Methods for Determining the Level of Autonomy to Design into a Human Spaceflight Vehicle: A Function Specific Approach

Ryan W. Proud*, Jeremy J. Hart†, and Richard B. Mrozinski#

NASA Johnson Space Center
2101 NASA Road 1, Houston, Texas, 77058, USA

ABSTRACT

The next-generation human spaceflight vehicle is in a unique position to realize the benefits of more than thirty years of technological advancements since the Space Shuttle was designed. Computer enhancements, the emergence of highly reliable decision-making algorithms, and an emphasis on efficiency make an increased use of autonomous systems highly likely. NASA is in a position to take advantage of these advances and apply them to the human spaceflight environment. One of the key paradigm shifts will be the shift, where appropriate, of monitoring, option development, decision-making, and execution responsibility from humans to an Autonomous Flight Management (AFM) system. As an effort to reduce risk for development of an AFM system, NASA engineers are developing a prototype to prove the utility of previously untested autonomy concepts. This prototype, called SMART (Spacecraft Mission Assessment and Re-planning Tool), is a functionally decomposed flight management system with an appropriate level of autonomy for each of its functions. As the development of SMART began, the most important and most often asked question was, "How autonomous should an AFM system be?" A thorough study of the literature through 2002 surrounding autonomous systems has not yielded a standard method for designing a level of autonomy into either a crewed vehicle or an uncrewed vehicle. The current focus in the literature on defining autonomy is centered on developing IQ tests for built systems. The literature that was analyzed assumes that the goal of all systems is to strive for complete autonomy from human intervention, rather than identifying how autonomous each function within the system should have been. In contrast, the SMART team developed a method for determining the appropriate level of autonomy to be designed into each function within a system. This paper summarizes the development of the Level of Autonomy Assessment Tool and its application to the SMART project. The conclusion from this on-going effort demonstrates that the Level of Autonomy Assessment Tool is a viable method for determining the appropriate level of autonomy for each function within a system.

KEYWORDS:

autonomy, autonomous, trajectory management, flight operations, mission operations, design phase, human-computer interaction, AFM, OODA, intelligent systems, decision-making, spacecraft

1. INTRODUCTION

Many system designers have long sought after the goal of complete autonomy. The common question that has been asked is, "Is my system smarter than yours?", rather than asking, "How autonomous should the system be?" Therefore, many tests exist to address the former question. Obstacle courses test mobile systems. Strategy games test intelligent systems. The Turing Test is used to determine if the testee is truly a human or a computer system. Each of these tests appears designed to test one, or a few, specific parameters after the system has been built. No tests have been found to determine how autonomous the system's functions should be during the design phase.

The question, "How autonomous should the system be?" is of primary importance to the designers of the next generation human spaceflight vehicle. Increased autonomy levels cost money and time during the design phase. Though, if implemented correctly, they increase safety and efficiency during the operations phase, which may lead to decreased overall lifecycle costs. The goal of the designers is to find the optimum level of autonomy that minimizes cost, maximizes safety, maximizes efficiency, can be completed on time, and maintains complete trust from its operators. This is not a new problem, but it can be solved in new ways for future spacecraft.

During the Apollo Program, NASA chose to put most of the decision-making power on the ground in human flight controllers' hands. At the time, computers were generally slow, they did not have enough memory, and there was not enough time to advance the technology before President Kennedy's end of the 1960's deadline. Therefore, humans

* Ascent Analyst, Mission Operations Directorate, Flight Design and Dynamics Division, NASA-Johnson Space Center, DM4, Houston, TX 77058
† GN&C Design and Analysis Engineer, Engineering Directorate, Aerosciences and Flight Mechanics Division, NASA-Johnson Space Center, EG4, Houston, TX 77058
# Descent Analyst and Space Shuttle Ascent Flight Controller, Mission Operations Directorate, Flight Design and Dynamics Division, NASA-Johnson Space Center, DM4, Houston, TX 77058

made the decisions whenever re-planning was necessary, and the astronauts had to rely on the ground-based flight controllers for help when problems arose, as in Apollo 13.

The era of the Space Shuttle presented a different paradigm. There was not a publicly mandated time deadline for the first launch. There was, however, a serious effort to minimize costs. Thus, when the choice had to be made for where to put the decision-making power, the answer was simple. Any change to the way NASA currently flew would cost too much money, and computer technology had not advanced enough to allow significant onboard processing capability. The decision was ultimately made to keep most of the decision-making ability in the hands of the ground flight controllers.

The next-generation human spaceflight vehicle has to answer the same question. However, computer advancements, the emergence of highly reliable decision-making algorithms, and the emphasis on efficiency make an increased use of autonomous systems very likely. SMART proposes to utilize these changes in the human spaceflight environment to shift responsibility from humans to computers, where appropriate. In order to determine where the shift from humans to computers is appropriate, The Level of Autonomy (LOA) Assessment Tool was developed. The LOA Assessment Tool seeks to optimize the specific level of autonomy for each function within an AFM based on a "common sense" method.

2. AN AUTONOMOUS FLIGHT MANAGEMENT SYSTEM

An AFM system is necessary for the next-generation human spaceflight vehicle to meet cost, safety, and mission requirements. In order to address how autonomous each of the AFM's functions should be, it is important to use an existing design program as a credible testbed. NASA's Orbital Space Plane (OSP) must incorporate some autonomy to launch, rendezvous, and land without human intervention in some contingency cases. Development of one AFM concept for the OSP is being performed by the SMART team.

With a crew onboard and flight controllers on the ground, many people question the need for increased autonomy. The argument is that with only a few launches per year, and the short duration of any flight, the human team can handle most of the in-flight problems. What this paradigm fails to acknowledge is three-fold. First, many of the most critical decisions that ensure crew and vehicle survival must be made in split seconds, where a computer might be safer, faster, and/or better suited to make a decision. Second, this ignores the OSP design reference missions to launch and rendezvous an uncrewed emergency rescue vehicle. This requirement drives the need for autonomous capability. Third, the argument assumes that the vehicles that fly today are not autonomous at all. This flawed assumption raises the need for a standard definition of what autonomy is and how to measure it.

