System requirements and architecture for time & space partitioning in spacecraft

K. Hjortnaes/J. Windsor

FSW-10

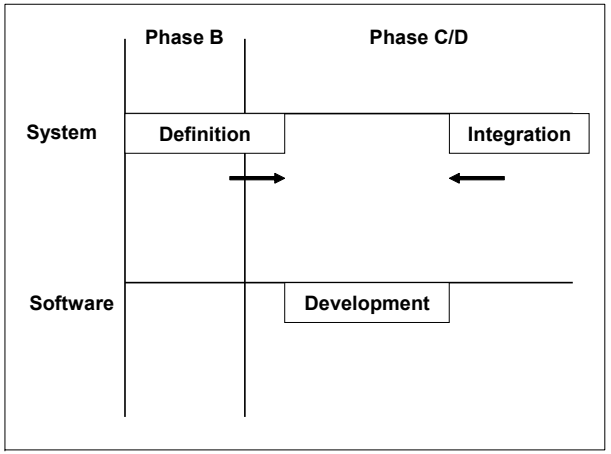
08-12-2010

ESA-ESTEC, TEC-SW Software Systems Division
Noordwijk, The Netherlands
Kjeld.Hjortnaes@esa.int
James.Windsor@esa.int

Introduction

- Trends in Spacecraft Avionics – growing complexity
- A potential solution: Integrated Modular Avionics
- Time and Space Partitioning
- Context
- Requirements and constraints
- Architectural Design
- Conclusion

Trends in Spacecraft Avionics
Growing Complexity



- Exponential growth of system functionality in software
  - Higher degree of autonomy
  - Push to get software ASAP for system AIV/AIT
- Higher levels of integration
  - Mass, volume & power savings
  - More powerful on-board computers
  - Uncorrelated functions integrated into a single application
- System Integrator responsible for
  - successful co-existence of software applications on same computer
  - Extensive system integration and V&V
  - All SW applications have same criticality class
  - Fault detection and containment is at SW level
  - Combinatorial explosion in numbers of test cases
  - Complete flight software must be subjected to functional validation as a single component

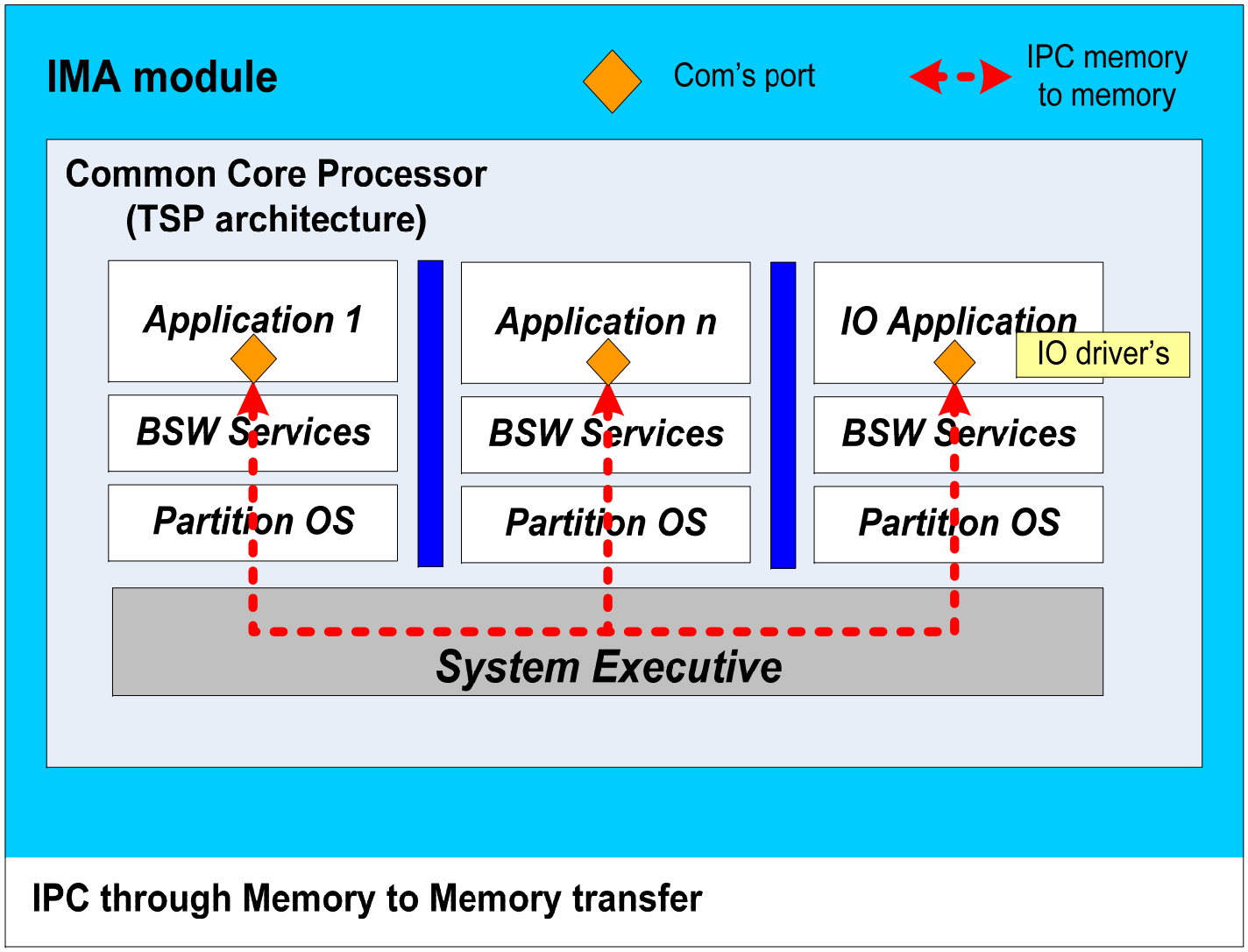
A potential Solution
Integrated Modular Avionics (IMA)

- The primary objective of IMA concept in the aviation domain is to achieve an integrated system architecture, which preserves the fault containment properties and development 'separation of concerns' of the federated systems.
- Benefits to the aviation domain
    - Saving on mass, volume and power
    - Maintenance and component obsolesce
    - Retaining multi vendor
    - Retaining parallel development
    - Retaining incremental validation
    - Fault containment (retaining federated system properties)
- IMA concept has evolved since early 1990s.
    - Used across most major airframe integrators
      e.g. Boeing (777 & 787) and Airbus (A380).
    - Supported by ARINC standards ((DO-297, DO-178B, DO-248, ARINC-629, ARINC-664 (AFDX), ARINC-653 (APEX API) etc.)

