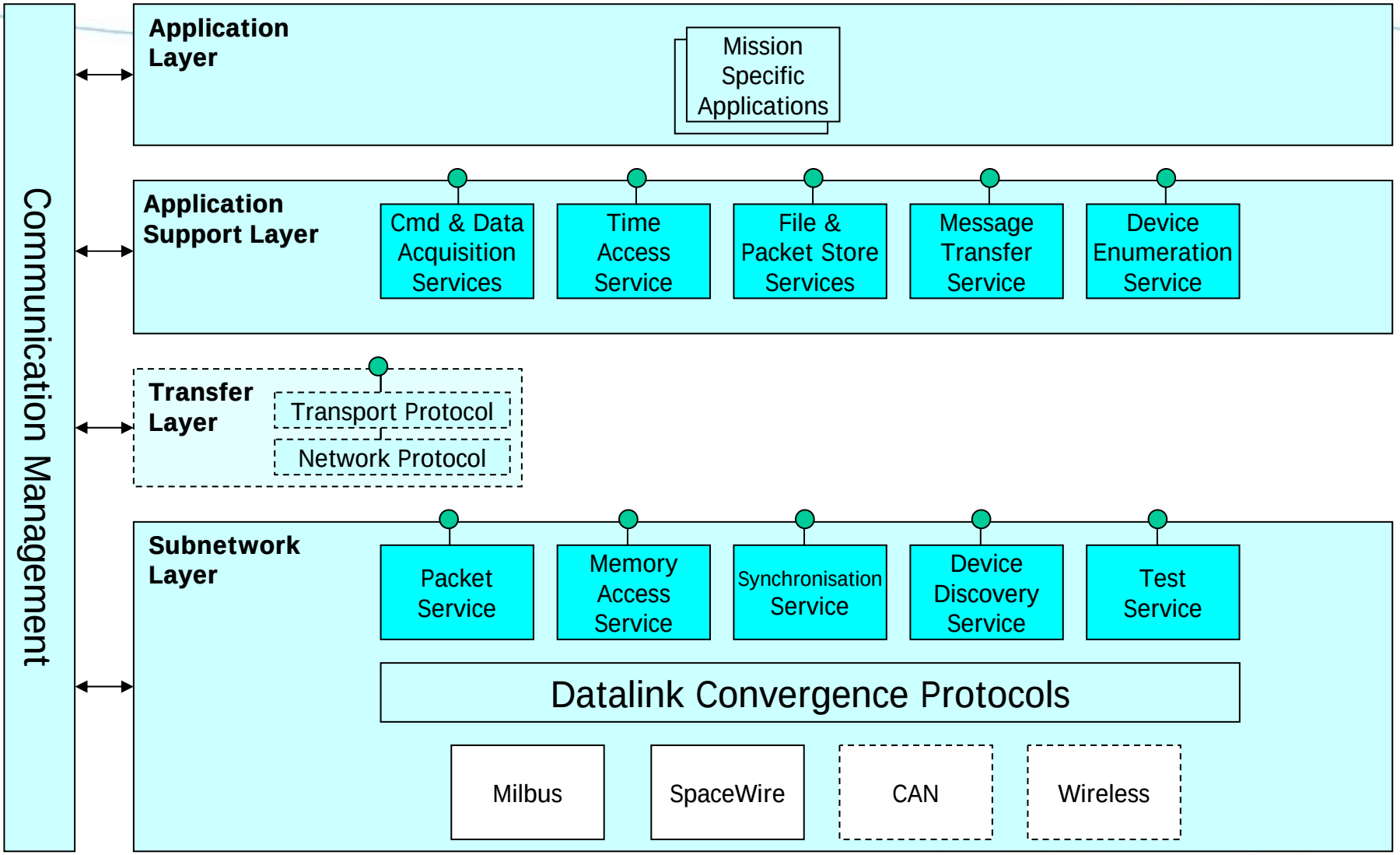
Update on SOIS Plug-and-Play: Device Virtualisation and Electronic Data Sheets

Stuart Fowell, SciSys

ECSS SpaceWire WG #17, 14th December 2011

SOIS Reference Communications Architecture



scisys

CCSDS SOIS Standardisation Status

- **Information Report** "Green Book" 850.0-G-1, Issue 1
  - Issue 2 about to be published

