

Modeling Temporal Processes in Early Spacecraft Design: Application of Discrete-Event Simulations for DARPA’s F6 Program

Gregory F. Dubos^{*}, Steven Cornford[†]

Jet Propulsion Laboratory, Pasadena, California, 91109

While the ability to model the state of a space system over time is essential during spacecraft operations, the use of time-based simulations remains rare in preliminary design. The absence of the time dimension in most traditional early design tools can however become a hurdle when designing complex systems whose development and operations can be disrupted by various events, such as delays or failures. As the value delivered by a space system is highly affected by such events, exploring the trade space for designs that yield the maximum value calls for the explicit modeling of time.

This paper discusses the use of discrete-event models to simulate spacecraft development schedule as well as operational scenarios and on-orbit resources in the presence of uncertainty. It illustrates how such simulations can be utilized to support trade studies, through the example of a tool developed for DARPA’s F6 program to assist the design of “fractionated spacecraft”.

I. Introduction

While the ability to model the state of a space system over time is essential during spacecraft operations, the use of time-based simulations remains rare in preliminary design. Many factors contribute to this late-stage-only usage, but it is predominately due to the lack of agility in building the associated models. Thus, projects wait until the mission, architecture and design parameters have converged before undertaking the construction of a full time-based simulation. However, many events that might occur throughout a spacecraft’s lifecycle (such as technology development issues, supply chain delay, or on-orbit failures) can ultimately impact the total value delivered by the system over a given time horizon. When exploring the trade space to find designs that deliver the maximum value (e.g., science return), modeling such events along the time dimension thus becomes relevant. Although combinations of ad hoc tools can be utilized for that purpose, the use of discrete-event simulations (DES) in early concept design can not only facilitate the simulation of operational scenarios and on-orbit resources consumption, but also provides valuable means to model the various phases of the spacecraft development schedule until launch.

Along these objectives, this paper presents work performed as part of JPL’s contribution to DARPA’s F6 program, through the formulation of a tool to assist the design and evaluation of “fractionated spacecraft”. A fractionated spacecraft is defined as a cluster of free-flying modules, wirelessly inter-connected and sharing resources, but capable of being developed, launched and replaced independently [1]. Unlike current monolithic spacecraft that are particularly exposed to program delays, market uncertainties and technical failures, fractionated spacecraft seem to offer adaptability and survivability features that may make them superior from a value standpoint. This paper shows how the use of rapidly reconfigurable discrete-event models was instrumental in exploring the large combinatorial space of fractionated spacecraft designs evaluated in the presence of uncertainty. Specifically, various types of risk (or “stimuli”) threatening the spacecraft were explicitly modeled and included, as proposed by DARPA: supply chain delays, funding profile fluctuations, technology development risk, change in user needs, technology obsolescence, launch failures, component failures, and orbital debris.

The remainder of this paper is organized as follows. Section II discusses the motivation behind the use of discrete-event simulations during early design activities and introduces the application of DES to DARPA’s F6 program. Section III provides an overview of the F6 DES model structure and discusses an example simulation of the nominal

^{*} Systems Engineer, NASA Jet Propulsion Laboratory, Mission Systems Concepts.

Corresponding author, Gregory.F.Dubos@jpl.nasa.gov.

[†] Senior Engineer, NASA Jet Propulsion Laboratory, Strategic Systems Office.

development and operations of a fractionated cluster. The unique capability of DES to explicitly simulate the emergence of unexpected events (identified as “risks”) is then presented in Section IV. Finally, Section V concludes this work by discussing further the opportunities offered by such simulation frameworks during early spacecraft design, along with some limitations.

II. The use of time-based simulation to explore the trade space during preliminary spacecraft design

A. Motivation

Time-based simulations (or “dynamic models”) of a space system, in which time is explicitly simulated, are not widely used during preliminary spacecraft design activities. However, at the early stages of a mission lifecycle, there still exist opportunities to gain insight about design options beyond the information traditionally obtained with standard methods. The following section examines current modeling needs that, if properly addressed, could enrich the preliminary design activities.

a) Capturing complex operational scenarios

Many analyses performed during preliminary spacecraft design are typically “static”: their focus is on subsystem sizing and the constitution of budgets (i.e. mass, power) in one single state (or at best a handful of states) representing the nominal behavior of the spacecraft. For example, it is routine to decompose the mission into a small set of “power modes” and then to use Excel-based tables which capture the power used, created and stored during various mission phases corresponding to these “power modes”, as illustrated in Table 1.

Table 1. Illustrative table showing power modes and power states during those modes

	Power State Table					
		Initial Check-out	Cruise	Orbit Insertion	Data Collection	Downlink
Instrument A	Stand-by (W)	-10	0	0		-10
	On (W)				-50	
Telecom SS	Stand-by (W)	-10	-10	-10	-10	
	On (W)					-40
CDS SS	(W)	-25	-25	-25	-25	-25
Other SS	(W)	-75	-75	-75	-75	-75
Solar Array	(W)	200	150	150	175	175
	NET POWER	80	40	40	15	25

For more complex systems that include various subsystems having different state evolutions over time, it becomes increasingly difficult to use such static models to obtain relevant budgets, due to the explosion of the number of scenarios or modes. In addition, unpredictable couplings can also make these static models unsuitable (e.g. opportunistic targeting and lack of knowledge of 1) where in the orbit this occurs or 2) resource states at that time).