Time and Space Partitioning

- The IMA architecture uses software Time and Space Partitioning to share a single computing platform between multiple applications
    - Ensures safety (and security) of S/C functions
- An application is a set of cooperating tasks which together perform a coherent function e.g. Attitude and Orbit Control
- Partition defined as an allocation of resources to an application
    - Memory space → spatial partitioning
    - CPU time → temporal partitioning
    - Access to I/O device and bandwidth
    - CPU privilege mode (System Partitions → Supervisor Mode)
    - Communication via ports and channels

Time & Space Partitioning Architecture



TSP Software Architecture

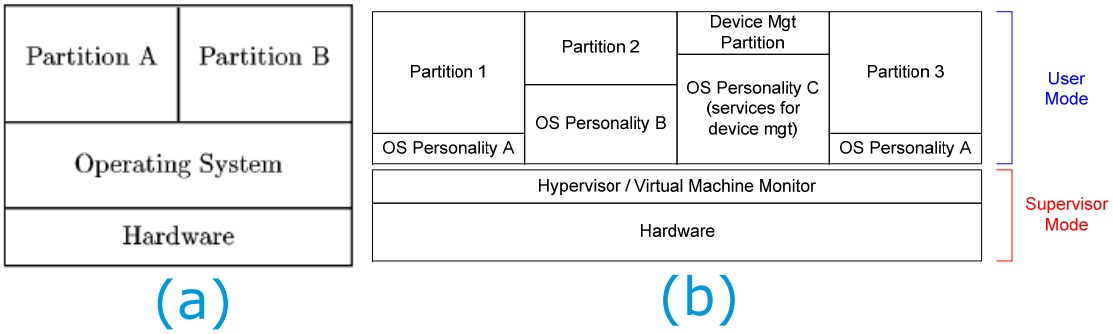
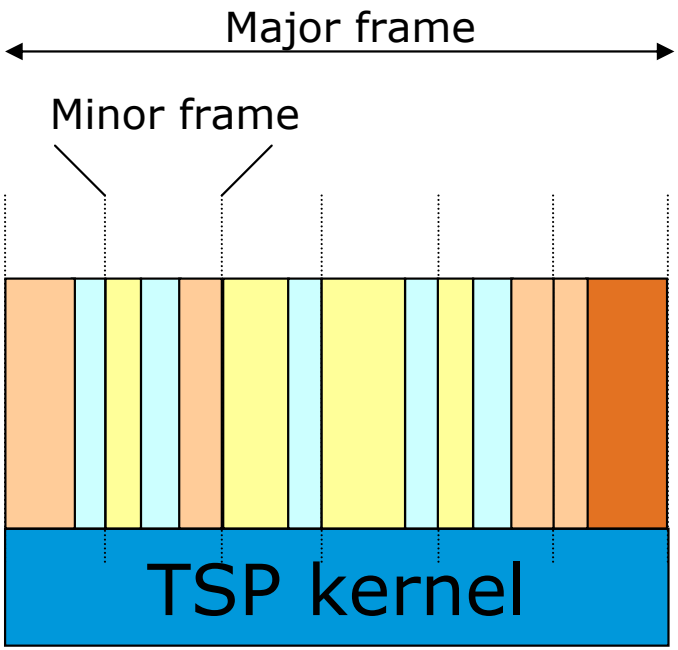


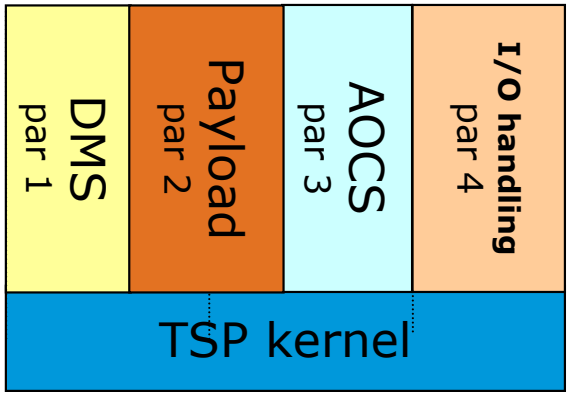
Diagram taken from "Partitioning in Avionics Architectures: Requirements , Mechanism and Assurance", John Rusby. 1999, Technical report NASA/CR-1999-209347, Computer Science Laboratory, SRI International, USA

a) Full Operating system provides all scheduling services. The partitions implement the application programs (in user-mode)

