Risk and Performance Assessment of Generic Mission Architectures: Showcasing the Artemis Mission

Clemens M. Rumpf
NASA Ames Research Center & STC & USRA
Moffett Field, CA 94035, USA
+1-650-604-0371
clemens.rumpf@nasa.gov

Oscar Bjorkman
Science & Technology Corp.
Moffett Field, CA 94035, USA.
+1-650-224-9772
oscarb@berkeley.edu

Donovan Mathias
NASA Ames Research Center
Moffett Field, CA 94035, USA
+1-650-604-0836
donovan.mathias@nasa.gov

Abstract*—Recently, NASA has initiated a strong push to return astronauts to the lunar vicinity and surface. In this work, we assess performance and risk for proposed mission architectures using a new Mission Architecture Risk Assessment (MARA) tool. The MARA tool can produce statistics about the availability of components and overall performance of the mission considering potential failures of any of its components. In a Monte Carlo approach, the tool repeats the mission simulation multiple times while a random generator lets modules fail according to their failure rates. The results provide statistically meaningful insights into the overall performance of the chosen architecture. A given mission architecture can be freely replicated in the tool, with the mission timeline and basic characteristics of employed mission modules (habitats, rovers, power generation units, etc.) specified in a configuration file. Crucially, failure rates for each module need to be known or estimated. The tool performs an event-driven simulation of the mission and accounts for random failure events. Failed modules can be repaired, which takes crew time but restores operations. In addition to tracking individual modules, MARA can assess the availability of predefined functions throughout the mission. For instance, the function of resource collection would require a rover to collect the resources, a power generation unit to charge the rover, and a resource processing module. Together, the modules that are required for a given function are called a functional group. Similarly, we can assess how much crew time is available to achieve a mission benefit (e.g. research, building a base, etc) as opposed to spending crew time on repairs. Here we employ the method on the proposed NASA Artemis mission. Artemis aims to return United States astronauts to the lunar surface by 2024. Results provide insights into mission failure probabilities, up- and downtime for individual modules and crew-time resources spent on the repair of failed modules. The tool also allows us to tweak the mission architecture in order to find setups that produce more favorable mission performance. As such, the tool can be an aid in improving the mission architecture and enabling cost-benefit analysis for mission improvement.

TABLE OF CONTENTS

1. INTRODUCTION

The United States National Air and Space Administration (NASA) has recently renewed its commitment to send astronauts to the lunar surface as part of the Artemis campaign by 2024. To support this effort we have developed the Mission Architecture Risk Assessment (MARA) tool. A given campaign architecture may be handed to MARA, which simulates this campaign in a discrete time step simulation. At each time step, discrete events — such as the probabilistic failure of a mission-critical module, or the availability of new crew following a launch — are evaluated. Over the course of thousands of Monte Carlo (MC) runs, these simulations provide insight into the statistical performance of a given architecture. The eventual goal of the MARA tool is to enable architecture optimization by providing sensitivity studies of architecture performance to design choices. In this paper, we model a notional Artemis campaign to showcase current capabilities of the MARA tool.

The Artemis program aims to place US astronauts on the lunar surface in 2024. An essential component of the program is the "Lunar Orbiting Platform - Gateway" space station [1]. Astronauts will be staged there for extended stays and also to descent to and return from the Lunar surface. The Orion crew capsule is another key component. This spacecraft will be the primary means for astronauts to travel between the Earth and the Gateway station. A yet to be specified lunar lander is planned to provide lunar descent and ascent capabilities from the Gateway station.

Previously, similar simulations have been performed as part of the Lunar Surface Systems (LSS) project [2], [3] or Mars campaign architectures [4]. These simulations depended on GoldSim, a commercial software that runs only on the Windows operating systems. In the current effort, we developed the code using the open source Python programming language with its various modules. As such, MARA is indifferent to operating systems, can readily be improved through in-house development, and has no restrictions associated with commercial software products.

