SAVOIR Communications Architecture

Final Presentation Days

ESA / ESTEC
08/12/2015

83230347-DOC-TAS-EN-003

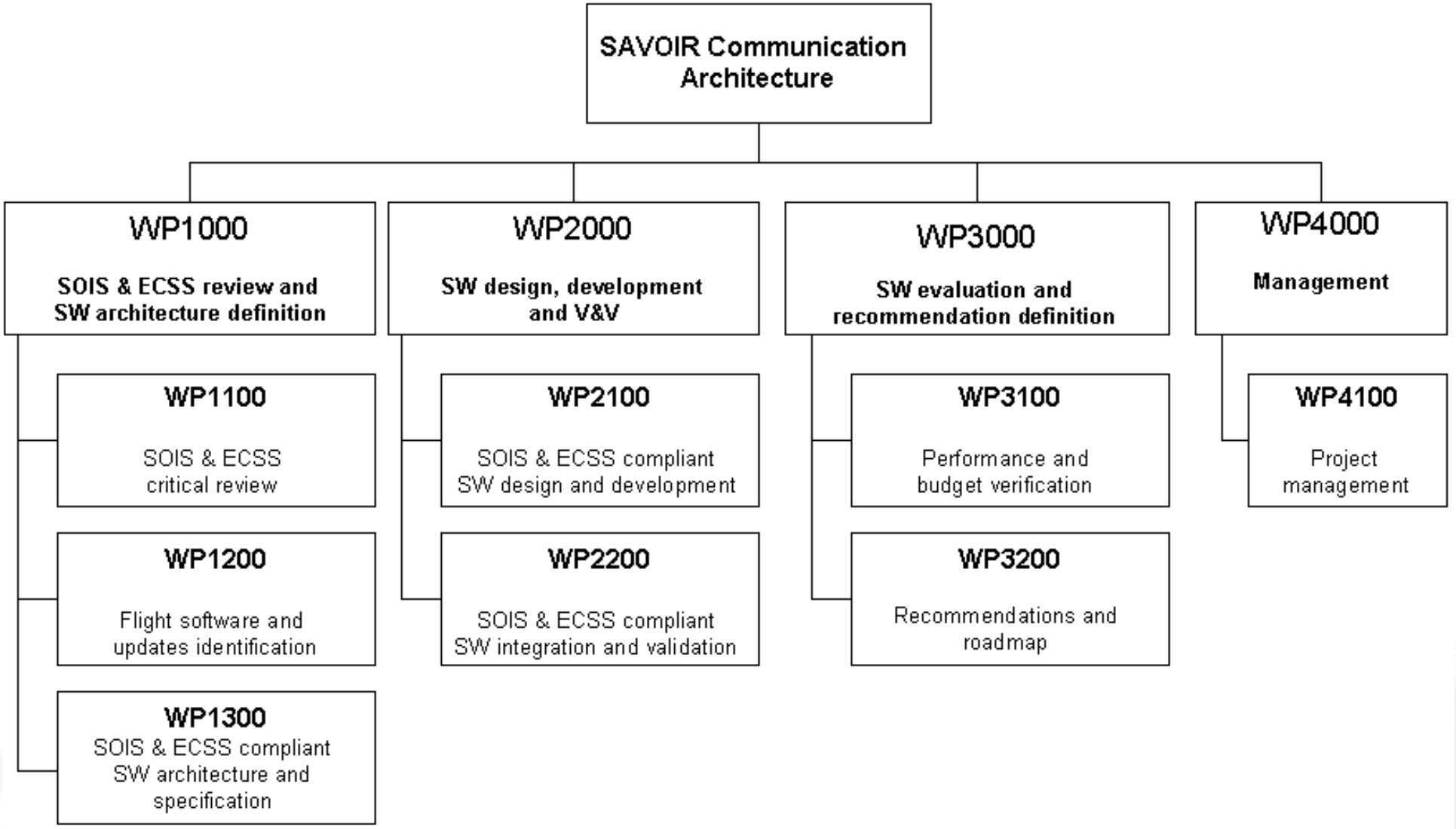
Outline

- On the SAVOIR Communications Architecture GSTP
  - Goals, WBS
- Short introduction on SOIS
- Identification of the software updates
- Architecture, design and implementation phase
- Validation phase
- Performance assessment phase
- Conclusions and recommendations