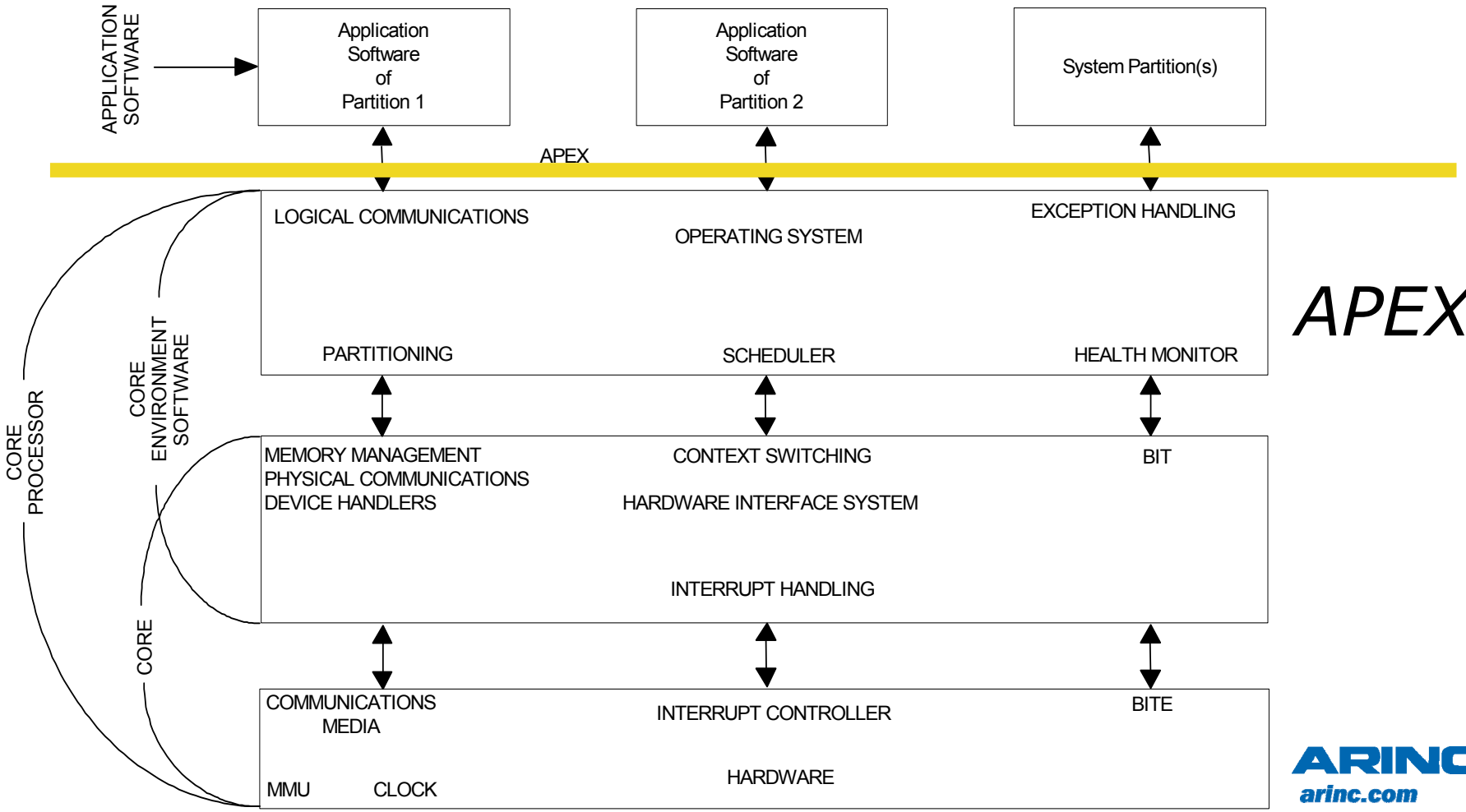
b) Virtualisation: the kernel is typically a µ-kernel, and Operating system services are integrated locally within the partition. The kernel provides the partition scheduling services.



AOCS – Attitude and Orbit Control System
DMS – Data Management System



ARINC 653 Structure - System Architecture



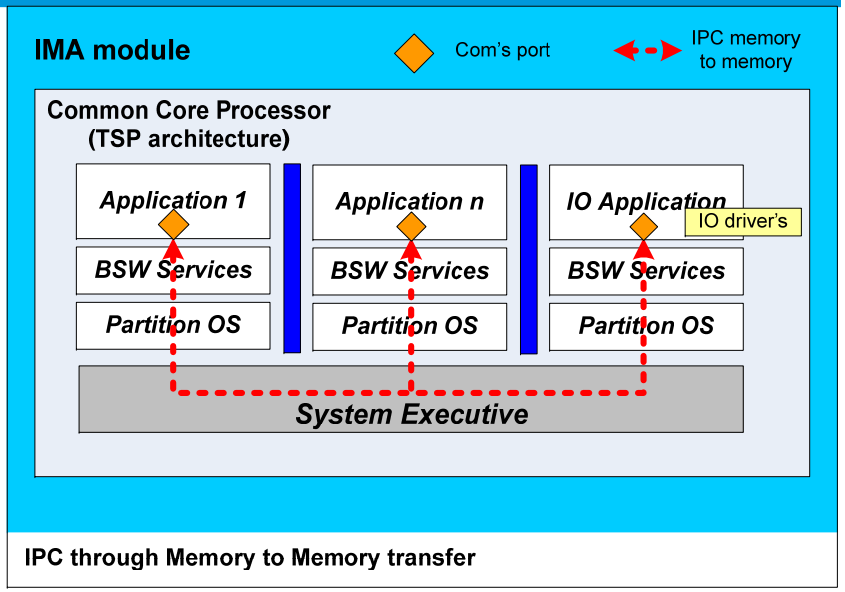
Communication Principles in TSP systems

Intra-partition communication:

- Communication between 2 tasks, internally in one partition

Inter-partition communication (IPC):

- Communication between 2 applications on the same computer



IPC through Memory to Memory transfer

Communication with I/O devices to ensure interface with external systems

- To ensure a robust partitioning, IPC uses Message Passing technique:
  - Physical sharing of memory is doable but not used in IPC especially because of Fault Containment violation risk.
  - **Message Passing technique** is recommended by the ARINC653 standard (APEX standard)

