Software Reuse in Safety Critical Ground Systems

Emmanuel Lesser

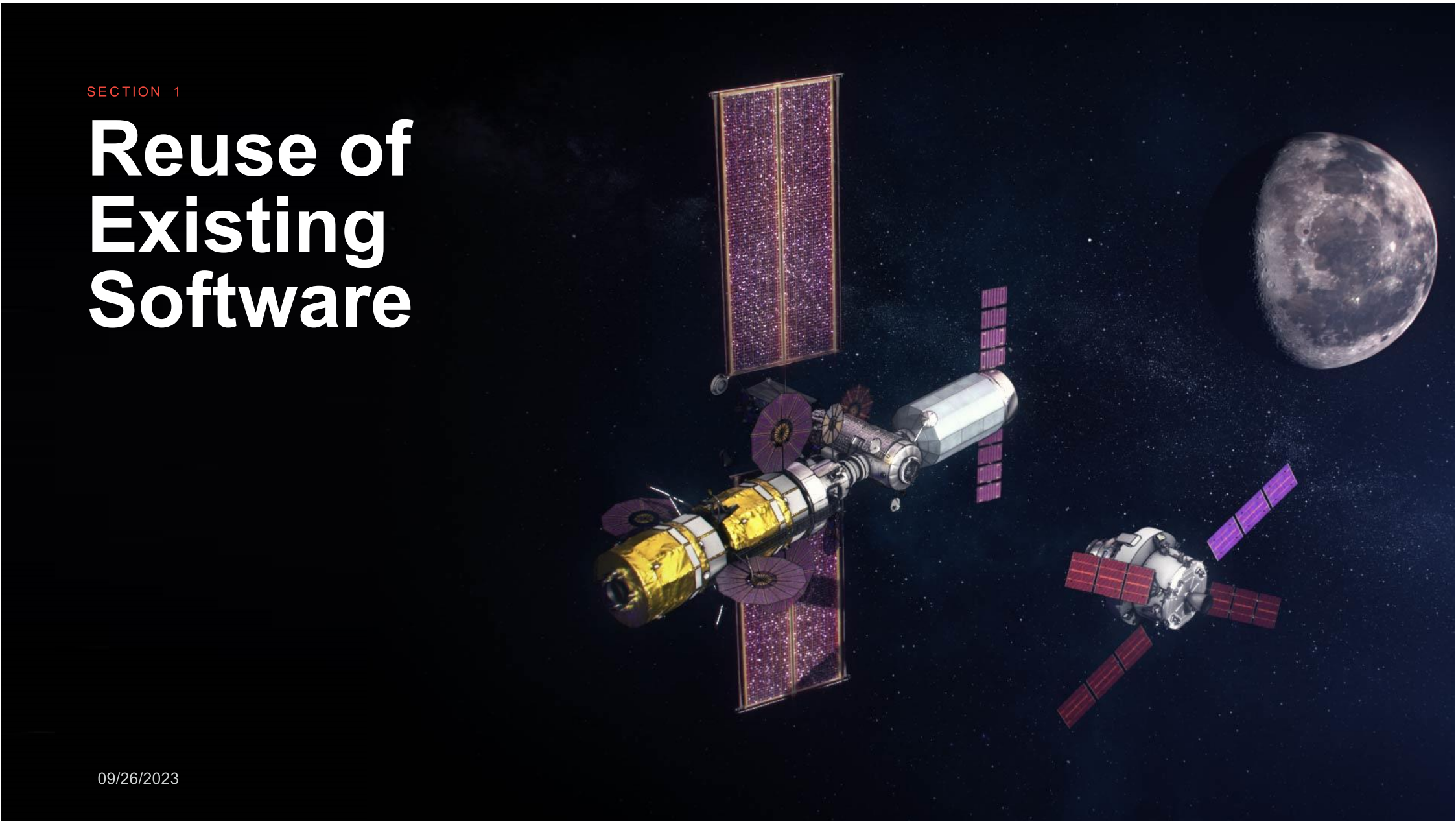
09/26/2023

MDA

TABLE OF CONTENTS

SECTION 1

Reuse of Existing Software



09/26/2023

Reuse of Existing Software

- Definitions

Existing Software (Gateway External Robotics System Product Assurance Requirements for Reuse of Existing Software, 4020749 Rev. B)

Any software developed outside of the Gateway project development as is or with adaptation. It includes software from previous developments provided by the supplier, software from previous developments provided by the customer, COTS, GOTS and MOTS software, freeware and open source software.

Reuse of Existing Software

- Definitions

Reuse (Systems and software engineering — Vocabulary, ISO/IEC/IEEE 24765:2017)

building a software system at least partly from existing pieces to perform a new application

Software Reuse (NASA Software Engineering Requirements, NPR 7150.2D)

A software product developed for one use but having other uses or one developed specifically to be usable on multiple projects or in multiple roles on one project. Examples include, but are not limited to, COTS products, acquirer-furnished software products, software products in reuse libraries, and pre-existing developer software products.

Reuse of Existing Software

- Examples
  - ✓ Application software inherited from a previous program
  - ✓ Hardware drivers, BSPs
  - ✓ Operating systems, hypervisors
  - ✓ Standard libraries
  - ✓ Libraries and drivers for FPGA soft-core processors
  - ✓ Code injected during auto generation (e.g. MBSE)
  - ✓ Desktop-type software like browsers, consoles, etc. when part of the operational system
  - ✓ Database software, when part of the operational system
  - ✓ …
  - ➢ *But not tools!*

Reuse of Existing Software

- Requirements

NASA Software Engineering Requirements, NPR 7150.2D

3.1.14 The project manager shall satisfy the following conditions when a COTS, GOTS, MOTS, OSS, or reused software component is acquired or used: [SWE-027]

a. The requirements to be met by the software component are identified.