3. MODELING AUTONOMOUS SYSTEMS

When defining how autonomous SMART should be, the team had different ideas of what the definition of autonomy is and how to measure it. As discussions progressed it was evident that a method for measuring autonomy was needed in order to proceed with the SMART prototype effort. Research into the autonomy community did not produce a viable solution for use during the design phase. The next few paragraphs summarize a few of the sources that were consulted, how they influenced the development to date, and the progress that this team has made towards developing the method.

The research began with Sheridan's *Telerobotics, Automation, and Human Supervisory Control* [5]. Many of the recent autonomy articles use this as a reference for an initial understanding of how humans and computers interact. Many of Sheridan's examples focus on telerobotics where the human is physically separated from the system but still issuing commands. While this does not exactly correlate with a crewed spaceflight vehicle scenario, it does address the role of ground-based flight controllers in the decision-making process. The most relevant information comes from Sheridan's trust development issues, such as reliability, robustness, familiarity, usefulness, and dependence. While these issues were not molded into a unified theory to determine if an autonomous system should be trusted, they are many of the central issues that define how autonomous a vehicle should be.

In addition, Sheridan proposes a 10 level scale of degrees of automation as seen in Table 1. This scale could be re-labeled as Sheridan's Levels of Autonomous Decision-Making and Execution. Clearly, Levels 2 through 4 are centered on who makes the decisions, the human or the computer. Levels 5-9 are centered on how to execute that decision. Levels 1 and 10 are appropriate bounds for either issue. The team sought to find if others had made the same functional split between decision-making and execution of a decision.

1)	The computer offers no assistance, human must do it all.
2)	The computer offers a complete set of action alternatives, and
3)	narrows the selection down to a few, or
4)	suggests one, and
5)	executes that suggestion if the human approves, or
6)	allows the human a restricted time to veto before automatic execution, or
7)	executes automatically, then necessarily informs the human, or
8)	informs him after execution only if he asks, or
9)	informs him after execution if it, the computer, decides to.
10)	The computer decides everything and acts autonomously, ignoring the human.

Table 1. Sheridan's scale of degrees of automation [5]

In 2000, Parasuraman, et al. provided a revised model for the levels of automation with *A Model for Types and Levels of Human Interaction with Automation* [4]. This model split the tasks that any human or system would ever have to perform into four categories: information acquisition, information analysis, decision and action selection, and action implementation. Information acquisition is the task of sensing, monitoring, and bringing information to a human's attention. Information analysis is performing all of the processing, predictions, and general analysis tasks. Decision and action selection result in making choices. For example, "Based on the available analysis, what should the system do?" Action implementation is acting on decisions or commanding new actions. Levels in this category include the computer asking for authority to proceed and allowing human overrides.

Internal analysis determined that the AFM functions fell into a similar four-tier system utilizing the terms monitor, analyze, decide, and act. Realization of the similarity between the two systems shifted the research into finding similar four-tiered systems. While Parasuraman, et al. were focused on the computer's process of decision-making, a similar four-tier system based solely on the human process of decision-making was found. Boyd's OODA (observe, orient, decide, and act) Loop [1], seen in Figure 1, was developed by the military to attempt to disrupt enemy decision-making processes.

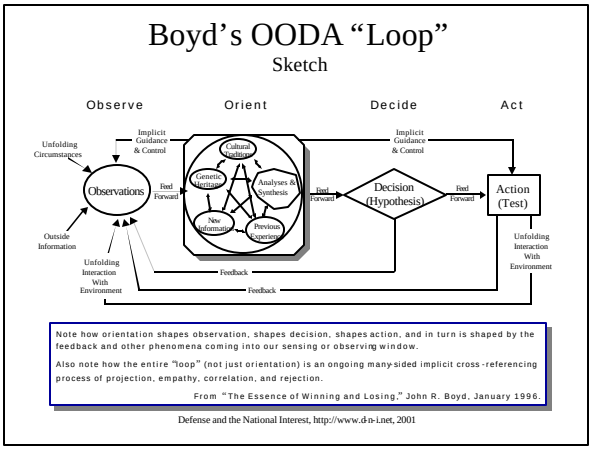

Figure 1. Boyd's OODA Loop [1]

Boyd's system adds two important characteristics to the four-tiered system, feedback and implicit control. Feedback is the concept that decisions do not necessarily have to become actions. Decisions themselves can spark new analysis tasks or requests for new observations. Implicit control is the process that runs in the background. For a human spaceflight vehicle the background trajectory process is GN&C (Guidance, Navigation, and Control). This process has implicit control of the vehicle and will continue to carry out its programmed commands for the vehicle unless instructed otherwise by a human or autonomous manager.

In a perfectly normal situation, SMART would never have to act, as the GN&C system would perform as expected until there is an off-nominal current or predicted future observation. It is only in this context that SMART would have to perform a complete OODA loop and send the act commands to the relevant parts of the vehicle. In order to capture these additional aspects of a decision-making system, the team chose to use the OODA terminology for the level of autonomy functional types. Using these types enables the ability to break down known functions and to determine how autonomous to design each one.

4. LEVEL OF AUTONOMY ASSESSMENT SCALES

After determining *how* to categorize the function types, the next question became how to determine the *level* of autonomy of a particular function. Based on research, including Clough's Unmanned Aerospace Vehicle studies [2 and 3], and the desire to rank each function type individually, an 8-level scale of autonomy for each OODA category, Table 2, was developed.

The level of autonomy scales are bounded by Levels 1 and 8, which correspond to complete human and complete computer responsibility, respectively (similar to Sheridan, Table 1). The team tailored each level of autonomy scale to

fit the tasks encompassed by function type (Observe, Orient, Decide, or Act). For example, the levels in the "Observe" column refer to gathering, monitoring, and filtering data; the levels in the "Orient" column refer to deriving a list of options through analysis, trend prediction, interpretation and integration; the levels in the "Decide" column refer to decision-making based on ranking available options; and the levels in the "Act" column refer to execution or authority to act on the chosen option.