Inter-partition communication

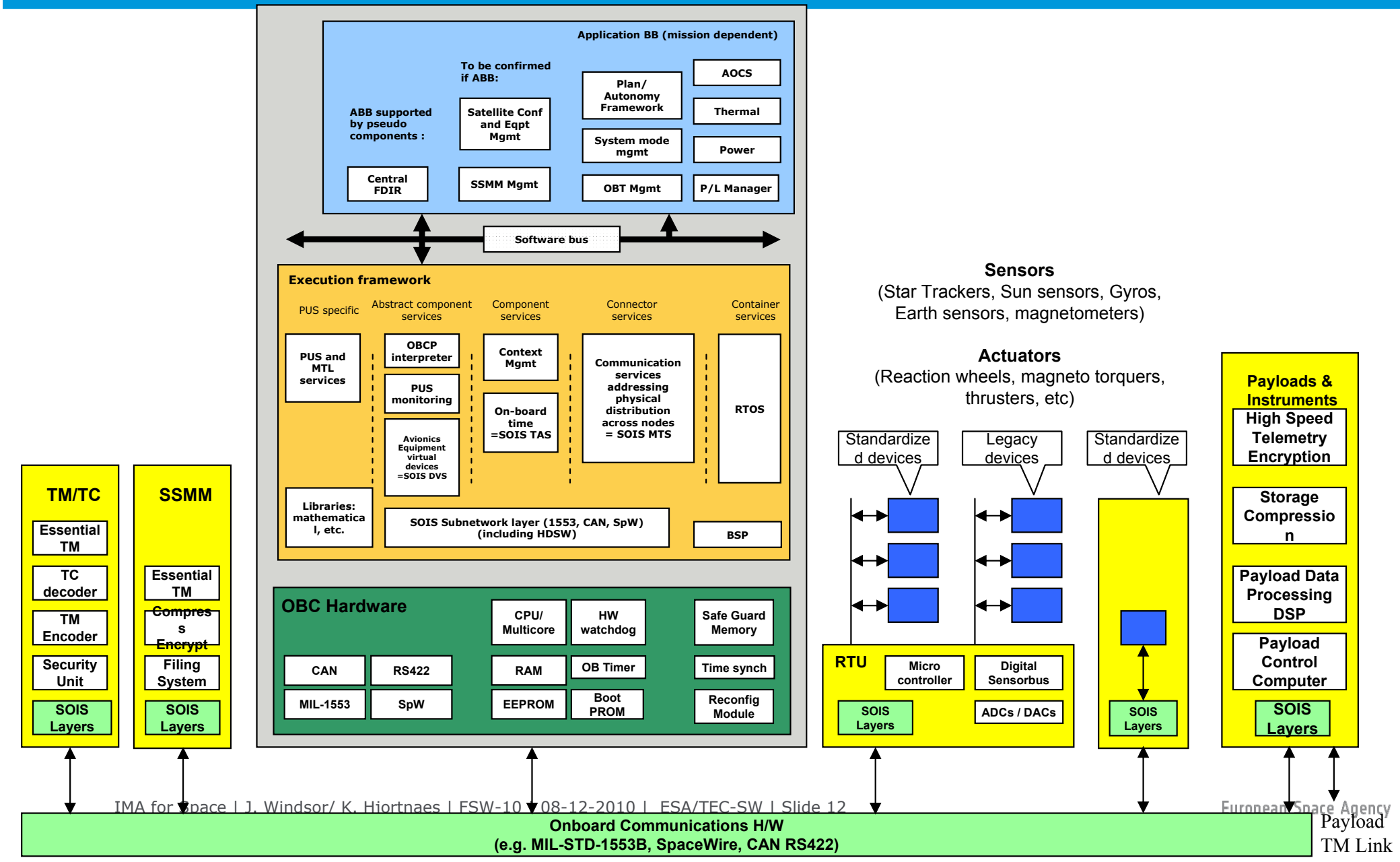
- This message passing process is based on “Ports” and “Channels”, and services provided by the SE
  - Channel: A channel is a logical link between one source and one or several destinations. It also describes the characteristics of the message to be sent
  - Port: Applications have access to channels via defined access points: the ports. A port provides the required memory locations that allow a specific application to either send or receive messages in a specific channel
- The ports and channels are statically defined at system configuration time via a Configuration Table. During this configuration phase, the relevant Port and Channel attributes are created (Name, Length of message…)