Goals of the GSTP

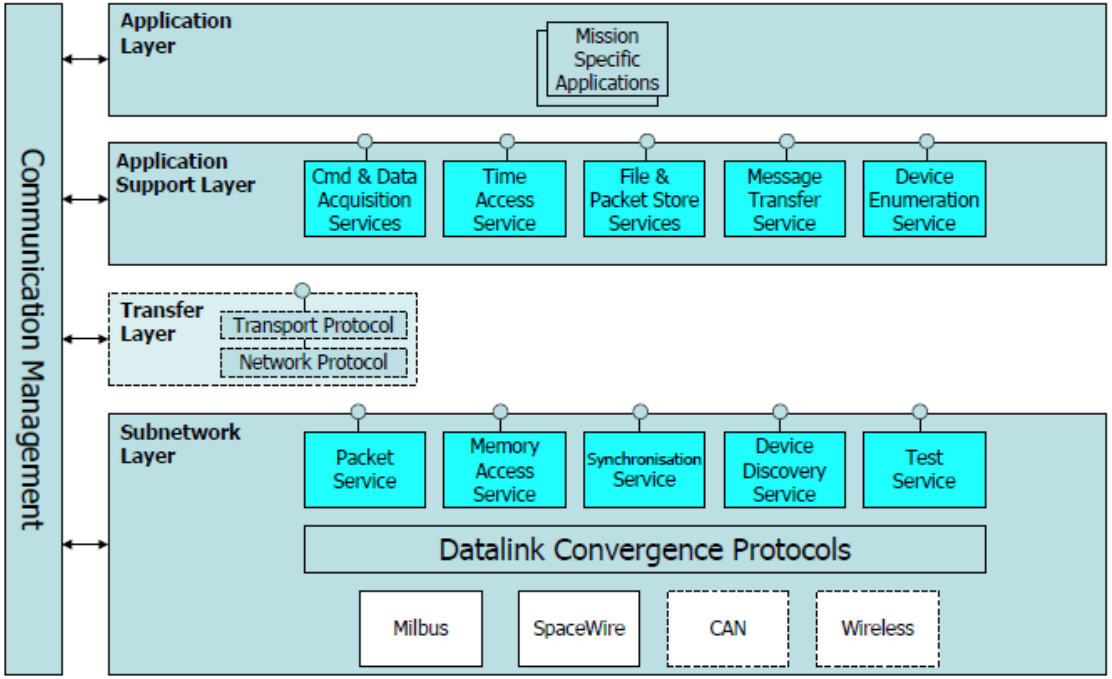
- SOIS recommended practices assessment
  - To evaluate the current status of the SOIS recommended practices and provide a critical assessment of them
- SOIS stack design and implementation
  - To identify an existent reference OBSW of TAS (FSW 1.0)
  - To identify what parts of the OBSW could be replaced with an implementation of the SOIS recommended practices
  - To design and implement a new OBSW encompassing an implementation of the SOIS communication stack (FSW 2.0)
- SOIS stack validation
  - To validate FSW 2.0 with appropriate validation tests
- SOIS stack performance assessment
    - To assess the performance of FSW 2.0
- To provide an evaluation of the SOIS implementation, recommendations on it and a roadmap for its adoption

Work Breakdown Structure



SOIS basic constituents

A static service architecture



OSI-level service primitives definition

READ.request (
MASAP Address,
Destination Address,
Transaction ID,
Memory ID,
Start Memory Address,
Size,
Priority,
Channel,
Authorisation (optional))

READ.indication (
MASAP Address,
Destination Address,
Transaction ID,
Memory ID,
Start Memory Address,
Size,
Priority,
Channel,
Data,
Result Metadata)

Service configuration (often non normative)

MIB

The reference mission for the GSTP (I)