The team ensured that individual levels in each scale are relatively consistent in magnitude of change between levels across the function types. Generally, the levels of autonomy can be broken down into three sections. In Levels 1-2, the human is primary and the computer is secondary. In Levels 3-5, the computer operates with human interaction. In Levels 6-

Level	Observe	Orient	Decide	Act
8	The computer gathers, filters, and prioritizes data without displaying any information to the human.	The computer predicts, interprets, and integrates data into a result which is not displayed to the human.	The computer performs ranking tasks. The computer performs final ranking, but does not display results to the human.	Computer executes automatically and does not allow any human interaction.
7	The computer gathers, filters, and prioritizes data without displaying any information to the human. Though, a "program functioning" flag is displayed.	The computer anlayzes, predicts, interprets, and integrates data into a result which is only displayed to the human if result fits programmed context (context dependant summaries).	The computer performs ranking tasks. The computer performs final ranking and displays a reduced set of ranked options without displaying "why" decisions were made to the human.	Computer executes automatically and only informs the human if required by context. It allows for override ability after execution. Human is shadow for contingencies.
6	The computer gathers, filters, and prioritizes information displayed to the human.	The computer overlays predictions with analysis and interprets the data. The human is shown all results.	The computer performs ranking tasks and displays a reduced set of ranked options while displaying "why" decisions were made to the human.	Computer executes automatically, informs the human, and allows for override ability after execution. Human is shadow for contingencies.
5	The computer is responsible for gathering the information for the human, but it only displays non-prioritized, filtered information.	The computer overlays predictions with analysis and interprets the data. The human shadows the interpretation for contingencies.	The computer performs ranking tasks. All results, including "why" decisions were made, are displayed to the human.	Computer allows the human a context-dependant restricted time to veto before execution. Human shadows for contingencies.
4	The computer is responsible for gathering the information for the human and for displaying all information, but it highlights the non-prioritized, relevant information for the user.	The computer analyzes the data and makes predictions, though the human is responsible for interpretation of the data.	Both human and computer perform ranking tasks, the results from the computer are considered prime.	Computer allows the human a pre-programmed restricted time to veto before execution. Human shadows for contingencies.
3	The computer is responsible for gathering and displaying unfiltered, unprioritized information for the human. The human still is the prime monitor for all information.	Computer is the prime source of analysis and predictions, with human shadow for contingencies. The human is responsible for interpretation of the data.	Both human and computer perform ranking tasks, the results from the human are considered prime.	Computer executes decision after human approval. Human shadows for contingencies.
2	Human is the prime source for gathering and monitoring all data, with computer shadow for emergencies.	Human is the prime source of analysis and predictions, with computer shadow for contingencies. The human is responsible for interpretation of the data.	The human performs all ranking tasks, but the computer can be used as a tool for assistance.	Human is the prime source of execution, with computer shadow for contingencies.
1	Human is the only source for gathering and monitoring (defined as filtering, prioritizing and understanding) all data.	Human is responsible for analyzing all data, making predictions, and interpretation of the data.	The computer does not assist in or perform ranking tasks. Human must do it all.	Human alone can execute decision.

Table 2. Level of Autonomy Assessment Scale

8, the computer operates independently of the human and the human has decreasing access to information and decreasing override capability. Understanding the differences between the levels is critical to interpreting them correctly. To understand a particular autonomy level requires referencing the entire scale to see how each level is different from the next, rather than focusing solely on a particular level.

The Level of Autonomy Assessment Scales were derived from a mix of research into external autonomy applications (Sheridan, Parasuraman, etc.) and the team's internal initial scale development. The intent of the scales is to help system designers easily and correctly identify the level of autonomy to design each function within their system. They are available for either identifying the level of autonomy of an existing function or for proposing an appropriate level of autonomy during the design of a new system. The OODA category aspect of this scale is advantageous because: 1) it allows more specific verbal description of the level of autonomy of a specific function than previous scales, and 2) it allows the function types to be weighted differently across a particular level. The second point is important to understanding the scale as a whole. A 5 in the Act column does not have the same costs, training requirements, or other assumptions as a 5 in the Orient column.

5. MAPPING LEVEL OF AUTONOMY ONTO A VEHICLE DESIGN

Once the team developed the Level of Autonomy (LOA) Assessment Scales, a method to map particular functions onto the scale was needed. Therefore, a common sense method for system designers to determine how autonomous each function should be was needed.

There is debate over the trustworthiness of any autonomous system. In a field similar to human spaceflight, a new aircraft may or may not be considered viable depending on an individual test pilot's subjective assessment. The aircraft testing community developed the Cooper-Harper scale and questionnaire to attempt to provide rationale for their subjective assessments of aircraft handling qualities. The logical extension of this process to spaceflight is to develop a similar scale and questionnaire to help determine how autonomous AFM functions should be.

The initial assumption is that the system designers, programmers, and operators would all contribute to appropriately determining LOA. Thus, some questions are geared towards training and developing trust in the system, some towards prototype design and implementation, and some towards testing and verification. This resulted in a robust questionnaire, or tool, that addresses all of the identified factors for determining LOA.

The initial tool had one major flaw. All of the responses in a given questionnaire were combined into one LOA output for mapping to the LOA Assessment Scales. While this was the original goal, the initial tool was ignoring at least one major aspect of the autonomy question. There is a clear difference between how much a human user trusts an autonomous system and how high the cost/benefit ratio directs the building of the system. If the human does not trust the system, then it does not matter how intelligent or cost-efficient the system is designed to be. Similarly, even though a system would be highly trusted to work fully autonomously, there is no guarantee that this is the most cost-effective method of performing the function. Therefore, the scale was split to determine two factors: 1) what LOA Trust Limit would ensure human trust in the function, and 2) what LOA C/B (Cost/Benefit Ratio) Limit would optimize the ratio of costs and benefits.