Assumed Artemis Timeline— The Artemis mission design remains in flux at the time of writing. As such, no certain information regarding the timeline and campaign profile are available. However, tentative information that is sufficient for our modelling purposes can be gathered from internet sources. We based the simulation design on a NASA website that showcases the Artemis program in its entirety [5]. According to that website, the official mission timeline is

summarized in Table 1.

Year	Artemis #	Description
2019	-	- First science/robotic deliveries using Commercial Lunar Payload Services (CLPS).
2020	1	- Unmanned flight test of the new Space Launch System (SLS) rocket, and Orion crew capsule.
2022	2	- Crewed flight of the SLS rocket and Orion crew capsule. Free return trajectory around the Moon.
2022	-	- A commercial launcher will deliver the first Gateway station element to space.
2023	-	- Delivery of a rover or similar mobility unit to the Lunar surface.
2023	-	- A second element will be added to the Gateway station.
2024	-	- Multiple launches will deliver the lunar descent/ascent vehicle to the Gateway station.
2024	3	- SLS and Orion will launch humans to Gateway who then proceed to land on the Moon.

Table 1. Publicly available information on Artemis program timeline. Some missions are on the critical path for crewed portions of the Artemis program and receive an Artemis mission number. Other missions have no mission number.

For the purpose of testing our tool and gathering preliminary results for validation, we created a mock mission manifest that mirrors the Artemis campaign from the start to the Artemis 3 mission. The manifest is summarized in Table 2. The need to create a mock flight plan arose because the simulation requires specific event times (day numbers) associated with each mission. MARA requires a detailed launch manifest that not only includes start times of missions but also the times of their returns or other milestones. In addition, the manifest holds information about the crew number and modules associated with each mission.

Day	#	Type	Modules	Crew
0	1	Launch	SLS, Orion, ESM	0
14	1	Return	Orion, ESM	0
365	2	Launch	CLV, PPE	0
730	2	Launch	SLS, Orion, ESM	4
740	2	Return	Orion, ESM	4
911	3	Launch	MHM, CLV	0
1308	4	Launch	CLV, LL	0
1430	5	Launch	SLS, Orion, ESM	4
1444	5	Descent	LL	4
1449	5	Ascent	LL	4
1459	5	Return	Orion, ESM	4

Table 2. Assumed Artemis program schedule based on currently available information.

In the mock manifest, missions 1, 2, and 5 correspond to Artemis 1, 2, and 3 as publicly defined [5] (see Table 1). Artemis 3 (# 5 in our scenario) is the mission that aims to land astronauts on the Moon. The remaining missions are general test and infrastructure buildup missions.

2. METHOD

Implementation

The MARA tool was developed entirely in Python 3.7. We utilized the SimPy library to set up the event-based simulation.

SimPy—SimPy is a discrete event simulation library [6]. This section explains some of the core elements that drive a SimPy simulation. Several concepts are key for SimPy simulations:

Environment: Each SimPy simulation runs within a simulation *environment* that is created at the beginning of each simulation. The *enviroment* is responsible for running the simulation and coordinating discrete events on the timeline when individual *processes* interact with each other.

Process: A *process* is a function that describes the behaviour of a component that is modelled within the simulation. For example, in a simulation that models the interaction of a driver and its car, both the driver and the car could sensibly each be modelled as a *process*. In our case, multiple modules (spacecraft or parts thereof) are part of the Artemis program. Each of these modules has a process that models its function and breakage behaviour. *Processes* can trigger *events* that can have consequences for other simulation components (for example spacecraft modules).

Resource: *Resources* are objects of limited capacity that can be assigned to *processes*. In our example, the only resource was a crew of four that was occasionally launched to space and became available to repair broken modules, conduct science, or perform maintenance tasks. If more than one module was broken at a given time, the time of an equivalent number of crew would be taken up to repair broken modules. If there was not enough crew available to repair all broken modules, a broken module would have to wait until one crew became available again.

Events: *Events* tell the simulation *environment* that a discrete milestone was reached. In our example, this might be the failure of a module based on a randomly generated trigger. Depending on the program, this could mean a change in behaviour for other modules as well.