⋮

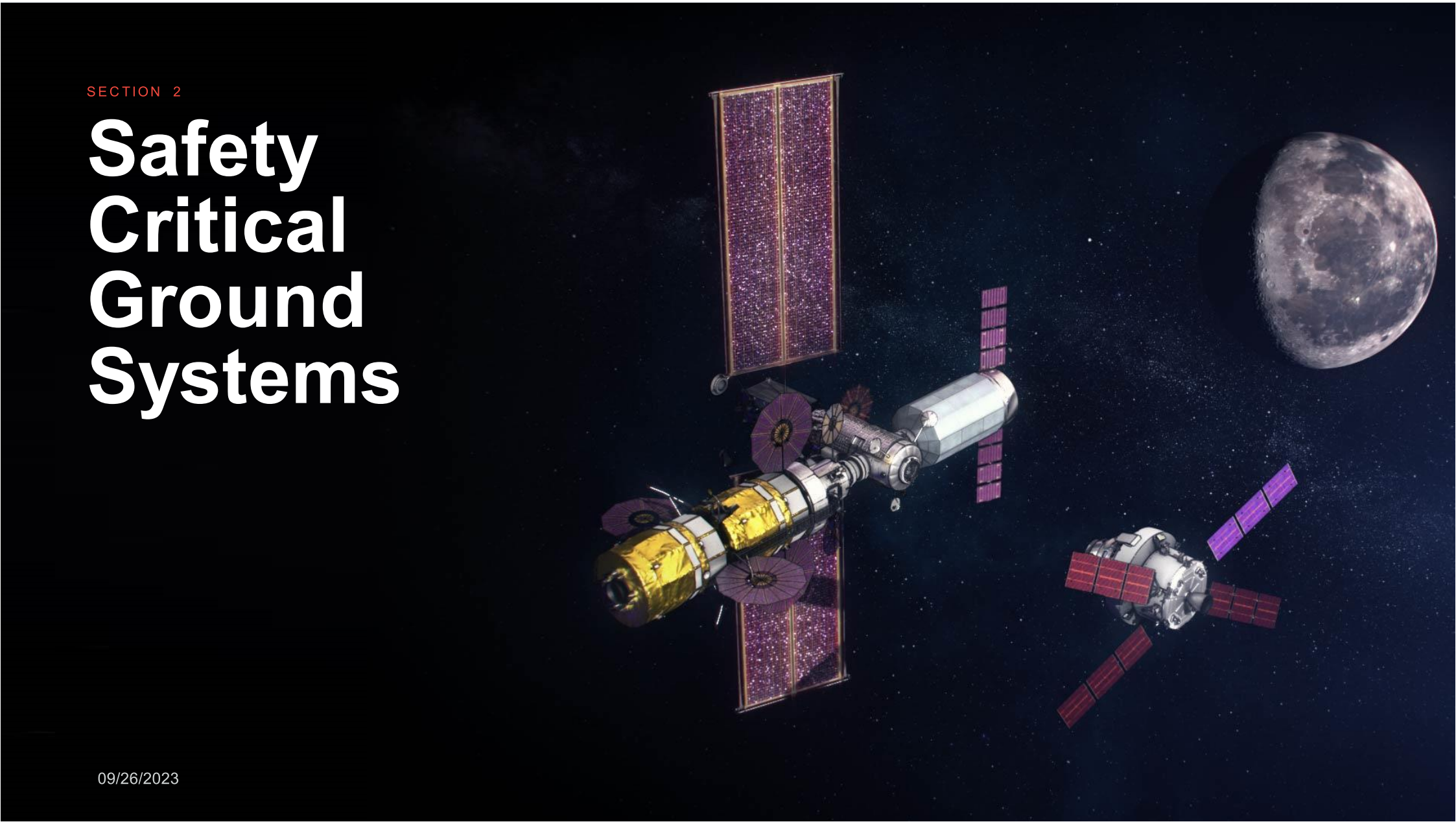
e. The software component is verified and validated to the same level required to accept a similar developed software component for its intended use.

⋮

Note: The project responsible for procuring off-the-shelf software is responsible for documenting, prior to procurement, a plan for verifying and validating the software to the same level that would be required for a developed software component. The project ensures that the COTS, GOTS, MOTS, reused, and auto-generated code software components and data meet the applicable requirements in this directive assigned to its software classification as shown in Appendix C.