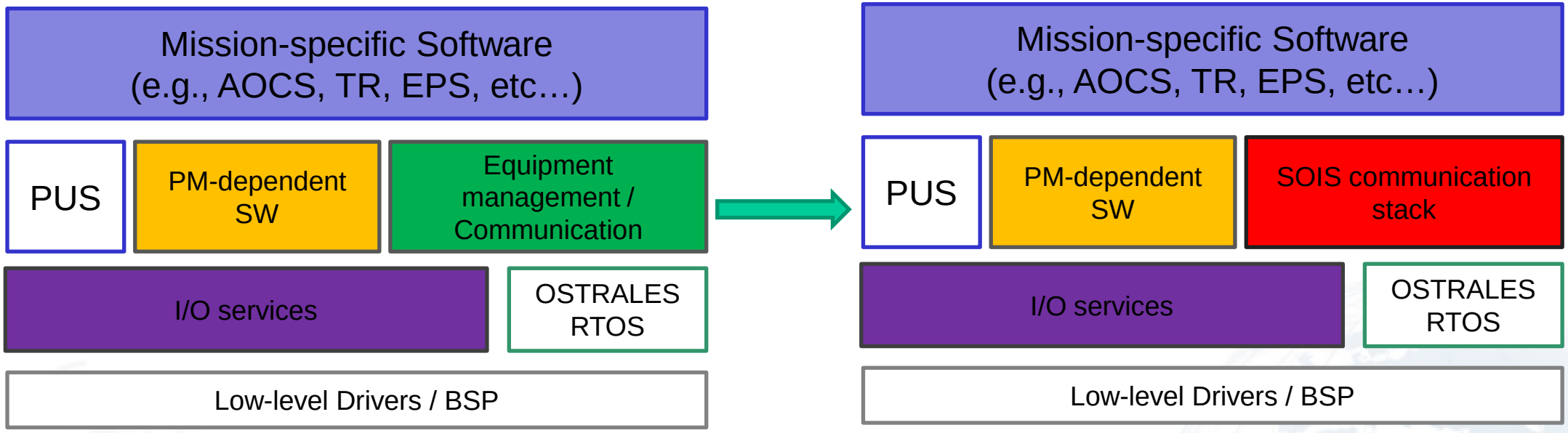
- AOCS Sensors and Actuators
  - 1) On dedicated SpW links
  - 2) Acquisitions / Commanding via remote I/O boards
    - Connected to the OBC via SpW
    - Use of acquisition lists
- Complex payload
  - With dedicated processor
  - Payload RTU comprising equipment with commanding / acquisition via the platform software
- Overall: 2 different SpW links, 1553B, discrete I/O links

The reference mission for the GSTP (II)

- At the time of the start of design activities in the GSTP
  - The architecture and design of the reference mission were already completed
- At the time of the start of implementation in the GSTP
  - The full OBSW was considered ~80% completed
  - All the relevant parts related to SOIS were already 100% completed
  - Relevant validation tests for the part addressed with SOIS were available from the reference mission
  - Relevant versions of TM/TC plan and SDB available

Identification of the software updates

- Identified scope of the study
  - To replace the full equipment management and communication layer of the reference mission with a communication stack based on the implementation of the SOIS recommended practices