- **Subnetwork Layer Services**

- Published "Magenta Books"
  - Packet Service 851.0-M-1
  - Memory Access Service 852.0-M-1
  - Synchronisation Service 853.0-M-1
  - Device Discovery Service 854.0-M-1
  - Test Service 855.0-M-1

- **Application Support Layer Services**

- Published "Magenta Books"
  - Time Access Service 872.0-M-1
- About to be Published
  - File and Packet Store Services 873-0-M-0
  - Message Transfer Service 875.0-R-1.1
  - Device Access Service 871.0-R-2.2
  - Device Data Pooling Service 871.1-R-2.2
- Under Development "Red Books"
  - Device Virtualisation Service 871.2-R-0.2
  - Device Enumeration Service 871.3-R-0.2

scisys

SOIS Proof of Concept TRP Project Status

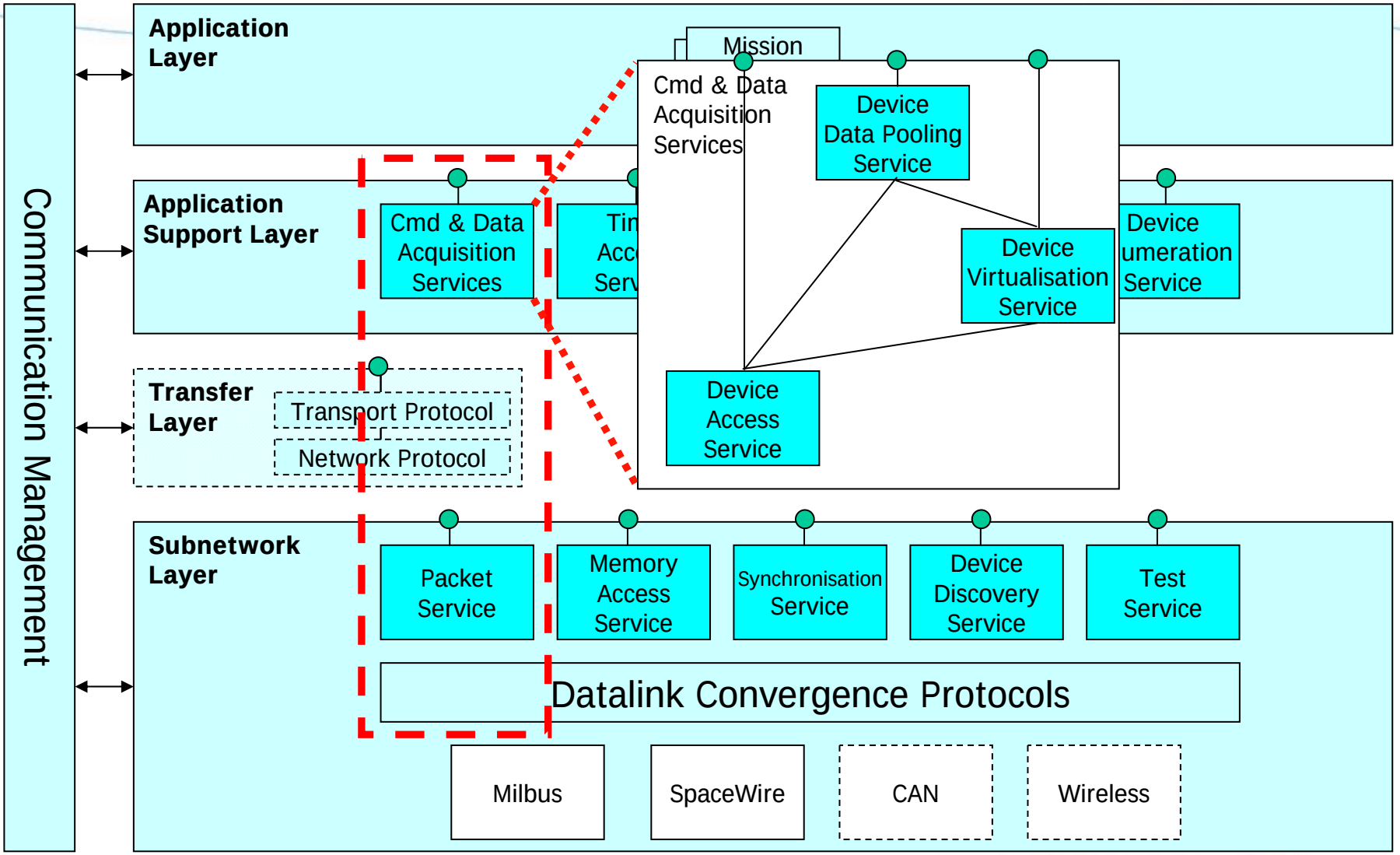
- ANSI C API Tech Note
  - ESA owned and distributable
  - Not CCSDS standard but…
- MISRA C/RTEMS Reference Implementation
  - SpaceWire Subnetwork Services
    - Delivered to ESA for acceptance on RASTA
  - MIL-STD-1553B Subnetwork Services
    - Developed & about to be delivered to ESA
  - Application Support Layer Services
    - Integrated with SpaceWire & delivered to ESA
    - About to be integrated with MIL-STD-1553B
  - Integration with PUS and CFDP
    - Next step

