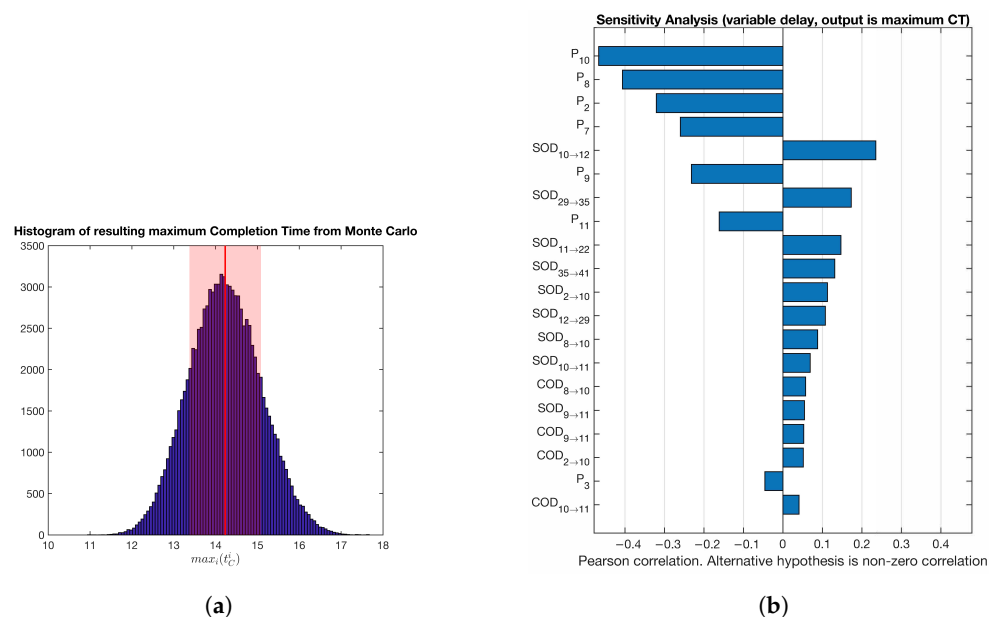


Figure 13. Sensitivity of maximum completion time to the model parameters and the punctuality variables. (a) Histogram of the resulting maximum completion time from Monte Carlo simulation. The shaded area is the 1σ standard deviation from the expected value. (b) Tornado diagram of Pearson's correlation coefficients for the 20 variables and parameters with which the maximum completion time exhibit the highest correlation. System numbers from Table 2.

4.4.3. Sensitivity of Total Development Time

We subsequently ran two more scenarios to evaluate the sensitivity of the total development time; that is, the sum of the time between beginning and completion for each activity, $\sum_i (t_C^i - t_B^i)$. The longest time needed to run the sensitivity analysis (second scenario) was 1.85 s. In the third scenario, similarly to the analysis run in the first scenario for the sensitivity of maximum completion time, we used a fixed punctuality $P_i = 70$ for all systems and technologies, while the SOD and COD parameters had a 20% uncertainty around the baseline value. The results of this analysis are shown in Figure 14.

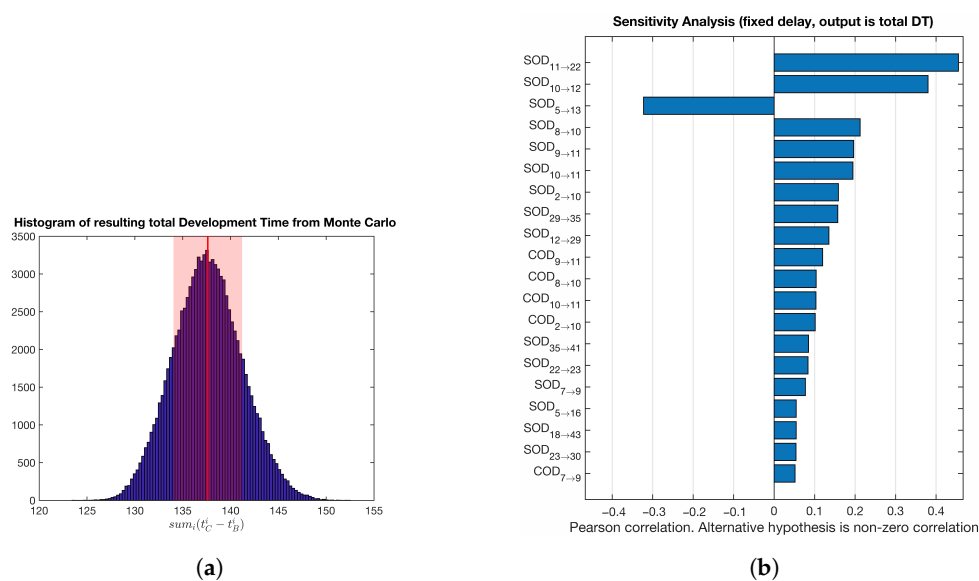


Figure 14. Sensitivity of total development time to the model parameters. (a) Histogram of the resulting total development time from Monte Carlo simulation. The shaded area is the 1σ standard deviation from the expected value. (b) Tornado diagram of Pearson's correlation coefficients for the 20 parameters with which the total development time exhibits the highest correlation. System numbers from Table 2.

