



In partnership with

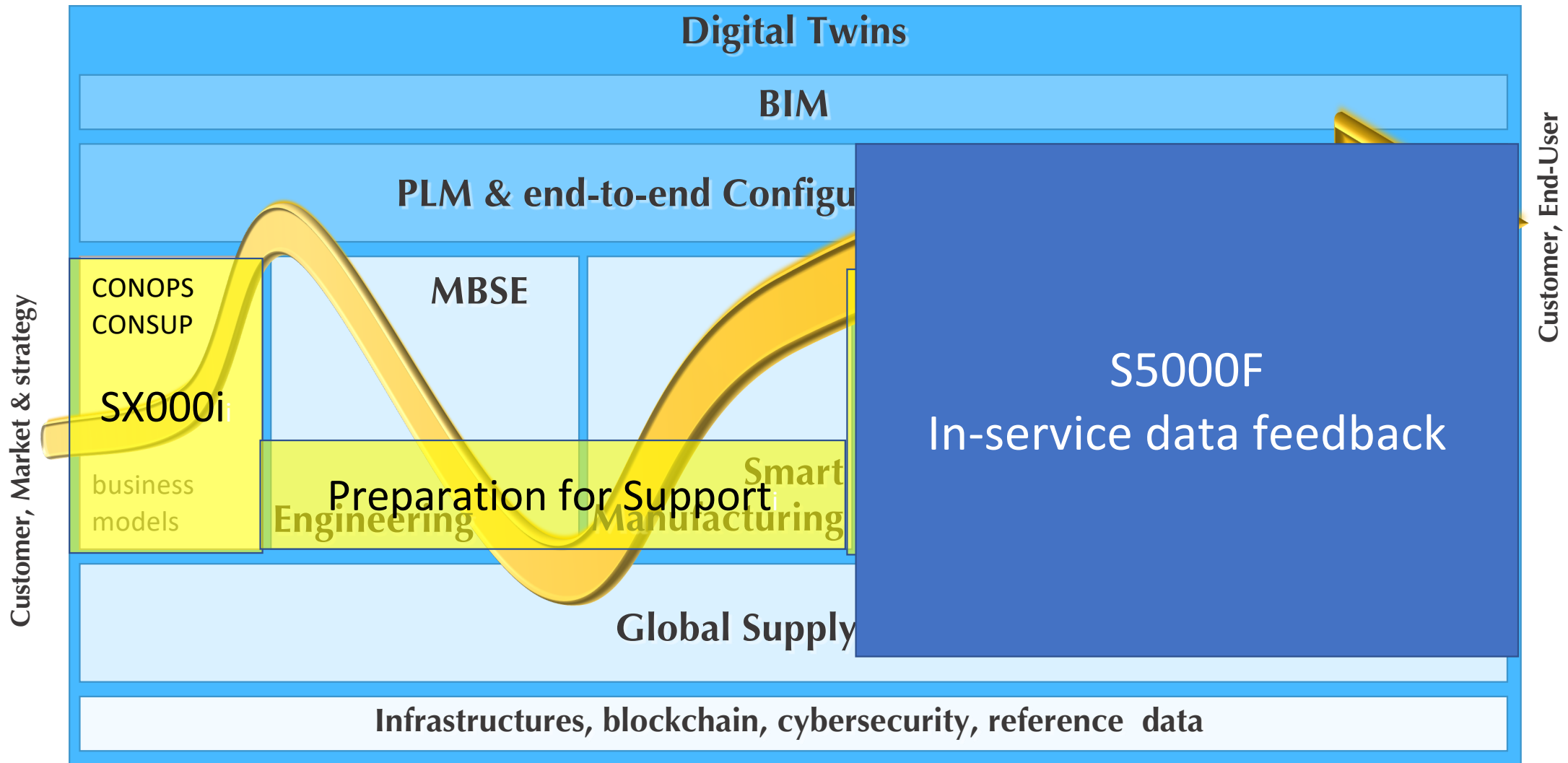


AFNeT Standards Days

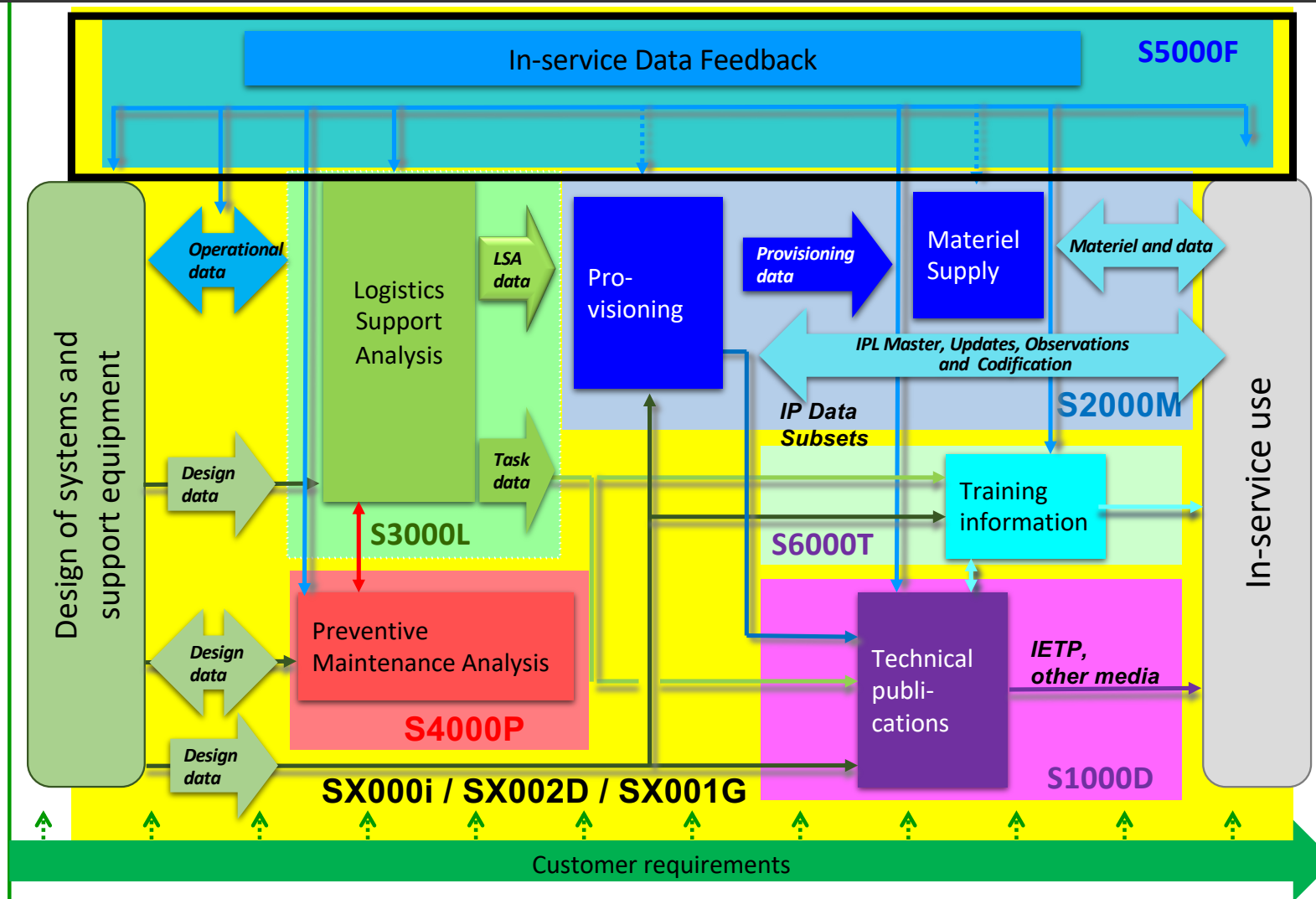
MRO/ILS – Episode 2: S5000F

by Ramón Somoza (Airbus Defence and Space)