The interested reader is referred to the SimPy website for practical examples of small SimPy simulations [6].

Input Files—The Python simulation requires three input files — a configuration file, a module file, and a manifest file. All files use the CSV (comma-separated values) format in plain text. Campaign details can be modified without the need to write new code.

Configuration File: The configuration file holds parameters for the simulation setup (e.g., number of Monte Carlo runs to perform in each analysis, the simulated mission time, if debugging messages should be displayed, etc.).

Manifest File: The manifest file describes the entire campaign of missions for the simulation. Each line in this file holds information such as start time of the mission, the names of the modules involved in this mission, number of crew launched or returned, amongst other variables.

Module File: This file holds information about the modules that are simulated in the campaign. Crucially, the failure rates are specified in this file.

Module Groups: We were interested in the times when certain mission-relevant functions could be performed. This file defined functions such as *habitation* and *lunar landing* and

listed the modules that needed to be operational in order for that function to be available.

Simulation Setup

SimPy requires a constant time step size with which it steps through the simulation to evaluate discrete events that may occur at each time step. This time step was set to correspond to 1 day in our simulation. A one day time step was chosen as a suitable compromise between speed of the simulation and sufficient resolution to capture the campaign timeline.

To increase computational speed, the MARA tool utilizes the Python's multiprocessing library and runs individual Monte Carlo (MC) runs in parallel. The results of each MC run are collected and combined after the MC runs terminated for result evaluation.

Modules

The Artemis mission as simulated in this work utilized the modules briefly described below. The corresponding failure rates are listed in Table 3. Launcher failure rates were assumed to be similar to those presented in the literature [7]. Note that the values presented in this table aim to be realistic but do not represent actual failure rates of the listed systems. The modules typically (except for launchers) have two failure rates. The operational failure rate describes the statistical chance of module failure while the module is in use. The passivated failure rate is applicable to those times when the module is not in active use such as a rover sitting on the lunar surface with no crew present.

PPE: The Power and Propulsion Element provides energy and station keeping capabilities to the Gateway space station.

MHM: The Minimal Habitation Module is attached to the Gateway space station. It is the first Gateway module to support human life there.

SLS: The Space Launch System is the United States flagship heavy lift launch vehicle and is planned to take an instrumental role in the Artemis program, lifting Gateway elements and launching astronauts on Lunar missions.

CLV: These are commercial launch vehicles. Some of the Gateway and Lunar surface elements are slated to be launched by them. The vehicles remain unspecified at the time of writing.

Orion: The Orion crew capsule is the space craft to transport crew to and from Gateway and for atmospheric re-entry.

ESM: The European Service Module provides power and propulsion to Orion.

LL: The Lunar Lander will carry astronauts to the surface of the Moon and return them to low lunar orbit.

Timeline and Delays—MARA replicates the timeline provided in the campaign manifest file (see Table 2). However, any mission that requires a launch from the Earth can delay the overall timeline. The delay was calculated based on a normal distribution with a mean of 180 days and a standard deviation of 30 days. The rationale for these delays is that a failed rocket launch will trigger an investigation into the root cause of the failure that traditionally takes several months. During the investigation that particular launch vehicle will be grounded. Switching to an alternative launch vehicle is similarly time consuming (or even impractical) as payloads are designed against the interface requirements of the intended launcher and its vibrational and acoustic launch environment.

Module	**Failure Rate** $\left[\frac{1}{\text{day}}\right]$ (operational/passivated)
PPE	$2.592 \times 10^{-2}/4.008 \times 10^{-5}$
MHM	$2.592 \times 10^{-2}/4.008 \times 10^{-5}$
SLS	$1 \times 10^{-2}*$
CLV	$1 \times 10^{-2}*$
Orion	$3.816 \times 10^{-3}/1.9344 \times 10^{-4}$
LL	$3.816 \times 10^{-3}/1.9344 \times 10^{-4}$

Table 3. Artemis campaign modules and their failure rates as used for this work. Failure rates labeled with asterisks (*) denote launcher failure rates and are interpreted as failure chance per launch.