MARC GSTP Project Status

- GenFAS PUS/FDIR Software
  - MISRA C/RTEMS Implementation
  - SOIS
    - SpaceWire Subnetwork Services
      - Subset of SpaceWire-RT
    - Application Support Layer Services
      - Time Access, Message Transfer, Device Access, Device Data Pooling, Packet Store Services
  - Integrated with PUS and FDIR
    - On MARC Hardware
- Currently undergoing System Validation by Astrium UK

SOIS Device Virtualisation & EDS

Location in SOIS Architecture



SOIS Plug-and-Play

- SOIS taking a broad definition of plug-and-play, encompassing design-time activities as well as “run-time” activities
- Goals
  - **Interoperability** – Application and device portability through isolation of the two, permitting flexibility and innovation in both
  - **Adaptivity** – Allow systems to adapt to change, probably more during the development process
  - **Rapidity** – Shorter development times, assisting design, implementation, integration and testing
- Use Cases
  - **Rapid Spacecraft Development** – Development Tool Chain
  - **Automated Integration & Test** – Adaptivity of EGSE
  - **Dynamic Fault Recovery** – Reconfiguration aspects of FDIR
  - **Dynamic Device Migration** – human and robotic missions

Current Approach to accessing to Sensors / Actuators

- Information for accessing devices currently provided in ICDs and Basic Software Device Drivers
  - Paper document
- What is the impact of how its delivered today?
  - Not standardised form
  - Not all information provided
  - Discrepancies between ICD and actual reality?
  - => Difficult to check for discrepancies
- How much device-specific functionality is built into applications?
  - Device Drivers, Equipment Management software function
  - How much changes between missions?
    - Affecting devices, applications, device drivers etc
- How are SOIS proposing to solve it?
  - **Device Virtualisation and Electronic Data Sheets**