<http://standardsdays.afnet.fr> - AFNeT Standards Days 2020 : 6 & 7 October 2020

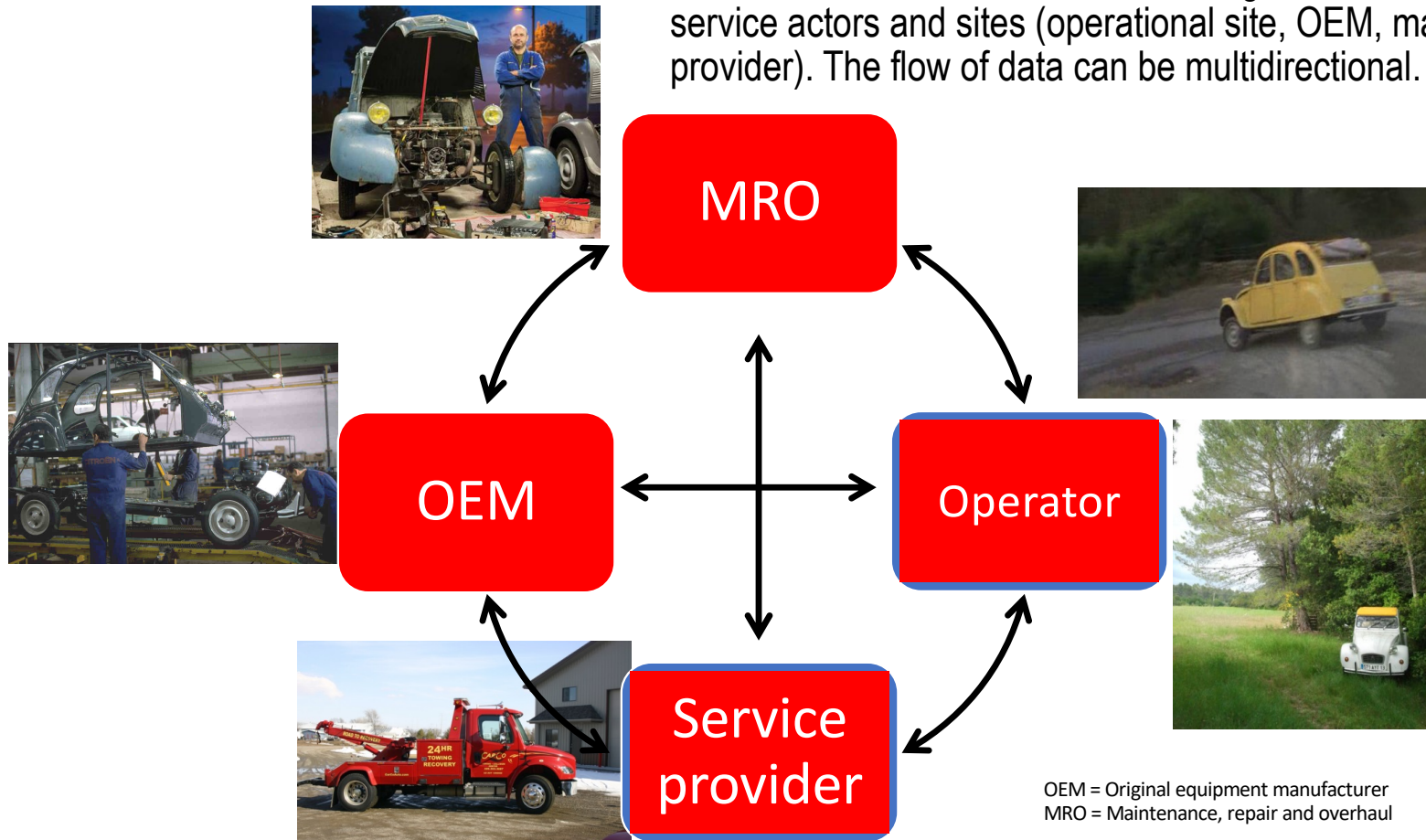


Context of S5000F



Introduction – what is in-service data feedback

In-service data feedback is an exchange of information between all in-service actors and sites (operational site, OEM, maintainer or service provider). The flow of data can be multidirectional.



OEM = Original equipment manufacturer
MRO = Maintenance, repair and overhaul

What is S5000F?



- S5000F, International specification for in-service feedback is a multinational specification developed jointly by the AeroSpace and Defence Industries Association of Europe (ASD) and Aerospace Industries Association (AIA) in the U.S.
- S5000F provides a standard data exchange mechanism for the feedback of in-service data for all kinds of products.
- It is product, organization, domain and tool-agnostic. Thus, it can be used for air, land, sea, space and domestic products, both for civil and military applications, and does not depend on proprietary solutions.
- Being tailorable, a partial implementation is feasible (there is no need to implement the whole specification). Extending its use later is trivial because exactly the same constructs are used.
- As its use is limited to data exchange, S5000F can be easily integrated with existing legacy systems by means of import/export functions.
- It is integrated and fully interoperable with the rest of the S-Series specifications, and work is undergoing to ensure compatibility with the new issue of the ISO 10303-239 (PLCS) standard (known as AP239).
- It is open to change requests from the rest of the world to address new needs.

Feedback data in S5000F are used for:

- Reliability, availability, maintainability, capability and testability
- Safety analysis
- Supply support
- Life Cycle Costing
- Warranty analysis
- Product health and usage monitoring
- Obsolescence management
- Software and mission data
- Integrated fleet management
- Configuration management
- Management of in-service contracts
- Non-predefined information (e.g., videos, audio, BIT files, etc)