The fourth scenario evaluates the sensitivity of the total development time to uncertainty in both the model parameters and the punctuality variables, similarly to the analysis run in the second scenario for the sensitivity of maximum completion time. The results of the analysis of the fourth scenario are shown in Figure 15.

The histograms of the resulting total development time $\sum_i(t_C^i - t_B^i)$ under uncertainty in the model parameters and punctuality variables shows that even this output is quite robust to uncertainty. Figure 14a shows that the value of one standard deviation in the third scenario is 3.6 years, with an expected value of around 137.64 years for the sum of all development times. When the uncertainty in the punctuality is also added, the value of one standard deviation grows to 6.44 years, as shown in Figure 15a.

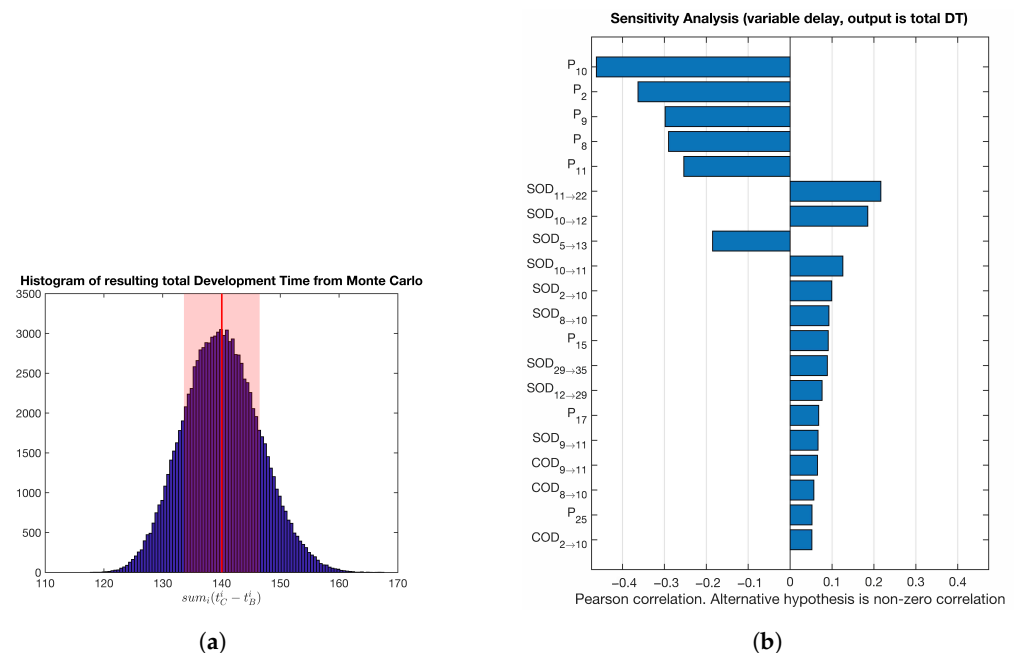


Figure 15. Sensitivity of the total development time to the model parameters and the punctuality variables. (a) Histogram of the total development time resulting from the Monte Carlo simulation. The shaded area is the 1σ standard deviation from the expected value. (b) Tornado diagram of Pearson's correlation coefficients for the 20 variables and parameters with which the total development time exhibits the highest correlation. System numbers from Table 2.

The tornado diagrams of the Pearson's correlation coefficients for the parameters and variables that have the highest impact on the total development time are shown in Figure 14b for scenario 3 and in Figure 15b for scenario 4. While the conclusions drawn from this kind of analysis are similar to what was stated for scenarios 1 and 2; these diagrams show two important differences:

- The parameter $SOD_{5 \rightarrow 13}$, showing the Strength of Dependency between the development of the PPE and the development of Canadarm 3, has a small negative correlation with the total development time. This means that a stronger dependency, i.e., a smaller overlap between these two activities, results in a shorter total development time (thus avoiding some wasteful overlap, without a strong impact on the final completion time).
- The punctuality variables with the most impact on the total time of developmental activities pertain to the HALO, the first commercial heavy lift rocket and the second government superheavy lift rocket.

Finally, Figure 16 highlights the systems whose punctuality has the highest correlation with the outputs of the sensitivity analysis and the developmental dependencies whose parameters have the highest correlation with the outputs of the sensitivity analysis. The

light orange represents a high correlation with the maximum completion time, purple represents a high correlation with the total development time, and red represents a high correlation with both outputs. This graphical representation shows some “paths” of model parameters that result in the relatively high sensitivity of the results.