An example of The Functional Level of Autonomy Assessment Tool for the Decide OODA category is located in Appendix A. It is a 35-question questionnaire broken into two sections. The first half is the LOA Trust Limit, and the second half is the LOA C/B Limit. Each section is broken down into sub-topics located in the left-hand column. The questions are answered using a 5-point scale, unless otherwise noted. The user will place a "1" in the appropriate column to answer based on their expertise. The following two columns contain relevant notes and a specific example explaining how to answer the questions. This questionnaire was developed in a spreadsheet, which allows automatic calculation of results based on underlying weighting and scaling algorithms that were developed. Each OODA function type has its own baseline scale for calculations, designed to map it correctly onto the LOA Assessment Scale. The two numbers, located at the bottom of the questionnaire, represent the output from the scoring algorithms for both the LOA Trust Limit and the LOA C/B Limit. The numbers are intended to directly correspond to the scales shown in Table 2. Though, interpreting specific design guidelines from the results cannot be performed yet. The underlying algorithms must be calibrated to ensure that an output of 5 from the questionnaire matches a 5 on the appropriate scale. Until this has been completed, the output can primarily be used as a relative scale where one function leans towards a higher LOA than another.

Some questions are designed such that answering "high" means very autonomous, while others have "low" as very autonomous. This is an effort to overcome the human propensity of filling out the questionnaire to achieve a desired result.

In use, the LOA C/B Limit and LOA Trust Limit must be compared. If the LOA C/B Limit is higher (more autonomous) than the LOA Trust Limit is, then even though it may be cost effective to implement an autonomous system at the LOA C/B Limit, humans would not trust the system at that level. Since trust is the limiting factor, either the system should be automated to the lower trusted level as determined by the LOA Trust Limit, or the trust issues should be resolved through test projects and education of the users until the LOA Trust Limit result increases to the LOA C/B Limit result.

If the LOA Trust Limit is higher than the LOA C/B Limit is, then even though humans would trust a more autonomous system, either high cost, minimal benefit, or both is the primary limiter in the LOA. Either the function should be designed to the lower LOA C/B Limit, or the cost and/or benefit issues should be resolved until the LOA C/B Limit result increases to the LOA Trust Limit result.

6. RESULTS

Initial results from the development can be captured in three categories: data trends, lessons learned, and future improvements. The data trends are taken from the results in Appendix B: Sample Results from the SMART Level of Autonomy Assessment Tool. A relative trend is that ascent functions tend to be more autonomous than other flight phases. This is reasonable given the time-critical nature of the ascent flight phase. Additionally, decide functions tend to be the least autonomous aspect of each of the flight phases. It has been the experience of NASA's human spaceflight program to have humans make the majority of the decisions for the system. Thus, this category requires the furthest leap in trust and the largest shift in paradigms to allow autonomous capability.

Lessons learned thus far are many and varied, and have led to identification of multiple future improvements that are under consideration. Due to the subjective nature of determining the appropriate level of autonomy in designing a system, personal biases and technical experiences could affect the results. To minimize this problem, a first step is to average the LOA Assessment Tool results from multiple users. How many users are required is an important question, but is probably not as important as selecting users with the right mix of backgrounds. The tool might be adapted to account for the background of the user and weight their answers appropriately. The wording of subjective questions, corresponding notes, and examples in this tool is critical, especially if the question refers to a length of time or milestone (which must be explicitly stated). At a minimum, a reference or anchor-point must be provided for the user to gauge an appropriate answer. It is critical that the tool be user-friendly. Answering the questions for many different functions can be tiresome, and so every effort should be made to assist the user in completing the questions easily and quickly. Statistical analyses of the results should be performed for each question as a method to discover tool functionality problems. Another method would be to allow users a space for comments for each question. An administrative team is required to properly manage the output, interface, questions, and results.

One of the most challenging obstacles is that the word *autonomous* is dangerous. The word *autonomous* is typically understood to mean complete computer responsibility. The LOA Assessment Tool development process has found that complete autonomy is often not practical. Cost, benefit, and trust issues limit real systems to practical, intermediate levels of autonomy that balance human and computer involvement. A broader understanding of this by the human spaceflight community would be extremely helpful in developing the required trust to operate at a level of autonomy in flight management systems that allows for reduced costs, increased safety, and expanded mission capability. Therefore, a system cannot be defined by autonomous or not autonomous, but a system can be defined by the levels of autonomy for each of its functions.

This version of the LOA Assessment Tool was designed to determine the division of labor between computers and humans for all levels of autonomy. Additional adaptations include adding questions that are designed to determine the division of labor between the ground and onboard. Questions must also be added that address sustaining engineering costs. This issue has proven hard to quantify, even subjectively, due to varied backgrounds and experiences with computer software. Finally, test cases will be used to calibrate and validate the underlying algorithms within the LOA Assessment Tool.

7. CONCLUSIONS

Computer advancements, the emergence of highly reliable decision-making algorithms, and the emphasis on efficiency make an increased use of autonomous systems very likely. However, for some human spaceflight applications, full autonomy is often not practical. Rather, a balance must be found between how much human operators trust automation, and how much benefit and cost savings automation provides. This balance typically results in an intermediate level of autonomy somewhere between full computer responsibility and full human responsibility.

The LOA Assessment Tool is a combination of the questionnaire and the underlying algorithms that produce an

analytical summary of the appropriate LOA for a particular function. The LOA Assessment Tool was developed to guide how autonomous each function within an AFM should be, and provides insight into whether the cost/benefit ratio or trust is the limiting factor. The development team first split each function within SMART, an AFM system, into a specific OODA (Observe, Orient, Decide, and Act) category. Second, the team answered a set of subjective questions within the tool for each function. Then, the tool utilized these answers to provide a LOA Trust Limit and a LOA C/B Limit for each function. The team then selects the lower of the two limits, and compares that value to the OODA-specific LOA Assessment Scale for the final assessment of the appropriate level of autonomy to design each AFM function. Determining LOA is an iterative process. Throughout the design, prototype, and testing phases the team can work to raise the lower of the two limits by increasing trust (through training, etc.), lowering costs or increasing benefits.

The LOA Assessment Tool was tested utilizing NASA's OSP program. Expected results were found. For example ascent functions are typically more automated than orbit functions.

Work is continuing to incorporate the lessons learned, then calibrate, validate, and baseline The LOA Assessment Tool. Once completed, the LOA Assessment Tool will be used to help develop an Autonomous Flight Management system that is trusted to operate in the most cost-effective and most beneficial manner possible.

REFERENCES

[1] Boyd, J. R. "The Essence of Winning and Losing," Excerpts in presentation format found on the web at http://www.defense-and-society.org/fcs/ppt/boyds_ooda_loop.ppt, 1996.