A switch would require re-analysis of launcher requirement conformity at a minimum and, more likely, time-consuming re-design work.

Broken Modules and Their Repair—Individual modules that are *operational* or *passivated* may break at any time step during the simulation. The failure rates specified in the module file determine the probability of such an occurrence on a given day. In the event of a failure, the module's status is switched from *operational* or *sleeping* to *broken*. It remains broken until crew becomes available to fix it. A broken module will count against the availability of functions such as *lunar landing* if that module is required for this function. Crew occupied with the repair of broken modules cannot conduct lower priority tasks, such as research. A broken module thus reduces the anticipated benefit of a mission. The repair time for any module was set to 5 days in this analysis.

Simulation Flowchart—Figure 1 visualizes MARA's simulation process. The simulation itself runs in a continuous loop that is only ended when the simulation end time is reached. Based on the data in the manifest file, new modules and crew are either introduced into the simulation or removed from it.

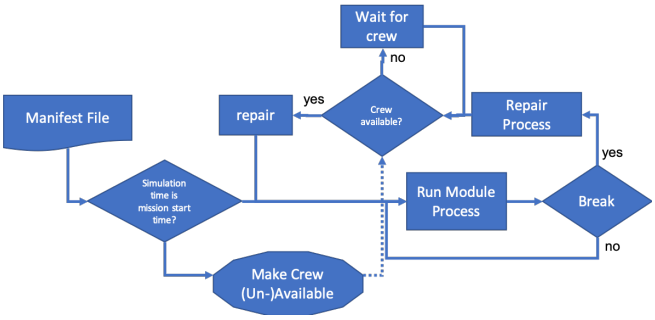


Figure 1. Simplified simulation flowchart for the Artemis simulation.

Performance Metrics

We use the following risk and performance characteristics to judge whether a given architecture performs better or worse than the baseline setup.

Delay Time—The MARA tool tracks delay times that accrue due to launch failures as mentioned in section 2. Each launch failure adds to the total delay of the program manifest. It can be expected that a better performing architecture yields less delays.

Science Crew Time—Modules fail and need to be repaired. Per failed module, one crew was assigned for 5 days to repair the module. This time could not be spent on mission beneficial tasks such as science. A robust program architecture minimizes the crew time spent on repairs and maximizes crew time spent on research.

Functional Group Availability Time—Two functional groups were defined for this analysis.

Habitation: The function of habitation was given when the MHM and PPE modules were available (not broken). This enabled crew to live aboard the Lunar Gateway station without the immediate need to return to the Earth.

Lunar Landing: This is the main objective of the Artemis program. A lunar landing is a complex undertaking under the Artemis program architecture. To be available, the following modules needed to be available: LL, PPE, MHM, Orion, and ESM.

3. RESULTS

The MARA tool was used to assess the performance of the Artemis baseline architecture as defined in Section 1. In a next step, we alter few program parameters and assess how these changes affect performance metrics. Each analysis consists of 10 000 MC runs and the presented statistics are based on these results.

Baseline Results

Delays—MARA calculated a 6.3 % chance that the baseline Artemis program will experience delays. In case of a delay, the mean expected delay is 182.1 days with a standard deviation of 52.5 days.

Figure 2 visualizes the delay simulation results. The grey lines in the background show the results from the individual MC runs over the entire simulation duration in years on the x-axis. A jump of a line to a new y-axis coordinate signifies that a delay occurred (usually launch failure). The remainder of that simulation will be delayed by the number of days marked on the y-axis. Some simulations experience more than one delay, which is represented by several jumps in a delay line. The mean delay shown in Figure 2 accounts for all simulation runs and is therefore low with 10.2 days. On the extremes, 99 % of delayed mission experienced delays longer than 6.3 days and shorter than 384.7 days.