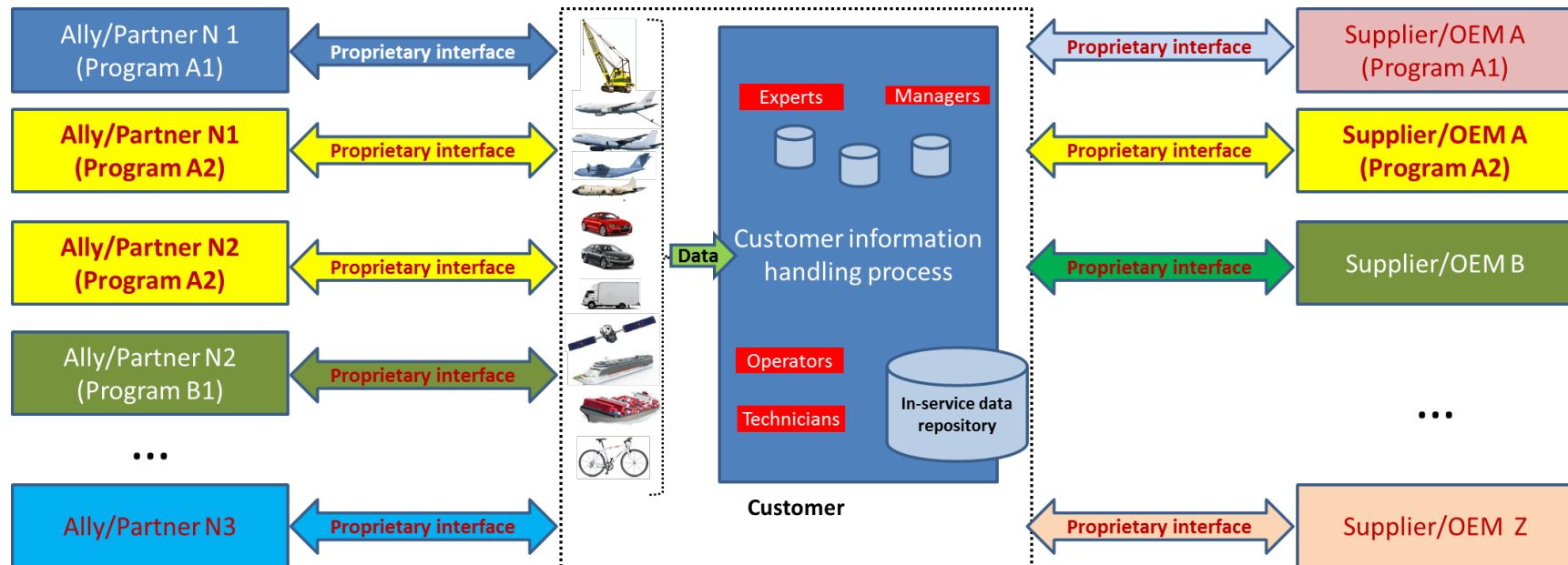


- The S5000F data model has been structured around 16 functional domains.
- Each domain is further subdivided into functional areas and units of functionality
- The data for the use cases defined in the chapters can usually be found in one single domain, which allows for quick targeting.
- S5000F Issue 2.0 has a total of 90 use cases and one or more use cases can be selected for implementation.
- The data (mandatory or optional) associated to each use case are defined in the specification.
- It is possible to add project-specific codes, and remove unnecessary (unused) ones. It is also possible to define project-specific use cases.
- The specification allows to add project-specific information without “breaking” the specification (i.e., maintaining the interoperability between specifications and projects).



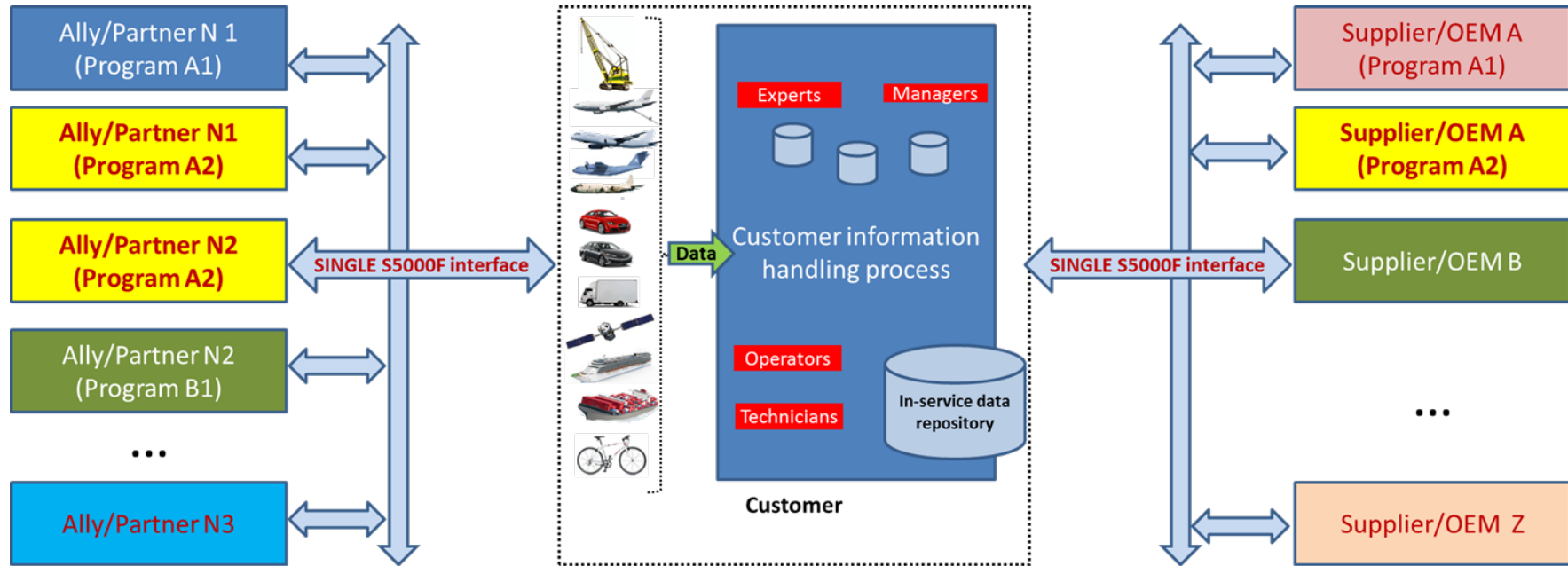
- Being a data exchange specification, the implementation of S5000F does NOT require changing the existing legacy IT systems. It may however require some adjusting of the internal processes for data validation.
- What should be done is to develop:
 - An export capability that generates the XML data to be submitted.
 - An import capability that allows to import the XML data received into the existing legacy systems.
- S5000F allows therefore to:
 - Replace a legacy system without affecting the communications with other parties
 - Migrate information from a legacy system to a new one
 - Consolidate information from multiple IT systems into a common repository (e.g., for data archiving or data analytics)
- The S5000F data model also provides the information about how all the data received are related with each other, thus greatly simplifying data warehousing and data analytics initiatives.
- S5000F only covers the data exchange format, so it can be used with multiple (even proprietary) data exchange protocols. It is also compatible with blockchain.

- It is likely that you have currently this situation (“as is”):