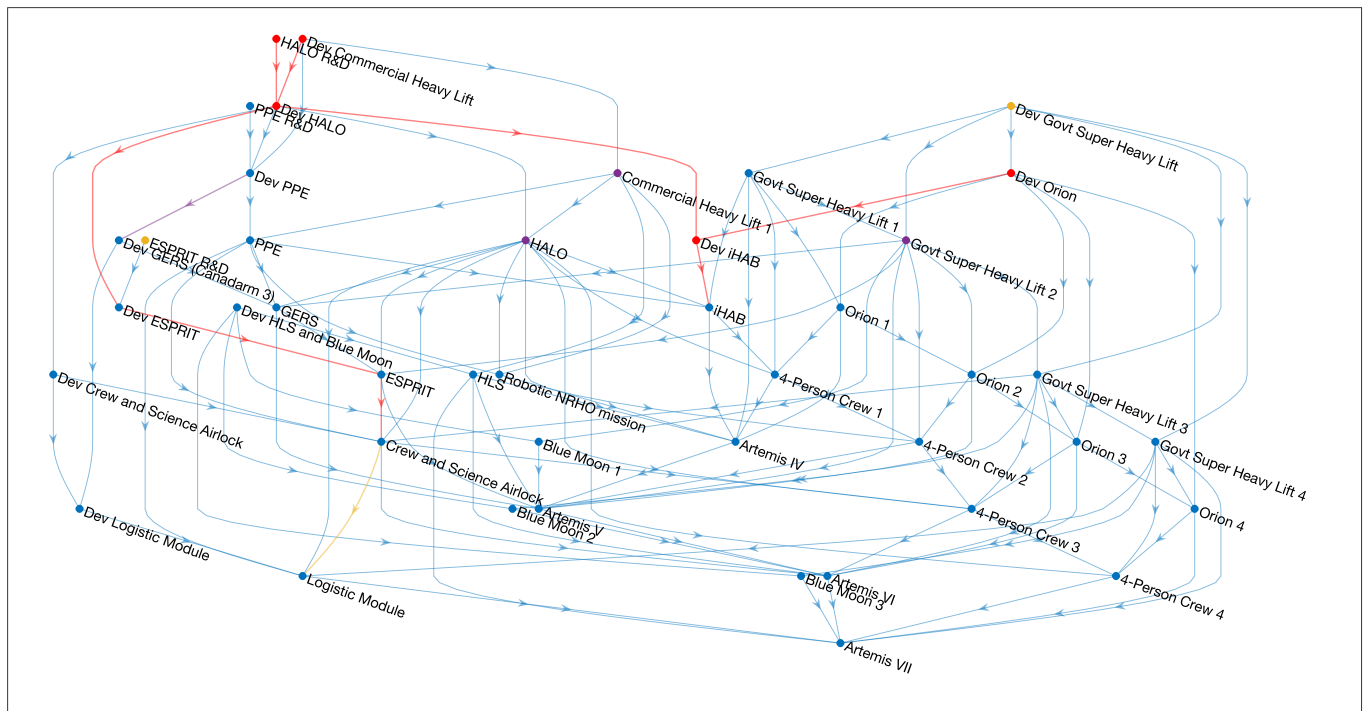


Figure 16. Graphic representation of the parameters and variables with a high correlation with the maximum completion time and total development time in the primary architecture for the Lunar Gateway System of Systems. Light orange represents a high correlation with the maximum completion time, purple represents a high correlation with the total development time, and red represents a high correlation with both outputs.

5. Conclusions, Contributions, and Future Work

Building upon the concepts of a parametric model of dependencies proposed by Garvey and Pinto [61,62] and by Guariniello and DeLaurentis [63] in the operational domain, the authors developed a method for analysis of the impact of the developmental dependencies between systems in an SoS. The SDDA methodology allows users to model the developmental interactions between systems, accounting for partial dependencies. This results in a simple model of complex and large systems at a relatively high level of abstraction based on a small number of input parameters. SDDA can thus be used to quickly analyze the impact of delays along a development network, identify criticalities, compare different development architectures, support decisions, and suggest policies to allow for a trade-off between delay absorption, time of completion, and use of resources.

Compared to traditional approaches to development and scheduling, the SDDA representation provides multiple advantages and improvements:

- SDDA has a very low computational cost; therefore, it can be useful in quickly analyzing many instances of an architecture and generating policy guidelines for a complex system with multiple dependencies.
- SDDA parameters have an intuitive meaning and may be easily related to the causes of observed results when analyzing the impact of delays. Therefore, with the SDDA model, the user may not only identify the criticalities of the system, but also possible ways to improve the development strategy.

- SDDA parameters may come from a variety of sources, including expert evaluation and historical data. One possible model used to identify the parameters is based on the information that the development of a system needs to receive from another system.
- A deterministic analysis with SDDA expands the PERT model to include partial dependencies and dynamic scheduling, accounting for the current punctuality with which the systems are being developed. The lead time and partial overlap of systems development are based on the model of dependencies and on punctuality.
- Stochastic analysis supports decision-making in the developmental domain. Based on some initial estimates, including the expected values and levels of uncertainty on the punctuality of each system, the user can analyze the effect of various policies and whether they are more or less risky. Multiple cases can be analyzed, including scenarios where the actual punctuality differs from the initial estimate to assess the expected outcome of each policy in terms of times of completion, delay absorption, and use of resources.
- The SDDA model is built in a way that can immediately accommodate extensions to include considerations of cost, the allocation of resources, technology prioritization, and development flexibility.

The advantages of the proposed methodology are compared against some of its limitations, which include the following:

- There is a risk of oversimplifying the uncertainty associated with SoS when modeling the developmental dependencies (a sensitivity analysis and accurate modeling of the most impactful parameters and variables are strongly suggested). At the current stage of the research, SDDA already provides better and more informed decision support than traditional methodologies. However, with potential advancements that could simplify the setup of the model and establish a database of standard developmental dependencies, more effort can be directed to the extraction of more accurate stochastic models from existing scenarios.
- A high upfront workload is necessary to identify the parameters of the model. This can be simplified by the implementation of standard libraries of developmental dependencies or by the use of information extraction methods for the existing data.
- There is a fast-growing search space, especially when additional studies are performed on top of the fast baseline analysis. Future work will include the use of other methodologies in combination with SDDA and will identify ways to appropriately address the scalability of SDDA beyond the baseline model.