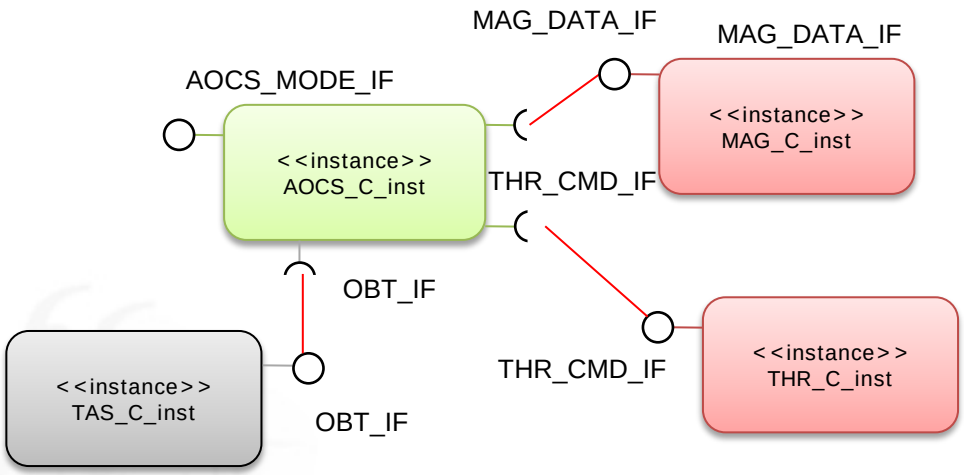


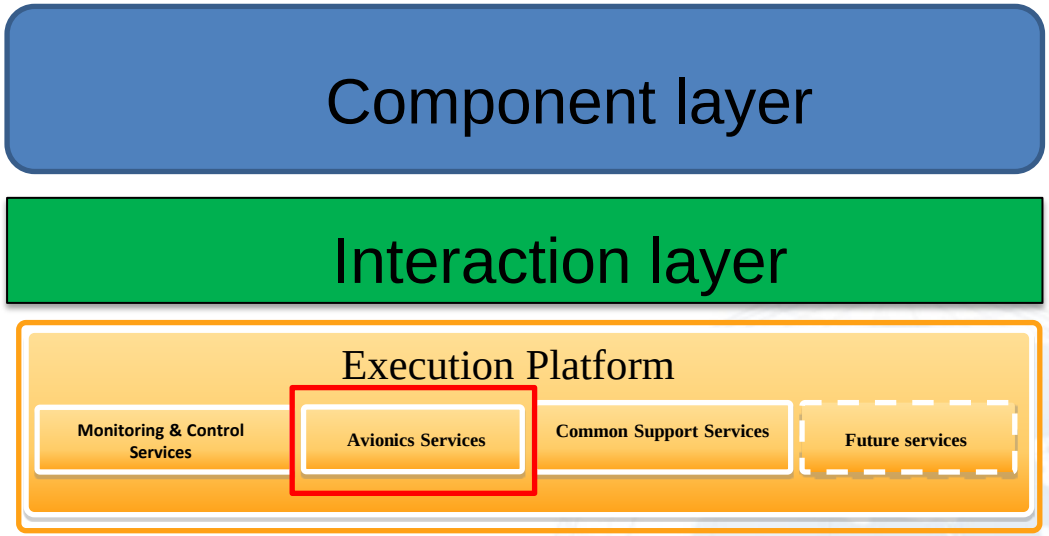
FSW 1.0

FSW 2.0

Principles, goals and challenges (I)

- We tried to implement the SOIS recommended practices in a manner that is compatible with
  - An OSRA-compliant development process
  - Users of the stack are mostly components designed with OSRA-like principles





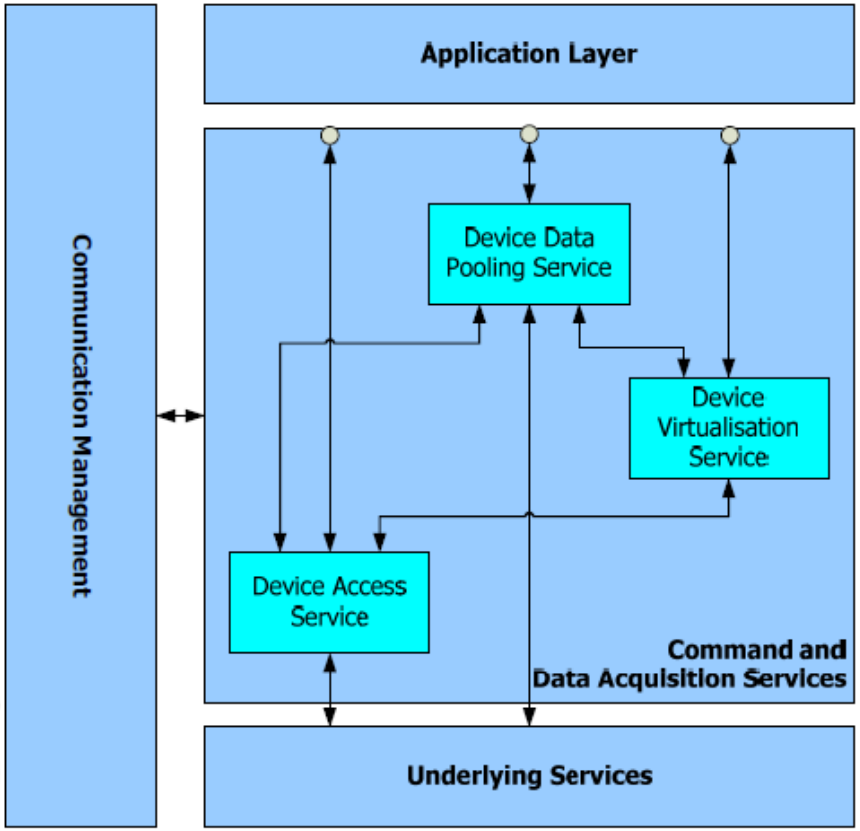
* Figures: Copyright ESA 2014-2015 – OSRA Training Material

Principles, goals and challenges (II)

