ARCHITECTURAL SIMULATION MODEL FOR OPERATIONS OF FUTURE LAUNCH SYSTEMS

Ghaith Rabadi, Ph.D., Old Dominion University
Trina Chytka, Ph.D., NASA Langley Research Center
Doug Morris, NASA Langley Research Center
Nancy White, NASA Langley Research Center
Resit Unal, Ph.D., Old Dominion University

Abstract

A significant portion of lifecycle costs for space launch vehicles is generated during the operations phase. These costs are largely determined by decisions made early during conceptual design. Therefore, operational considerations are an important part of vehicle design and concept analysis especially early in the design phase. Given a mix of launch vehicle concepts, NASA would like to estimate the operational and support cost necessary to meet the missions' launch schedule. In this paper, we present a discrete-event simulation model to estimate the total cost, including the operational cost, to accomplish a set of missions.

Introduction

Research indicates that operations costs can account for a large percentage of the total life-cycle costs of reusable space transportation systems. These costs are largely determined by decisions made early during conceptual design. Therefore, operational considerations are an important part of vehicle design and analysis that need to be modeled and studied early in the design phase. However, this is a difficult and challenging task due to uncertainties of operations definitions, dynamic nature of the processes, lack of analytical models, and scarcity of historical data during the conceptual design phase.

In this paper, a set of future missions to transport cargo (or payloads) to space is considered. The missions and their relevant information will be referred to as the "manifest". The cargo may have different weights and destinations. There may be different types of space launch vehicles that can accomplish these missions but at different costs. The vehicles go through processing steps on the ground before launching. After launch, and depending on whether the elements are reusable or expendable, some elements come back for re-processing before they are used on another vehicle to fly new missions. NASA would like to assess the cost of executing a certain manifest using a specific set of launch vehicles to meet the missions' launch dates. To answer this question, it is necessary to have a way to estimate the cost of flying the missions when the type of vehicles as well as the level of resources required to support the vehicles are given. Once this is done, the second phase would be to develop an optimization model that uses the cost model to find the best resource levels that minimize the total cost over a defined life cycle as well as meet the missions' launch dates. An optimization approach will be necessary because as the number of missions, vehicles, and recourse levels increase, the problem becomes explosively combinatorial, and identifying optimal solutions would not be an easy task. We address in this paper the first phase only using a simulation-based model to estimate the cost of accomplishing the manifest given that we know which vehicle types are going to fly which missions as well as the level of resources necessary to process these vehicles.

Methodology

Discrete-event simulation, the approach used in this paper, is the modeling of a system as it evolves over time by a representation in which the state variables change instantaneously at separate points in time (Law and Kelton, 2000). It is used to assess the operational cost of future launch vehicle designs to meet the manifest schedule. The simulation model, hereinafter refer to as the Architectural Model, was developed using the discrete-event simulation modeling environment Arena from Rockwell Software.

Discrete-event simulation is widely used to study complex systems because of its flexibility, and ability to capture the dynamics of the system. It is also considered a stochastic modeling technique where the inputs to the model can take the form of statistical distributions, and events can be of a probabilistic nature. Traditionally, discrete-event simulation has been used with manufacturing and service applications. Nonetheless, it has been lately used with space applications. Mollaghasemi et al (2000), Cates et al (2001), and Rabadi et al (2001) have developed simulations to model the ground processes of the Space Shuttle. Steele et al (2002) presented a generic simulation model the ground processes of reusable launch vehicles assuming that all vehicles belong to the same family. Ruiz-Torres and Zapata (2000) presented an approach to predict operational characteristics of a future space transportation system. The model utilizes expert knowledge to predict the operational requirements of a vehicle concept design including the ground activities, flows, resources, and costs. The model incorporates simulation in order to include spaceport characteristics as alternative flows, processing variability, and other random events.

Our work is general enough to be applied to different families of launch systems to estimate the cost, and operation and support requirements of new launch system architectures at high modeling level.

Before discussing the simulation, we first define a framework based on which the simulation model was developed. Other engineering management tools can be developed based in the same framework.