This work presents a case study of a primary and alternative architecture for NASA's Lunar Gateway exploration SoS to demonstrate the application of SDDA, both the deterministic and in the stochastic version. The results show how SDDA can be used to support project management decisions by providing a quantitative analysis of the expected completion time of tasks, as well as the outcomes of different policies in the execution of tasks. SDDA can therefore provide improvements over existing scheduling methodologies, directly addressing the characteristics of Systems of Systems, including uncertainty in development delays, stakeholder preferences regarding acceptance and development priorities, and managerial independence. Because of the domain-agnostic nature of SDDA, the method can be applied in many different fields with applications in the System of Systems domain; for example, smart grids, quantum communication technologies, air transportation, multi-domain defense scenarios, multi-stakeholder satellite constellations, and Urban Air Mobility.

Author Contributions: Conceptualization, C.G. and D.D.; methodology, C.G.; software, C.G.; validation, C.G. and D.D.; formal analysis, C.G.; investigation, C.G. and D.D.; resources, D.D.; data curation, C.G.; writing—original draft preparation, C.G.; writing—review and editing, C.G. and D.D.; visualization, C.G.; supervision, D.D.; project administration, D.D.; funding acquisition, D.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract HQ0034-13-D-0004 RT 108-155.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

COD	Criticality of Dependency
CPM	Critical Path Method
DBD	Decision-Based Design
DSM	Design Structure Matrix
ESPRIT	European System Providing Refueling, Infrastructure and Telecommunications
FDNA	Functional Dependency Network Analysis
FMEA	Failure Mode and Effects Analysis
GERS	Gateway External Robotic System
HALO	Habitation and Logistics Outpost
HLS	Human Landing System
I-HAB	International Habitation module
MAXIT	Maximum Independent Development Time
MINIT	Minimum Independent development Time
ML	Machine Learning
NASA	National Aeronautics and Space Administration
NDS	Nearly Decomposable Systems
NRHO	Near-Rectilinear Halo Orbit
PDF	Probability Density Function
PERT	Project Evaluation and Review Technique
PPE	Power and Propulsion Element
R&D	Research and Development
SDDA	Systems Developmental Dependency Analysis
SLS	Space Launch System
SME	Subject Matter Expert
SOD	Strength of Dependency
SODA	Systems Operational Dependency Analysis
SoS	System of Systems
TRL	Technology Readiness Level

Appendix A

Table A1 lists the systems in the alternative architecture of the Lunar Gateway SoS.

Table A1. Systems and technologies in the alternative architecture for the Lunar Gateway System of Systems. The nodes of the development network represent research and development, system design, system production, crew training, and mission logistics.

Node No.	System/Technology	Category	Node No.	System/Technology	Category
1	PPE R&D	R&D	23	Robotic NRHO mission 2	Logistics
2	HALO R&D	R&D	24	Commercial Heavy Lift 3	Production
3	ESPRIT R&D	R&D	25	ESPRIT	Production
4	Dev Crew and Science Airlock	Sys design	26	Crew and Science Airlock	Production
5	Dev PPE	Sys design	27	Robotic NRHO mission 3	Logistics
6	Dev Logistic Module	Sys design	28	Govt Super Heavy Lift 1	Production
7	Dev Govt Super Heavy Lift	Sys design	29	Orion 1	Production
8	Dev Commercial Heavy Lift	Sys design	30	Blue Moon 1	Production
9	Dev Orion	Sys design	31	4-Person Crew 1	Training
10	Dev HALO	Sys design	32	Artemis IV	Logistics
11	Dev I-HAB	Sys design	33	Govt Super Heavy Lift 2	Production
12	Dev ESPRIT	Sys design	34	Orion 2	Production
13	Dev GERS (Canadarm 3)	Sys design	35	Blue Moon 2	Production
14	Dev HLS and Blue Moon	Sys design	36	4-Person Crew 2	Training
15	Commercial Heavy Lift 1	Production	37	Artemis V	Logistics
16	PPE	Production	38	Govt Super Heavy Lift 3	Production
17	HALO	Production	39	Orion 3	Production
18	HLS	Production	40	Blue Moon 3	Production
19	Robotic NRHO mission 1	Logistics	41	Logistic Module	Production
20	Commercial Heavy Lift 2	Production	42	4-Person Crew 3	Training
21	I-HAB	Production	43	Artemis VI	Logistics
22	GERS	Production			

Tables A2 and A3 list the MINIT, MAXIT, and level of uncertainty for the systems respectively in the primary and alternative Lunar Gateway architecture.

Table A2. Minimum and Maximum Independent development Time, and level of uncertainty for the systems in the primary Lunar Gateway architecture.

System	MINIT	MAXIT	Unc.	System	MINIT	MAXIT	Unc.
PPE R&D	2	2.4	2	4-Person Crew 1	0.5	1.5	2
HALO R&D	2	4	2	Artemis IV	0.4	0.8	1
ESPRIT R&D	1.6	4	3	Govt Super Heavy Lift 2	1	2	1
Dev Crew and Science Airlock	2	3	2	Orion 2	1.4	2.1	1
Dev PPE	3.6	4.5	3	Blue Moon 1	3	4.8	2
Dev Logistic Module	2.2	3	1	GERS	1.4	2.3	1
Dev Govt Super Heavy Lift	2.1	6	3	ESPRIT	2.8	3.5	3
Dev Commercial Heavy Lift	2	5	3	4-Person Crew 2	0.5	1.5	2
Dev Orion	2.3	4	2	Artemis V	0.4	0.8	1

Table A2. Cont.