[2] Clough, B. "Metrics, Schmetrics! How do you Track a UAV's Autonomy?," in Proceedings of AIAA 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles, 2000.

[3] Clough, B. "Relating Autonomy to a Task-Can it be Done?," in Proceedings of AIAA 1st Technical Conference and Workshop on Unmanned Aerospace Vehicles, 2000.

[4] Parasuraman, R., Sheridan, T. B., and Wickens, C. D. "A Model for Types and Levels of Human Interaction with Automation," in Proceedings of IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, Vol. 30, No. 3, 2000.

[5] Sheridan, T. B. *Telerobotics, Automation, and Human Supervisory Control*. The MIT Press. 1992.

APPENDIX A: DECIDE FUNCTION EXAMPLE FROM THE FUNCTIONAL LEVEL OF AUTONOMY ASSESSMENT TOOL

Function Name	Scale Type (Ob, Or, D, or A)							
	Decide							
	Question	Answer = 1 in most applicable column						
1	LOA Trust Limit							
		High (yes)	Med-High	Med	Med-low	Low (no)	Question notes	Question examples
Ability	What is the expected ability of developers to correctly design the function for all possibilities within the design phase deadlines?						Expected ability of designers to completely define the world of possibilities that this function will face, before the final deadline. Ability is defined as able to do the job, not the designer's ability level.	Designers would have low ability to design the hardest or newest functions.Thus, hard or new functions would be low on the ability scale. Though, easy functions or functions that we have been doing for years would be high in ability.
	What is the expected ability of programmers to correctly implement the design within the implementation deadlines?						Expected ability of software writers to completely code the design that the developers handed them, regardless of the size of the world that was defined in the design phase, before the final deadline. Ability is defined as able to do the job, not the programmer's ability level.	If the developers are only going to design for a limited set of cases, it makes it more likely that the software writers will be able to code it up by the deadline.
Difficulty	What is the expected effort of developers to correctly design the function for all possibilities within the design phase deadlines?						This is the same as the above questions, but the focus is not on "how good will the design be?' but on "how hard will it be to design?"	Hard (high) designs start from scratch or incorporate new ideas where expectations are not well-defined. Easy (low) designs have already been used in this context.
	What is the expected effort of programmers to correctly implement the design within the implementation deadlines?						This is the same as the above questions, but the focus is not on "will I have good software in time?' but on "how hard will it be to get the software done in time?"	The code itself may be relatively simple for if-then statements, but the effort to completely code the flight rules into if-then statements might be hard to do in the time allotted (especially since all of the rules do not exist yet). This would be a medium.
Robustness	What is the likelihood of an "outside-the-box" scenario occurring?						Probability statistics or sigmas.	It would be difficult to program the Orbit Attitude Time Line for all contingencies at all attitudes. What is the probability that we will find ourselves in one of these attitudes?
	How well will/can the function be designed to manage "outside-the-box" scenarios?						Question of returning an answer of "Unconverged" vs. "Unconverged with an explanation. In addition, what has been done in the mean time to protect the crew and the vehicle."	Often, this is a timing issue. On Ascent, there may not be enough time to get the ground operators to come up with a solution for an outside-the-box scenario, but on Orbit you have more time. So orbit would be able to manage these scenarios easier. The point is that Ascent has to define its world of possibilities better, thus making it harder to do, and ultimately pushing towards less autonomy when compared to Orbit
Experience	How autonomous (what level) has the function been shown to perform?						On this five point scale, what is the current maximum level of autonomy for a human spaceflight vehicle?	If the function has never been done before (trajectory shaping) it would be low.

	Has the function been completed solely by a human during the flight phase itself?						Either crew or ground operators doing this dynamically during the particular phase.	Abort boundaries are predicted dynamically during a flight today through Human interaction with the Abort Region Determinator. Without the tool, the human is reliant on the pre flight cue-card because the human can't come up with a predicted solution fast enough. This answer would be low because the Human does not do this "solely". Though, what Abort mode to ultimately choose is decided solely by
Understandability	How understandable of a mental model of the function can a human create, including how the function works, what the output means, how to interact with the function?						Are the concepts, themselves, involved with this function complicated?	Understanding how a neural network comes to a decision is very complex. This would be a low. If-then statements would be easier to understand. This would be a high.
	What is the level of human understanding required to accurately decide when an override is necessary?						What level of understanding would a human need to have in order to determine if the output from this function is out of family?	If EVAR is running and gives a number of 15, when the human is expecting a 0.8, the human would probably understand enough about the function to override
	If an override is performed, what is the ability of a human to come up with a solution themselves?						If a human is going to override, they better have a good answer to put in its place.	If a human was expecting the 0.8 in EVAR and found a 15, would he take the site off the list, or would he be able to update the priority table with the correct number (determined through the human's own math).
Art vs. Science	How much would a human have to infer what the computer "really meant" or what the computer will do in the future?						This is truly an Art vs. Science question. If performing this function is an art form of human fudge factors, and post-processing mental tweaks, then it should be hard to automate. Though if the function is purely scientific, with a definite answer that needs little human interaction to change it to be the "correct" answer, then the function should be easier to automate.	Trending data shows that an engine is going to be shut off in 20s. The function which determines what Abort to declare is currently deciding that Moron is the top priority. Though the system might not take into account that the faulty engine will still run for an additional 20 seconds, or that the secondary engines will ignite to increase performance. With the addition of these factors, a human knows that Zaragoza is really the best site to go to. Thus, 20 seconds later when the engine is shut down and the secondary engines light off, the computer changes the decision to Zaragoza. In this case the human had to infer what the computer "really meant" when it was choosing Moron (med-high).
Familiarity	How familiar, friendly, and natural will the output feel to the user?						Based on the designer's (the person who should be filling this out) understanding of the current implementation of this function, is the output going to be something new, or will it be the same type of output that humans deal with today.	Basing abort decisions on velocity is very familiar to human ground operators and crew. SAFM changed the measure of landing site ability to a single number that is evaluated based on its relation to 1. This was a foreign concept and ultimately not trusted by the users
Correctness	What is the probability that the computer could come up with an answer that is "more accurate" than a human?						Both a human and a computer can come up with an answer that is "right". A human may be able take the big picture and incorporate it into coming up with the better answer. A computer may be able to run optimization algorithms to come up with a better answer.	Choose an Insertion Altitude pre-launch = low. Design a trajectory that maximizes ascent performance = high.
Training	How much training would be required for a human to perform this function instead of performing the function highly autonomously?						Can a human do this without computer help?	Making abort decisions without a computer requires a lot of training. This would be a high. Monitoring engine chamber pressure would be a low