Time-based simulations provide a mean to tackle that issue: once the behavior of each subsystem (e.g. subsystem power consumption) is defined, the state at the system-level (e.g., bus available power) is computed at every instant based on its previous value and on the combination of the states of all the subsystems. Instead of pre-calculating the system state by assuming a given scenario (or mode), the simulation thus follows the course of time and aggregates the subsystems’ states. In effect, the combinatorial complexity that was previously the burden of the engineer is replaced with frequent state updates that are now the burden of the machine. Note that in the latter case, the added benefit is the ability to observe the state of the system at every instant of the mission and not only for a finite number of situations.

b) Accounting explicitly for system development times

In early design stages, creating a reasonable schedule of the development (or “implementation”) of the spacecraft is often a challenge. It typically relies on historical data, institutional and managerial experience, margins (e.g. one month per year) and a lot of educated guesses. Some attention is paid to schedule when one design (or a few at most) has been selected, but when the trade space is still open and many designs are still being considered concurrently, most engineering efforts are generally directed towards the evaluation of technical performance, neglecting development schedule. The approach presented in this paper builds schedule estimates from the bottom-up, (i.e. from individual components delivery times to spacecraft integration and launch), in which schedule *itself* becomes a parameter of the trade study.

c) Designing for uncertainty

Over the course of the development and operations of a spacecraft, there exist many external events that can affect programmatic aspects of the mission and/or degrade system performance. These possible events are traditionally identified as “risks” during preliminary studies, and their likelihood and impact are assessed using methods that are mostly qualitative. For each combination of risks that materialize, there exists a different future under which the system will come into being and operate. It thus becomes difficult to use static models to capture these possible scenarios resulting from various possible combinations of events. As quantifying the joint and cascading effect of certain risks on system performance and program cost and schedule appears unfeasible with traditional static methods, it is common to add several layers of “design margins” intended to absorb all possible uncertainties. This design approach presents the disadvantage of excessively inflating parameters such as mass, power and cost. The ability to observe the impact of various combinations of risks as they unfold over time appears crucial to help identifying where margins would remain critical, and where margins might be unnecessary.

d) Enabling value-centric design

Finally, recent research has advocated a shift from purely cost-driven design of space systems to more value-centric design approaches [2]. As the value delivered by a space system over time is affected by the three aspects discussed previously (namely system development, operational scenarios, and emergence of unexpected events), dynamic methods that capture these elements show great potential to assist the evaluation of design alternatives according to value-centric principles.

B. Discrete-event simulations

Many approaches exist to model the way a particular system behaves over time. Identifying the nature of the system studied is then essential to choose the proper strategy to simulate how it evolves through time. Many systems are typically modeled using laws of physics that often involve differential equations, assuming their state changes continuously over time. Such systems can thus be described with continuous variables such as position, velocity, temperature, etc. Their state evolution can be adequately simulated using *time-driven simulations*, in which a master clock is regularly advancing, one step at a time. There exists another class of systems however, with states that only take a finite number of values (e.g., on/off) and that can be subject to events disrupting their evolution. In this case, the asynchronous behavior of the system is better modeled with an *event-driven simulation*, in which the clock jumps from one event to the next scheduled event [3].

Although certain variables describing the state of a space system can be seen as continuously changing (e.g., position, velocity, available power, available propellant), many other spacecraft processes occur at discrete points in time (e.g., launch, entry to umbra region) or involve discrete quantities (e.g., number of packets transmitted to the ground). The existence of such processes makes it difficult to use time-driven simulation to model the behavior of the spacecraft, and calls instead for the use of discrete-event simulations. Furthermore, a discrete-event simulator becomes the natural candidate to model the emergence of unexpected events (or “risks”) that can disturb the normal development and operations of a space system.

C. Application of DES for DARPA’s F6 program

The Broad Agency Announcement (BAA) for System F6 issued by DARPA in October 2010 laid out the key features of a desired design tool intended to support “the widespread development and employment of fractionated architectures”. Specifically, the purpose of such a tool was a) to help understand if/when the business case for a fractionated spacecraft closes; and b) in that case, to help identify the design features (e.g., degree of fractionation, distribution of functional components, amount of redundancy, etc.) of the most promising fractionated architectures from a value perspective. Special emphasis was to be put on the modeling of technical and market uncertainties that could arise during the spacecraft development as well as during on-orbit operations. Such uncertainties included for example technology development risks, supply chain delays, changes in user needs, program funding fluctuations, launch failures, component failures, orbital debris, and technological obsolescence. In the light of the points discussed in beginning of this section, the use of a discrete-event simulation engine as a key component of a design tool for F6 appeared justified.

Many discrete-event simulation frameworks have been developed over the years, both commercially and in academia. Examples of popular ones include OMNet++, SimEvents, ns-2, JiST, ARENA, and SimPy, with different foci, such as supply chains or networks. Several reasons motivated the selection of the SimPy simulation framework to develop the DES component of the tool developed for the F6 program. SimPy is an open-source discrete-event simulation framework that builds on the Python language [4]. Its object-oriented nature facilitated the inclusion of the DES model into the larger F6 design tool, that was itself built according to object-oriented principles (in this case, in SysML). The choice of an object-oriented programming language also presented major benefits in terms of model scalability, since generic behaviors only need to be defined once to be used multiple times with ease, and an existing model can be rapidly reconfigured to be executed in different conditions. This essential feature enables the use of discrete-event simulation, not only to model the performance of a single-point design, but also *to perform a trade space exploration among an unlimited number of design alternatives*. More specifically, the trade space of fractionated spacecraft clusters that can be explored includes not only parametric variants (e.g., various data rates values), but also structural variants (e.g., architectures with different numbers of spacecraft). Creating a new instance of an executable DES model for a completely different cluster configuration thus becomes almost as simple as “dragging and dropping” new nodes (spacecraft and/or ground stations) into an existing version of the DES model. Additionally, the reliance on an open-source language with transparent functions and behaviors enabled modifications or extensions of the existing simulation engine to meet specific needs, enhancing the framework flexibility.