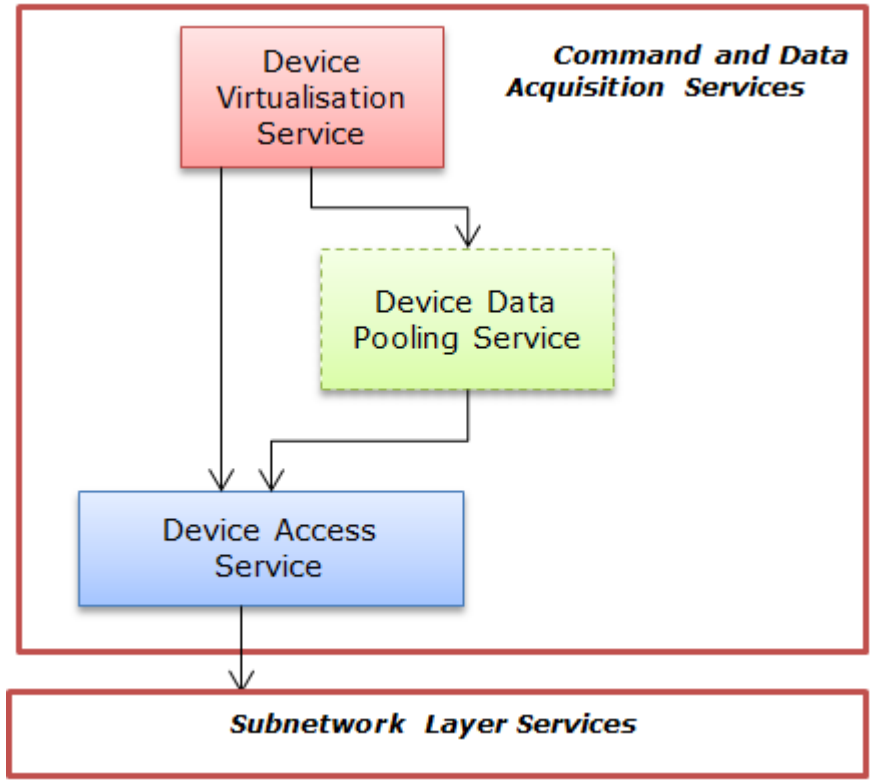
- Goals and challenges
  - Can SOIS easily support current and future avionics technologies and network topologies?
    - Use of RTUs, equipment on communication links more complex than a point-to-point SpW
    - Support for arbitrary redundancy and cross-strapping
  - How can we relate to existing FDIR mechanisms and policies?
  - How do we manage aspects related to the dynamic architecture?
  - How well can we plug the SOIS implementation into an existing TM/TC dispatching system?
  - How can everything be satisfactorily put in place in an existing model-based development process?

Application Support Layer (I)

Original SOIS CDAS architecture



Modified CDAS architecture



Application Support Layer (II)

- Advantages of the new CDAS architecture
  - Device Data Pooling Service (DDPS)
    - Used to decouple access to acquired data from the acquisition
      - In application of the separation of concerns principle
    - Essential to manage acquisition lists on I/O boards or remote RTUs
  - User applications access data systematically via DVS
    - They shall not be concerned about how this data is acquired and on the aggregation in terms of acq lists of those data
  - Access to the subnetwork layer services performed only via DAS
    - Architectural choice to organize systematically communications
    - May incur a penalty in performance

Application Support Layer (III)

Major Service	Service	Use in the GSTP	Comments
Device Virtualisation Service (DVS)	-	Implemented	Refer to modified CDAS architecture.
Device Data Pooling Service (DDPS)	-	Implemented	Refer to modified CDAS architecture. Used to manage acquisition lists.
Device Access Service (DAS)	-	Implemented	Refer to modified CDAS architecture.
Device Enumeration Service (DES)	-	Implemented	Quite unusable as presently defined. Large extension to support redundancy and cross strapping (make it become a full-fledged equipment manager).
File and Packet Store Services (FPSS)	File Access Service / File Management Service	Assessed	No file system used in reference mission. Mapping to CNES “FMS” ~OK.
File and Packet Store Services (FPSS)	Packet Store Access Service	Designed	Considered too low-level, incomplete, sometimes cumbersome to implement. Forces clients to know internal organisation of PS.
File and Packet Store Services (FPSS)	Packet Store Management Service	Not considered	Not considered, as reference mission is operated with PUS ver. A
Time Access Service (TAS)		Designed	Should be extended to account for different types of time sources (wall-clock time vs. logical time in TSP, different time precisions)
Message Transfer Service (MTS)		Not considered	No value added compared to existing mechanisms. Link with AMS brings unnecessary complexity.

Subnetwork Layer (I)