Context

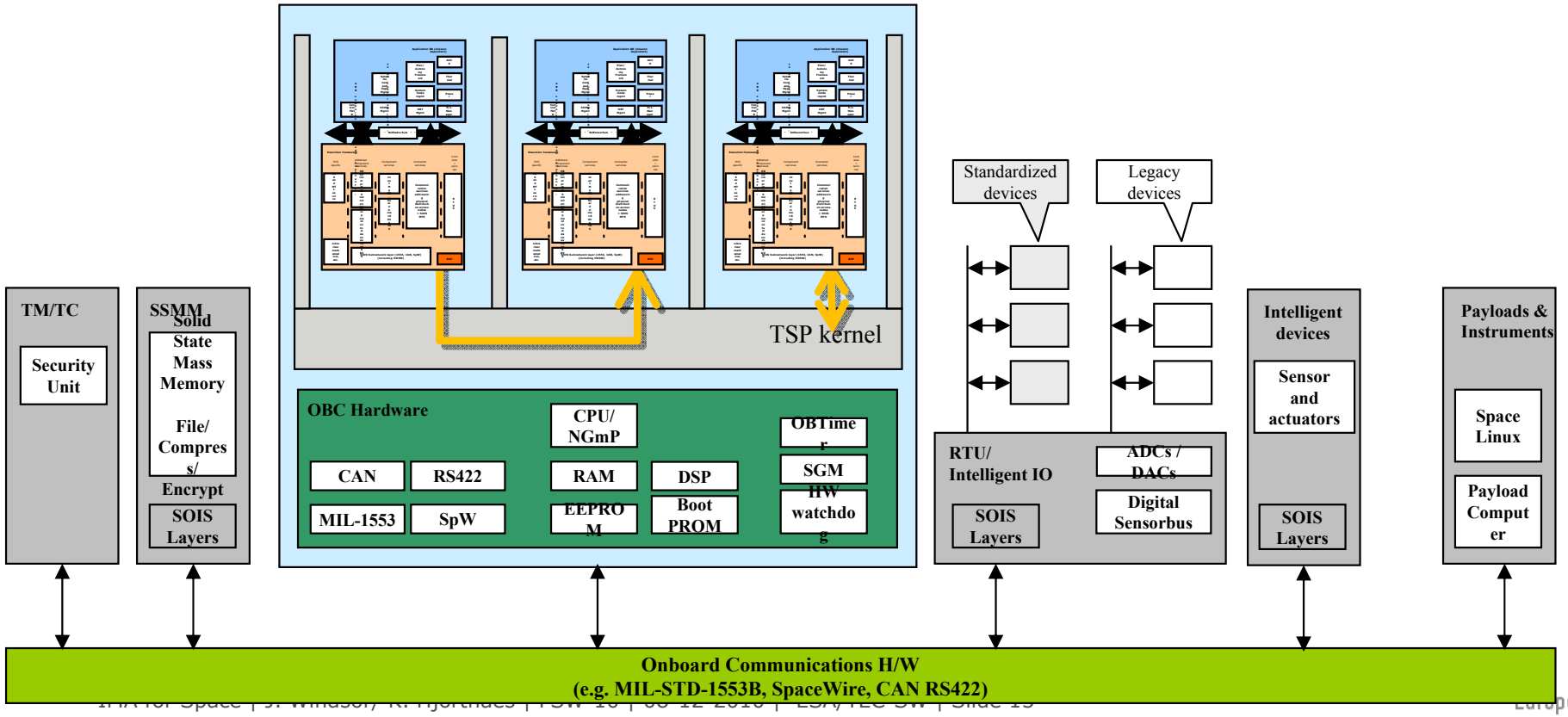
SAVOIR initiative

- Reference architecture
- SOIS communication stack

Conceptual Reference Architecture and Building Blocks

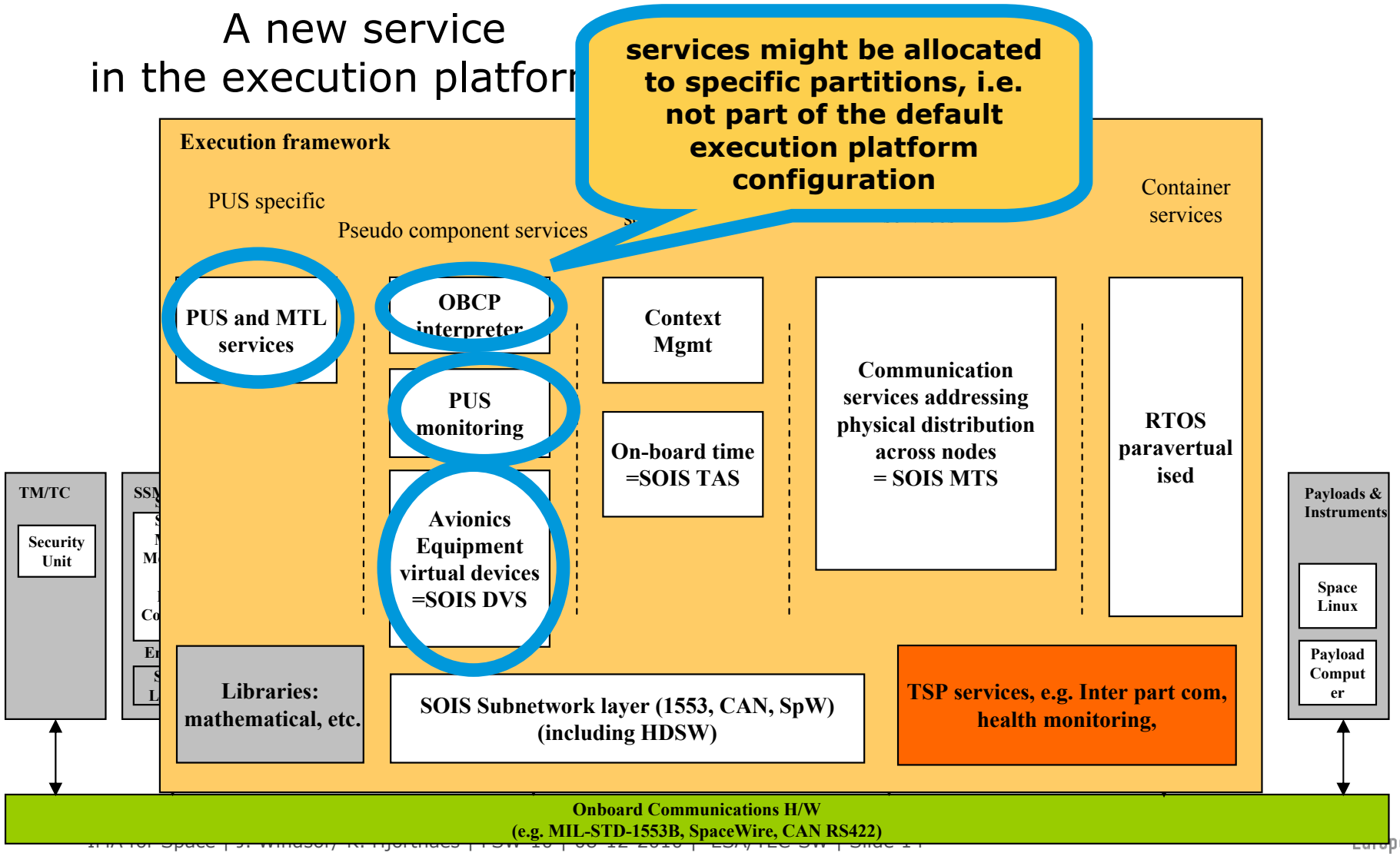


Time & Space Partitioning and Software Reference Architecture

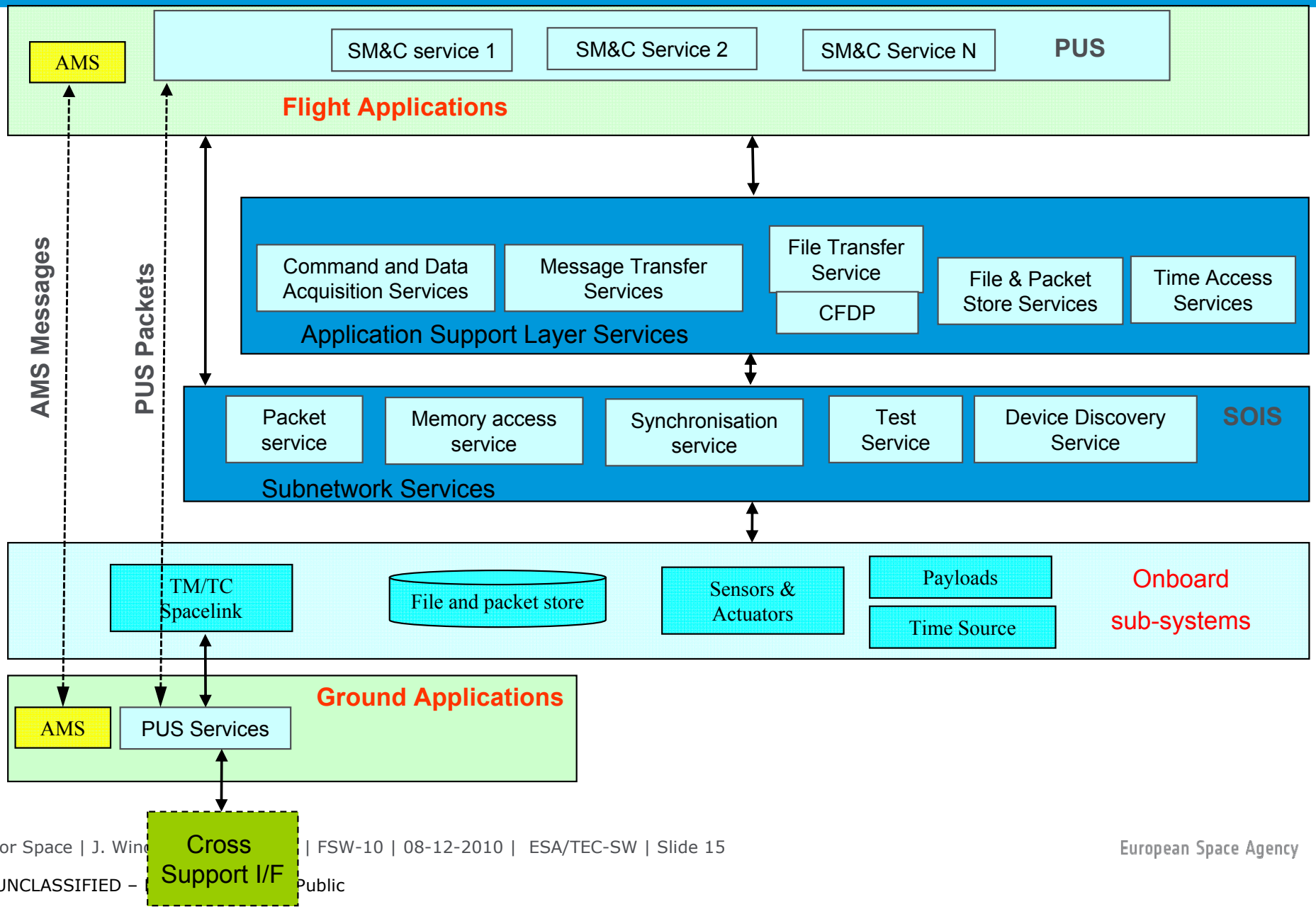


Time & Space Partitioning and Software Reference Architecture