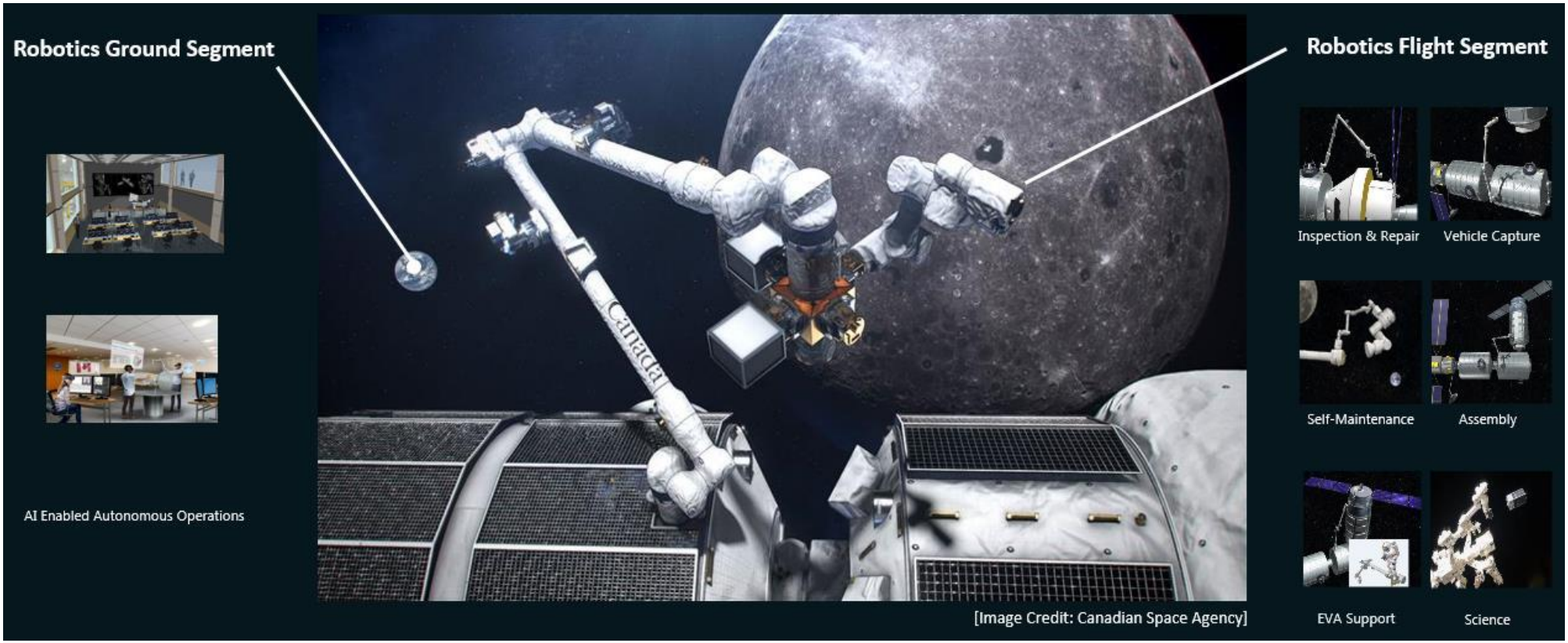
SECTION 2

Safety Critical Ground Systems



09/26/2023

Safety Critical Ground Systems



Safety Critical Ground Systems

[Image Credit: Canadian Space Agency]

AI Enabled Autonomy Infrastructure addresses 8hrs / week operations window

Mission Planning, Training, Rehearsal and Execution

Monitoring, Analysis and Trending

Simulation

Sustaining Engineering – System Maintenance and Enhancement

Public Outreach

Remote Access

Ground-Based Systems

- Operational Control Facility
- Operational Engineering Facility
- Sustaining Engineering Facility
- Training Facility
- Public Outreach Systems
- External Robotics Network Interface
- System Development GSE

Safety Critical Ground Systems

- Hazards
  - Action overrides
  - Cyber attacks
  - Emergency, Warning, Caution, & Advisory (EWCA) Alerting
    - Including display of time-critical telemetry to operators
  - Transmission of critical telecommands and data to the Flight Segment
  - Extra-Vehicular Activities (EVA) support
  - Functions supporting Flight Segment autonomy:
    - Mission planning (task & path planning)
    - Collision Avoidance model verification and certification
  - Hardware-in-the-Loop (HIL) Simulations

Safety Critical Ground Systems

- Fault Tolerance requirements
  - The Gateway Program mandates Single Fault Tolerance (1-FT) against Catastrophic hazards
  - This requirement is valid for any Gateway element, including Ground Systems
  - In effect, the Flight and Ground Segments are part of the same, integrated Computer Based Control System (CBCS)
  - 1-FT is difficult to achieve for software due to common mode failures
  - GP 10024 (Gateway Program Hazard Analysis Requirements) foresees 3 options:
    a. System Failure Tolerance
    b. Recover/Repair
    c. Risk Acceptance (Exemption B)

Gateway Program System Requirements, GP 10000

CATASTROPHIC HAZARD: Any hazard that may result in: loss of life or permanently disabling injury; loss of Gateway; loss of crew-carrying Vehicle; a condition that requires safe haven; or a loss of a major Ground Facility.