Crew Time—The crew is on station for an average of 44.5 days with a standard deviation of 26.6 days. In general, four crew are present in space during those days for a total average of 171.2 crew-days. However, one crew was always assigned to maintenance tasks and other crew conducted repairs when necessary. Thus, the total number of crew-days available for non-repair and maintenance related tasks was 114.7 days with a standard deviation of 75.0 days. Figure 3 visualizes the availability of the crew over the duration of the Artemis program. The two spikes at about 2 and 4 years correspond to the two crewed launches in the manifest. Some of the grey lines representing individual simulations extend significantly beyond the scheduled crew availability times. Those occurrences correspond to delayed missions that make crew available later than anticipated. The concentration of available crew after the scheduled launch times is due to mission delays that occur mainly from launch failures with a mean delay of 180 days. While most of the missions are carried out as intended, the delayed missions produce a minor crew availability spike about half a year after the scheduled mission time.

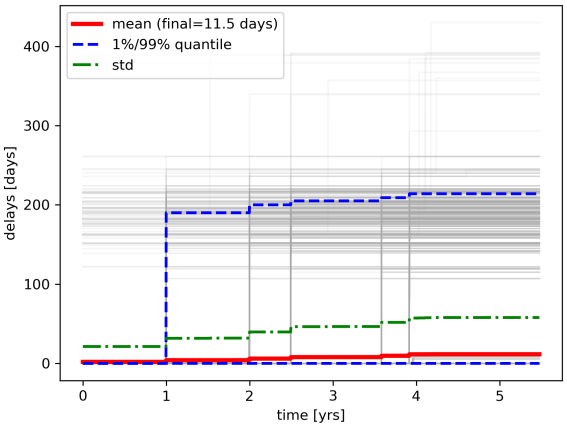


Figure 2. The delays encountered due to launch failures over the Artemis program timeline. Light grey lines show individual simulation results while the thicker lines show mean, standard deviation and 1/99% numbers.

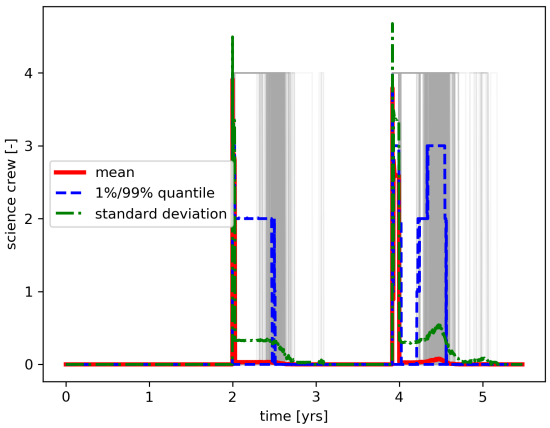


Figure 3. The free crew over time. These crew members are available for science tasks or infrastructure buildup tasks. They are the crew that are not occupied by safety critical tasks such as maintenance and module repair work.

Functional Groups—The functional group of habitation required the availability of the MHM and PPE module. This condition was met for an average of 34.3 days with a standard deviation of 24.0 days. In 99 % of the cases, habitation was available for more than 8 days and less than 134 days. Breaking modules are the cause for a reduced habitation availability duration. When crew became available, the function could be restored by repairing broken modules over the course of 5 days per repair. This is the reason why the function of habitation is available on average for a shorter time than the average time of crew in space. Figure 4 visualizes the timeline of habitation functional group availability.

Lunar landing functionality was given when modules LL, PPE, MHM, Orion, and ESM were available. This condition was met for an average of 19.0 days with a standard deviation

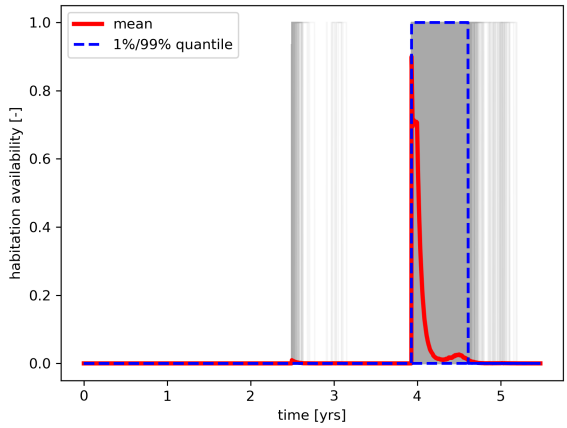


Figure 4. Habitation functional group availability. A value of 1 means full availability and 0 means no availability. The mean value across all MC runs may take any value between $[1, 0]$.

of 14.2 days. In 99 % of the cases, lunar landing was available for more than 6 days and less than 26 days. Figure 5 visualizes the timeline of habitation functional group availability.