III. Overview of the F6 DES model

The following section presents the general structure of the F6 discrete-event simulation (DES) model. It was designed to simulate the entire lifecycle of a spacecraft cluster: a) from the conception of the individual spacecraft to the launch, i.e. the Implementation phase, and b) from the launch of a spacecraft to its termination, i.e. the Operations phase.

In the following, a distinction is made between “motherships” and “daughterships”. Daughterships are spacecraft that carry one or more payloads (which produce data) and can crosslink data to motherships within the same fractionated cluster. In addition to having all the capabilities of daughterships[‡], motherships are spacecraft that possess the ability to downlink data to ground stations.

A. Cluster implementation

Each individual spacecraft, or space node of the fractionated cluster, is assumed to be composed of various subassemblies as summarized on Table 2. The spacecraft infrastructure relates to the hardware enabling the basic capabilities of any spacecraft (bus). The F6 Tech Package refers to the hardware allowing crosslinking between spacecraft of a fractionated cluster. The downlinker relates to the hardware enabling communication between a mothership and the ground. Finally, the payloads are the data producers.

[‡] In this study, motherships without payloads are also allowed.

Table 2. List of components for each type of spacecraft for implementation purposes

Daughtership components	Mothership components
Spacecraft infrastructure	Spacecraft infrastructure
F6 Tech Package	F6 Tech Package
Payload(s)	Payload(s)
	Downlinker

The various subassemblies composing a spacecraft are developed in parallel on distinct production lines, following a sequence traditionally referred to as “Design, Build, Assembly and Test” (DBAT). Each of these four phases has a specific duration that constitutes an input of the DES model. The following assumptions are made:

- Each phase starts immediately after the previous one has been completed, and not before.
- Every spacecraft containing a given subassembly is in a queue to receive one unit of that specific subassembly type once it is ready. Only one unit is produced at a time.
- The Design phase is only conducted once, for the first unit produced by a given production line.
- Schedule learning curves are applied to each phase duration, and represent the reduction in schedule resulting from learning effects associated with the development of multiple identical units on the same production line.

Once all the subassemblies composing a cluster node have been delivered, they are integrated to form a single spacecraft. This spacecraft is then integrated into a launch vehicle, provided there is one ready to be launched (Figure 1).

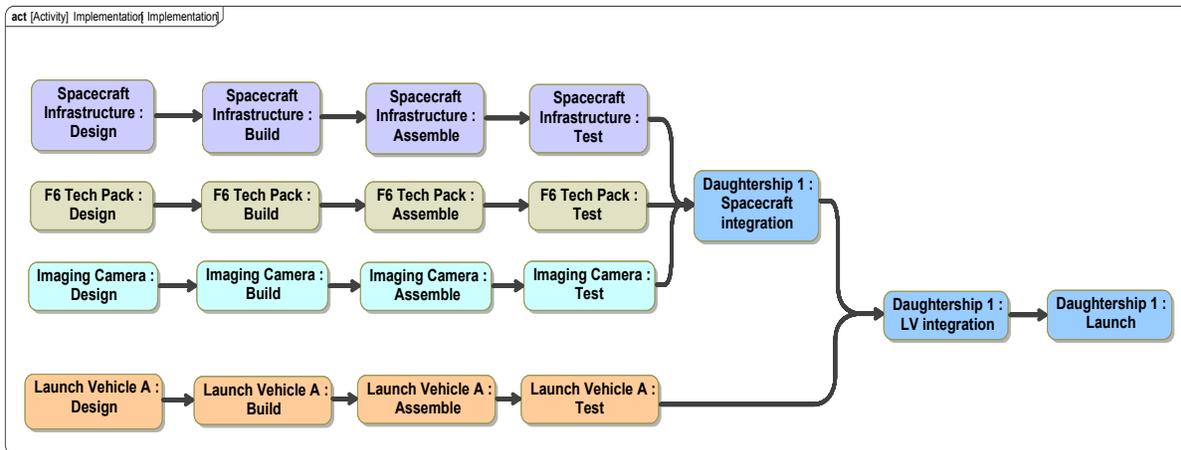


Figure 1. Symbolic representation of the implementation of a spacecraft (daughtership)

B. Cluster operations

For each spacecraft composing a fractionated cluster, the DES model simulates three main processes occurring while on orbit: power management, propellant management and data management.

Although these processes are simulated in parallel via distinct logical loops, a certain degree of coupling is reflected in the model. For example, data-related processes are suspended on a given spacecraft if the spacecraft energy level

reaches a lower limit (Figure 2). They will remain inactive until the energy goes back up. In a similar fashion, a spacecraft will be considered non-operational once its propellant level reaches the limit corresponding to the amount of fuel necessary to perform de-orbit maneuvers.

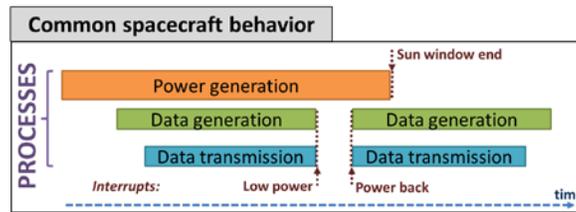


Figure 2. Symbolic representation of the model of fractionated spacecraft operations

Power management: Energy is produced via the spacecraft solar arrays at a rate that is entered as a model input. The production of energy follows a cycle that repeats every orbit, as the spacecraft enters and exits regions with no incident solar light (umbra). The energy is then stored inside batteries whose storage capability constitutes another model input. Power is drawn from the batteries by the spacecraft infrastructure, the F6 Technology Package and the payloads, at rates that are entered as inputs.