Safety Critical Ground Systems

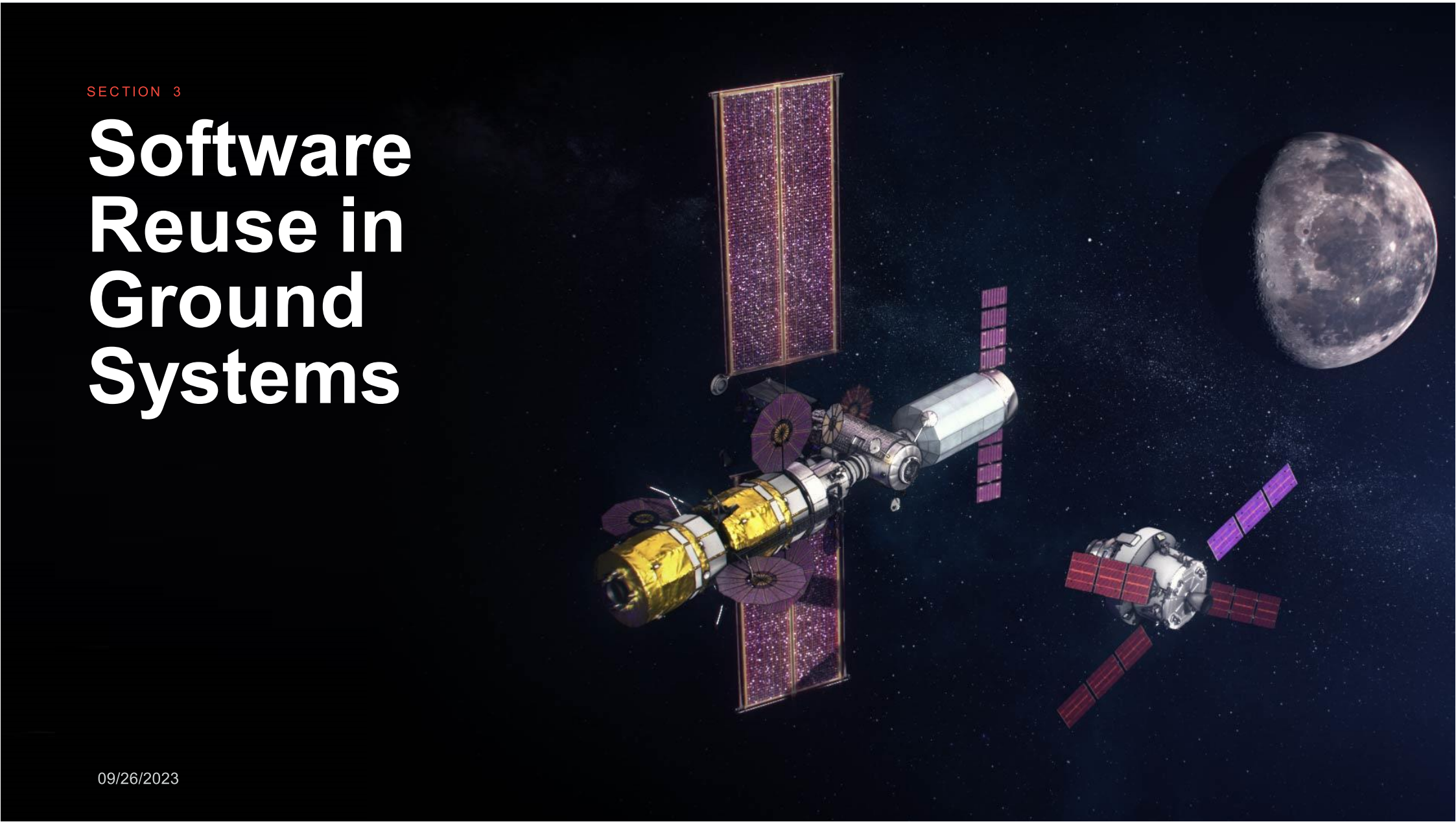
- Classes

MDA SW Class		Safety Impact (Software Risk Index is defined in 3.2 Software Risk Index)	Dependability (Worst case non-safety software impact)
I	Human-Rated Space Software (Flight or Ground)	Software Risk Index 1: High Risk: Software controls catastrophic hazards	-
II	Non-Human Space-Rated Software (Flight or Ground)	Software Risk Index 2: Elevated Risk: Software control of catastrophic hazards is reduced but still significant, or software controls critical hazards.	Loss of mission. Major mission degradation.
III	Mission Support Software (Flight or Ground)	Software Risk Index 3: Moderate Risk: Software control of less significant hazards	Minor mission degradation.
IV	Ground Support Software	Safety: Software Risk Index 4, 5: Zero or Low Risk: No impact on safety or negligible hazard or little software control.	No direct impact to operations.

NASA class		New MDA Class		ECSS class (ESA)	
A	Human rated space sw systems	I	Space or ground Safety Critical	A	Human rated
B	Non-Human rated space sw systems or large scale aeronautics vehicles	II	Space or ground Mission Critical	B	Loss of mission
				C	Major mission degradation
C	Mission support software or aeronautic vehicles, or major engineering/research facility sw	III	Space or Ground Mission support software	D	Minor mission degradation or any other effect
D	Basic science/engineering design and R&T sw	IV	Ground support software		
		V	Engineering support software		
E	Design Concept, Research, Technology, General purpose sw	VI	Proof of Concept Research related solution		
F	General purpose computing, business and IT software				