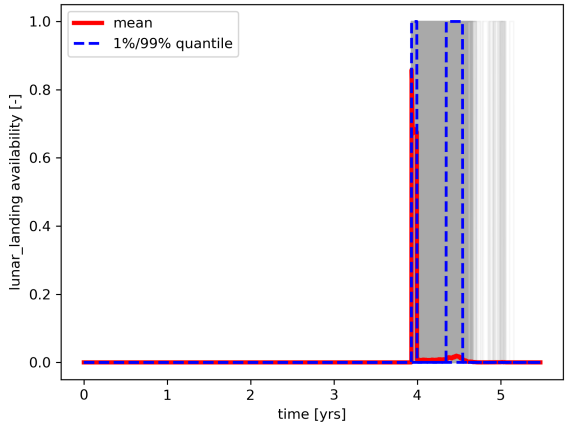


Figure 5. Lunar landing functional group availability. A value of 1 means full availability and 0 means no availability. The mean value across all MC runs may take any value between $[0, 1]$.

Given that lunar landing is the primary goal of the Artemis campaign, a more thorough look at the lunar landing availability outcome is warranted. Figure 6 shows a histogram of observed lunar landing durations. The distribution contains markings for statistical parameters such as mean, and median, as well as for the 1 and 99 percentile values. Few extreme cases allow for unplanned and potentially unrealistic scenarios in which lunar landing is available for 91 to 296 days. Only 0.99 % of cases exhibited a lunar landing availability longer than 91 days. These rare cases skew the results such that the 99 percentile value lies outside the values shown in this distribution range. They also shift the mean by one day compared to the median and bias it towards a longer duration. With this observation, it might be prudent for mission planners to rely on the median value of 18 days for the most likely lunar landing duration.

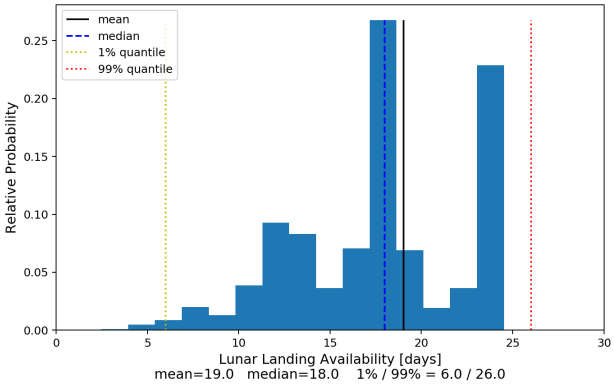


Figure 6. Histogram showing the duration distribution for which the functional group of lunar landing is available. The histogram does not show potentially unrealistic scenarios with lunar landing availability durations between 91 and 296 days which made up only 0.99 % of runs.

Alternative Mission Scenario Results

MARA enables assessment of alternative mission scenarios. To showcase this capability, the original mission scenario has been altered. Instead of two short crew stays at the Lunar Gateway space station (mission # 2 and 5 in Table 2), one extended mission is flown. In practical terms, we removed the first return mission that would usually return the crew to Earth (at time 740 in Table 2) and the launch mission that sends the second crew to space (at time 1430 in Table 2).

This change makes the crew available for an extended period and allows continued maintenance of modules prone to failure. Specifically, the PPE module statistically fails every 39 days. In the original mission design this means that the crew would almost certainly be occupied repairing this module as soon as they arrive on orbit. The repair takes up a larger portion of the short on-orbit time. On average, the crew was on orbit for 44.5 days with 114.7 crew-days available for science. Each day an average of $\frac{114.7}{44.5} = 2.58$ crew was available for science.