Sun windows/eclipse periods are entered using a given cycle that repeats every orbit. (Real ephemeris data could be entered). Figure 3 illustrates the depth of discharge of the battery of a daughtership, with eclipse periods arbitrarily set to 45 min out of a 90-min orbit, and the initial charge set to half the battery capacity.

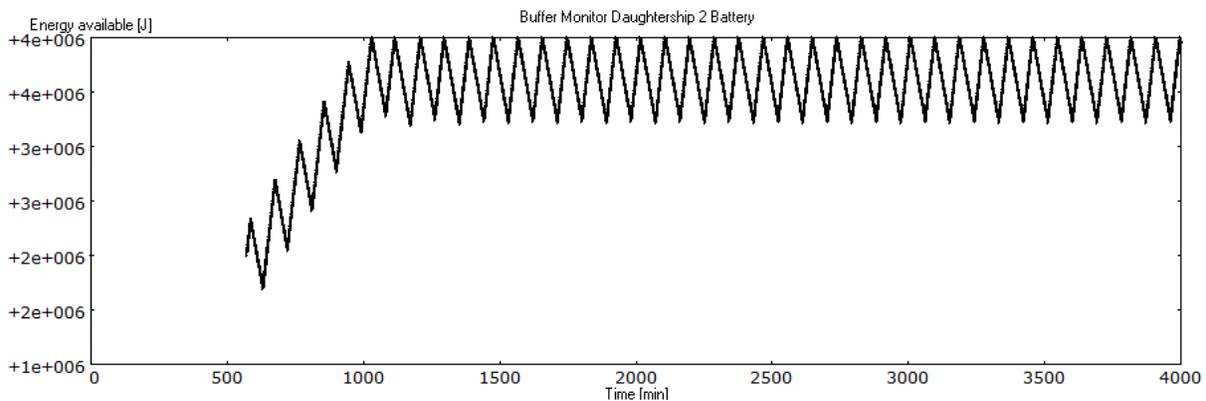


Figure 3. Depth of discharge of a daughtership

Propellant management: The DES model assumes liquid propulsion, with propellant being stored in one (or more) tank(s) whose total storage capacity is a model input. In the nominal mode of operations, propellant is consumed in a quasi-continuous manner through station keeping maneuvers that aim at maintaining the spacecraft on the desired orbit.

Data management: On each spacecraft, data is generated by payloads (if any) at rates selected as inputs, and is then stored on the spacecraft "hard drive", whose capacity is also entered as a model input. When a daughtership has established a connection with a mothership, it can cross-link its data and thus allow for more data to be generated by the payload(s) and stored on-board. During a crosslink, the mothership's hard disk gets filled with the incoming data from the daughterships to which it is connected. Onboard storage is thus reduced until the mothership gets the opportunity to downlink its onboard data to ground stations. Such opportunities are modeled by downlink windows that occur periodically. (More realistic windows can be implemented if real ephemeris data is utilized). A mothership will refuse incoming connections from daughterships if its hard disk is full. In this case, the corresponding daughterships will attempt to crosslink to other motherships if these exist in the cluster. Ultimately, the model outputs the total amount of data (for each type) downlinked to the ground stations over time.

C. Nominal case example (no uncertainties)

Consider for example a fractionated cluster composed of three spacecraft organized as follows:

Mothership 1: containing a Spacecraft Infrastructure, an F6 Tech Package and a Downlinker

Daughtership 1: containing a Spacecraft Infrastructure, an F6 Tech Package, an Imaging Camera producing data of type “Blue”, and a Mapping Camera producing data of type “Red”

Daughtership 2: containing a Spacecraft Infrastructure, an F6 Tech Package, an Imaging Camera producing data of type “Blue”, and a Radiometer producing data of type “Green”

Figure 4 shows that once Daughtership 1 is launched (approximately one month after the launch of Mothership 1), it starts collecting “Blue” and “Red” data and storing it on its hard drive. As data is generated by the payloads but also crosslinked to the mothership, the levels of onboard data oscillate very rapidly[§] from low values to higher values (hundreds of Mbytes). Note also that since the rate of data generation of the Imaging Camera is greater than that of the Mapping Camera, blue data tends to occupy more space than red data.

Daughtership 2 is launched approximately one week after the launch of Mothership 1**, taking advantage of the shorter development of its specific payload (Green) and launch vehicles compared to those of Daughtership 1. As shown on Figure 5, Daughtership 2 shows a similar behavior than that of Daughtership 1 after its launch, but its storage capacity (~500 MBytes) is quickly reached and distributed among Blue data and Green data. The levels of onboard data never go back to zero, mostly due to a total rate of data generation that is much greater than the crosslink rate.

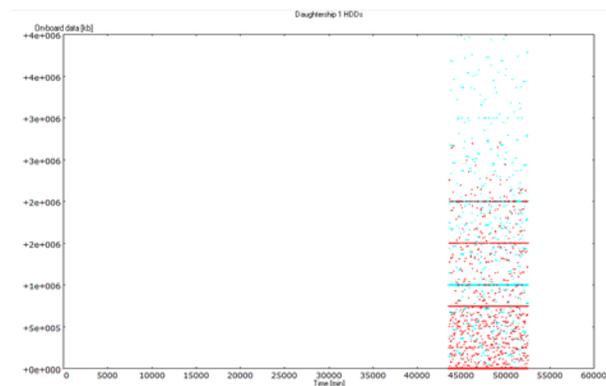


Figure 4. Data onboard Daughtership 1 over time (elapsed since Mothership 1 launch)