System	MINIT	MAXIT	Unc.	System	MINIT	MAXIT	Unc.
Dev HALO	4	6	3	Govt Super Heavy Lift 3	1	2	1
Dev iHAB	4	6	2	Orion 3	1.4	2.1	1
Dev ESPRIT	4.8	5.5	2	Blue Moon 2	1.5	2.8	1
Dev GERS (Canadarm 3)	1	2	1	Crew and Science Airlock	2.5	3	2
Dev HLS and Blue Moon	3	6	3	4-Person Crew 3	0.5	1.5	2
Commercial Heavy Lift 1	1	1.4	3	Artemis VI	0.4	0.8	1
PPE	2.5	3.1	3	Govt Super Heavy Lift 4	1	2	1
HALO	3	3.8	2	Orion 4	1.4	2.1	1
HLS	2	3.4	1	Blue Moon 3	1.5	2.8	1
Robotic NRHO mission	1	4	3	Logistic Module	2	5	3
Govt Super Heavy Lift 1	2	3	2	4-Person Crew 4	0.5	1.5	2
Orion 1	1.8	2.5	2	Artemis VII	0.4	0.8	1
iHAB	4	5.5	2				

Table A3. Minimum and Maximum Independent development Time, and level of uncertainty for the systems in the alternative Lunar Gateway architecture.

System	MINIT	MAXIT	Unc.	System	MINIT	MAXIT	Unc.
PPE R&D	2	2.4	2	Robotic NRHO mission 2	0.8	1.2	2
HALO R&D	2	4	2	Commercial Heavy Lift 3	0.8	1	1
ESPRIT R&D	1.6	4	3	ESPRIT	2.8	3.5	3
Dev Crew and Science Airlock	2	3	2	Crew and Science Airlock	2.5	3	2
Dev PPE	3.6	4.5	3	Robotic NRHO mission 3	0.8	1.2	1
Dev Logistic Module	2.2	3	1	Govt Super Heavy Lift 1	2	3	2
Dev Govt Super Heavy Lift	2.1	6	3	Orion 1	1.8	2.5	2
Dev Commercial Heavy Lift	2	5	3	Blue Moon 1	3	4.8	2
Dev Orion	2.3	4	2	4-Person Crew 1	0.5	1.5	2
Dev HALO	4	6	3	Artemis IV	0.4	0.8	1
Dev iHAB	4	6	2	Govt Super Heavy Lift 2	1	2	1
Dev ESPRIT	4.8	5.5	2	Orion 2	1.4	2.1	1
Dev GERS (Canadarm 3)	1	2	1	Blue Moon 2	1.5	2.8	1
Dev HLS and Blue Moon	3	6	3	4-Person Crew 2	0.5	1.5	2
Commercial Heavy Lift 1	1	1.4	3	Artemis V	0.4	0.8	1
PPE	2.5	3.1	3	Govt Super Heavy Lift 3	1	2	1
HALO	3	3.8	2	Orion 3	1.4	2.1	1
HLS	2	3.4	1	Blue Moon 3	1.5	2.8	1
Robotic NRHO mission 1	1	4	3	Logistic Module	2	5	3
Commercial Heavy Lift 2	0.8	1	2	4-Person Crew 3	0.5	1.5	2
iHAB	4	5.5	2	Artemis VI	0.4	0.8	1
GERS	1.4	2.3	1				

References

- Maier, M.W. Architecting principles for systems-of-systems. *Syst. Eng.* **1998**, *1*, 267–284. [\[CrossRef\]](#)
- Wiest, J.D.; Levy, F.K. *A Management Guide to PERT/CPM*; Prentice-Hall: Englewood Cliffs, NJ, USA, 1977.
- Raz, T.; Michael, E. Use and benefits of tools for project risk management. *Int. J. Proj. Manag.* **2001**, *19*, 9–17. [\[CrossRef\]](#)
- White, D.; Fortune, J. Current practice in project management—An empirical study. *Int. J. Proj. Manag.* **2002**, *20*, 1–11. [\[CrossRef\]](#)
- Ford, M. Attitudes towards Project Management Tools and Effectiveness on Today's Complex Programs. Ph.D. Thesis, Colorado Technical University, Colorado Springs, CO, USA, 2017.
- Yeazitzis, T.; Weger, K.; Mesmer, B.; Clerkin, J.; Van Bossuyt, D. Biases in Stakeholder Elicitation as a Precursor to the Systems Architecting Process. *Systems* **2023**, *11*, 499. [\[CrossRef\]](#)