	How much training would be required for a human to interface with a tool using this function based on current understanding of the implementation of this function?						Can a human do this task with some help from the computer?	Making abort decisions with a computer's aid still requires training, but not as much. This would be a med-high to med. Monitoring engine chamber pressure would be a low.
	How much verification would be required for this function to be trusted to perform fully autonomously?						Not intended to be Verification and Validation costs, but it is intended to be how many cases and examples would have to proven to work correctly for the function to be trusted to work fully autonomously	Adapting the trajectory based on launch day winds would be high, because of the unpredictable nature of the winds.
Override	Is an override capability required (yes or no)?						This will limit the autonomy scale to allow override if yes is chosen	It may be required for the computer to allow human override of abort decisions. This would be a yes
Deterministic	How deterministic is the output from this function?						If the output is not deterministic (i.e. the output is not a specific answer) then it cannot be highly automated	If the function does not converge, it may be programmed to output "Unconverged" after a certain time. This is a deterministic answer. It is when the same inputs produce two different outputs that it would not be deterministic.

2	**LOA Cost/Benefit Limit**							
		High (yes)	Med-High	Med	Med-low	Low (no)	**Question notes**	**Question examples**
Usefulness	How critical is this function to an overall Autonomous Flight Management system?						While the function itself might not be that critical, other functions might require this function to be done highly autonomously in order to work highly autonomously themselves.	While predicting future apogee and perigee could be done by a human, it might be needed to work autonomously by the system for the anytime deorbit processor. Thus, the answer would be high.
	How useful would automating this function be?						Gut feeling. This function would be useful for the computer to do instead of the human.	Automating Pre-launch design is inherently useful for the OSP due to the small changes that will occur flight-to-flight (i.e. it is always flying the same mission)
Time	How much time is available to perform function, considering flight phase, circumstances, possible contingencies, etc.?						Each flight phase has a different scale of time. On Ascent and Entry, many decisions may be required in milliseconds. This is faster than a human could possibly provide an answer, trending towards more autonomy.	Deciding to shut down an engine, given that an observe function is predicting a catastrophic failure imminently is a split second decision. This would be low (or low amount of time available). A relatively high amount of time might be available for some Orbit attitude time line decisions.
Criticality	What is the criticality of this function to vehicle safety?						Vehicle specific.	Attitude time line would be med-high due to the need to point radiators away from sun to protect the vehicle.
	What is the criticality of this function to crew safety?						Human crew specific.	Pad abort decisions would be high .
Costs	How many lines of code are expected? low <= 1000 med-low <= 10,000 med <= 50,000 med-high <= 100,000 high >100,000						Arbitrary scale based on a few conversations with Howard Hu and Rich Ulrich. There has to be a set scale in order to answer this question.	SAFM = 10,000 ARD > 100,000 Shuttle FSW code = 23,000
	How much time to design the function is expected?						Question of man-hours. The deadline for completion is set, and this question asks will this function be done in time.	Determine the In-Plane Launch time would be a low. This is a function that is well understood today, and it could be designed for OSP quickly. More complicated architectural questions like how negotiation for propellant occurs between flight phases would probably take a long time to design correctly.

	How much time to implement the software for this function is expected?						Same as previous question. Focused on writing the code, not the design phase.	low = 1/10 of the available time med = 1/2 of the available time high = all of the available time
	What is the level of required verification and validation?						This is the software V&V question. How many runs will be needed to prove that the algorithms work.	Trajectory Shaping would be very high because the possible answers and scenarios are infinite. This would be a high. Predicting the primary engine cutoff state is bounded by the GN&C program that flies the vehicle. This would be a low.
	What is the required skill level of software writers?						How hard will the function be to program?	Simple prediction equations would be lows. High-level decision making logic like neural nets would be high.
Efficiency/task management	To what degree would automating this function increase the efficiency of a human or another part of the AFM?						Would this increase the efficiency of whoever interfaces with this function, be it human or other function?	Having the computer filter incoming observed information, to only show relevant data, would increase efficiency by allowing the human to analyze the relevant data more often or increase the number of parameters that are examined.
Mental workload	To what degree would automating this function decrease a human's mental workload?						Would the human still have to worry about this function? Could this be automated well-enough that the human does not have to think about it anymore.	Anything that decreases the amount of calculations that the human would perform during the flight phase would be high to med-high.
Boredom	How repetitious is the function (level of frequency)?						Answer based on the flight phase, and the number of cycles a second the function would be performed.	Determine if Pad Abort is necessary must be evaluated every cycle per second. This would be highly repetitious.
	How mundane (does not utilize the skills of the operator) is the function?						If the task bores the human that is forced to perform it, the tendency is for errors to increase. Thus, mundane tasks should be automated.	Simple tasks that have a human watch a number to see if it goes over a limit is highly mundane.