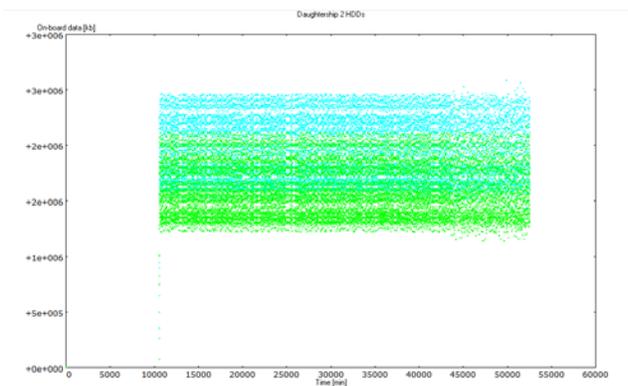


Figure 5. Data onboard Daughtership 2 over time (elapsed since Mothership 1 launch)

Figure 6 illustrates the crosslink process initiated between Mothership 1 and both Daughterships. Specifically, Blue data and Green data generated by the Daughtership 2 payloads is transferred to Mothership 1 following the launch of Daughtership 2. As soon as Daughtership 1 is launched, Red data appears on the hard drive of Mothership 1, and the level of blue data goes up as both Daughterships carry an Imaging Camera that produce blue data and transfer such data to Mothership 1.

Finally, the total data downlinked to the ground per data type can be observed on Figure 7. Both Blue and Green data is returned after the launch of Daughtership 2. The launch of Daughtership 1 results in the return of Red data to the ground, a subsequent moderate reduction in the rate of return of Green data (as the onboard storage capacity of Mothership 1 is then shared by an additional data type), and finally a major increase in Blue data return (due to the operation of an additional Imaging Camera).

[§] This extremely rapid oscillation relative to the time scale explains the multiple lines that appear to be parallel (horizontal) for a given data type.

^{**} In this example, development schedules have been artificially shortened to allow “zooming-in” on the operations phase.

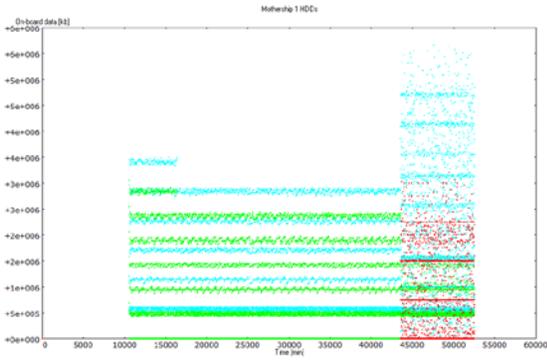


Figure 6. Data onboard Mothership 1 over time (elapsed since Mothership 1 launch)

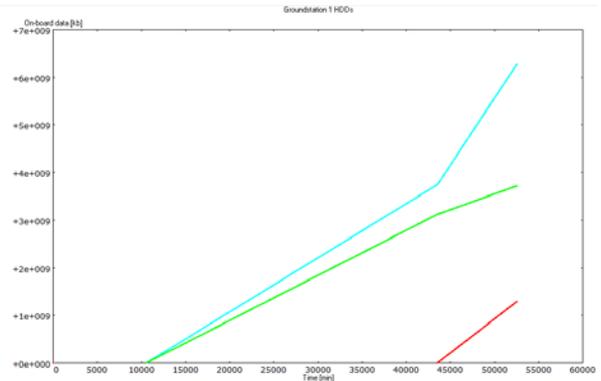


Figure 7. Data downlinked to Groundstation 1 over time (elapsed since Mothership 1 launch)

One of the features offered by such a DES model is the ability to directly investigate the connections between certain design parameters and the value obtained from the operation of the spacecraft. Such interactions are particularly observable on limiting cases. For example, Figure 8 illustrates the impact of poor power design on a mothership: the generation of energy being insufficient, the battery energy level rapidly drops and oscillates around low values (black curve). This limits the ability to operate the payload instruments and to generate data, as reflected by the reduction in data downlinked to the groundstation over time (lower green curve).

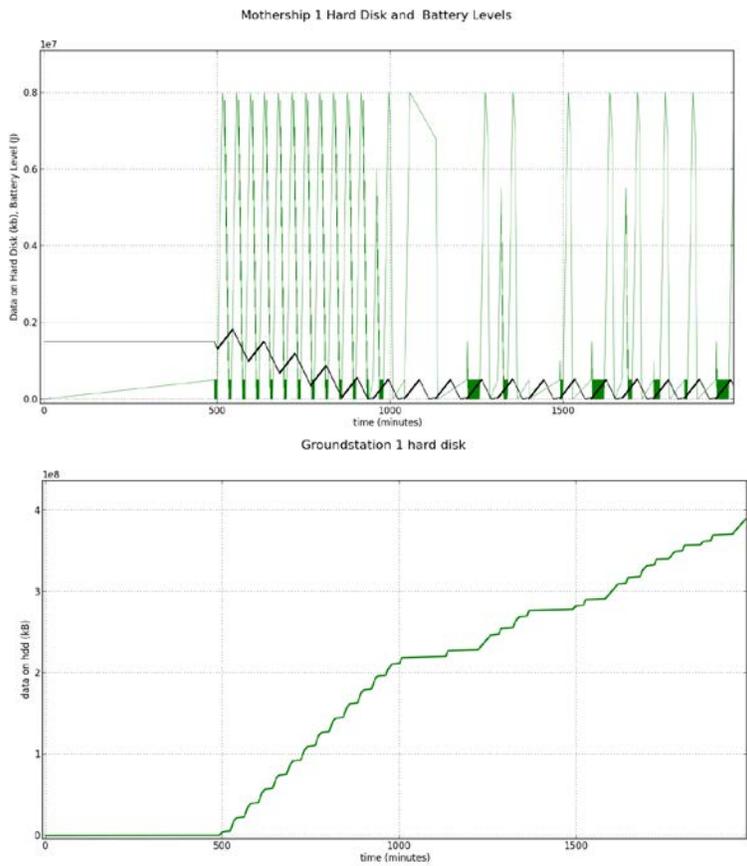


Figure 8. Hard disk and battery level on a mothership (top) and corresponding data downlinked to the ground (bottom) in case of poor power design