7. Trif, S.R.; Curşeu, P.L.; Fodor, O.C. Power differences and dynamics in multiparty collaborative systems: A systematic literature review. *Systems* **2022**, *10*, 30. [\[CrossRef\]](#)
8. Guariniello, C.; DeLaurentis, D. Dependency analysis of system-of-systems operational and development networks. *Procedia Comput. Sci.* **2013**, *16*, 265–274. [\[CrossRef\]](#)
9. DeLaurentis, D.A.; Marais, K.; Davendralingam, N.; Han, S.Y.; Uday, P.; Fang, Z.; Guariniello, C. *Assessing the Impact of Development Disruptions and Dependencies in Analysis of Alternatives of System-Of-Systems*; Technical Report; Systems Engineering Research Center: Hoboken, NJ, USA, 2012.
10. Abbott, B.P. Littoral Combat Ship (LCS) Mission Packages: Determining the Best Mix. Ph.D. Thesis, Naval Postgraduate School, Monterey, CA, USA, 2008.
11. Fleischer, M.; Liker, J.K. *Concurrent Engineering Effectiveness: Integrating Product Development Across Organizations*; Hanser Gardner Publications: Cincinnati, OH, USA, 1997.
12. Birchler, D.; Chrisle, G.; Groo, E. Investigating Concurrency in Weapons Programs. In *Defense AT&L*; Defense Acquisition University: Fort Belvoir, VA, USA, 2010; pp. 18–21.
13. Kamrani, A.K.; Azimi, M. *Systems Engineering Tools and Methods*; CRC Press: Boca Raton, FL, USA, 2010.
14. DeLaurentis, D.; Mane, M. System development and risk propagation in systems of systems. In Proceedings of the Seventh Annual Acquisition Research Symposium, Seaside, CA, USA, 11–13 May 2010.
15. Mane, M.; DeLaurentis, D.A. Network-level metric measuring delay propagation in networks of interdependent systems. In Proceedings of the 2010 5th International Conference on System of Systems Engineering, Loughborough, UK, 22–24 June 2010; pp. 1–6.
16. Mane, M.; DeLaurentis, D.; Frazho, A. A Markov Perspective on Development Interdependencies in Networks of Systems. *J. Mech. Des.* **2011**, *133*, 101009. [\[CrossRef\]](#)
17. Fang, C.; Marle, F. A simulation-based risk network model for decision support in project risk management. *Decis. Support Syst.* **2012**, *52*, 635–644. [\[CrossRef\]](#)
18. Gaonkar, R.S.; Viswanadham, N. Analytical framework for the management of risk in supply chains. *IEEE Trans. Autom. Sci. Eng.* **2007**, *4*, 265–273. [\[CrossRef\]](#)
19. Yassine, A.; Joglekar, N.; Braha, D.; Eppinger, S.; Whitney, D. Information hiding in product development: The design churn effect. *Res. Eng. Des.* **2003**, *14*, 145–161. [\[CrossRef\]](#)
20. Silva, S.; Crispim, J. Performance measurement and management in complex environments: A system of systems approach for the public sector. *Prod. Plan. Control.* **2024**, 1–21. [\[CrossRef\]](#)
21. Cheng, Z.; Ma, Y. A network-based assessment of design dependency in product development. In Proceedings of the 2014 International Conference on Innovative Design and Manufacturing (ICIDM), Montreal, QC, Canada, 13–15 August 2014; pp. 137–142.
22. Narayanan, S.; Balasubramanian, S.; Swaminathan, J.M.; Zhang, Y. Managing uncertain tasks in technology-intensive project environments: A multi-method study of task closure and capacity management decisions. *J. Oper. Manag.* **2020**, *66*, 260–280. [\[CrossRef\]](#)
23. Browning, T.R. Applying the design structure matrix to system decomposition and integration problems: A review and new directions. *IEEE Trans. Eng. Manag.* **2001**, *48*, 292–306. [\[CrossRef\]](#)
24. Eppinger, S.; Browning, T. *Design Structure Matrix Methods and Applications*; MIT Press: Cambridge, MA, USA, 2012; p. 334.
25. White, C.; Eaton, C.; Bates, M.; Perner, D.; Mesmer, B.L. Exploring Dynamic Preferences in Systems Engineering. In Proceedings of the Conference on Systems Engineering Research, Tucson, AZ, USA, 25–27 March 2024; Springer: Berlin/Heidelberg, Germany, 2024; pp. 49–65.
26. Hazelrigg, G.A. A Framework for Decision-Based Engineering Design. *J. Mech. Des.* **1998**, *120*, 653–658. [\[CrossRef\]](#)
27. Hazelrigg, G.A. *Fundamentals of Decision Making for Engineering Design and Systems Engineering*. Niels Corp, 2012. Available online: <https://isbnsearch.org/isbn/9780984997602> (accessed on 4 March 2025).
28. Muller, G.; Giudici, H. Social Systems of Systems Thinking to Improve Decision-Making Processes Toward the Sustainable Transition. In Proceedings of the Conference on Systems Engineering Research, Tucson, AZ, USA, 25–27 March 2024; Springer: Berlin/Heidelberg, Germany, 2024; pp. 341–353.
29. Mihret, Z.; Axelsson, J.; Jee, E.; Bae, D.H. Policy-Guided Collaboration for Enhancing System of Systems Goal Achievement. In Proceedings of the 18th Annual IEEE International Systems Conference, Montreal, QC, Canada, 15–18 April 2024.
30. Elfrey, P.R.; Zacharewicz, G.; Ni, M. Smackdown: Adventures in simulation standards and interoperability. In Proceedings of the 2011 Winter Simulation Conference (WSC), Phoenix, AZ, USA, 11–14 December 2011; pp. 3958–3962.
31. Liu, P.; Wu, Y.; Li, Y.; Wu, X. An improved FMEA method based on the expert trust network for maritime transportation risk management. *Expert Syst. Appl.* **2024**, *238*, 121705. [\[CrossRef\]](#)
32. US DoD. *MIL-P-1629—Procedures for Performing a Failure Mode Effects and Criticality Analysis*; Department of Defense: Washington, DC, USA, 1949.