SECTION 3

Software Reuse in Ground Systems



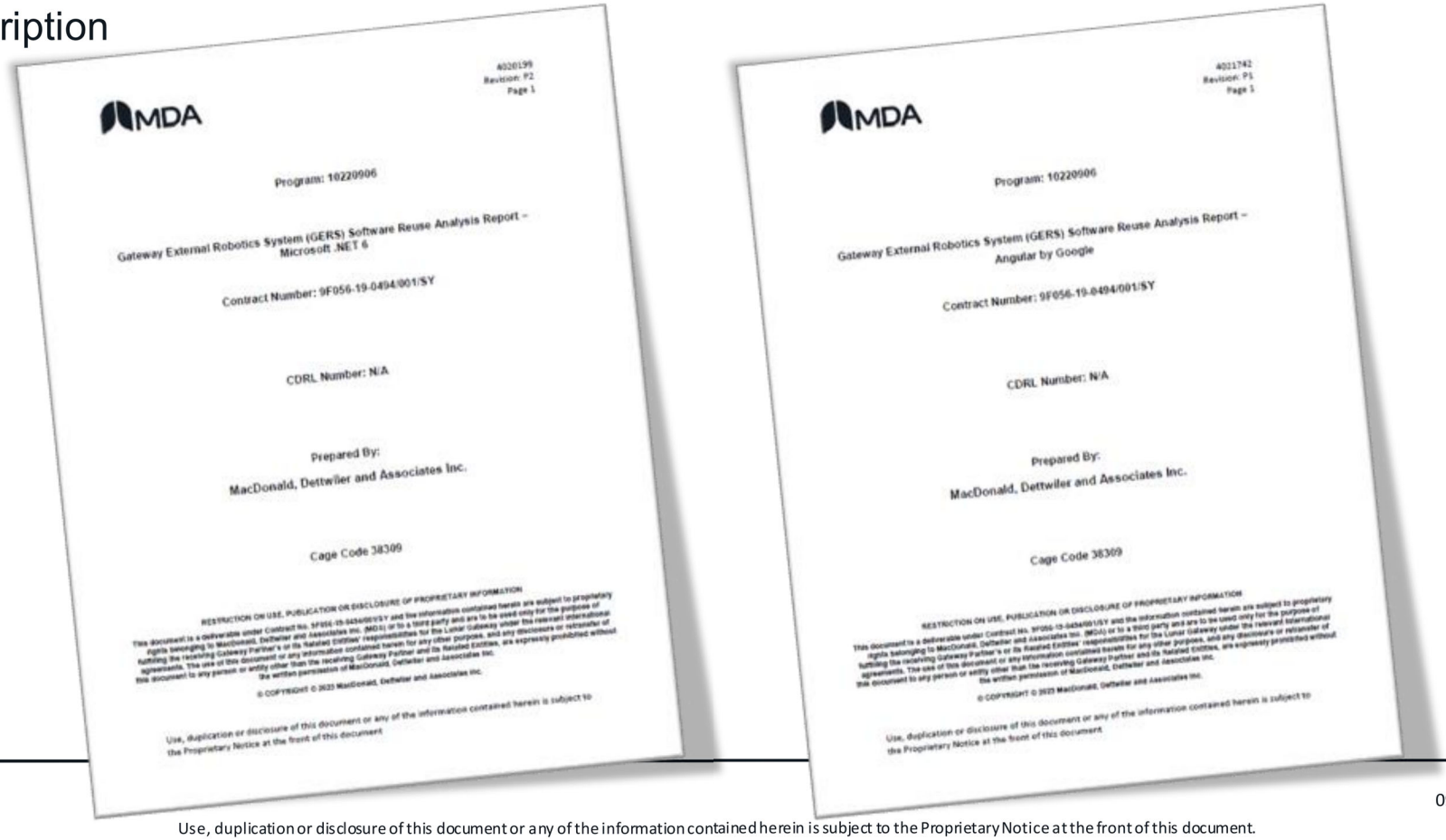
09/26/2023

SW Reuse in Ground Systems

- Large ground operation systems rely heavily on software reuse for fundamental functions, including:
  - Operating systems like Windows or Linux
  - Development frameworks like .NET and Java
  - Visualization frameworks like Angular and React
  - Audiovisual frameworks like FFMPEG and VideoLAN
  - Browsers like Chrome or Firefox
  - Virtualization platforms like Docker and Kubernetes
  - Databases like MySQL and MongoDB
  - ….

SW Reuse in Ground Systems

- Problem description



4020195
Revision: P2
Page 1

MDA

Program: 10220906

Gateway External Robotics System (GERS) Software Reuse Analysis Report – Microsoft .NET 6

Contract Number: 9F056-19-0494/001/SY

CDRL Number: N/A

Prepared By:
MacDonald, Dettwiler and Associates Inc.

Cage Code 38309

RESTRICTION ON USE, PUBLICATION OR DISCLOSURE OF PROPRIETARY INFORMATION

This document is a deliverable under Contract No. 9F056-19-0494/001/SY and the information contained herein are subject to proprietary rights belonging to MacDonald, Dettwiler and Associates Inc. (MDA) or to a third party and are to be used only for the purpose of fulfilling the receiving Gateway Partner's or its Related Entities' responsibilities for the Lunar Gateway under the relevant international agreements. The use of this document or any information contained herein for any other purpose, and any disclosure or retransfer of this document to any person or entity other than the receiving Gateway Partner and its Related Entities, are expressly prohibited without the written permission of MacDonald, Dettwiler and Associates Inc.

Use, duplication or disclosure of this document or any of the information contained herein is subject to the Proprietary Notice at the front of this document

4021742
Revision: P1
Page 1

MDA

Program: 10220906

Gateway External Robotics System (GERS) Software Reuse Analysis Report – Angular by Google

Contract Number: 9F056-19-0494/001/SY

CDRL Number: N/A

Prepared By:
MacDonald, Dettwiler and Associates Inc.

Cage Code 38309

RESTRICTION ON USE, PUBLICATION OR DISCLOSURE OF PROPRIETARY INFORMATION

This document is a deliverable under Contract No. 9F056-19-0494/001/SY and the information contained herein are subject to proprietary rights belonging to MacDonald, Dettwiler and Associates Inc. (MDA) or to a third party and are to be used only for the purpose of fulfilling the receiving Gateway Partner's or its Related Entities' responsibilities for the Lunar Gateway under the relevant international agreements. The use of this document or any information contained herein for any other purpose, and any disclosure or retransfer of this document to any person or entity other than the receiving Gateway Partner and its Related Entities, are expressly prohibited without the written permission of MacDonald, Dettwiler and Associates Inc.