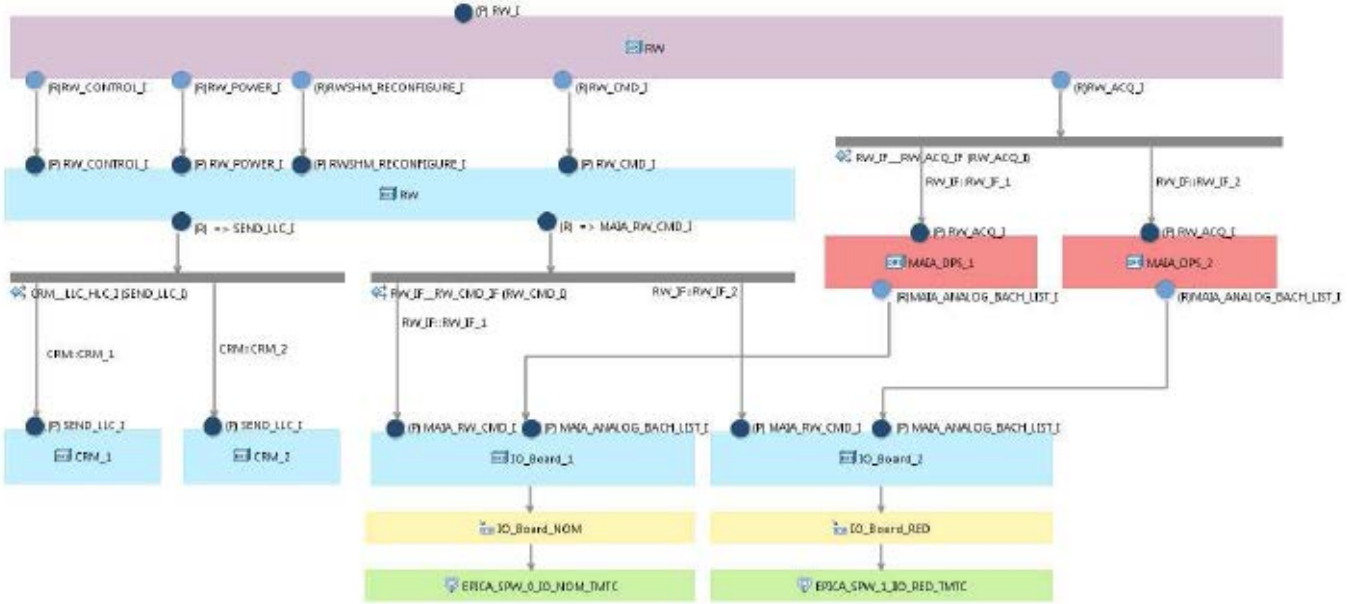
- PS / MAS were implemented to support all communication links
- The relationship between DAS and PS / MAS was quite debated internally during the project
  - Whether buffering of requests at PS / MAS was desirable
  - Implementation-specific interpretation of Channel / Priority to map towards underlying links
    - Mandatory for any complex link or for scheduled communications
  - Whether the DAS / PS / MAS chains are adequate when much of the communication effort is performed by intelligent controllers
  - Relationship between DAS and PS / MAS and their task executors was arbitrarily chosen
    - Lack in SOIS of any view for establishing a dynamic architecture, when necessary
    - An implementation in line with separation of concerns was chosen (e.g., declaration of entry-points, separate allocation to tasks)

Subnetwork Layer (II)

Major Service	Use in the GSTP	Comments
Packet Service (PS)	Implemented	See previous slide.
Memory Access Service (MAS)	Implemented	See dedicated slide.
Test Service (TS)	Implemented, not used in validated binary	Implemented and tested for all communication links but not activated in validated binary.
Device Discovery Service (DDS)	Implemented, not used in validated binary	It indicates rechability of devices. Rather incomplete as building block for an “extended DES”. Implementation not obvious for devices behind an RTU. Implemented yet not used in the validated binary for the reference mission (impact on communication traffic).
Synchronisation Service (SS)	Not implemented	

Application of Model-Driven Engineering to SOIS

- The SOIS implementation was realized by using a (partial) MDE process
  - Model for static service architecture definition, layering and assembly
  - Configuration of relationship between services and redundancy / cross-strapping
  - (Partial) code generation from a DSL



Validation phase

- FSW 2.0 was extensively validated in the GSTP
  - All the original XCheck tests of the equipment layer of the original mission were re-run on the full binary of FSW 2.0
    - 22 XCheck Validation Tests
      - 4-8 validation test cases each
      - For communication links, I/O boards and RTUs, AOCS equipment, payload side processor and instruments
      - For communication test, re-configuration

Performance assessment (I)

- Memory budget
  - +9% in EEPROM size w.r.t. FSW 1.0
  - +12% in RAM size w.r.t. FSW 1.0
  - The increased memory budget is also due to
    - A part of the previous equipment layer of FSW 1.0 was still necessary in order simplify the coupling with TM/TC dispatching
    - Increased space due to certain duplication or increased needs for buffers at different levels of the communication architecture (e.g., DAS and PS / MAS and data convergence layer)
  - We estimate that a more optimized implementation could achieve +5% in EEPROM / +10% in RAM

Performance assessment (II)