Device Virtualisation – Concept

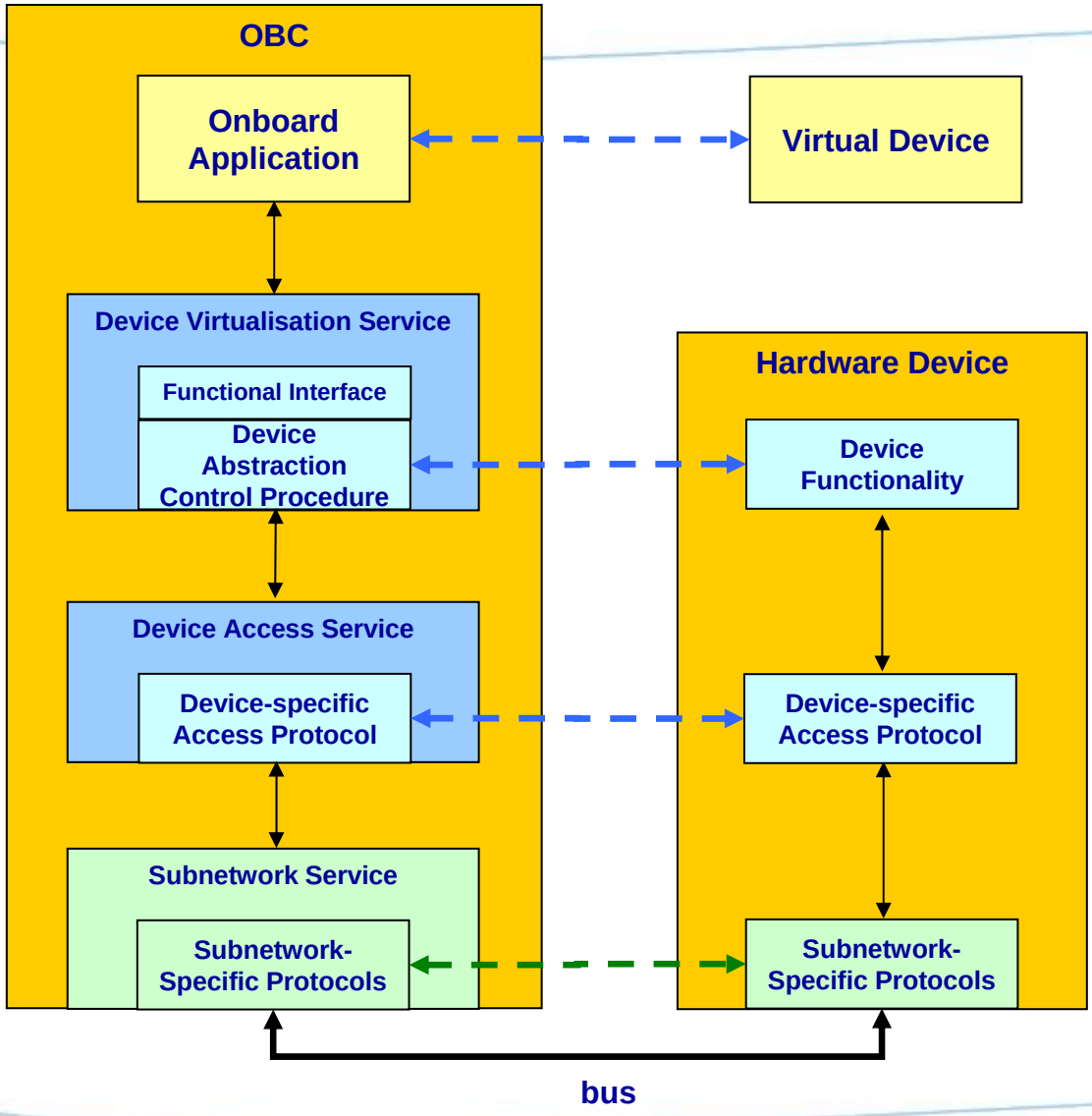
- What is Device Virtualisation?
  - Applications interface with devices at a **Functional Interface**
    - Mapped onto access protocols
    - Not directly using protocols or device representation of commands and data values
    - Aim is for single **Generic Functional Interface per device type**

- What does Device Virtualisation do for you?
  - Swappable devices = reuseable applications
  - Standard access protocols = reuseable devices

- BUT only works if Generic Functional Interface isolates Applications from device-specific characteristics
  - What if the differing characteristics of the different devices affect the algorithms of the applications?
  - Can they be input parameters to configure e.g. control loop?
  - Can't substitute for system engineering

scisys

Interfaces, Protocols and Services



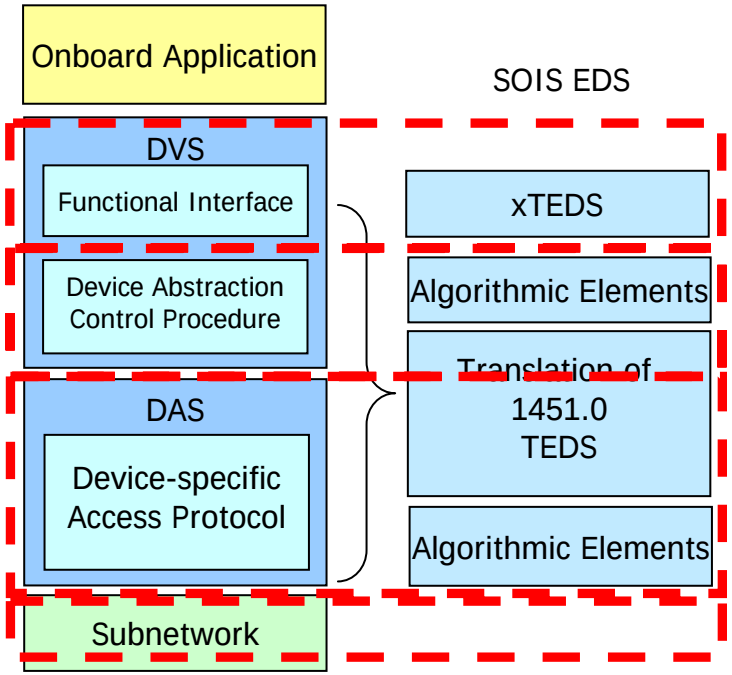
- **Generic Functional Interface**
  - Functionality common to a device type
- **Device Abstraction Control Procedure**
  - How the functional interface is mapped onto the device-specific access protocols
  - Type conversions, operations, state-machine
- **Device-specific Access Protocol**
  - How to command and acquire raw data for specific devices using subnetwork-specific protocols
  - State machine
- **Subnetwork-specific Protocol**
  - How to transfer data to/from device across subnetwork
  - QoS: ack, retransmit, priority etc
  - E.g. ECSS 1553 and SpaceWire