The Architectural Model Framework. Before implementing the simulation model, it is necessary to design a framework to determine the modeling level of detail and capability. This framework includes the following components:

1. *Launch Systems Structure***:** The following structure definition is used in the development of the model:

- Element: is the basic building block for a Launch Vehicle Configuration (i.e., Orbiter, Booster, Cargo Carrier, etc.).
- Configuration: is a unique combination of elements from a Launch Family to form a vehicle capable of performing a specific mission.
- Family: A Launch Vehicle and all of its variants, where each variant is a configuration.
- Architecture: an aggregation of individual Launch Families that represent the total launch capability available to support a manifest.

To put these definitions in perspective, one can take the Space Shuttle as an example, where it includes three *element* types: an Orbiter, External Tank (ET) and a Solid Rocket Booster (SRB). The integration of an Orbiter, an ET and two SRBs results in a *Configuration*. Another configuration would emerge if for example an Expendable Cargo Carrier element were added to the first configuration. Both configurations, however, belong to the same family – the Shuttle Family. If a future launch vehicle (i.e., a new configuration) consists of only two elements, say an Orbiter and a Booster, but from a technology that is different from the Shuttle Family, then these elements will belong to another family. The collection of families that can accomplish a manifest represents an *Architecture*.

Within this framework, it is important to note the following design requirements:

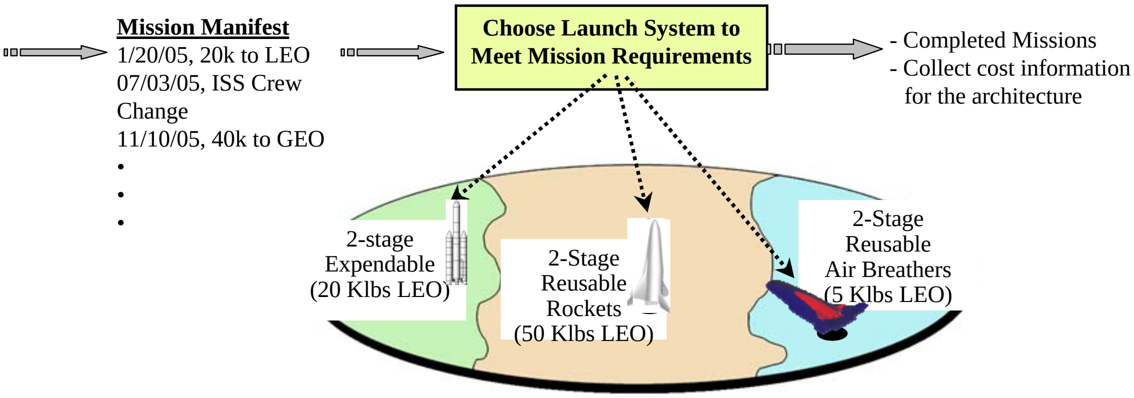
- Elements can be either expendable or reusable. A Configuration (i.e., a vehicle) may consist of expendable Elements, reusable Elements, or a mixture of both.
- There is a predefined set of generic Element types for all Architectures
- There could be an unlimited number of Elements in a Configuration
- There could be an unlimited number of Configurations in an Architecture
- There could be an unlimited number of Families within an Architecture

2. *Manifest-driven Process***:** The objective of the Architectural Model is to be able to compare the ability of alternate space transport architectures to meet a given manifest. Specific attributes targeted are cost and schedule. The model is driven by manifest requirements where given missions requirements, the model will determine the likelihood that the missions launch dates can be met and at what cost. Mission requirements include payload to orbit weight, destination, launch date, and Family and Configuration selection to execute the mission. Possible mission destinations include Geosynchronous Earth Orbit (GEO), Low Earth Orbit (LEO), the International Space Station (ISS), Planetary, and Polar missions.

Exhibit 1 shows an example of an architecture that may have 3 Configurations (or Vehicles) that belong to three different Families: a two-stage expendable vehicle capable of carrying 20,000 pounds to LEO or 10,000 pounds to GEO; a two-stage man rated reusable vehicle capable of placing 50,000 pounds in LEO or 20,000 pounds to GEO; and a two-stage man rated reusable Air Breather capable of placing a 5,000 pound payload to LEO.

The manifest includes missions that need to deliver payloads with certain weight to certain destination at certain times. Based on vehicle selection, the model should produce information about the likelihood of meeting the manifest launch dates as well as the cost of accomplishing these missions.