In contrast, the longer duration crewed mission had crew on orbit for 734.7 days with 2026.6 crew-days available for science. That is a science crew per day ratio of $\frac{2026.6}{734.7} = 2.76$ crew corresponding to a 7.0 % increase in productivity. Figure 7 visualizes the available crew over time. The longer crew time on orbit allows for available crew numbers to stabilize while dealing with maintenance and the occasional module failure.

Similarly, a longer crew presence allows the operations of modules in conjunction with crew to reach steady state. This is visualized in the functional availability of habitation shown in Figure 8.

Given that the PPE module needed 17.6 repairs and the MHM module 12.6 repairs on average while crew was on orbit, habitation had a steady availability expectation of about 75 % during that time. Steady habitation availability is desirable as this function is crucial for the survival of crew in space. Furthermore, it means that module failures can be addressed

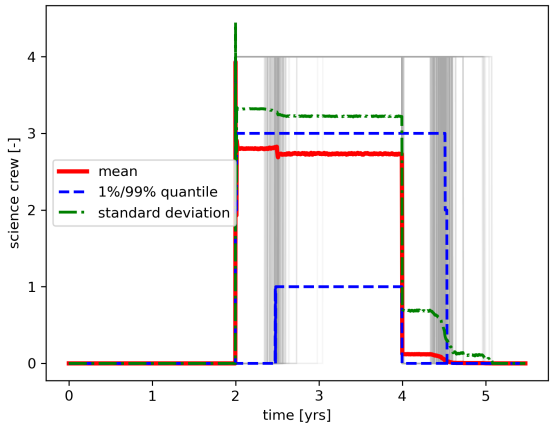


Figure 7. Free crew over the mission time in the alternative program design.

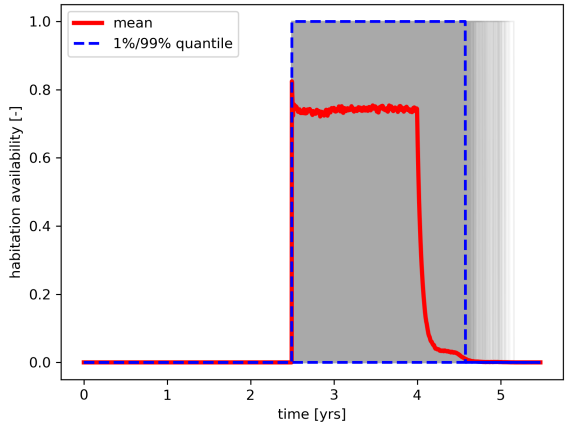


Figure 8. Availability of habitation function in the alternative program design.

promptly as opposed to a situation when an unattended module fails in space. An unattended, failed module might worsen the general program situation in space. For example a failed Gateway station could drift away from its operational module, exasperating the need to repair the module with the need to move it back into place. A permanent crew presence allows to correct for abnormal situations quickly without risking situations cascading further into undesirable conditions.

4. Summary and Conclusions

A new space mission architecture tool, called MARA has been developed. It can simulate a predefined space program manifest while taking into account the failure rates of its constituent parts. To showcase the tool, we modeled a plausible timeline for the Artemis program that follows the sparse information available to the public. The simulation results presented here are based on 10 000 Monte Carlo runs for each simulated manifest.

Given our failure rate numbers, the manifest we modelled faces a 6.3 % chance of experiencing a delay until the Moon landing in 2024. In the case of a delay, the expected delay time would be about half a year (182.1 days) and almost certainly less than 384.7 days (99 % confidence interval).

Furthermore, MARA's capability to predict the availability of predefined functions has been shown. The functions of habitation on the Lunar Gateway space station and the function of lunar landing were investigated. Results show significant variability in the availability duration for these functions, which might be the cause for concern for mission planners.