scisys

Electronic Data Sheets

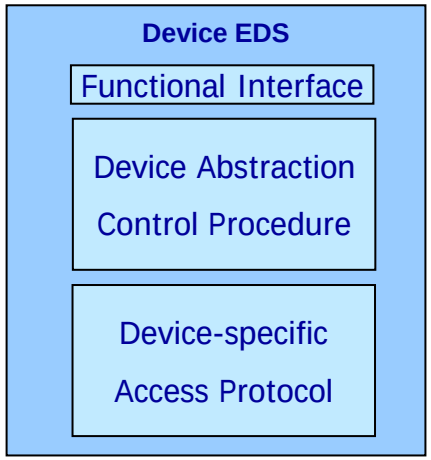
- What are Electronic Data Sheets (EDS)?
  - Electronic specification of information normally found in ICDs
    - But also covers generic Functional Interface
- Doesn't have to be used but provides:
  - Uniformity of information
  - Inputs for tools:
    - Automatic coding/configuration of device drivers (SOIS)
    - Simulation of devices
    - Test tool for devices
    - Automatic import into Spacecraft database
    - Etc
  - Online dynamic discovery of device capabilities
  - Enabler for standardisation of functional interfaces across device types
- What does EDS do for you?
  - System Engineer
    - Define requirements on interfaces
    - Allows conformance testing of devices to ensure they meet interfaces
    - Configuration support during integration of spacecraft
  - Application Developer
    - Allows conversion of functional interface into API, software model, etc
    - Automatic generation of device driver
  - Device Manufacturer
    - Drives testing of device interface
    - Avoids discrepancies between ICD and device, ICD and device driver

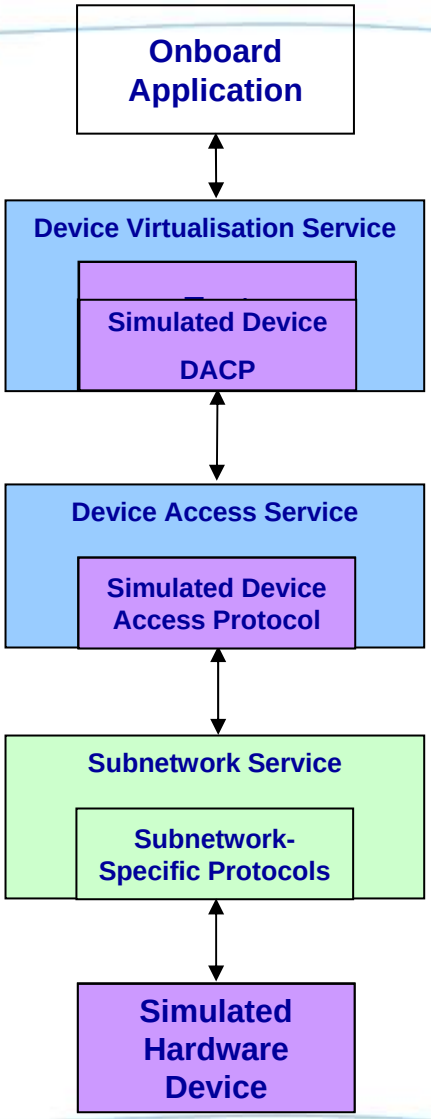
EDS Structure



- EDS elements mapping to DVS and DAS data
  - Candidate technologies
  - Functional Interface
  - Device Abstraction Control Procedure
  - Device-specific Access Protocol
- Who provides/uses what?
  - System Engineer
  - Application Developer
  - Device Manufacturer
  - Device Driver Developer (no longer exists?)