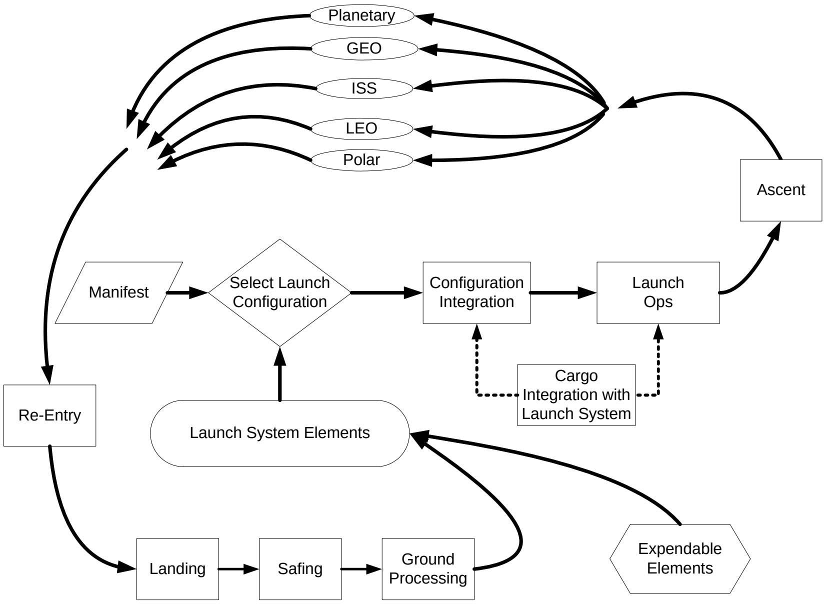
Exhibit 1. An Example of An Architecture Driven by Manifest Requirements.



3. *Process Flow*: Following the process flow illustrated in Exhibit 2, the model would use the manifest to make demands on systems (vehicles and their support resources) to be ready to launch at a predetermined date specified on the manifest. Once the elements are available and ready for integration, they are moved to the Configuration Integration process for assembly. Upon completion, the vehicle is moved to the Launch Operations process. Cargo Integration takes place either during Configuration Integration or during Launch Operations. On the launch date, the vehicle launches to the Ascent phase then to an orbit as demanded by the manifest. Depending whether the vehicle includes expendable or reusable elements, expendable elements are disposed and only reusable elements land back. Upon landing, the vehicle is safed (i.e., checked out and secured). The elements then go through ground processing to be available for new missions. As reusable elements are processed and released, new missions seize then and acquire new expendable elements as necessary.

Note that in all stages, the delay times are based on statistical distribution to reflect uncertainty existing in the real system. For example, the launch vehicle may not always be ready for service at the launch date. The consequence of not meeting the launch date is assessed via a delay cost. By the same token, a mission may arrive to the launch pad earlier than its launch date. No mission is allowed to launch before its launch date, and therefore, the mission will be kept on the launch pad until its launch date. While sitting on the launch pad cost will be incurred since launch resources remain tied up with the mission until launch.

Exhibit 2. Process Flow Chart.



The Architectural Model Design

In this section we discuss the different design aspects of the simulation model as follows.

1. Launch Elements. A generic set of expendable and reusable elements shown in Exhibit 3 is predefined in the model.

Exhibit 3. Launch Elements.

REUSABLE ELEMENTS	EXPENDABLE ELEMENTS
ORBITER	EXPENDABLE ORBITER
BOOSTER	EXPENDABLE BOOSTER
OMV (ORBITAL MANEUVERING VEHICLE)	EXPENDABLE OMV
OTV (ORBITAL TRANSFER VEHICLE)	EXPENDABLE OTV
CARGO CARRIER	EXPENDABLE CARGO CARRIER
MANNED CARRIER	EXTERNAL TANK
SRB (SOLID ROCKET BOOSTER)	

Note that these elements are generic enough to handle most future vehicle concepts that NASA may consider. Also, in this model we are interested in the functionality of an element rather than its design detail. For example, an Orbiter is a reusable element that can carry cargo and humans to space, accomplish the mission and come back. As long as we know its time and cost information, the model does not require other specifications such as size, material, or manufacturer.