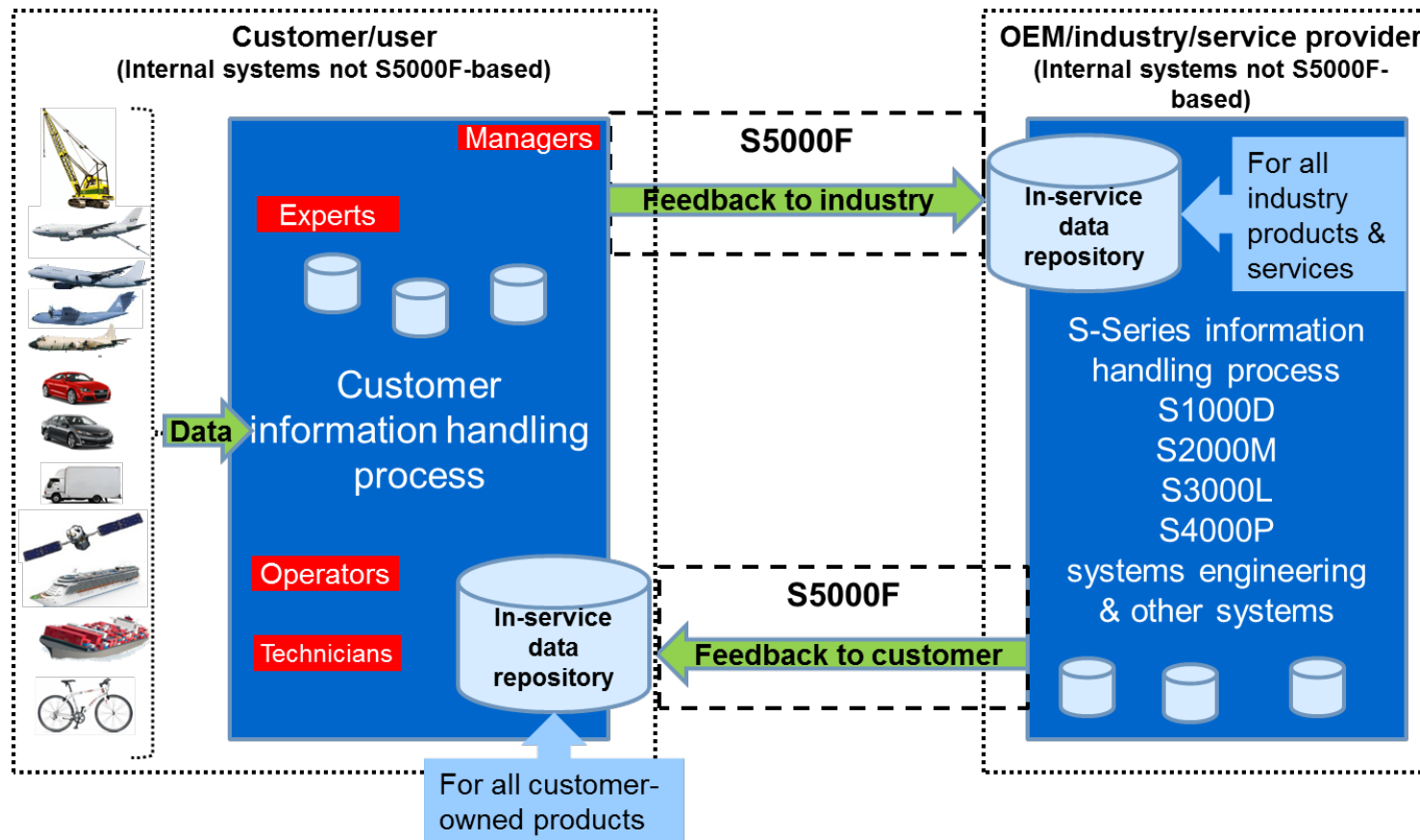


- Different interfaces for different suppliers and/or partners, sometimes even different interfaces with a same supplier/partner for a different program!
- This plethora of interfaces is expensive, complicates a lot the data integration and is the nightmare of the IT technicians that need to keep those interfaces operational, not to speak about what will happen when you (or one of your partners/suppliers) modernize the IT systems.

- And this is the situation where you would like to be (“to be”):



- One single interface for each and every system, which is common for all.
- Stable and defined international I/F, vendor-independent and publicly available.
- Easy IT integration (can be also used between internal systems)
- All data in a same common format, hence very suitable for big data initiatives.



- The internal format of the in-service repository “can” be defined based on S5000F, but is not mandatory, and can therefore be any legacy internal system (no changes necessary).

- S5000F provides some textual description of the feedback domains, but its core is the data model.
- It has been developed with Enterprise Architect, using the common rules and Common Data Model used for other ASD specifications
- It uses business objects, so that it can be understood by the business.
- Simplifying, you could state that a class (a block in the data model diagram) is the equivalent of a database table, and an attribute (an item within a block) is a field in that table. Relationships (arrows between the blocks) indicate the relationships between the classes.
- If you want a more detailed understanding of how the data model functions, there is a document **SX004G, Unified Modeling Language (UML) model reader's guidance**, which can be downloaded for free from <http://www.sx000i.org>
- The data model is huge (over a 100 diagrams), 541 classes and interfaces, and 1289 attributes, as it covers ALL in-service data.
- The good news is that you will always use just a small subset.
- The smallest known implementation has **just 4 attributes** + the message data itself! And yes, it is a valid S5000F implementation.

Sample S5000F information



Serialized Product Variant

Product name	Product variant	PV Identifier	PV serial number	Manufacturing date	Entry into service	Fleet	Operating base	Owner	Operator
ASD/AIA Bike	Mountain Bike	MTB-2000M	42	10/05/2013	12/05/2013	Sevilla Biking Tours	Calle Sierpes, Sevilla, Spain	City of Sevilla	Sevilla Biking Tours
ASD/AIA Bike	Mountain Bike	MTB-2000M	43	11/05/2013	12/05/2013	Granada Bike Hire	Granada Bike Hire central office	Bike Leasing AG	Granada Bike Hire
ASD/AIA Bike	Mountain Bike	MTB-2000M	44	12/05/2013	13/05/2013	Granada Bike Hire	Granada Bike Hire central office	Bike Leasing AG	Granada Bike Hire
ASD/AIA Bike	Mountain Bike	MTB-2000M	45	13/05/2013	13/05/2013	Schultz Bike Hire	Schultz Bike Hire, Garmisch-Partenkirchen, 82467, Germany	Bike Leasing AG	Schultz Bike Hire
ASD/AIA Bike	Mountain Bike	MTB-2000M	46	14/05/2013	02/06/2013	Private_Customers	Home of customer, Munich, Germany	Hans-Dieter	Hans-Dieter
ASD/AIA Bike	Mountain Bike	MTB-2000M	47	15/05/2013	15/06/2013	Granada Bike Hire	Granada Bike Hire central office	Bike Leasing AG	Granada Bike Hire

Actual serialized product variant (PV) configuration

Product name	Product variant (PV)	PV identifier	PV serial number	Part name	Part identifier	Serialized part serial number	Location	Installation date
ASD/AIA Bike	Mountain Bike	MTB-2000M	46	Brake System	MTB-BRS800	84623	DA1	15/05/2013
				Front Brake	MTB-BRS800-801			
				Front Brake Lever	MTB-BRS800-401			
				Front Brake Tube	MTB-BRS800-403			
				Front Wheel Brake	MTB-BRS800-405			
				Front Brake Caliper Assy	WB-10000-401			
				Rear Brake	MTB-BRS800-802			
				Rear Brake Lever	MTB-BRS800-402			
				Rear Brake Tube	MTB-BRS800-404			
				Rear Wheel Brake	MTB-BRS800-406			
				Rear Brake Caliper Assy	WB-10000-401			

Germany	Bike Leasing AG	Schultz Bike Hire
---------	-----------------	-------------------

Change embodiment

Change authorization	Change embodiment requirement id	Change embodiment requirement date	Change embodiment requirement type	Technical order priority	Technical Order required implementation date
MTB-2000M-CH1602	CH1602	31/12/2016	Mandatory	N/A	N/A