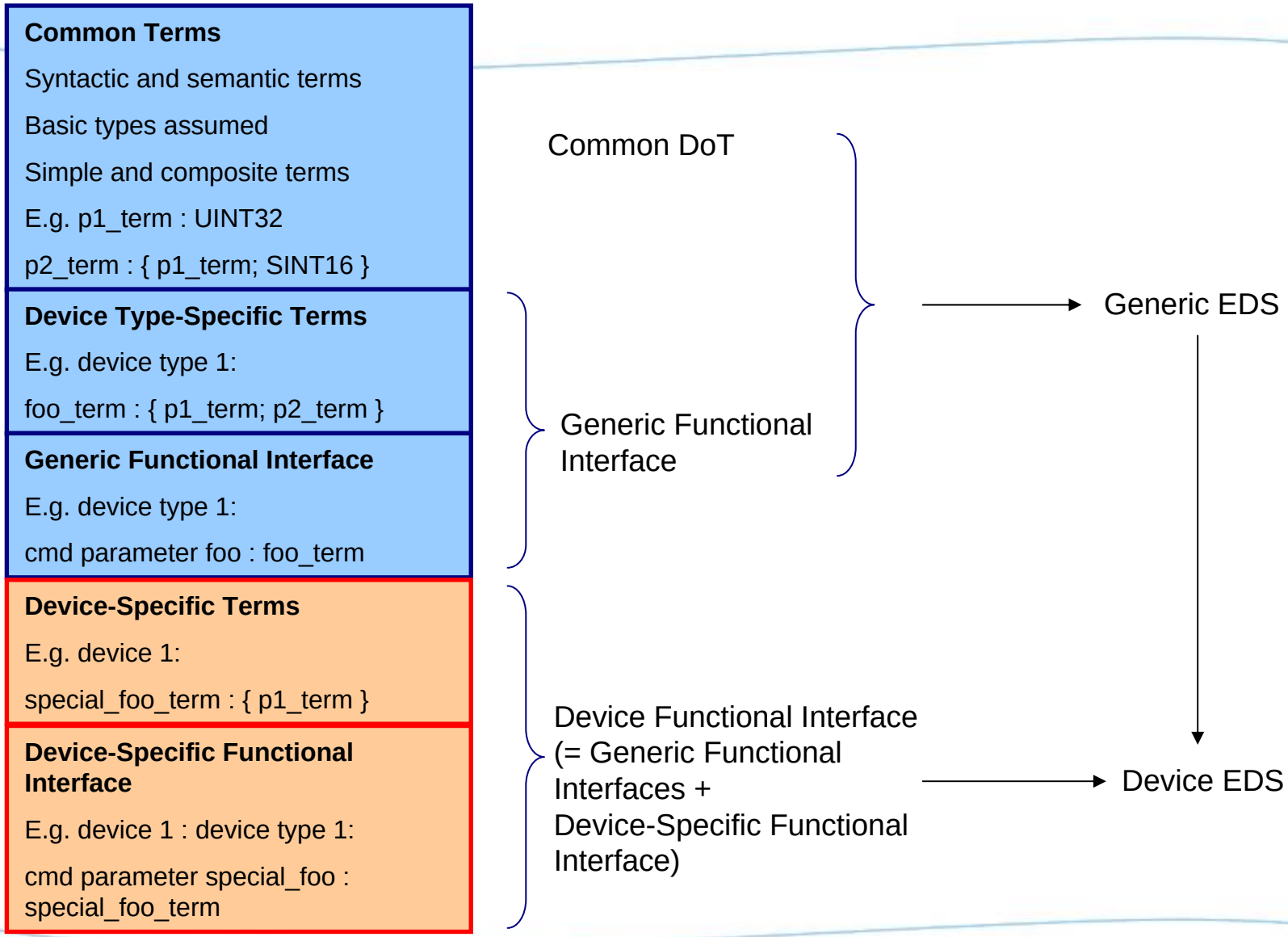
Worked Example: Development Process with EDS