*2. Manifest***.** The manifest may include more information than what the model needs. However, what is discussed here are only those inputs required by the simulation model. A manifest sample is shown in Exhibit 4. The manifest input is stored in a Microsoft Excel Spreadsheet that is maintained separately from the model. The separation of model logic from model inputs is a good design practice to ensure configurability, scalability, and modularity. We further discuss each column of the manifest sample depicted in Exhibit 4.

Exhibit 4. Manifest Sample.

Mission	*PL Weight*	*Dest.*	*Release Day*	*Launch Day*	*Delay Penalty per Day per mission*	*Family*	*Config*
1	30000	1	0	30	0	1	1
2	50000	2	20	60	0	1	1
3	30000	1	50	130	0	1	2
4	30000	1	120	180	0	2	3
5	30000	1	170	235	0	2	3
6	50000	2	225	290	0	1	2
7	30000	1	280	340	0	2	4
8	50000	2	330	395	0	1	1
9	50000	2	385	445	0	1	1

Mission No: is a serial number for each mission to be able to track a specific mission if necessary.

PL Weight: is the payload weight. This input is needed to assign the appropriate Configuration (or vehicle) to a certain mission. The user makes this assignment external to the simulation model.

Destination: is where the mission is headed. This mission attribute is also needed to assign the appropriate Configuration to a certain mission and must be entered by the user. The destination is an attribute entered to the simulation. Arena, however, accepts only numeric values for attributes rather than a string. A one-to-one mapping between the destination and their numeric values is accomplished within the spreadsheet and is depicted in Exhibit 5.

Exhibit 5. Destination Code.

DESTINATION	CODE
ISS	1
LEO	2
GEO	3
PLANETARY	4
POLAR	5

Release Day: is the date when the mission is released to the system to start the flow. This date can be arbitrarily entered by the user, or can be calculated by estimating the time needed in each facility and subtracting that from the Launch Day. This is rather a simple heuristic to generate reasonable mission release times. More sophisticated methods can be applied including heuristics to take into consideration real-time dynamics of the system.

Launch Day: is a planned launch day for a specific mission. A mission can not be launched prior to this date. The model allows for launch delays; however, there is a penalty associated with such delays. From Exhibit 4 you can observe that launch days are a cumulative sequencing of calendar days to a launch event.

Delay Penalty per Mission: is a daily penalty for missing the launch date. It is important to note that the launch delay (tardiness) cost may come from two sources: delay per Configuration and delay per mission. The delay per Configuration represents the cost of occupying or using resources past launch date. The delay penalty per mission is the daily penalty for being late to the customer. The default values for delay per mission were set to zero, but the field is kept for future use.

Mission Support Cost: Mission support cost starts accumulating a day prior to launch until the end of mission. In this model, the mission support cost factor is a linear combination of mission support cost per mission and mission support cost per Configuration. As mentioned earlier, the mission support cost per mission has been set to zero, but the field is kept for future use.

Family: is the Family number to which the vehicle flying a certain mission belongs.

Configuration: is the vehicle number that will fly a mission. Note that Configuration numbers are unique regardless to which Family they belong. For example, if there are two families and each one has two configurations, then Configurations 1 and 2 will belong to Family 1 and Configurations 3 and 4 will belong to Family 2.

*3. Processing Times***.** Processing times are the times spent by vehicles in different facilities. Processing times have two levels of fidelity. In the Integration, Cargo Integration, and Launch Operations areas, the processing times are per Configuration. However, when reusable elements go to Orbit and then land, they go to different facility types, and therefore, the processing times are per element. Both Configurations and Elements seize the necessary facilities based on the families to which they belong.

Processing times were mostly stochastic and based on triangular and uniform statistical distributions provided by process experts.

*4. Cost Information***.** The model consists of a generic set of processes that use top-level information from studies of individual launch systems to assess the costs of using an Architecture to meet the manifest requirements. The model includes the following cost components:

Acquisition Cost: The cost of acquiring a new element.

Element Cost per Use: is the "cost per use" of an element. For expendable elements, it is the same as the acquisition cost. The cost per use for reusable elements is derived by dividing the element's design life by the number of flights it is expected to fly.