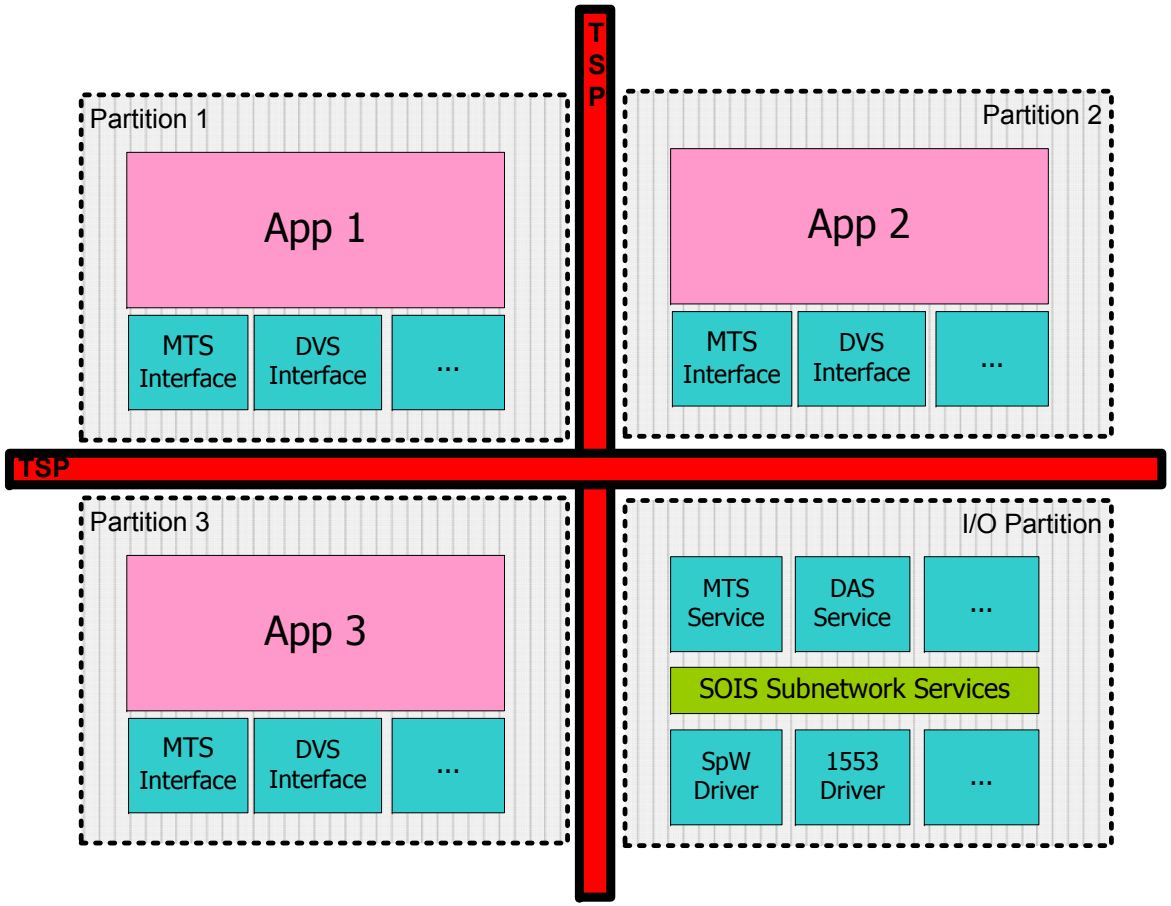
A new service in the execution platfor



SOIS, PUS, AMS and SM&C – Relationships



Example: I/O Partitions and SOIS



- One or more I/O partitions
- Implementations of SOIS service in I/O partition
- Interfaces to services in application partitions
- Allow full reuse of SOIS services
- Transparent to applications