In addition to providing quantitative information for a single-point design, the F6 DES model enables the exploration of the trade space by varying spacecraft design parameters such as battery size or payload power consumption. It then becomes possible to observe the impacts of such variations on the amount of data downlinked for example. Figure 9 represents the performance in terms of total data returned (in color) of 160 different fractionated architectures, for which battery capacity, solar array charging rate and payload power consumption of a specific daughtershship are varied along each axis^{††}.

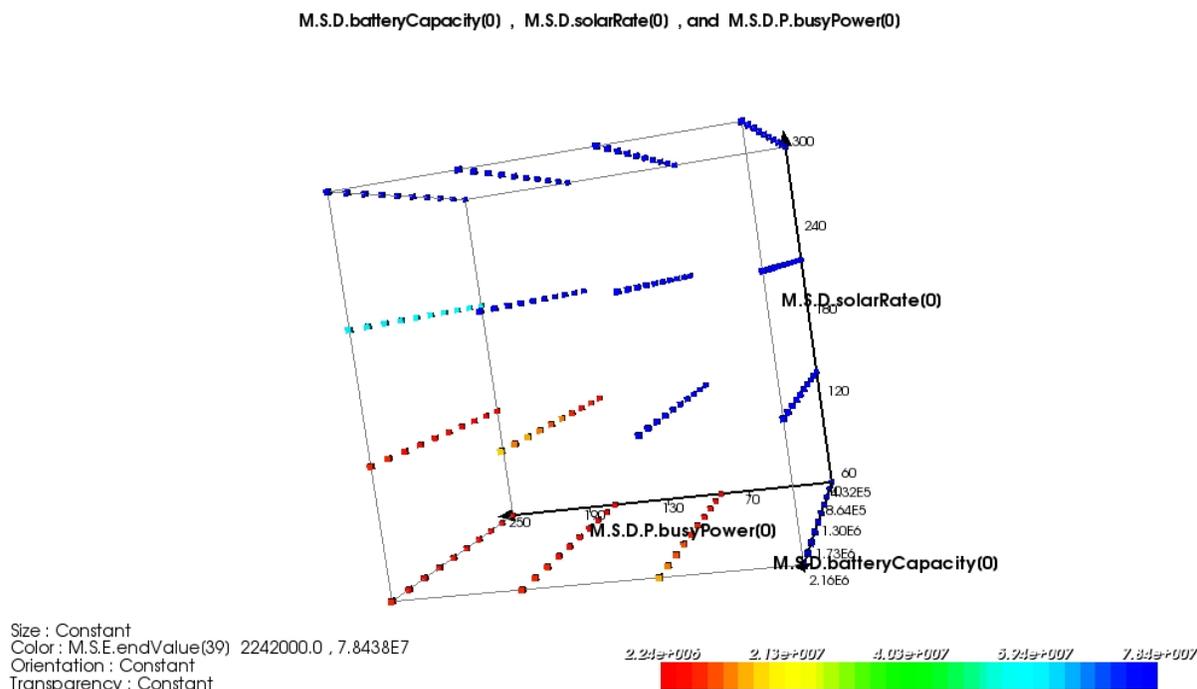


Figure 9. Representation of the trade space obtained when varying selected power-related parameters

IV. Evaluating designs in the presence of uncertainty

The research on fractionated space architectures was in part motivated by the observation that traditional monolithic spacecraft lack the intrinsic adaptability and survivability needed to cope with the many uncertainties that put the value they deliver at risk [5]. As part of the JPL-led effort responding to the DARPA F6 solicitation, the DES model presented here constitutes a key element to quantify the value delivered by a space architecture (monolithic or fractionated) in the presence of uncertainty. Various “anticipated or unanticipated stimuli, events or changes”, occurring either during the development of the space architecture or during its life on orbit, have been explicitly captured in the DES model. The following section briefly describes the way each type of uncertainty (or stimulus) was implemented in the model along with two possible responses of the owner of the fractionated cluster to that event. Note that the decision point is assumed to coincide with the emergence of the event.

A. Sources of uncertainty

Supply chain delay: one (or more) production line(s) is behind the nominal schedule to produce the corresponding subassembly. This results in the slippage of the schedule of each spacecraft of which this subassembly is a

^{††} The commercial software ModelCenter by Phoenix Integration was used in this example to set up the design of experiments resulting in the population of this trade space and to visualize the design points.

component. The two possible responses to this event are: a) reorder the queues of the production lines so that a spacecraft containing a delayed subassembly has the lowest priority. As a result, spacecraft that do not depend on the delayed subassembly will not have their schedule affected by the delay; b) maintain the initial queue order of the production lines.

Change in user needs: at a given time, the need for a particular type of payload disappears, after the start of the development of one or more spacecraft intended to carry that payload. The two possible responses to this event are: a) if the integration of the subassemblies (including payloads) has not started when the change in user needs occurs, the irrelevant payload is replaced in all spacecraft that initially carried it, with a different type of payload meeting some user needs; b) the irrelevant payload remains in place in the spacecraft intended to carry it, resulting in little or no value produced on orbit.