Processing cost: is a rate per configuration per facility multiplied by the processing time at that facility. This cost is generated for the Integration, Cargo Integration, Launch Operations, and Ground Processing areas. As the mission goes through the first three facilities (Integration, Cargo Integration, and Launch Operations in Exhibit 2), the cost factors are per configuration; however, when the reusable Elements land back, the processing cost at the Ground Processing Area will be the cost factor per Element multiplied by the processing time an Element spends at that processing facility. It is very important not to confuse the term "Processing Cost" with the "Ground Processing Cost", which is only one of the four processing areas.

Delay (tardiness) cost: is the daily cost for missing the launch date. This cost is broken down to a cost factor per mission multiplied by the number of tardy days, and a cost factor per configuration multiplied by the number of tardy days. Mathematically, the Delay cost per mission DC_i can be defined as:

$$DC_i = \alpha_i \max(A_i - P_i, 0) + \beta_{ic} \max(A_i - P_i, 0) \quad (1)$$

where

DC_i : Delay cost for mission i

A_i : Actual launch day for mission i

P_i : Planned launch day for mission i

α_i : Delay penalty per day per mission i

β_{ic} : Mission delay penalty per day per configuration c

The total delay cost would then be:

$$DC = \sum_{i=1}^{n} DC_i \quad (2)$$

Where n is the number of missions in a manifest. The parameters P_i, α_i, and n are inputs to the model (see Exhibit 4). β_{ic} is also an input to the model but stored in a separate spreadsheet that has Configuration information. The actual launch day for a mission A_i is decided by the simulation, and therefore, the value of the Delay cost DC is generated by the simulation.

Mission Support Cost: is Mission Support Cost factor per Configuration multiplied by the time one day prior to launch until landing.

Manifest Cost: is the summation of all previous costs for all missions. This cost can be used as a measure of performance to compare different architectures.

Simulation Model Verification and Validation

Simulation verification is the process of debugging the model to ensure it behaves as expected. Steps taken to verify the model include testing every added functionality independently to make debugging easier and more efficient. Second, structured meetings with NASA's experts on the processes were conducted to check the model's logic. Finally, the complete model was run with several scenarios and the logic was traced using animation to ensure that the model doing what it was built to do.

The most common way for simulation validation is to compare its output with historical data. The Architectural Model, however, is meant to be used for the operational design of future launch systems, for which historical data does not exist. As a result, we had to find other methods of validation. As a preliminary method of validation, several NASA experts inspected and validated the results of several hypothetical scenarios that were run through the model. Additionally, the Space Shuttle's historical data was used to populate the model with cost factors and processing times. A ten-year Shuttle manifest was generated with an average of seven flights per year and was run through the populated model. The rationale behind using the Shuttle scenario for validation purposes is that it is the only existing reusable launch vehicle and if the model is robust enough, it should handle the Space Shuttle as one of the possible launch systems to accomplish the manifest.

As an initial step for validation, the model was populated with deterministic data instead of stochastic data and its results were compared to a spreadsheet cost model. The model was then run with statistical distributions based on the Shuttle's historical data. The cost breakdown within the simulation model was based on the framework discussed earlier. The infrastructure cost and fixed cost for the shuttle scenario was added post the simulation runs. The results of running the shuttle scenario produced an average cost of 3.5 to 4.0 Billion dollars a year, which is comparable to the Shuttle's program budget over the past several years indicating the preliminary validity of the Architectural Model. This result was brought before three NASA expert who shared the same conclusion. It is important to notice that further model validation is necessary especially when we have a mix (i.e., architecture) of launch families and not only one family (i.e., the Shuttle family). Therefore, more experiments need to be conducted with more than one family in the model. This can be accomplished by using historical data of existing launch families such as the Atlas and Delta families in addition to the Shuttle family. This work is still in progress and more validation will be carried out in the near future.

Conclusions and Future Work

In this paper, the Architectural Model, a discrete-event simulation model, is presented to assess the operation and support costs for future launch systems to fly a set of missions. The model also helps investigate the capability and likelihood of launch system families to meet a manifest schedule. The model was designed to be robust enough to handle many possible launch vehicle designs. To validate the model, several NASA experts reviewed preliminary simulation results and concurred the outputs were reasonable. In addition, Space Shuttle historical data was used for preliminary validations. The results thus far indicate the validity and usefulness of the model. More simulation validation needs to done however before it can be used to test potential future conceptual launch systems.