Service Bulletin

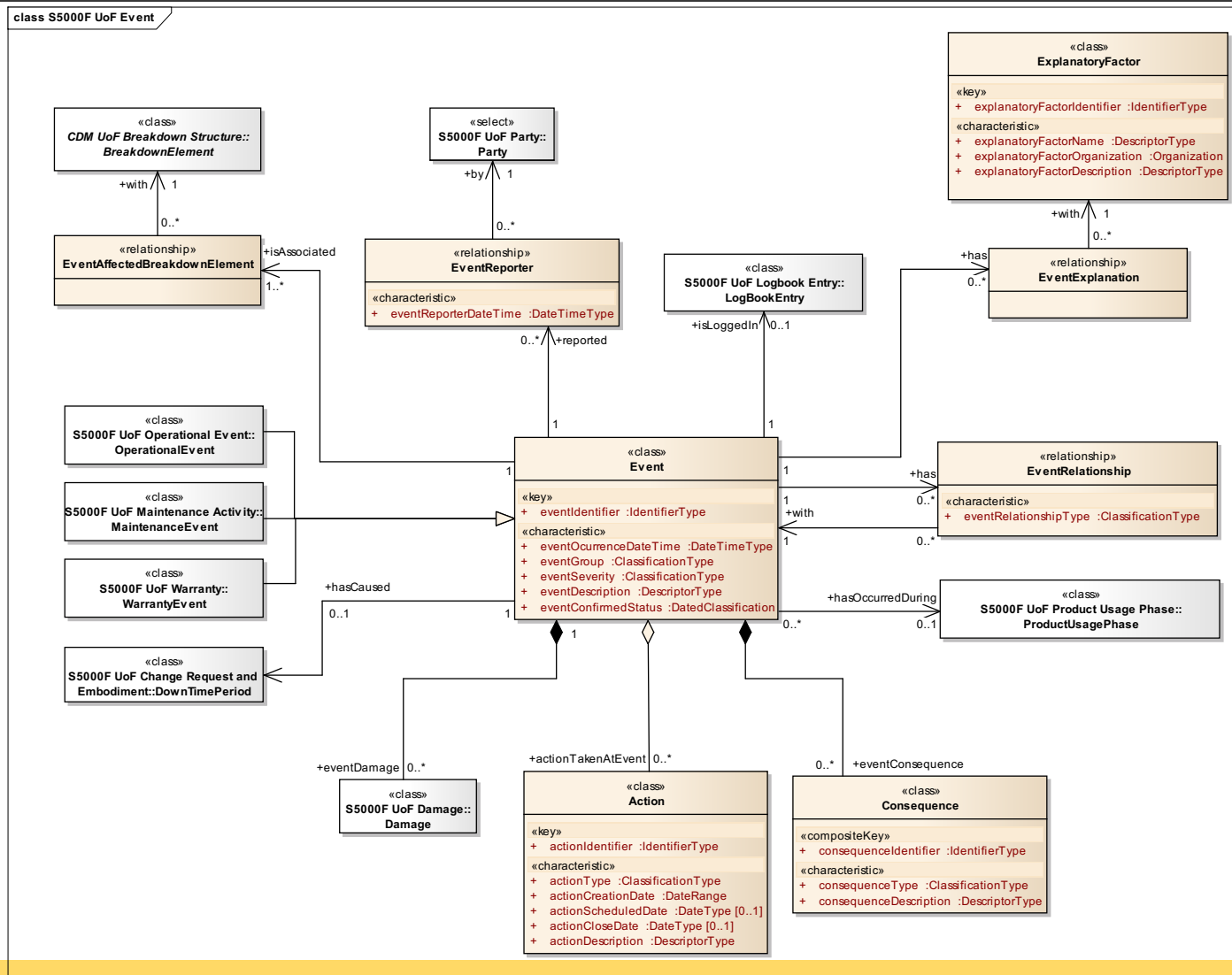
Safety Instructions

Document Id	Associated to safety issue	Title	Description	Status	Creation date	Document type	Safety criticality
SIN20160803A	SISS20160730A	MTB-2000M Brake safety instructions	Special safety instructions associated to MTB-2000M brake MTB-BRS800-801	Approved	14/08/2016	Special safety instructions	Major

Safety Instruction Id	Safety action identifier	Type	Description	Priority	Release date	Required implementation date
SIN20160803A	1	Mandatory	The BRAKE PAD SET must always be installed using graphite gliding paste on clean gliding surfaces (also inside the Brake Caliper Housing)-	High	14/08/2016	30/08/2016
SIN20160803A	2	Mandatory	Pending formal MB manual update, add manually the following statements to the MB manual: - „Severy injury may result in case of installation of the BRAKE PAD SET without graphite gliding paste on indicated surfaces (see FIGURE...).“ - „Keep BRAKE DISK surface free from any gliding paste or oil contamination during maintenance. Clean BRAKE DISK after brake maintenance.“	High	14/08/2016	30/08/2016

Title	Document type	Description	Status	Date	Type	Priority	Embodiment Limit	Cost
07 MTB-2000M brake bolt replacement	SB	Replace ISO4762-M4X40 STEEL 10.9 bolt (pre-mod) by ISO4762-M4X40-A2 bolt (post-mod)	Approved	29/09/2016	Mandatory	High	31/12/2016	Free

How does the data model look like?



How does the XSD look like? – Message



```
<xsd:complexType name="message">
  <xsd:annotation>
    <xsd:appinfo>SX001G:Message</xsd:appinfo>
  </xsd:annotation>
  <xsd:sequence>
    <!-- UML-Attributes -->
    <xsd:element name="msgStatus" type="messageContentStatus" minOccurs="0"/>
    <xsd:element name="msgType" type="messageContentType" minOccurs="0"/>
    <xsd:element name="msgDate" type="messageCreationDateTime" minOccurs="0"/>
    <xsd:element name="msgId" type="messageIdentifier" minOccurs="1"/>
    <xsd:element name="msgLang" type="messageLanguage" minOccurs="0"/>
    <!-- UML-CompAssocs -->
    <xsd:group ref="messageContentNonAbstractClasses" minOccurs="0"/>
    <!-- UML-EmbeddedAssociations -->
    <xsd:element name="msgContext" type="messageContextRef" nillable="true" minOccurs="0"/>
    <xsd:element name="msgPty" type="messagePartyRef" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="relatedMsg" type="messageRelationshipRef" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <!-- UML-Extends -->
    <xsd:element name="secs" type="securityClassificationItem" nillable="true" minOccurs="0"/>
    <xsd:element name="docs" type="documentReferencingItem" nillable="true" minOccurs="0"/>
    <xsd:element name="projAttrs" type="projectSpecificExtensionItem" nillable="true" minOccurs="0"/>
    <xsd:element name="rmks" type="remarkItem" nillable="true" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="uid" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:ID">
        <xsd:annotation>
          <xsd:appinfo>uidPattern:msg</xsd:appinfo>
        </xsd:annotation>
        <xsd:pattern value="msg[1-9][0-9]*"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="crud" type="crudCodeValues" default="I"/>
</xsd:complexType>
```

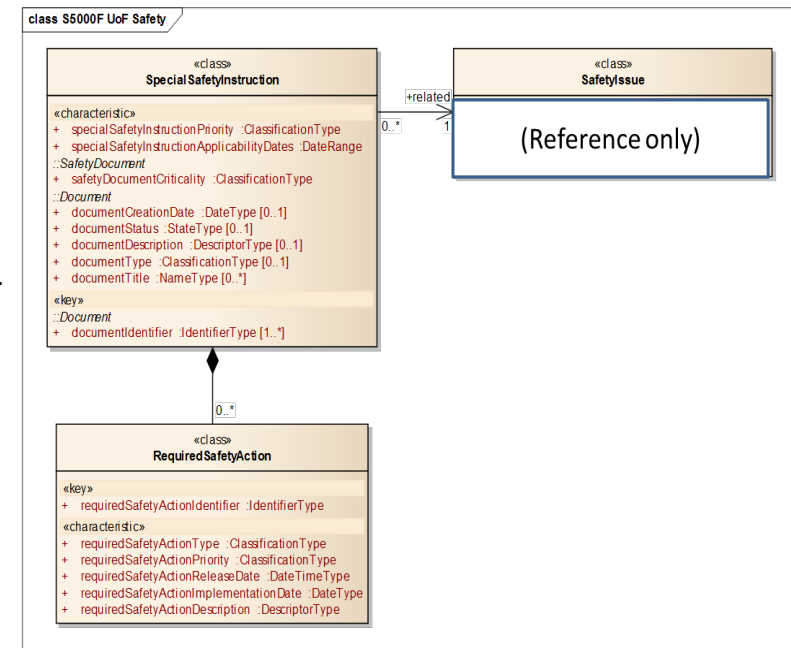
Example message content for use case UC50503 (Provide Special Safety Instructions):

```
<xsd:complexType name="messageContent">
  <xsd:annotation>
    <xsd:appinfo>SX001G:MessageContent</xsd:appinfo>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="SpecialSafetyInstructions" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="doc" type="specialSafetyInstruction" minOccurs="0" maxOccurs="unbounded"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

How does the XSD look like? – Actual content

Example message content for use case UC50503 (Provide Special Safety Instructions):