33. US DoD. *MIL-STD-1629A—Military Standard: Procedures for Performing a Failure Mode Effects and Criticality Analysis*; Department of Defense: Washington, DC, USA, 1980.
34. Malcolm, D.G.; Roseboom, J.H.; Clark, C.E.; Fazar, W. Application of a technique for research and development program evaluation. *Oper. Res.* **1959**, *7*, 646–669. [\[CrossRef\]](#)
35. Kelley, J.E., Jr.; Walker, M.R. Critical-path planning and scheduling. In Proceedings of the Eastern Joint IRE-AIEE-ACM Computer Conference, Boston, MA, USA, 1–3 December 1959; pp. 160–173.
36. Fulkerson, D.R. Expected critical path lengths in PERT networks. *Oper. Res.* **1962**, *10*, 808–817. [\[CrossRef\]](#)
37. Robillard, P.; Trahan, M. Expected completion time in PERT networks. *Oper. Res.* **1976**, *24*, 177–182. [\[CrossRef\]](#)
38. Cinicioglu, E.N.; Shenoy, P.P. Solving stochastic PERT networks exactly using hybrid Bayesian networks. In Proceedings of the 7th Workshop on Uncertainty Processing, Mikulov, Czech Republic, 16–20 September 2006; pp. 183–197.
39. Azaron, A.; Tavakkoli-Moghaddam, R. Multi-objective time—Cost trade-off in dynamic PERT networks using an interactive approach. *Eur. J. Oper. Res.* **2007**, *180*, 1186–1200. [\[CrossRef\]](#)
40. Hameri, A.P.; Heikkilä, J. Improving efficiency: Time-critical interfacing of project tasks. *Int. J. Proj. Manag.* **2002**, *20*, 143–153. [\[CrossRef\]](#)
41. Khamooshi, H.; Golafshani, H. EDM: Earned Duration Management, a new approach to schedule performance management and measurement. *Int. J. Proj. Manag.* **2014**, *32*, 1019–1041. [\[CrossRef\]](#)
42. Muriana, C.; Vizzini, G. Project risk management: A deterministic quantitative technique for assessment and mitigation. *Int. J. Proj. Manag.* **2017**, *35*, 320–340. [\[CrossRef\]](#)
43. Sharma, S. *Operation Research: Pert, Cpm & Cost Analysis*; Discovery Publishing House: New Delhi, India, 2006.
44. Lientz, B.; Rea, K. *Project Management for the 21st Century*; Routledge: London, UK, 2007.
45. Brown, D.E.; Marin, J.A.; Scherer, W.T. A survey of intelligent scheduling systems. In *Intelligent Scheduling Systems*; Springer: Berlin/Heidelberg, Germany, 1995; pp. 1–40.
46. Smith, S.F. Reactive scheduling systems. In *Intelligent Scheduling Systems*; Springer: Berlin/Heidelberg, Germany, 1995; pp. 155–192.
47. Zaveri, J.S.; Emdad, A.F. Intelligent scheduling systems: An artificial-intelligence-based approach. In *Manufacturing Decision Support Systems*; Springer: Berlin/Heidelberg, Germany, 1997; pp. 204–216.
48. Ambriz, R. *Dynamic Scheduling with Microsoft Office Project 2007: The Book by and for Professionals*; J. Ross Publishing, Inc.: Ft. Lauderdale, FL, USA, 2008.
49. Boehm, B.; Lane, J.A.; Koolmanojwong, S.; Turner, R.N. *The Incremental Commitment Spiral Model: Principles and Practices for Successful Systems and Software*; Addison-Wesley Professional: Boston, MA, USA, 2014.
50. Turner, R.; Lane, J.A.; Ingold, D.; Madachy, R. A Lean Approach to Improving SE Visibility in Large Operational Systems Evolution. *INCOSE Int. Symp.* **2013**, *23*, 412–427. [\[CrossRef\]](#)
51. Senses, S.; Kumral, M. Trade-off between time and cost in project planning: A simulation-based optimization approach. *Simulation* **2024**, *100*, 127–143. [\[CrossRef\]](#)
52. Satic, U.; Jacko, P.; Kirkbride, C. A simulation-based approximate dynamic programming approach to dynamic and stochastic resource-constrained multi-project scheduling problem. *Eur. J. Oper. Res.* **2024**, *315*, 454–469. [\[CrossRef\]](#)
53. Fazel Zarandi, M.H.; Sadat Asl, A.A.; Sotudian, S.; Castillo, O. A state of the art review of intelligent scheduling. *Artif. Intell. Rev.* **2020**, *53*, 501–593. [\[CrossRef\]](#)
54. Xie, L.L.; Chen, Y.; Wu, S.; Chang, R.D.; Han, Y. Knowledge extraction for solving resource-constrained project scheduling problem through decision tree. *Eng. Constr. Archit. Manag.* **2024**, *31*, 2852–2877. [\[CrossRef\]](#)
55. Krishnan, V.; Eppinger, S.D.; Whitney, D.E. A model-based framework to overlap product development activities. *Manag. Sci.* **1997**, *43*, 437–451. [\[CrossRef\]](#)
56. Erdem, F.; Bilgiç, T. Airline delay propagation: Estimation and modeling in daily operations. *J. Air Transp. Manag.* **2024**, *115*, 102548. [\[CrossRef\]](#)
57. Huang, P.; Guo, J.; Liu, S.; Corman, F. Explainable train delay propagation: A graph attention network approach. *Transp. Res. Part Logist. Transp. Rev.* **2024**, *184*, 103457. [\[CrossRef\]](#)
58. Willumsen, P.; Oehmen, J.; Stingl, V.; Geraldi, J. Value creation through project risk management. *Int. J. Proj. Manag.* **2019**, *37*, 731–749. [\[CrossRef\]](#)
59. Belvedere, V.; Cuttaia, F.; Rossi, M.; Stringhetti, L. Mapping wastes in complex projects for Lean Product Development. *Int. J. Proj. Manag.* **2019**, *37*, 410–424. [\[CrossRef\]](#)
60. Wang, L.; Kunc, M.; Bai, S.J. Realizing value from project implementation under uncertainty: An exploratory study using system dynamics. *Int. J. Proj. Manag.* **2017**, *35*, 341–352. [\[CrossRef\]](#)
61. Garvey, P.R.; Pinto, C.A. Introduction to functional dependency network analysis. In Proceedings of the MITRE Corporation and Old Dominion, Second International Symposium on Engineering Systems, Cambridge, MA, USA, 15–17 June 2009; MIT: Cambridge, MA, USA, 2009; Volume 5.