After more thorough validation of the Architectural Model is complete, a natural extension of the model would be to do simulation optimization. Currently, the user needs to specify the resource levels such as the number of elements and facilities capacity. With a simulation optimization approach, the user needs to provide only the manifest input data and the optimization system should find the optimal or near-optimal settings of the resource levels. Scheduling and mission release heuristics can also be developed to have more optimized flow of missions, which will affect the optimal level of resources required.

References

Cates, G. R., Mollaghasemi, M., Rabadi, G., and Steele, M. J. "Macro-Level Simulation Model of Space Shuttle Processing", *Military, Government and Aerospace Simulation Proceeding*, Vol. 33, No. 4 (2001), pp. 143-148.

Law, A.M., and Kelton, W. D., *Simulation Modeling and Analysis*, McGraw Hill (2000).

Mollaghasemi, M., Rabadi, G., Cates, G. R., Correa, D. Steele, M. J., and Shelton, D., "Simulation Modeling and Analysis of Space Shuttle Flight Hardware Processing", *Proceeding of Harbour, Maritime & Multimodal Logistics Modeling and Simulation Workshop* (2000), pp. 59-62.

Rabadi, G., Mollaghasemi, M., Cates G. R., and Steele,M. J., "A Discrete-Event Simulation Model for the Space Shuttle Ground Processing Operations at Kennedy Space Center", *Simulation and Modeling for Acquisition, Requirements and Training (SMART) Conference* (2001).

Ruiz-Torres, T., and Zapata, E., "Simulation Based Operational Analysis of Future Space Transportation Systems", *In J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, eds.*, *Proceedings of the Winter Simulation Conference* (2000), pp. 1123 – 1131.

Steele, M. J., Mollaghasemi, M., Rabadi, G., and Cates, G. R, "Generic Simulation Models of Reusable Launch Vehicles". *In E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes (eds.), Proceedings of the Winter Simulation Conference* (2002), pp. 747 – 753.

About the Author(s)

Ghaith Rabadi is an Assistant Professor at the Department of Engineering Management and Systems Engineering at Old Dominion University. He received his Ph.D., and M.S. degrees in Industrial Engineering from the University of Central Florida in 1999 and 1996 respectively. He received his B.S. degree in Industrial Engineering from the University of Jordan in 1992. His major research interest is in the areas of scheduling, mathematical programming, simulation and optimization. He has been involoved with NASA projects since year 2000.

Trina Marsh Chytka works for NASA in the Vehicle Analysis Branch at the Langley Research Center in Hampton, Virginia. Prior to NASA, she worked for Northrop Grumman Newport News Shipbuilding in the Submarine Engineering Division. Trina Marsh Chytka graduated from Old Dominion University with a degree in Civil Engineering in 1997; received a Master's degree in Engineering Management from Old Dominion University in 2000; and earned her Ph.D. in Engineering Management and Systems Engineering in 2003.

W. Doug Morris received his Bachelor of Science in Aerospace Engineering from North Carolina State University in 1965 and his Master of Science in Oceanography from Old Dominion University in 1979. Currently, Mr. Morris is a research engineer at the Langley Research Center in Hampton, Virginia. He leads a research effort whose objective is to develop concepts, tools, and the associated technology data bases which will aid in the design of future space transportation vehicles that offer significant improvement in operational efficiency and economy.

Nancy H. White received a B.S. in Mathematics from Longwood College and an M.S. in Applied Science from the College of William and Mary. She was employed at the NASA Largely Research Center for more than 30 years prior to her retirement in 2003.

Resit Unal is a Professor in the Department of Engineering Management. He has B.S. in Electrical Engineering and received his M.S. and Ph.D. in Engineering Management from the University of Missouri, Rolla in 1983 and 1987 respectively. Dr. Unal is involved in research at NASA Langley Research Center in developing multidisciplinary design optimization approaches for space transportation systems. His research interests include Multidisciplinary Design Optimization, Robust design, Quality engineering, and Response Surface Methods.