```
<xsd:complexType name="specialSafetyInstruction">
  <xsd:annotation>
    <xsd:appinfo>SX001G:SpecialSafetyInstruction</xsd:appinfo>
  </xsd:annotation>
  <xsd:sequence>
    <!--UML-Attributes-->
    <xsd:element name="docDate" type="documentCreationDate" nillable="true" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="descr" type="documentDescription" nillable="true" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="docId" type="documentIdentifier" minOccurs="1" maxOccurs="unbounded"/>
    <xsd:element name="docstatus" type="documentStatus" nillable="true" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="docTitle" type="documentTitle" nillable="true" minOccurs="0" maxOccurs="1"/>
    <xsd:element name="docType" type="documentType" nillable="true" minOccurs="0"/>
    <xsd:element name="docCrit" type="safetyDocumentCriticality" nillable="true" minOccurs="0"/>
    <xsd:element name="safeAppDate" type="specialSafetyInstructionApplicabilityDates" nillable="true" minOccurs="0"/>
    <xsd:element name="ssInstr" type="specialSafetyInstructionPriority" nillable="true" minOccurs="0"/>
    <!--UML-EmbAssocs-->
    <xsd:element name="safeiss" type="safetyIssue" nillable="true" minOccurs="0"/>
    <!--UML-CompAssocs-->
    <xsd:element name="reqsa" type="requiredSafetyAction" nillable="true" minOccurs="0" maxOccurs="unbounded"/>
    <!--UML-Interfaces-->
    <xsd:group ref="applicabilityAssignmentItem"/>
    <xsd:group ref="securityClassificationItem"/>
  </xsd:sequence>
  <xsd:attribute name="uid" use="optional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:ID">
        <xsd:pattern value="doc[1-9][0-9]*"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>
  <xsd:attribute name="crud" type="crudCodeValues" default="I"/>
</xsd:complexType>
```



- Messages can provide guidelines about what to do with the information:

- Insert
- Delete
- Update
- Select
- Replace

- You can create tailored messages for a specific use case. Tailoring guidelines (and an example) can be found in Chapter 19.
- Guidelines on how to implement the XML can be found in **SX005G, S-Series ILS specifications XML schema implementation guidance**, which can be downloaded for free from <http://www.sx000i.org>

```
<multiValuedExampleClass>
  <classId>
    <id>1</id>
  </classId>
  <classProp>
    <vdtm>REQ</vdtm>
    <unit>FH</unit>
    <value>15</value>
  </classProp>
  <classProp>
    <date>2016-10-15</date>
    <vdtm>MEA</vdtm>
    <unit>FH</unit>
    <value>20</value>
  </classProp>
</multiValuedExampleClass>
```

S5000F has 90 pre-defined use cases which can be used “as-is” or tailored. Some examples:

- Testability:
 - UC50310: Can product be tested
 - UC50311: Fault diagnosis, fault identification
- Maintenance:
 - UC50401: Report Manufacturer Maintenance Schedule
 - UC50402: Report Product User Maintenance Program
 - UC50403: Report Maintenance Performed
 - UC50404: Report Product Performance
 - UC50405: Report New Modifications
 - UC50406: Report Technical Queries
 - UC50407: Report Shop Findings
 - UC50408: Report Structural Damage
- Safety:
 - UC50501: Report Safety Issue
 - UC50502: Report Safety Warning
 - UC50503: Provide Special Safety Instructions

S5000F on-going Projects :

- Airbus Defence and Space (2 projects)
- SIMAT : Referential study
- NHI S5000F : Study and POC implementation
- Dassault Aviation for the Rafale (4 Projects)
- French MoD : DMAé (Brasidas Project)
- Shipdex (shipping industry)
- 2MORO (MRO software vendor)

- **More to follow!**



Note: All projects are working with S5000F Issue 2.0 data model and XSDs.