Use, duplication or disclosure of this document or any of the information contained herein is subject to the Proprietary Notice at the front of this document

SW Reuse in Ground Systems

- Problem description
  - The integrated software product, including all reused software components, supports functions whose malfunction may lead to Catastrophic hazards
  - While the custom-developed parts of the system may be qualified to Class I, the integrated system remains classified at the lowest class of any of its components
  - How can we (delta-)qualify software products like Windows, .NET, Docker, …?
  - Most of the reused software products were never intended for safety-critical applications
  - Can we actually rely on these software products to safeguard human life?
  - At the same time, the in-house development of these software products is not feasible and may arguably also lead to less reliable outcomes

SECTION 4

Qualification Techniques

09/26/2023

Qualification Techniques

- Technique 1: System Failure Tolerance
  - The usage of unqualified Existing Software can be acceptable if the availability of one or more compensating provisions is ensured at System level, to allow for the correct operation of the system in case the Existing Software fails.
  - For this option to be acceptable, three necessary conditions must be demonstrably met:
    1. The compensating provision(s) (e.g. a back-up software product) must be qualified to the applicable SW Class;
    2. The compensating provision(s) may not incorporate the Existing Software in question; and
    3. Upon failure of the Existing Software, the compensating provision(s) must be able to act within time to effect, if applicable.

Qualification Techniques

- Technique 1: System Failure Tolerance
  - Consists of developing a fully qualified backup system for the Ground Segment Software.
  - This backup system shall:
    - Be as minimal and ‘thin’ as possible, by only and strictly implementing the Ground Segment software functions identified as being safety-critical and/or of Class I or II;
    - Be classified at the highest Software Class of the functions it implements;
    - Only and strictly make use of Existing Software (if any) that is already pre-qualified to the corresponding Software Class. This shall apply to any and all Existing Software, including operating systems, hardware drivers, frameworks, libraries bundled with programming languages, etc.;
    - Be command-line based (i.e. contain no GUI);
    - Be able to execute within the applicable timing constraints on commonly available desktop hardware, including in the worst-case, credible resource loading scenario; and
    - Be developed and qualified to the applicable Software Class.

Qualification Techniques

- Technique 1: System Failure Tolerance

Pros	Cons
• Provides a high level of confidence • Lowers the criticality class of the main Ground Segment software • Maintainable over the long lifetime of the program • Portable	• Expensive • Technically difficult • Integration challenges • Requires additional training for the operators

Qualification Techniques

- Technique 2: Product Service History (PSH)
  - PSH is defined as “the utilization of the information about previous in-service experience of the component that is relevant to the new intended application and that can constitute evidence of product dependability.” (ECSS-Q-HB-80-01, Rev. A)
  - Can be achieved by:
    a) collecting existing information about previous in-service experience;
    b) generating and collecting in-service experience; or
    c) a combination of (a) and (b).
  - In-service experience for PSH is only acceptable if the context of previous use is known, well-defined and similar to the context of usage of the new intended application.
  - In case of differences between the previous context(s) of usage, the gaps shall be documented and identified, so that delta-qualification activities can be established, if needed.

Qualification Techniques

- Technique 2: Product Service History (PSH)
  - Consists of:
    1. Investigating and collecting any existing service history of the individual reused packages used in the Ground System. In case product versions are different, the delta and its potential impact on safety and dependability shall be documented.
    2. Building a mock Ground System which uses all these packages, in a configuration that mimics the Ground System as closely as possible. Any deltas and their potential impact on safety and dependability shall be documented.
    3. Generating service history for the product built in (2), by deploying it in a context similar to the operational context of the GERS GS.
    4. Collecting and documenting adequate service history, in accordance with the guidelines of ECSS-Q-HB-80-01, Rev. A, section 7.6 and Annex B, and DOT/FAA/AR-01/125. The service history shall be collected over an adequate length of time.

Qualification Techniques

- Technique 2: Product Service History (PSH)
  - Generation of service history in parallel to development using analogue environments:

MISSION CONTROL

LUNARES RESEARCH STATION

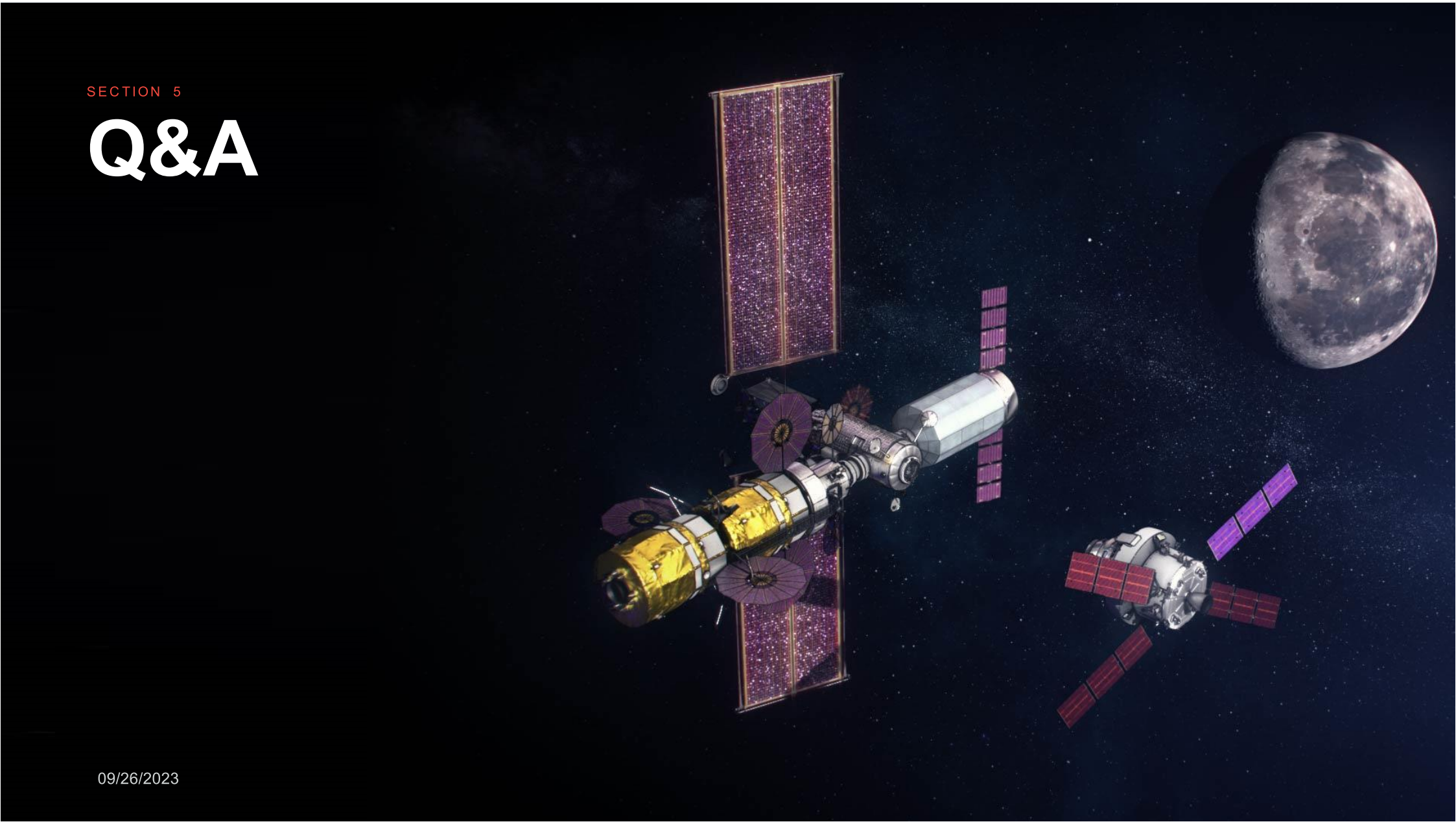
Qualification Techniques

- Technique 2: Product Service History (PSH)

Pros	Cons
• Single system • No additional training • No integration difficulties • Issues with operational system uncovered before commissioning	• Requires a baselined architecture early in the development • Provides less confidence from a safety perspective