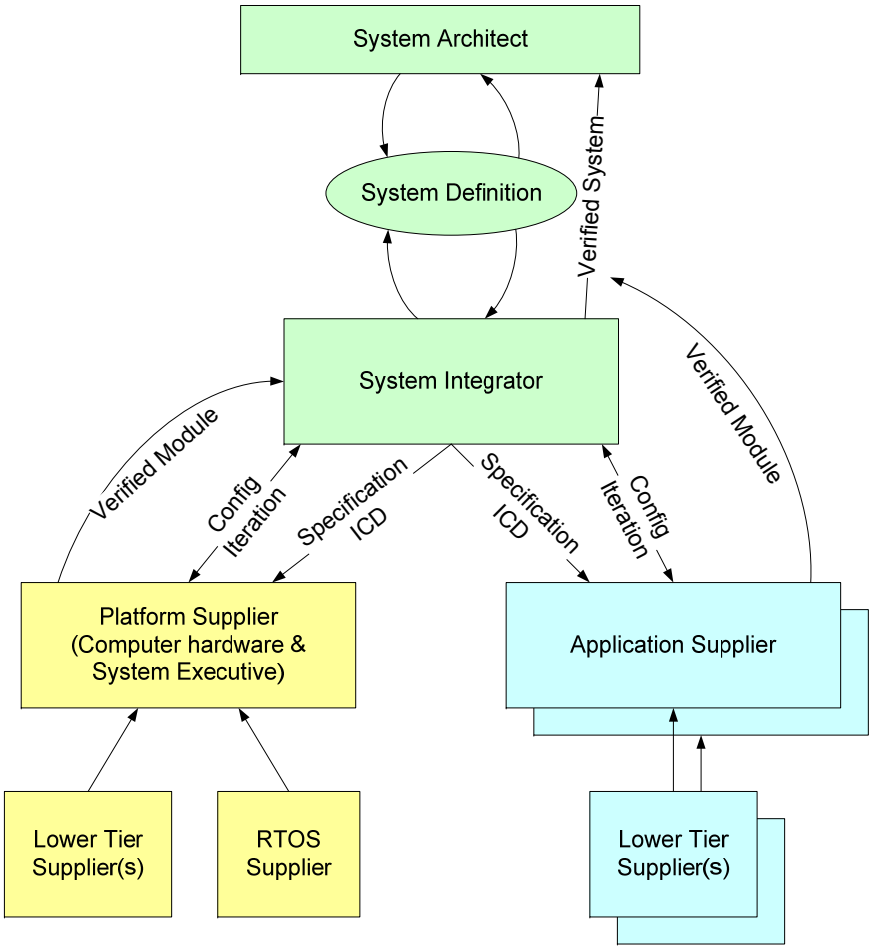
Requirements & Constraints

Constraints

- In assessing IMA for Space the following constraints are considered
  - Clear role definition – who does what in the software lifecycle.
  - Multi vendor support
  - Shall be implementable on existing hardware platforms
    - Low on computing resources
    - Small memory
  - Shall facilitate the emerging product philosophy as defined under the SAVOIR initiative.
  - Shall be compliant to the SOIS communication philosophy
  - Shall identify the differences in context between space and aeronautic applications e.g.
    - Maintaining a running system

Role definition

- System integrator
  - System Architect
  - Component integrator
- TSP-platform supplier
  - MMU/BIOS supplier
  - TSP system Executive supplier
- Application supplier (functional chains)
  - Hosted applications only
  - Hosted applications & peripheral hardware.



IMA for Space (IMA-SP) System Requirements

- Real time execution environment for applications
  - Support for the CORDET Reference Architecture
- Virtualisation of the avionics platform
  - Provides a safe separation of computing resources
  - Support for SOIS
- Enforces separation at run-time of the resources allocated to the applications
- Supports application-to-application & application-to-spacecraft functional interfaces
- IMA paradigm enables following quantities
  - Independent lifecycle for each application
  - Ensures continuous operational availability of the software functions with non-propagation of faults

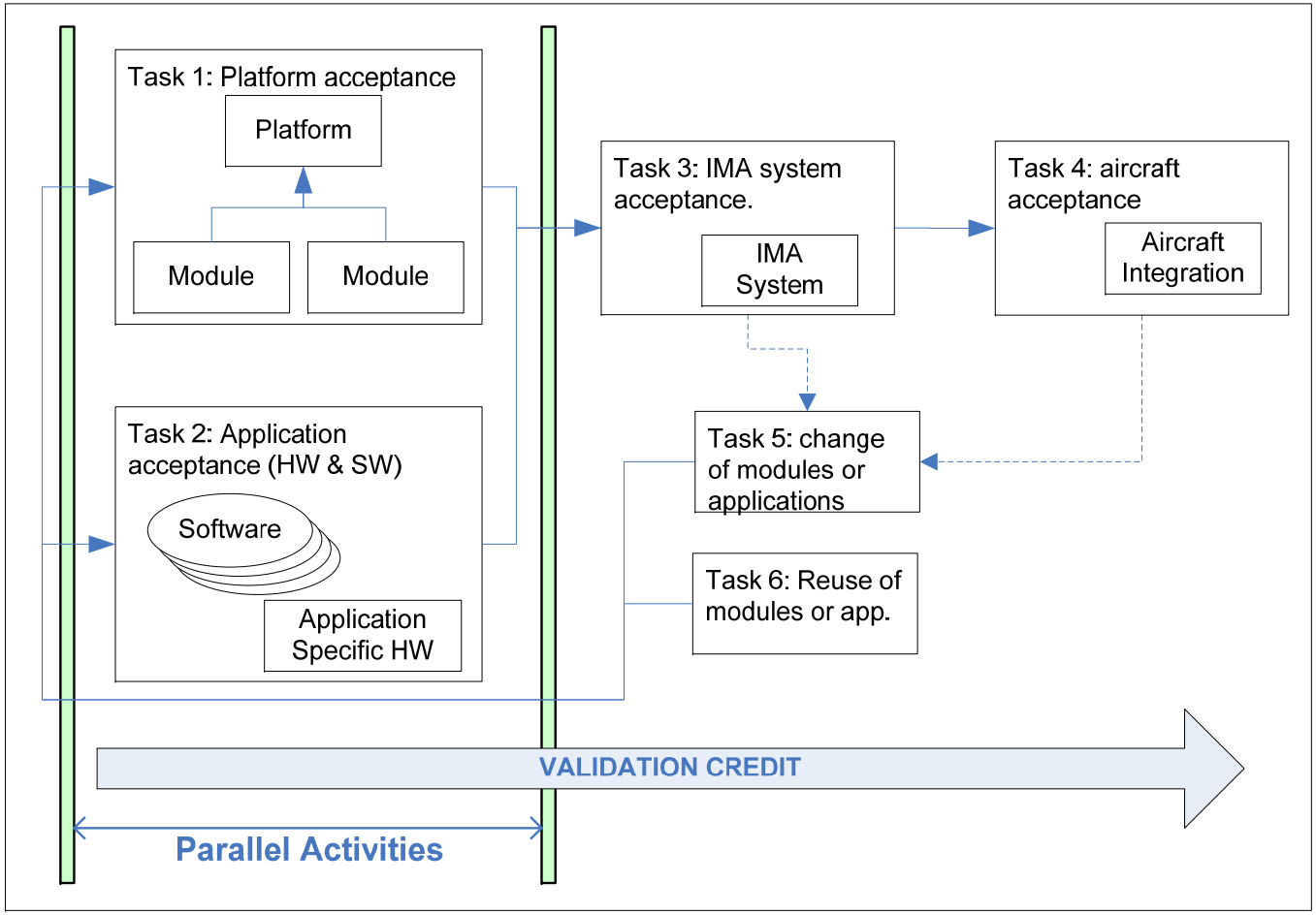
System Requirements
Independent lifecycles for each applications

- Separate function lifecycle
  - Limits inter-dependency of the lifecycles of the applications
  - Maximum flexibility and efficiency to application stakeholders

  Challenge is to allocate resources early enough to allow for decoupling of application lifecycles

- Incremental system lifecycle
  - Accumulate qualification credits in independence from the integrated system
- Criticality
  - Integration of applications with differing criticality levels in one computing platform.