This is the smallest known S5000F implementation.

- **Requirement:** It is necessary to report torque values of screws during manufacturing and also to exchange the same torque values after verification.

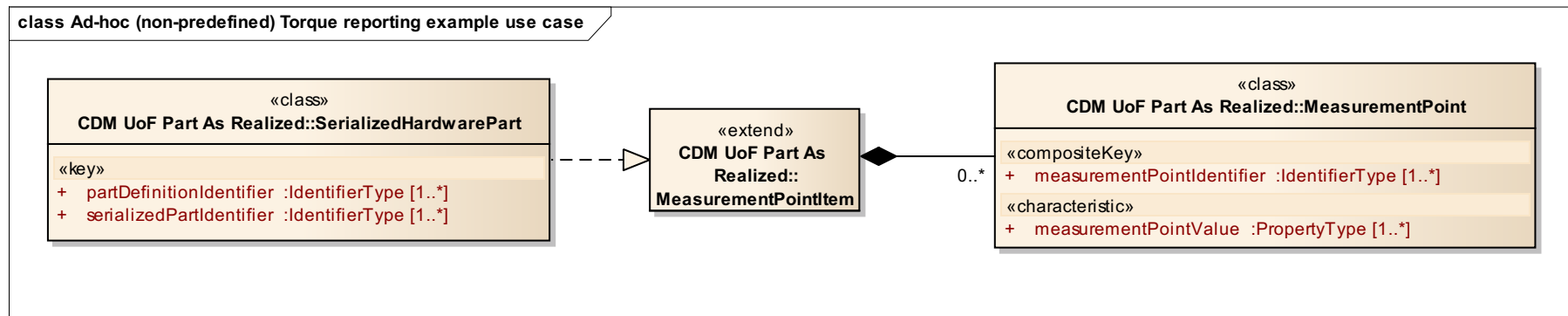
Vérification du couple de serrage des fixations chapes sur carter BTP

Tableau 1 : valeur enregistrée durant la vérification				
Chape Avant	Vis (1)	Vis (2)	Vis (3)	Vis (4)
4.2 - 4,62 daN.m 372 - 408 lbf.in	27 Nm	42 Nm	42 Nm	42 Nm
Chape Arrière Gauche	Vis (5)	Vis (6)	Vis (7)	Vis (8)
4.2 - 4,62 daN.m 372 - 408 lbf.in	18 Nm	46.2 Nm	46.2 Nm	42 Nm
Chape Arrière Droite	Vis (9)	Vis (10)	Vis (11)	Vis (12)
4.2 - 4,62 daN.m 372 - 408 lbf.in	46.2 Nm	46.2 Nm	42 Nm	46.2 Nm

Tableau 2 : valeur si remise au couple après vérification				
Chape Avant	Vis (1)	Vis (2)	Vis (3)	Vis (4)
4.2 - 4,62 daN.m 372 - 408 lbf.in	46.2 Nm	42 Nm	42 Nm	42 Nm
Chape Arrière Gauche	Vis (5)	Vis (6)	Vis (7)	Vis (8)
4.2 - 4,62 daN.m 372 - 408 lbf.in	42 Nm	46.2 Nm	46.2 Nm	42 Nm
Chape Arrière Droite	Vis (9)	Vis (10)	Vis (11)	Vis (12)
4.2 - 4,62 daN.m 372 - 408 lbf.in	46.2 Nm	46.2 Nm	42 Nm	46.2 Nm

Note: the fact that S5000F is intended for in-service data exchange does not imply can it can be used **only** for that purpose!

- First, we identify the classes that represent the data to be exchanged.
- These classes are:
 - **SerializedHardwarepart** (representing the part on which the data are reported).
 - **MeasurementPoint** (representing the values for each individual measurement)
 - The **MeasurementPointItem** <<extend>> interface to a MeasurementPoint (not implemented in XSD, but required to link the two classes indicated before).
- Thus, the required classes are as indicated below



For clarity, here we indicate how the data maps to the data model.

There is just ONE part number with 12 measurement points (the three items correspond to different views of the same part)

There are TWO blocks of data for EACH part (reported and measured)

Obviously, we must report the values for each individual part, so we use a serialized hardware part class

class Ad-hoc (non-predefined) Torque reporting example use case

Verification du couple de serrage des fixations chapes sur carter BTP

	Vis (1)	Vis (2)	Vis (3)	Vis (4)
Chape Avant 4.2 - 4,82 daN.m 372 - 408 lb/in	27 Nm	42 Nm	42 Nm	42 Nm
Chape Arrière Gauche 4.2 - 4,82 daN.m 372 - 408 lb/in	18 Nm	46.2 Nm	46.2 Nm	42 Nm
Chape Arrière Droite 4.2 - 4,82 daN.m 372 - 408 lb/in	46.2 Nm	46.2 Nm	42 Nm	46.2 Nm

	Vis (1)	Vis (2)	Vis (3)	Vis (4)
Chape Avant 4.2 - 4,82 daN.m 372 - 408 lb/in	46.2 Nm	42 Nm	42 Nm	42 Nm
Chape Arrière Gauche 4.2 - 4,82 daN.m 372 - 408 lb/in	42 Nm	46.2 Nm	46.2 Nm	42 Nm
Chape Arrière Droite 4.2 - 4,82 daN.m 372 - 408 lb/in	46.2 Nm	46.2 Nm	42 Nm	46.2 Nm

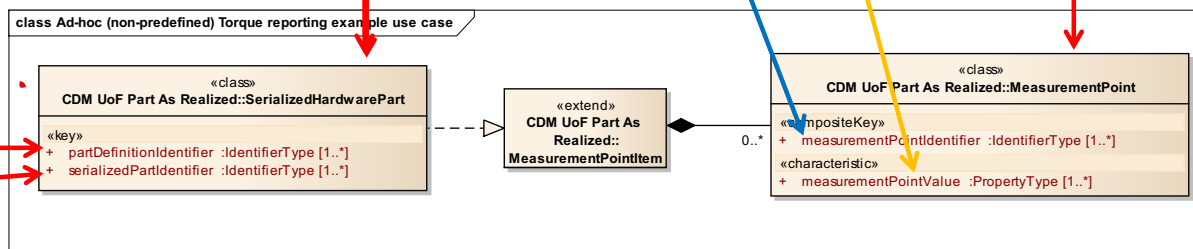
Each entry corresponds to one MeasurementPoint

The screw number or name of the entry is used as the identifier.

The value goes into the value attribute

Serial part identified by:

P/N
S/N



- So how do we distinguish between reported and measured values?
- How to we report the units?

MeasurementPointValue is a **PropertyType**

The **valueDetermination** allows to define the context of the measure.

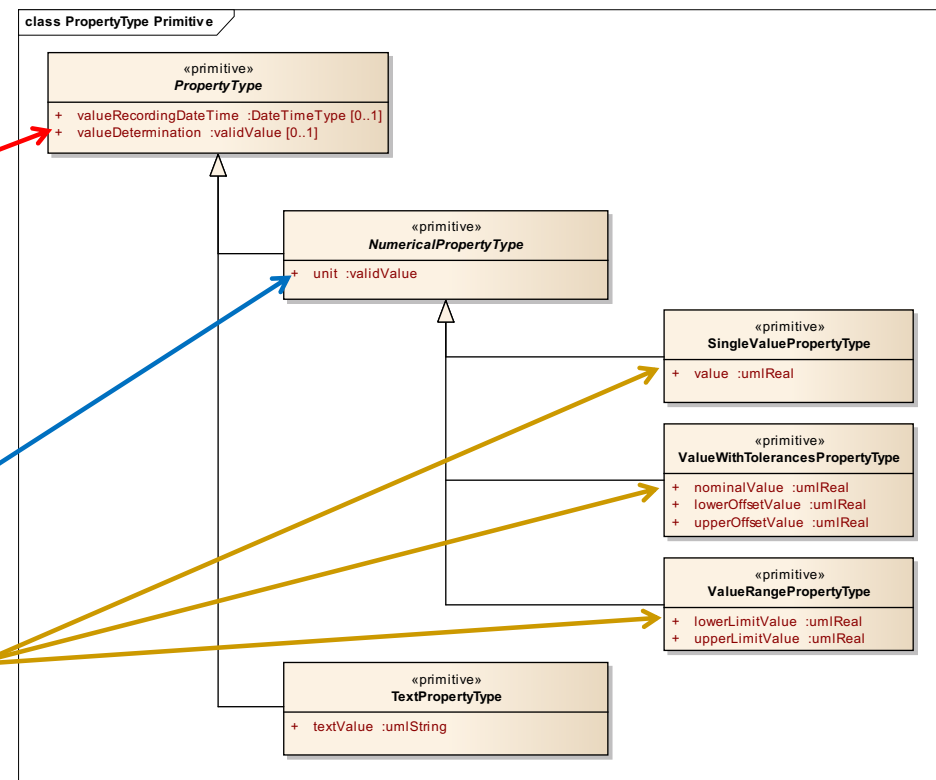
Eg:

SET for reported values

MEAS for measured values

The **unit** is inherent to any NumericType and can be also set

And the actual numeric **value** can be a single value, a value with tolerances or a value range – whatever you need!
(and *without* actually changing the message definition)



How the XML looks like