Finally, MARA may be used to compare alternative program manifests with the current baseline. Comparisons can be useful to terse out mission architecture choices that improve program robustness and/or program performance. Here, we modified the baseline mission to feature one extended crew presence at the Lunar Gateway with eventual visit to the lunar surface instead of two short stays at the Lunar Gateway. Results showed that a 7.0 % increase in crew productivity could be achievable with such a measure. It also allowed for the operations of modules and crew to reach steady state.

5. Future Work

MARA is still in prototype stage. Next steps include further development to the software design in order to increase its applicability to the largest variety of space missions possible. Some parameters such as the crew capacity on orbit are currently fixed. In the future, it should be possible to dynamically change the number of available crew.

Over the coming months and years, the design of the Artemis program should become more concrete. We are interested in updating the current manifest information to reflect higher fidelity information, thereby increasing the relevancy of the results produced with MARA.

Acknowledgments

Support for this work was provided by the NASA Postdoctoral Program (NPP) as well as Science and Technology Corporation (STC).

References

[1] J. C. Crusan, R. M. Smith, D. A. Craig, J. M. Caram, J. Guidi, M. Gates, J. M. Krezel, and N. B. Herrmann, "Deep space gateway concept: Extending human presence into cislunar space," *IEEE Aerospace Conference Proceedings*, vol. 2018-March, pp. 1–10, 2018.

[2] S. Go, D. Mathias, H. Nejad, and F. Thomson, "Integrated Risk Assessment for Lunar Surface Systems," NASA, Tech. Rep., 2009.

[3] S. Go, D. L. Mathias, F. Thomson, and B. Ramamurthy, "Mass-Constrained Availability for Lunar Exploration," in *10th International Probabilistic Safety Assessment and Management Conference.*, 2010.

[4] T. A. Manning, H. Nejad, and C. Mattenberger, "Near-Earth phase risk comparison of human Mars campaign architectures," in *Proceedings - Annual Reliability and Maintainability Symposium*, 2013.

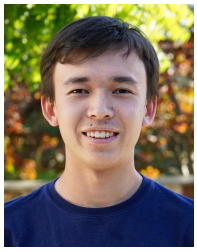
[5] NASA, "Explore Moon to Mars," 2019. [Online]. Available: https://www.nasa.gov/specials/moon2mars

[6] O. Lunsdorf and S. Scherfke, "SimPy," 2013. [Online]. Available: https://simpy.readthedocs.io/en/latest/index.html

[7] S. A. Motiwala, D. L. Mathias, and C. J. Mattenberger, "Conceptual Launch Vehicle and Spacecract Design for Risk Assessment," NASA, Tech. Rep. June, 2014.

BIOGRAPHY

Clemens Rumpf *received his Dipl.-Ing. degree in aerospace engineering from TU Braunschweig in 2012 and his Ph.D. in aerospace engineering from the University of Southampton in 2017. Following a NASA Postdoctoral Program fellowship, he is now working as an aerospace engineer at NASA Ames Research Center. In addition to mission analysis, his work covers asteroid impact risk assessment and detecting meteors in weather satellite data as part of the ATAP project. Previously, he has worked as a navigation engineer at the German Aerospace Center (DLR) and as a space system engineer at the European Space Agency (ESA).*



Oscar Bjorkman *currently studies Computer Science at the University of California, Berkeley. He graduated in 2019 from Los Altos High School in Los Altos, California. The summers of 2018 and 2019 he interned in the NASA Advanced Supercomputing Division at NASA Ames Research Center. There he researched future unmanned flight initiatives and used Python to perform engineering risk assessment for use of autonomous drones. He created a GUI using Electron visualizing the risk assessment of flights over maps and developed a utility for assessing the viability of a new lunar mission plan.*

Donovan Mathias *currently serves as the Deputy Chief of the NASA Advanced Supercomputing Division at NASA Ames Research Center. Donovan's work combines physical and system models for improved risk assessment in areas of human spaceflight, sample return missions, urban air mobility, and asteroid impact scenarios. Donovan received B.S. and M.S. degrees in Aeronautical Engineering from California Polytechnic State University, San Luis Obispo and a Ph.D. in Aeronautics and Astronautics from Stanford University.*