Incremental Validation / Qualification Credits



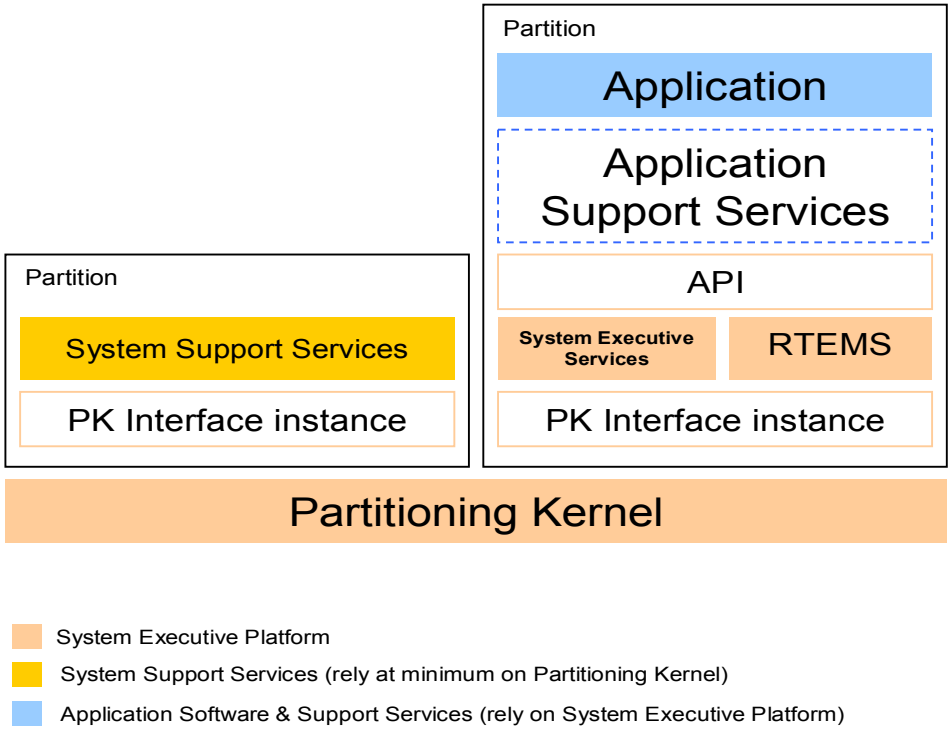
System Requirements
Operational availability of applications

- Adaption to contingent situations
  - Failure in an application is detected/recovered without impacting availability of other applications
    - Automated recovery (restart/switch to backup) or stop
- Operational maintenance
  - Partition management: stop, unload, load, start, patch, pause
  - *Problems with functional couplings between application*
    - Architecture must allow partition 'disconnect'
- Adaption to mission schedule
  - Start/stop applications based on mission phase

Architecture

IMA-SP Architectural Design

- Application Support Services
  - Provides application services which are not available from either the kernel or RTEMS, e.g. SOIS support
- API – provides ARINC 653 services to the application
- System Executive Services – implements API services not included in RTEMS or kernel
- System Support Services
  - executes in dedicated partition(s)
  - 'standardised' system support functions
  - e.g. IO handling, failure detection & recovery, platform functions (power, thermal), system services requiring supervisor privileges (e.g. patch/load mgt)
- System Executive Platform
  - Partition Kernel, RTEMS, System Executive, Services, & API



IMA-SP Platform Requirements (General) I

Defines requirements on the IMA-SP Platform based on needs of the applications (the user)

- Mode requirements
  - Shall implement various operational modes
  - Privileged applications shall be used to switch between modes
  - Mode transitions shall be deterministic
- Time services
  - Access to the current Spacecraft Time
  - Wait until a given Spacecraft Time event occurs
  - Wait for a given time period
- Access to on board data stores, e.g.
  - Housekeeping parameters
  - On board state vectors
  - Mission Time Line
  - Images for patches

IMA-SP Platform Requirements (Failure Handling) II

- Allocation of basic physical resources
  - Predictable CPU resource and memory allocation scheme
  - Failure of an application shall have no impact on integrity and availability of physical resources
- Flight software maintenance
  - Central maintenance service with memory dump and patch operations
  - Application software image reprogramming (partial or full)
  - Kernel patching → patch non volatile memory & computer reset
- Fault protection service
  - Manages system state, individual partition state, interface with external reconfiguration HW
  - Partial system reconfiguration
    - Failure detection & recovery at partition level
  - Full system reconfiguration
    - Hardware level

IMA-SP Platform Requirements (Observability) III

- System Interface
  - Allows interaction with platform services or other applications through well defined interfaces
- On Board Events
  - Raise and access on board operational events
  - Enables synchonisation across applications
  - Events can equal Alarms → failure detection
- System and Software Observability
  - Produces set of monitoring and control parameters for ground
    - Partition state, IMA-SP platform configuration
    - Operational events/alarms, reconfigurations

Conclusion

- The IMA architecture spin-in is a strategy that addresses
  - Separation of concern for the development phase
  - Incremental validation
  - Facilitates design for re-use
  - Fault containment
  - Allows different criticality / security classes to coexist within the same computer.
  - Management of the growth of software functionality

- A good option to handle growing software complexity

acknowledgement

This presentation reflects ongoing R&D work done by the European Space Agency in corporation with European aerospace industries as

-**Astrium Satellite,**

-**Thales-Alenia-Space,**

-**SciSys,**

-**Spacebel,**

-**SysGO,**

-**University of Valencia,**

-**GTD,**

-**GMV-Skysoft**

THANK YOU

Kjeld Hjortnaes, James Windsor
European Space Agency
James.Winsor@esa.int
Kjeld.Hjortnaes@esa.int