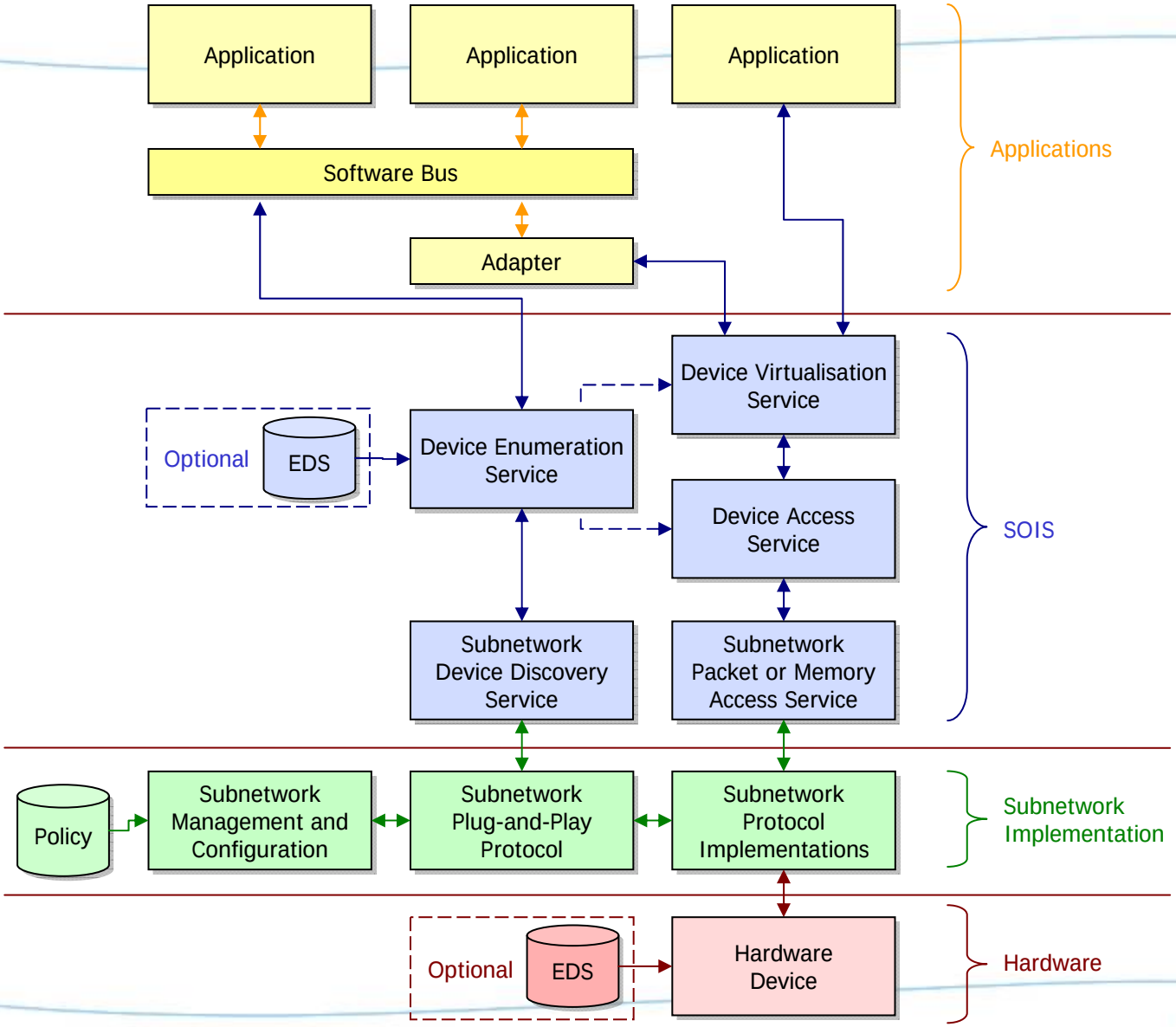


1. System Specification
    1. Application, Bus and Device Functional Specification
    2. SOIS, Device Virtualisation and EDS Technology Selection
    3. Device Functional Interface Selection
2. Component Development:
    i. Device development & testing against EDS
    ii. Application development & testing against Functional Interface
3. Progressive integration:
    i. Auto-generation of device drivers & integration with device
    ii. Integration with application
4. Run-time

Functional Interfaces, DoT & EDS



SOIS Plug-and-Play Architecture



Book Structure & Relationship