- Timing performance was assessed with a complex performance test (representative of a high-load mission scenario)
  - PFSW initialization from PM boot
  - TM encryption activated and HK packets enabled so as to reach the maximum downlink bandwidth
  - Equipment necessary for a nominal, high-load AOCS scenario switched on by TC
  - PCDU switched on by TC
  - High-load NOM AOCS scenario on *closed-loop* is activated
  - Payload processor and payload-side instruments activated
  - Battery Management and Thermal Regulation switched on
  - FDIR on OBC with platform and payload applications monitoring switched on
- Timing Performance assessment based on comparison of results by FSW 1.0 and FSW 2.0

Performance assessment (III)

- CPU performance
  - +9% CPU load overall
  - Applications that are clients of the communication stack take some overhead
    - Slightly higher than expected (13%-25%)
    - But for some of them the absolute overhead is low, making the relative comparison not very significant
  - Tasks involved in activities related to the lower-level part of the communication architecture have an overhead between 5% and 20%
    - Increased number of data copies, especially for 1553B-related communications
  - Execution of TCs that require traversing the full communication stack take a considerable performance penalty
  - It is fair to assess that a part of the performance penalty is due to fitting the SOIS stack in an already existing architecture

Performance assessment (IV)

- Spacewire and 1553B traffic unchanged w.r.t. the original mission
  - As DDS and TS were only implemented and tested but not activated in the final validated binary
    - In order not to perturb bus / traffic schedule and keep the comparison of performance as fair as possible

Conclusions (I)

- SOIS *as it is currently defined*, does not provide significant advantages to our current communication architecture
  - The non-recurrent cost of modification is too high compared to the observed advantages
- The SOIS static architecture has some interesting ideas on layering and separation of concerns that may help improve our reference architecture
  - In particular w.r.t. to the DVS / DAS / DDPS / SL layering
- Little or no value added found in SOIS service primitives specification
  - Too generic, inflexible, they hinder modularity and complexity minimization in the SOIS implementation

Conclusions (II)

- Can SOIS easily support current and future avionics technologies and network topologies?
  - Lack of knowledge on avionics, redundancy and cross-strapping in SOIS is a limiting factor
  - Mapping to scheduled links requires implementation-dependent interpretation of the recommended practices
  - Support for RTUs and their redundancy schema requires quite creative interpretation of the recommended practices
  - Lack of generalized support for equipment management / configuration would make the implementation rather non portable
  - QoS and relationship to message communication classes (e.g., absolute-time message delivery, maximum latency constraints, use of virtual channels) shall be reviewed

Conclusions (III)

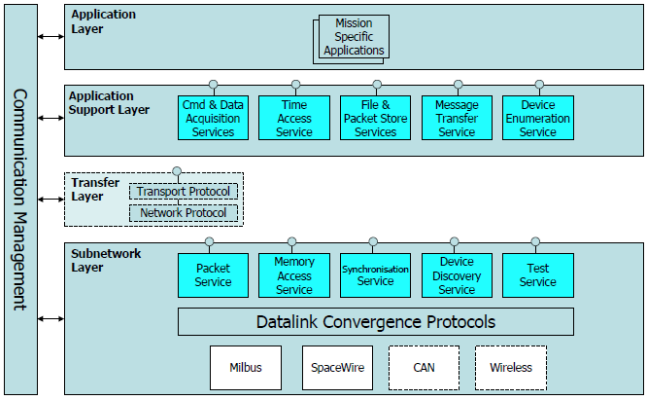
- How can we relate to existing FDIR mechanisms and policies?
  - Either use directly the existing mechanisms of the OBSW architecture (inherently non-portable)
  - Or provide a cleaner model of raised errors, later bound to the FDIR mechanisms of the hosting architecture
- How do we manage aspects related to the dynamic architecture?
  - Buffer allocation, entry points for tasks, support for schedule constraints definition were all implemented with implementation-specific solutions
- How well can we plug the SOIS implementation into an existing TM/TC dispatching system?
  - Concern similar to relationship to FDIR
  - Compatibility to the adopted style of commandability and observability should be analysed

Conclusions (IV)

- How can everything be satisfactorily put in place in an existing model-based development process?
  - SOIS and MDE are partly at odds as they try to optimize in two orthogonal directions
    - (i) SOIS, by providing an inflexible interface forces all clients to align on it and hides the complexity in the *implementation* of the services
    - (ii) MDE keeps flexibility for clients and tries to introduce context-dependent adaptation points in the generated code
      - As well as optimizations which are possible thanks to the model-level knowledge of the provided / requested services and avionics

Recommendations (I)