LOA Trust Limit	LOA Cost/Benefit Limit
0.00	0.00

APPENDIX B: SAMPLE RESULTS FROM THE SMART LEVEL OF AUTONOMY ASSESSMENT TOOL

Function Number	Function Name	LOA Trust Limit	LOA C/B (Cost/Benefit) Limit	Function type observe=1 orient =2 decide =3 act=4	Minimum
1 - Prelaunch	Ground Updates	4.00	5.63	1	4.00
2 - Prelaunch	Vehicle System Monitoring	4.00	5.38	1	4.00
3 - Prelaunch	Non-Trajectory-Related Flight Operations Monitoring	4.00	5.50	1	4.00
4 - Prelaunch	LW/LT Function Objective Determination	4.71	5.25	3	4.71
5 - Prelaunch	Selection of Nominal Insertion Target Altitude	5.22	5.75	3	5.22
6 - Prelaunch	Nominal Insertion Propellant Requirement	5.32	5.00	2	5.00
7 - Prelaunch	Monitor Launch Window expansion amount	4.00	5.50	1	4.00
8 - Prelaunch	Degraded Insertion Target Altitude - Launch Window Expansion	4.60	4.75	3	4.60
9 - Prelaunch	Launch Window Expansion Propellant Requirement	5.22	5.00	2	5.00
10 - Prelaunch	Predict Ascent Performance Margin	5.53	5.88	2	5.53
11 - Prelaunch	Degraded Insertion Target Altitude - Ascent Performance Loss	4.81	4.63	3	4.63
12 - Prelaunch	Ascent Performance Loss Propellant Requirement	5.22	5.00	2	5.00
13 - Prelaunch	Determine Phasing Windows	5.74	5.25	2	5.25
14 - Prelaunch	Determine Intersection Point or Point of Closest Approach	6.00	5.25	2	5.25
15 - Prelaunch	Determine Zero Wedge Angle Time (Analytic In-Plane Time)	6.00	6.00	2	6.00
16 - Prelaunch	Determine In-Plane Time	6.00	5.38	2	5.38
17 - Prelaunch	Determine Planar Window	6.00	5.75	2	5.75
18 - Prelaunch	Determine Planar/Phase Window Overlap	5.94	6.13	2	5.94
19 - Prelaunch	Evaluate Candidate LW/LTs Against Constraints	4.60	4.75	2	4.60
20 - Prelaunch	Rank Available Launch Targets	4.71	4.75	3	4.71
21 - Prelaunch	Store Available Launch Targets	6.35	7.13	4	6.35
22 - Prelaunch	Optimum Launch Target	5.32	6.00	4	5.32
23 - Prelaunch	Objective Function Selection	4.50	4.75	3	4.50
24 - Prelaunch	Nominal Ascent Flight Constraints	4.91	6.00	2	4.91
25 - Prelaunch	Planetary Environment Data	4.00	6.38	1	4.00
26 - Prelaunch	Vehicle Data	4.00	6.38	1	4.00
27 - Prelaunch	Trajectory Data	4.00	6.25	1	4.00
28 - Prelaunch	Staging Constraint Parameters	4.00	6.38	1	4.00
29 - Prelaunch	Mathematical Model of Objective Function	5.01	5.50	2	5.01
30 - Prelaunch	Selection of New Mathematical Model of Objective Function	5.12	5.50	3	5.12
31 - Prelaunch	Nonlinear Optimizer Tuning Properties Selection	5.74	6.25	3	5.74
32 - Prelaunch	Initial Trajectory Generation	4.91	5.00	4	4.91
33 - Prelaunch	Trajectory Optimization Analysis	4.81	5.88	2	4.81
34 - Prelaunch	Trajectory Optimization Decision	5.32	6.00	3	5.32
35 - Prelaunch	Feasibility and Convergence Check	5.22	6.13	2	5.22
36 - Prelaunch	Trajectory Performance and Constraint Evaluation	5.53	6.50	2	5.53
37 - Prelaunch	Optimized Trajectory	5.84	6.13	4	5.84

38 - Prelaunch	Candidate Landing Site Status	4.00	4.75	1	4.00
39 - Prelaunch	Evaluate Landing Site Coverage	5.63	5.50	3	5.50
40 - Prelaunch	Evaluate Abort Coverage	5.12	4.75	3	4.75
41 - Prelaunch	Trajectory Shaping for Insufficient Abort Coverage	4.91	3.88	3	3.88
42 - Prelaunch	Trajectory Shaping for Marginal Abort Coverage	4.81	3.75	3	3.75
43 - Prelaunch	Ascent Target Redesign	5.43	4.00	4	4.00
44 - Prelaunch	Continuous Abort Coverage Evaluation	5.53	5.00	2	5.00
45 - Prelaunch	Vehicle Internal Systems Flight Rules	4.00	4.63	1	4.00
46 - Prelaunch	Vehicle Consumables Flight Rules	4.00	4.63	1	4.00
47 - Prelaunch	Ground-Based Systems Flight Rules	4.00	4.63	1	4.00
48 - Prelaunch	Space-Based Systems Flight Rules	4.00	4.63	1	4.00
49 - Prelaunch	Trajectory - Guidance Flight Rules	4.00	4.63	1	4.00
50 - Prelaunch	Trajectory - Control Flight Rules	4.00	4.63	1	4.00
51 - Prelaunch	Trajectory - Navigation Flight Rules	4.00	4.63	1	4.00
52 - Prelaunch	Pad Abort Monitor	4.00	7.38	1	4.00
53 - Prelaunch	Pad Abort Evaluation	5.74	6.25	2	5.74
54 - Prelaunch	Pad Abort Decision	5.84	6.50	3	5.84
55 - Prelaunch	Pad Abort Execution	6.76	6.63	4	6.63
56 - Ascent	Ascent Elements	5.84	7.00	2	5.84
57 - Ascent	Ascent Goal Assessment	6.00	7.63	2	6.00
58 - Ascent	Predict Ascent Propellant Usage	6.00	6.38	2	6.00
59 - Ascent	Projected Vehicle Conditions	6.00	7.50	2	6.00
60 - Ascent	Secondary Engine Insertion Burn(s) Target Solution(s)	5.22	5.75	3	5.22
61 - Ascent	Command Secondary Engine Insertion Burn(s)	6.87	7.13	4	6.87
62 - Ascent	Insertion Goal Assessment	4.71	5.25	2	4.71
63 - Ascent	Predict Insertion Propellant Usage	5.94	6.50	2	5.94
64 - Ascent	Ground Inputs	4.00	7.38	1	4.00
65 - Ascent	Vehicle System Monitoring	4.00	7.13	1	4.00
66 - Ascent	Flight Mode Applicability	5.57	6.75	3	5.57
67 - Ascent	Flight Mode Constraints	6.00	6.63	2	6.00
68 - Ascent	Current Flight Mode Evaluation	6.35	7.13	3	6.35
69 - Ascent	Hypothetical Flight Mode Evaluation	6.25	6.63	3	6.25
70 - Ascent	Calculation of Margins	6.00	7.13	2	6.00
71 - Ascent	Prioritize Flight Modes	6.56	7.63	3	6.56
72 - Ascent	SMART Commands (time critical)	6.25	6.63	4	6.25
73 - Ascent	Command Decisions (non-time critical)	6.04	5.00	4	5.00
74 - Ascent	Command Displays	5.43	6.38	4	5.43
75 - Ascent	Propellant Status Determination	4.00	6.38	1	4.00
76 - Ascent	Propellant Prediction	5.12	6.25	2	5.12
77 - Ascent	Propellant Allocation	4.09	4.88	3	4.09
78 - Ascent	Propellant Management Execution	6.35	6.00	4	6.00
79 -Orbit	create initial trajectory objectives (given mission objectives)	4.50	3.88	3	3.88
80 -Orbit	create initial trajectory constraints	4.19	3.88	3	3.88
81 -Orbit	define traj events	5.32	3.88	3	3.88
82 -Orbit	resolve traj conflicts	4.40	4.13	3	4.13
83 -Orbit	targeting	6.00	4.88	2	4.88