Funding Fluctuation Profile: additional funding becomes available during the development of one or more spacecraft of the fractionated cluster. The two possible responses to this event are: a) allocate the extra funding to one (or more) specific subassemblies to accelerate their development, i.e. compress the time remaining in the production schedule; b) no acceleration of the development of the corresponding subassemblies.

Component failure: a spacecraft of the fractionated cluster experiences a component anomaly resulting in total failure, at a certain time after its launch. The two possible responses to this event are: a) the replacement of the failed spacecraft. The development of an identical spacecraft is initiated, by scheduling the production of new units of the necessary subassemblies as well as using existing subassemblies previously produced and available in inventory; b) no replacement, and loss of the associated on-orbit capability.

Launch failure: a launch vehicle experiences a failure leading to the total loss of the spacecraft it was carrying onboard. The two possible responses to this event are: a) the replacement of the failed launch vehicle and each spacecraft that it was carrying; b) no replacement, and the associated spacecraft never join the on-orbit cluster.

Orbital Debris: the on-orbit spacecraft of a fractionated cluster encounter orbital debris that threaten to collide with some (or all) of them. The two possible responses to this event are: a) a cluster-wide avoidance maneuver (scatter-regather) that consumes extra propellant, reducing the on-orbit lifetime of the spacecraft; b) no maneuver, and total loss of the spacecraft hit by the debris.

Technology Development Risk: a subassembly is not sufficiently mature to be used on one (or more) spacecraft, i.e. its Technology Readiness Level (TRL) is too low to ensure that its development (i.e., DBAT) will be on schedule. This type of uncertainty and its corresponding responses are currently modeled like a supply chain delay.

Technology Obsolescence: at a given time, the technology characterizing a particular type of payload is outperformed by a competing new technology. This type of uncertainty and its corresponding responses are currently modeled like a change in user needs.

B. Examples

Orbital Debris

In this example, an encounter with orbital debris occurs at approximately $t = 3$ months following the mission start. In this particular future, the decision is made to perform an avoidance maneuver by scattering then re-gathering the spacecraft of the fractionated cluster. Figure 10 illustrates how this maneuver consumes a significant amount of propellant, resulting in a reduction of the spacecraft on-orbit lifetime, which in turn lowers the total amount of data returned over a given period.

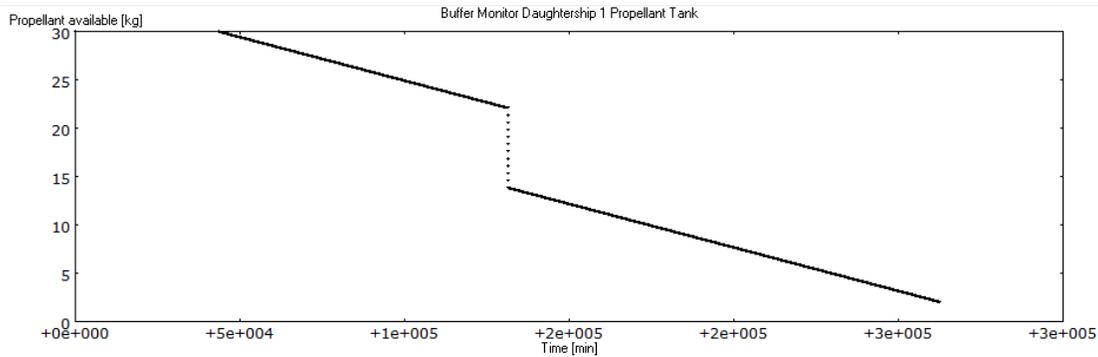


Figure 10. Evolution of the propellant available of a daughtership with a maneuver to avoid orbital debris

Supply Chain Delay

In this example, a mapping camera designed for Daughtership 1 is behind schedule, due to a contractor delay resulting in a schedule slip of approximately 1.5 year for the camera. An option to *Reorder* the queues of the production lines is exercised as soon as the delay is identified: Daughtership 1 carrying the mapping camera will receive other subsystems last to minimize schedule impact on other spacecraft under development. As a result, Daughtership 2 no longer waits to be launched due to delay in the development of Daughtership 1, allowing the downlink of data to start as early as $t = 1020$ days after the mission start (against $t = 1201$ days if no action is taken). The use of the DES model allows quantifying the gain in total data downlinked that is achieved over the life of the mission (+12 % or +2.6 TByte) via this response to uncertainty (see Table 3)..

Table 3. Launch timeline in a supply chain delay scenario (for both responses)

Spacecraft	Launch date [days after mission start]	
	No response	Reorder
Mothership 1	849	849
Daughtership 1	1201	1260
Daughtership 2	1260	1020
Total data downlinked after 5 years [TByte]	21.3	23.9

Component failure

Building on the nominal case presented in the previous section, this example shows the impact of a daughtership failure on the total data returned to the ground station. The failure of Daughtership 2 occurs approximately 11 days following its launch. In this particular future, the decision is made to replace the spacecraft as soon as the failure occurs. For the purpose of this example, the launch of the replacement spacecraft is assumed to occur 2 weeks following the failure. The gap in the blue and green curves in Figure 11 shows the interruption in data collection due to the spacecraft failure. This ultimately results in a lower amount of cumulative data downlinked over a given time horizon compared to the nominal case.