A static service architecture



OSI-level service primitives definition

READ.request (
MASAP Address,
Destination Address,
Transaction ID,
Memory ID,
Start Memory Address,
Size,
Priority,
Channel,
Authorisation (optional))

READ.indication (
MASAP Address,
Destination Address,
Transaction ID,
Memory ID,
Start Memory Address,
Size,
Priority,
Channel,
Data,
Result Metadata)

Service configuration
(often non normative)

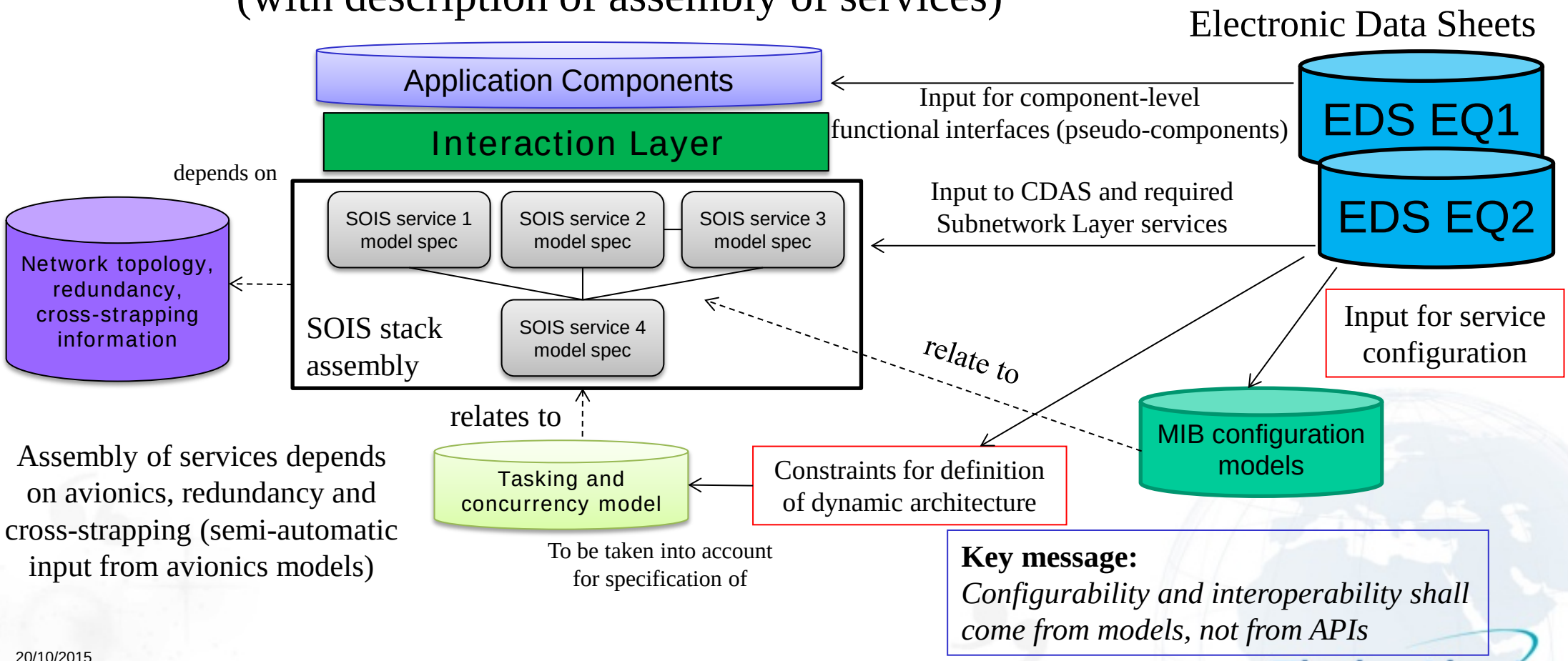
MIB

- Our proposal: re-center the effort for SOIS on
- 1) architecture definition;
- 2) compatibility of the approach with modern OBSW development paradigms (model-driven, component-oriented);
- 3) support of current and future avionics, with associated design process

Recommendations (II)

- SOIS shall not be defined in isolation. Think about the bigger picture!

A static *model-based* service architecture for use with the OSRA (with description of assembly of services)



Key message:
Configurability and interoperability shall come from models, not from APIs

Recommendations (III)

- Envisaged evolutions of the SAVOIR Communication Architecture shall be driven by an “ESA member states” WG
  - Similar to SAVOIR-FAIRE, but with a mandate also on EDS, equipment functional interfaces and relevant avionics aspects (e.g., communication protocols)

- Feedback to CCSDS should be provided, yet we are not confident that big changes to the recommended standards will be easily accepted