84 -Orbit	propagate trajectory	6.00	6.00	2	6.00
85 -Orbit	evaluate mission/traj objectives	4.81	3.88	2	3.88
86 -Orbit	evaluate target collision risk	5.43	4.75	2	4.75
87 -Orbit	evaluate debris conjuction risk	5.74	4.00	2	4.00
88 -Orbit	evaluate plume impignment on ISS	5.63	4.38	2	4.38
89 -Orbit	evaluate D/O handover - Entry team handles all subfunctions, post-plan?	5.22	3.75	2	3.75
90 -Orbit	determine orbit plan senstivities	4.50	4.13	2	4.13
91 -Orbit	modify trajectory objectives (if possible)	4.19	3.88	3	3.88
92 -Orbit	create initial ATL objectives (given mission objectives)	4.60	3.75	3	3.75
93 -Orbit	create initial ATL constraints	4.19	3.75	3	3.75
94 -Orbit	define ATL events	5.32	3.88	3	3.88
95 -Orbit	resolve ATL & traj conflicts	4.40	3.38	3	3.38
96 -Orbit	evaluate thermal	5.63	5.00	2	5.00
97 -Orbit	modify ATL objectives (if possible)	3.99	3.88	3	3.88
98 -Orbit	post-generated integrated plan assessment	4.09	4.25	2	4.09
99 -Orbit	propellent	5.63	5.75	2	5.63
100 -Orbit	power - (definitely if solar involved) (a lesser driver if no solar arrays)	5.53	5.75	2	5.53
101 -Orbit	manage space ephemeris - (support traj constraints - lighting)(supports ATL constraints - pointing)	6.00	5.25	2	5.25
102 -Orbit	monitor GN&C sensors	4.00	5.13	1	4.00
103 -Orbit	monitor GN&C effectors	4.00	5.25	1	4.00
104 -Orbit	monitor non-GNC systems	4.00	5.13	1	4.00
105 -Orbit	monitor consumables (prop)	4.00	5.75	1	4.00
106 -Orbit	diagnose impact of degraded system (human derived flt rules live here) (flt phase dependant)	4.71	2.88	2	2.88
107 -Orbit	recommend modifications to mission objectives to VMS	4.09	3.75	3	3.75
108 -Orbit	monitor mission performance	4.00	5.13	1	4.00
109 -Orbit	assess orbit debris avoidance maneuver	4.09	3.88	2	3.88
110 -Orbit	assess if can include secondary objectives (added capablity)	4.81	4.13	2	4.13
111 -Orbit	track completed objectives (mission & traj & atl)	5.63	6.13	3	5.63
112 -Entry	Deorbit Opportunities Evaluator	5.84	4.88	2	4.88
113 -Entry	Phasing Maneuvers	3.99	4.50	2	3.99
114 -Entry	Ditch Opportunity	5.53	4.88	2	4.88
115 -Entry	Deorbit Opportunities Prioritization	5.22	5.13	3	5.13
116 -Entry	Pre-Deorbit Update Monitoring	4.00	5.88	1	4.00
117 -Entry	Deorbit Targeting Processor	5.12	4.88	2	4.88
118 -Entry	Deorbit Priorities Evaluation	3.00	3.88	3	3.00
119 -Entry	Deorbit Targeting Reconfiguration Supervisor	4.29	4.75	2	4.29
120 -Entry	Landing Site Selection	4.09	3.75	3	3.75
121 -Entry	Landing Site Output	5.43	5.13	4	5.13
122 -Entry	Vehicle Monitor	4.00	5.88	1	4.00
123 -Entry	Vehicle Reconfiguration	5.22	5.00	4	5.00
124 -Entry	Deorbit Target Recomputation	4.29	4.63	3	4.29
125 -Entry	Go for Deorbit Ignition	5.32	4.63	4	4.63
126 -Entry	Deorbit Burn Trajectory Monitor	3.78	5.50	1	3.78
127 -Entry	Deorbit System Health Monitor	4.00	6.00	1	4.00
128 -Entry	Deorbit Burn Abort/Reconfiguration Determination	3.99	5.00	2	3.99
129 -Entry	Pullout Propagator	5.63	6.63	2	5.63
130 -Entry	Pre-Pullout Target	3.00	4.25	3	3.00
131 -Entry	Vehicle Capability Footprint	5.63	6.25	2	5.63
132 -Entry	Landing Site Proximity	6.00	6.63	2	6.00
133 -Entry	Landing Site Prioritization	5.32	5.25	3	5.25
134 -Entry	Descent Update Monitoring	4.00	5.38	1	4.00
135 -Entry	Descent Predictor	5.22	5.13	2	5.13
136 -Entry	Descent Evaluator	4.40	4.88	3	4.40
137 -Entry	Descent Trajectory Optimization	3.00	4.13	2	3.00

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
SEP 2003		**00-00-2003 to 00-00-2003**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Methods for Determining the Level of Autonomy to Design into a Human Spaceflight Vehicle: A Function Specific Approach	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **NASA Johnson Space Center,2101 NASA Road 1,Houston,TX,77058**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
PerMIS?03, Performance Metrics for Intelligent Systems, 16-18 Sep 2003, Gaithersburg, MD

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Same as Report (SAR)**	**14**	