SECTION 5

Q&A



09/26/2023

Q&A

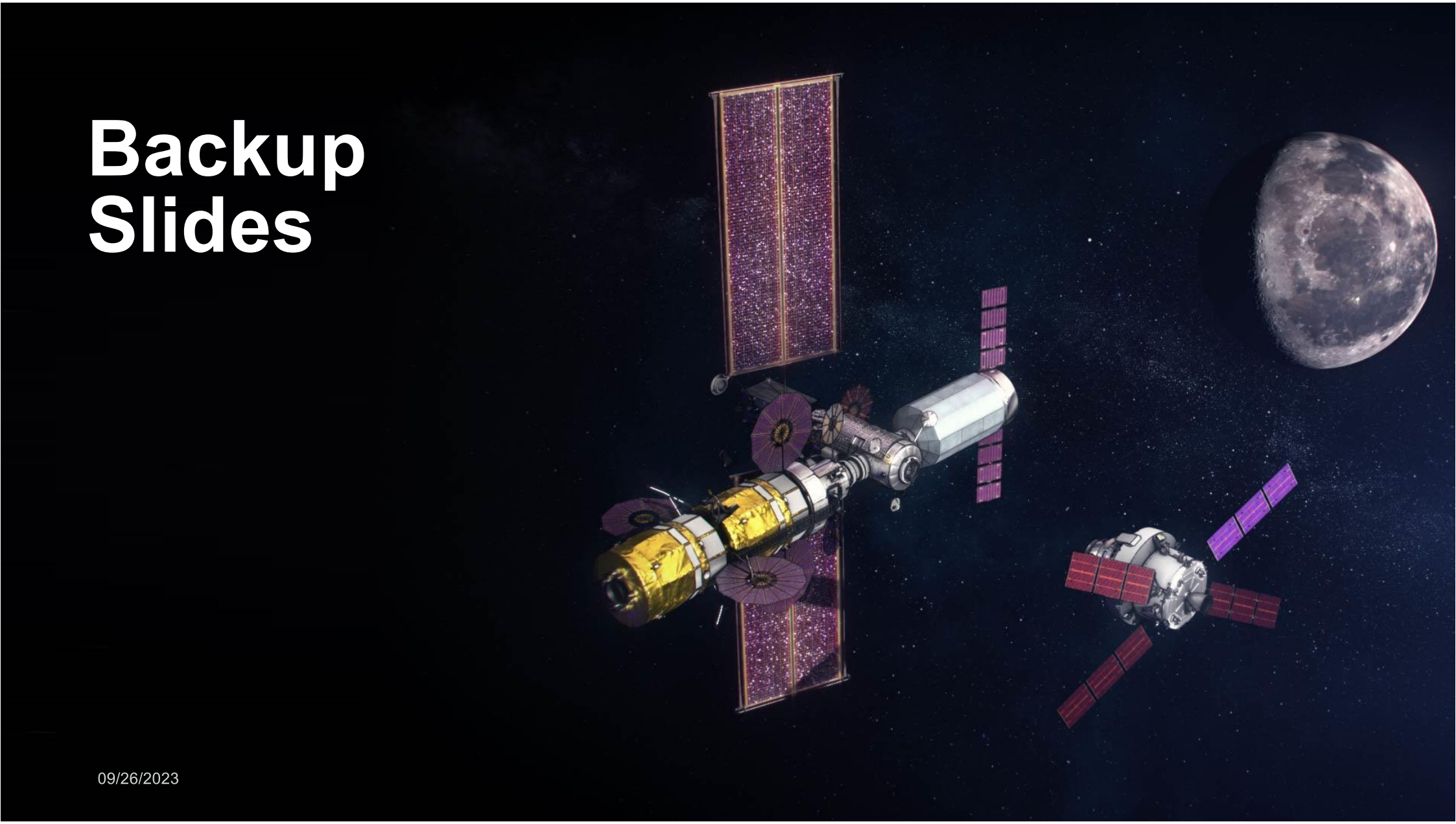
THANK YOU

CONTACT INFO:

SAFETY & MISSION ASSURANCE

EMMANUEL.LESSER@MDA.SPACE

Backup Slides



09/26/2023

Reuse of Existing Software

- Definitions

Existing Software (Space product assurance – Software product assurance, ECSS-Q-ST-80C Rev.1)

> any software developed outside the business agreement to which this Standard is applicable, including software from previous developments provided by the supplier, software from previous developments provided by the customer, COTS, OTS and MOTS software, freeware and open source software

Existing Software (Gateway External Robotics System Product Assurance Requirements for Reuse of Existing Software, 4020749 Rev. B)

> Any software developed outside of the Gateway project development as is or with adaptation. It includes software from previous developments provided by the supplier, software from previous developments provided by the customer, COTS, GOTS and MOTS software, freeware and open source software.

Acronyms

OTS = Off-The-Shelf
COTS = Commercial Off-The-Shelf
GOTS = Government Off-The-Shelf
MOTS = Modified Off-The-Shelf

Reuse of Existing Software

- Requirements

Space product assurance – Software product assurance, ECSS-Q-ST-80C Rev.1

6.2.7.7

a. Corrective actions shall be identified, documented in the reuse file and applied to the reused software not meeting the applicable requirements related to the aspects as specified in clauses 6.2.7.2 to 6.2.7.6.

Gateway Subcontractor Software Product Assurance Requirements, 4017520 Rev. B

GERS-SCPA-SW-04250 The Subcontractor shall ensure that any reused or OTS software complies with the same PA requirements applicable to newly developed software.

The SW Reuse checklist, per Appendix B, shall be used to perform software reuse analysis.

[Related Deliverables: Software assurance records]

GERS-SCPA-SW-04260 The Subcontractor shall ensure that any reused or OTS software is tested to the same level as newly developed software.

NASA Software Engineering Requirements, NPR 7150.2D

3.1.14 The project manager shall satisfy the following conditions when a COTS, GOTS, MOTS, OSS, or reused software component is acquired or used: [SWE-027]

a. The requirements to be met by the software component are identified.

⋮

e. The software component is verified and validated to the same level required to accept a similar developed software component for its intended use.

⋮

Note: The project responsible for procuring off-the-shelf software is responsible for documenting, prior to procurement, a plan for verifying and validating the software to the same level that would be required for a developed software component. The project ensures that the COTS, GOTS, MOTS, reused, and auto-generated code software components and data meet the applicable requirements in this directive assigned to its software classification as shown in Appendix C.

Acronyms

OTS = Off-The-Shelf
COTS = Commercial Off-The-Shelf
GOTS = Government Off-The-Shelf
MOTS = Modified Off-The-Shelf
OSS = Open Source Software

Reuse of Existing Software

- Qualification methods

Gateway External Robotics System Product Assurance Requirements for Reuse of Existing Software, 4020749 Rev. B

In case certain requirements cannot be met for an Existing Software product, a number of actions can be undertaken:

A. Selection of an alternative Existing Software product;
B. In-house production of a comparable software product per applicable SW PA requirements;
C. Demonstration of adequacy using PSH, possibly as part of an IOD, as applicable;
D. Performing delta-qualification activities, as documented by the Existing Software Reuse Analysis. These activities may include, but are not limited to: reverse-engineering software deliverables, performing audits of the supplier(s), developing and executing additional tests, and performing static analysis on the source code to determine code quality and adherence to (secure) coding standards;
E. Ensure the availability of one or more compensating provisions at System level, to allow for the correct operation of the system in case the Existing Software fails. For this option to be acceptable, three necessary conditions must be demonstrably met:
    1. The compensating provision(s) (e.g. a back-up software product) must be qualified to the applicable SW Class;
    2. The compensating provision(s) may not incorporate the Existing Software in question; and
    3. Upon failure of the Existing Software, the compensating provision(s) must be able to act within time to effect, if applicable.

It is the responsibility of the responsible software Project Engineer to determine which of these actions to select, based on the cost-effectiveness and feasibility of the available options.

Acronyms

PSH = Product Service History
IOD = In-Orbit Demonstration