62. Pinto, C.A.; Garvey, P.R. *Advanced Risk Analysis in Engineering Enterprise Systems*; CRC Press: Boca Raton, FL, USA, 2012.
63. Guariniello, C.; DeLaurentis, D. Supporting design via the system operational dependency analysis methodology. *Res. Eng. Des.* **2017**, *28*, 53–69. [[CrossRef](#)]
64. DeLaurentis, D.A.; Moolchandani, K.; Guariniello, C. *System of Systems Modeling and Analysis*; CRC Press: Boca Raton, FL, USA, 2022.
65. Sadin, S.R.; Povinelli, F.P.; Rosen, R. The NASA technology push towards future space mission systems. In *Space and Humanity*; Elsevier: Amsterdam, The Netherlands, 1989; pp. 73–77.
66. Mankins, J.C. *Technology readiness levels. A White Paper*; Advanced Concepts Office, NASA: Washington, DC, USA, 1995.
67. Correia, E.; Garrido-Azevedo, S.; Carvalho, H. Supply Chain Sustainability: A Model to Assess the Maturity Level. *Systems* **2023**, *11*, 98. [[CrossRef](#)]
68. Costa, J.M.; Feitosa, R.M.; de Souza, C.R. Tool support for collaborative software development based on dependency analysis. In Proceedings of the 6th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2010), Chicago, IL, USA, 9–12 October 2010; pp. 1–10.
69. Schulz, A.; Kotson, M.; Meiners, C.; Meunier, T.; O’Gwynn, D.; Trepagnier, P.; Weller-Fahy, D. Active dependency mapping: A data-driven approach to mapping dependencies in distributed systems. In *Theory and Application of Reuse, Integration, and Data Science*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 169–188.
70. Simon, H.A. *The Sciences of the Artificial, Reissue of the Third Edition with a New Introduction by John Laird*; MIT Press: Cambridge, MA, USA, 2019.
71. Gantt, H.L. *Work, Wages, and Profits: Their Influence on the Cost of Living*; The Engineering Magazine: New York, NY, USA, 1911.
72. Crusan, J.; Bleacher, J.; Caram, J.; Craig, D.; Goodliff, K.; Herrmann, N.; Mahoney, E.; Smith, M. NASA’s gateway: An update on progress and plans for extending human presence to cislunar space. In Proceedings of the 2019 IEEE Aerospace Conference, Big Sky, MT, USA, 2–9 March 2019; pp. 1–19.
73. Fuller, S.; Lehnhardt, E.; Zaid, C.; Halloran, K. Gateway program status and overview. *J. Space Saf. Eng.* **2022**, *9*, 625–628. [[CrossRef](#)]
74. Lehnhardt, E.; Travis, T.; Connell, D. The Gateway Program as Part of NASA’s Plans for Human Exploration Beyond Low Earth Orbit. In Proceedings of the 2024 IEEE Aerospace Conference, Big Sky, MT, USA, 2–9 March 2024; pp. 1–6.
75. Chavers, G.; Watson-Morgan, L.; Smith, M.; Suzuki, N.; Polsgrove, T. NASA’s Human Landing System: The Strategy for the 2024 Mission and Future Sustainability. In Proceedings of the 2020 IEEE Aerospace Conference, Big Sky, MT, USA, 7–14 March 2020; pp. 1–9.
76. Watson-Morgan, L.; Chojnacki, K.T.; Gagliano, L.; Holcomb, S.V.; Means, L.; Ortega, R.; Percy, T.K.; Polsgrove, T.; Woods, D.; Vermette, J.P. NASA’s Human Landing System: A Sustaining Presence on the Moon. In Proceedings of the 74th International Aeronautical Congress (IAC), Baku, Azerbaijan, 2–6 October 2023.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.