```

<msgId>
  <id>2</id>
</msgId>
<ucTorqueReport>
  <serialPart>
    <partId>
      <id>Chape</id>
    </partId>
    <serialId>
      <name>20</name>
    </serialId>
    <mpoints>
      <mPoint>
        <mPointId>
          <id>vis1</id>
        </mPointId>
        <mPointVal>
          <vdtm>SET</vdtm>
          <unit>NM</unit>
          <value>27</value>
        </mPointVal>
        <rmks>
          <rmk>
            <text>
              <descr>This is an anomalous value</descr>
            </text>
          </rmk>
        </rmks>
      </mPoint>
      <mPoint>
        <mPointId>
          <id>vis1</id>
        </mPointId>
        <mPointVal>
          <vdtm>MEAS</vdtm>
          <unit>NM</unit>
          <value>46.2</value>
        </mPointVal>
      </mPoint>
    </mpoints>
  </serialPart>
</ucTorqueReport>
  
```

Part Number
 Serial Number
 Screw Identifier
 Indicates *reported* value
 Unit
 Actual value
 Note that we added an optional remark
 Screw Identifier
 Indicates *measured* value
 Unit
 Actual value

Vérification d'un couple de serrage des fixations chapes sur carter B

Tableau 1 : valeur enregistrée durant la vérif

Chape Avant	Vis (1)	Vis (2)	Vis (3)
4.2 - 4,62 daN.m 372 - 408 lbf.in	27 Nm	46.2 Nm	46.2 Nm
Chape Arrière Gauche	Vis (5)	Vis (6)	Vis (7)
4.2 - 4,62 daN.m 372 - 408 lbf.in	18 Nm	46.2 Nm	46.2 Nm
Chape Arrière Droite	Vis (8)	Vis (9)	Vis (10)
4.2 - 4,62 daN.m 372 - 408 lbf.in	46.2 Nm	46.2 Nm	42 Nm

Tableau 2 : valeur si remise au couple après

Chape Avant	Vis (1)	Vis (2)	Vis (3)
4.2 - 4,62 daN.m 372 - 408 lbf.in	46.2 Nm	46.2 Nm	42 Nm
Chape Arrière Gauche	Vis (5)	Vis (6)	Vis (7)
4.2 - 4,62 daN.m 372 - 408 lbf.in	42 Nm	46.2 Nm	46.2 Nm
Chape Arrière Droite	Vis (8)	Vis (9)	Vis (10)
4.2 - 4,62 daN.m 372 - 408 lbf.in	46.2 Nm	46.2 Nm	42 Nm

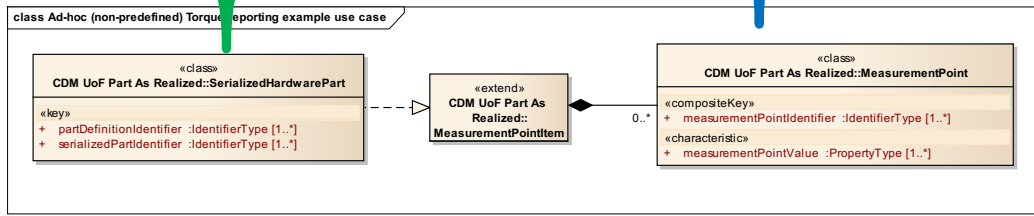
Reading the XML into Excel, it looks like this:



id	id2	name	id3	vdtm	unit	value	descr	id4	code	id5	code6	id7
2	Chape	20	vis1	SET	NM	27	This is an anomalous value	NH90 S/N 77			U	1
2	Chape	20	vis1	MEAS	NM	46,2		NH90 S/N 77			U	1
2	Chape	20	vis2	SET	NM	42		NH90 S/N 77			U	1
2	Chape	20	vis2	MEAS	NM	42		NH90 S/N 77			U	1
2								NH90 S/N 77	S	DoD	U	1
2								NH90 S/N 77	R	AH	U	1

You can recognize the data model

Message													
MessageContent													
Use case UC50000 inServiceDataFeedback													
SerializedHardwarePart													
MeasurementPoint													
MessageContext													
MessageParty													
SerializedProductVariant													
Party													
MessageRelationship													
messageIdentifier	partDefIdentifier	serializedPartIdentifier	Identifier	determination	unit	value	remarktext	serProductIdentifier	messagePartyType	organizationIdentifier	relatedMessage		
id	id2	id3	id4	vdtn	unit	value	descr	id5 (compound key)	code	id6	relatedMsg		
2	Chape	Chape20	vis1	SET	NM	27	This is an anomalous value	NH90 S/N 22	S	DoD	update to 1		
			vis1	MEAS	NM	46,2			D	AH			
			vis2	SET	NM	42							
			vis2	MEAS	NM	42							



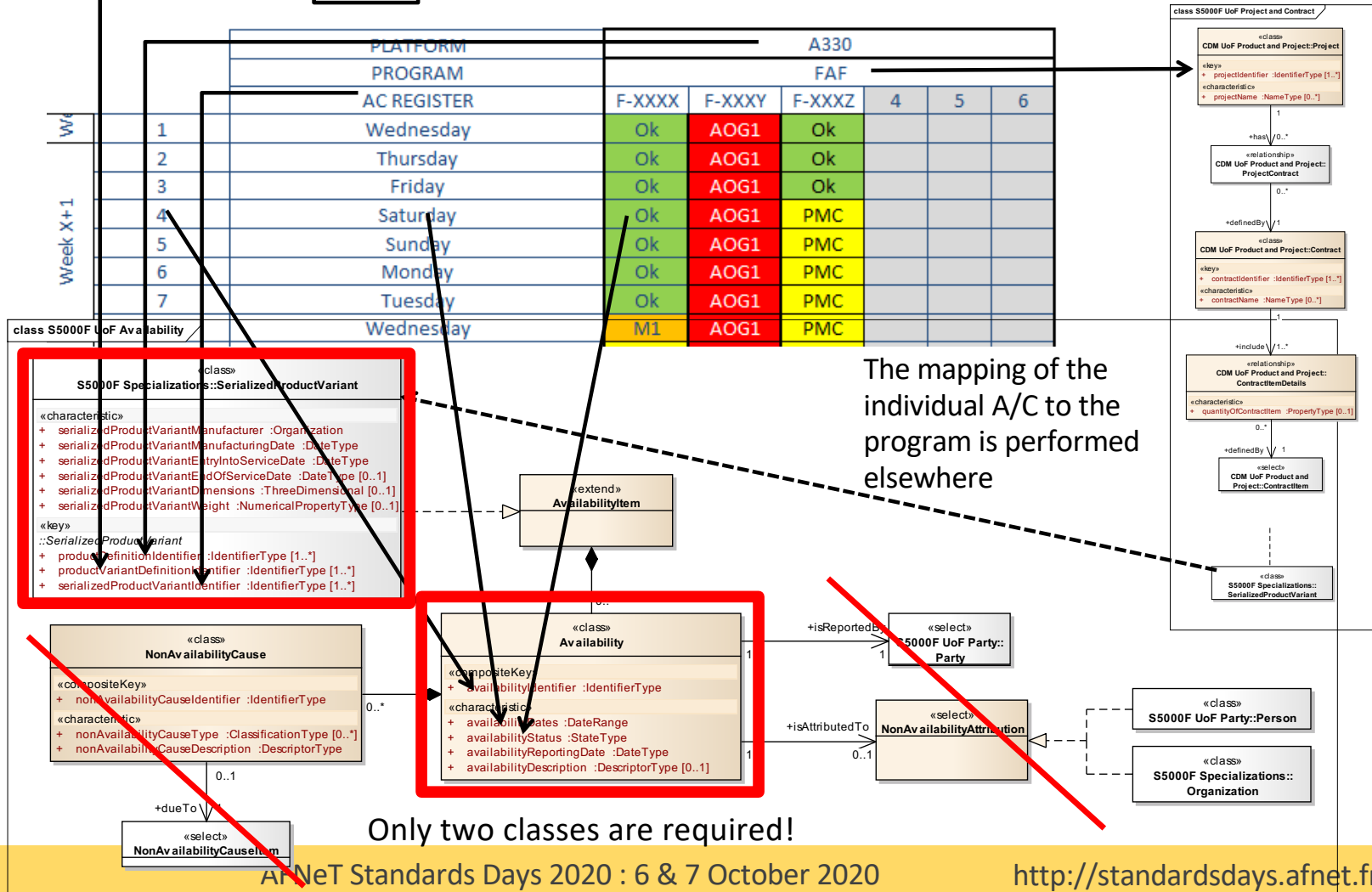
Complete message

Another example – how to report aircraft availability

MRTT Fleet Availability (Month - year)

		PLATFORM			A330			
		PROGRAM			FAF			
		AC REGISTER	F-XXXX	F-XXXY	F-XXXZ	4	5	6
Week X+1	1	Wednesday	Ok	AOG1	Ok			
	2	Thursday	Ok	AOG1	Ok			
	3	Friday	Ok	AOG1	Ok			
	4	Saturday	Ok	AOG1	PMC			
	5	Sunday	Ok	AOG1	PMC			
	6	Monday	Ok	AOG1	PMC			
	7	Tuesday	Wednesday	M1	AOG1	PMC		

(Not real data)



- S5000F covers all in-service information that was identified by the development team, and is open to add aspects that might have not been considered.
- It is an international specification that is product, organization, domain and tool-agnostic and be therefore used anywhere.
- It is not necessary to implement the whole specification, and implementations can be performed step-wise.
- It is fully compatible and interoperable with the S-Series of ILS specifications and can be therefore use for efficient product support data exchange and analysis.
- It separates the data exchange from the tools, enabling the usage of heterogeneous legacy IT systems and the consolidation of information in a data warehouse for analytics projects.
- It is already in use for multiple projects.
- Download it for free at <http://www.s5000f.org>

Questions
&
Answers



6th and 7th October

<http://standardsdays.afnet.fr>