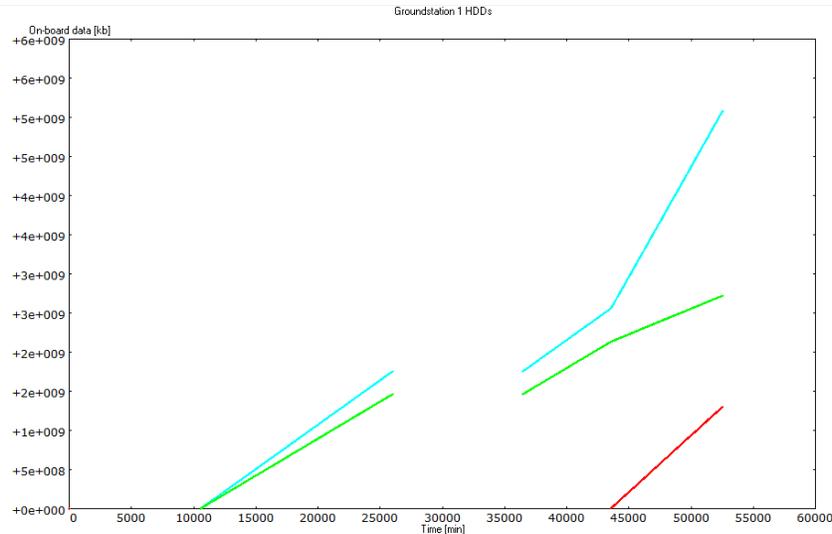


Figure 11. Data downlinked over time in case of a daughtership failure

In the scenario of a replacement of a failed spacecraft, the DES model allows exploring the impact of programmatic variables (e.g. durations of DBAT) on the duration of the interruption (or reduction) of data collection. It is then possible to test whether a given requirement on the total value delivered over a given time horizon will be met for various values of the DBAT durations, launch vehicles lead times, use of spares in storage, etc.

V. Conclusion

This paper has described the utility and demonstrated the feasibility of early-phase time-based simulations, which constitute a key component of the design/trade tool developed for DARPA's F6 program. The main features of the DES model presented previously can be summarized as follows:

- **Unique scope:** both the development and the operations of a fractionated cluster have been modeled and are simulated, as both have an impact on the total value returned by the architecture during a given period. Various uncertainties relative to each lifecycle phase have also been modeled, along with possible actions to take (responses).
- **Scalability:** this feature was a critical requirement for the exploration of different fractionated architectures. This was achieved by building the model engine according to object-oriented principles, facilitating the creation of new model instances by simply adding new nodes to the architecture (spacecraft modules or ground stations) without requiring any structural change to the model engine. As a result, it was possible to generate a large variety of DES model instances corresponding to the many fractionated design alternatives in an automatic manner (from SysML rules), enabling the efficient and comprehensive exploration of the trade space.
- **Flexibility:** the model structure possesses sufficient modularity to easily change the types of risks and the possible associated responses without affecting the rest of the code. As a result, many futures that are mission-specific can be considered with ease.

Although the implementation of the F6 DES tool demonstrated the benefits of using time-based simulations for the design and exploration of the trade space of fractionated systems, these benefits also extend to the more traditional monolithic spacecraft. Generally, time-based simulations (and DES in particular) offer valuable opportunities when performed in the early stages of the design process that include:

- The ability to rapidly conduct “what if” analyses, and therefore to design the system to make it robust to the various uncertainties that can unfold in many different futures. Rather than only applying margins blindly,

both programmatic elements and technical parameters can be selected to minimize the effect of such uncertainties on the mission outcome.

- Providing further insight on the impact of residual risks that remain once the design has converged, beyond the qualitative assessment that is more common during preliminary studies. Specifically, once a risk has been identified (e.g., component not ready because of low TRL, risk of component failure, etc.), the effect of this risk can be directly simulated to provide a quantitative estimate of reduction in value return or in schedule reserves for example. (The impact of a failure on total value returned will vary if it occurs near the end of a mission vs. at the beginning).
- The ability to identify bottlenecks, thresholds or saturation points for data- and power-related processes (for example), in relation with design parameters. For example, time-based simulations can help determine how often the onboard memory might become full, for how long, and the impact on value returned.
- Simulating whole campaigns of multiple spacecraft (and not just a single mission), and evaluating the “business case” for various spacecraft lifetimes and replacement strategies.

The use of time-based simulations in early spacecraft design faces however several limitations:

- The runtime can be long, depending on the time resolution of the processes modeled. This can further limit the number of designs in the trade space that can be evaluated in a reasonable amount of time.
- Beyond the simple “what if” analyses that investigate the emergence of an event assumed to occur at a given time, a more thorough risk analysis relying on probabilistic methods would require the use of Monte-Carlo methods that could further increase the total runtime.
- Modeling inaccuracies can propagate throughout the simulation and be magnified over time.

Despite such limitations, time-based simulations provide unique information regarding the behavior of a space system under the presence of uncertainty, which can prove very valuable in the early stages of the design process.

Acknowledgements

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and funded through the DARPA’s F6 program. The authors would like to thank Owen Brown and Paul Eremenko for their support and other helpful discussions, Martin Feather for his suggestions and Tyler Ryan for his help in generating the data.

References

- [1] DARPA. [Online]. Available: http://www.darpa.mil/Our_Work/tto/Programs/System_F6.aspx. [Accessed 13 August 2012].
- [2] O. Brown, P. Eremenko and P. Collopy, “Value-centric Design Methodologies for Fractionated Spacecraft: Progress Summary from Phase 1 of the DARPA System F6 Program”, AIAA-2009-6540, in *AIAA Space 2009 Conference*.
- [3] C. G. Cassandras and S. Lafortune, *Introduction to Discrete Event Systems*, Norwell, Massachusetts, USA: Kluwer Academic Publishers, 1999.
- [4] K. Muller, “documentation for SimPy,” 2012. [Online]. Available: http://simpy.sourceforge.net/SimPy_Manual/index.html. [Accessed 12 07 2012].
- [5] Defense Advanced Research Projects Agency, “System F6 Broad Agency Announcement, DARPA-BAA-11-01,” 2010.