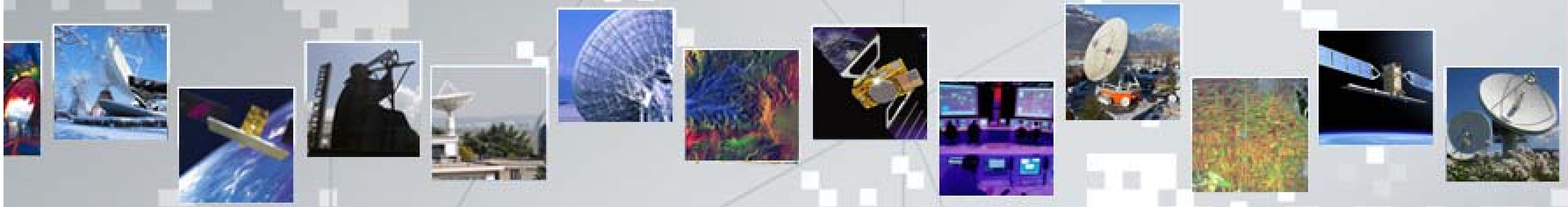


EVOLUTION OF SMP2 INTO ECSS SMP

SESP 2017, 28/03/2017

Peter Fritzen, Alberto Ingenito (Telespazio VEGA),
Robert Blommestijn, Vemund Reggestad, Anthony Walsh (ESA)

TELESPAZIO
a LEONARDO and THALES company

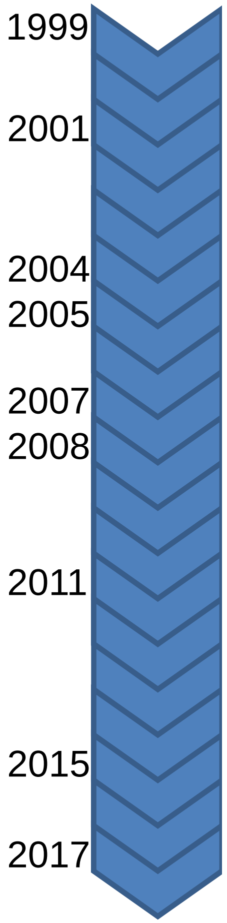
Agenda

- History of SMP1, SMP2 and ECSS SMP
- Major Changes done since SMP2 Issue 1.2
  - Concepts removed from the SMP2 Specification
  - Concepts added in the draft of E-ST-40-07
  - Clean-Up and Rationalisation
- First Results from a Migration of SIMULUS Models

History of SMP1, SMP2 and ECSS SMP

History of SMP1, SMP2 and ECSS SMP

- SMP1 Specification was released by ESA in 1999
- SMP2 Configuration Control Board founded in 2001
- SMP2 Issue 1.0 was released in 2004
- SMP2 Issue 1.2 was released in 2005
- First ECSS WG founded in 2007
- First Draft of ECSS SMP completed in 2008
- First Draft of ECSS SMP released as TM in 2011
- Second ECSS WG founded in 2015
- Second Draft of ECSS SMP for internal review



Major Changes done since SMP2 Issue 1.2

Concepts removed from the SMP2 Specification

Concepts removed from the SMP2 Specification Level 2 support, namely Assembly and Schedule files

- In SMP2, it is possible to define a hierarchy of Models in an external XML File, called an Assembly
  - Can be used to initially define a Simulation
  - Can be used to export and import Simulation
- In an additional file (called Schedule), the events put on the Scheduler service can be defined/exported/imported

- This concept has been dropped from ECSS E-ST-40-07
  - Exchange of Simulation between different environments is no longer standardised

- Justification: No mandate from ECSS to include Level 2

Concepts removed from the SMP2 Specification Platform Independent Component Model

- In SMP2 and E-TM-40-07, the Component Model (set of interfaces to use for communication) has been defined in CORBA IDL, with a specific Platform Mapping to the C++ Platform
- In the draft of ECSS SMP, the interfaces are defined in C++ 11 only
- Justification: No plan to support other Platform next to C++

Concepts removed from the SMP2 Specification
Mandatory versus Optional Services

- SMP2 has the concepts of Mandatory and Optional Services
  - The Resolver is defined as an optional service

- In the draft of ECSS SMP, all Simulation Services are mandatory

- Justification: Use of optional services could affect portability.

Major Changes done since SMP2 Issue 1.2

Concepts added in the draft of E-ST-40-07

Concepts added in the draft of E-ST-40-07 Configuration File

- SMP2 has a single file (the Assembly) to define hierarchy and state
  - Initial Values of Simulation Models are defined in the Assembly

- In the draft of ECSS SMP (and in E-TM-40-07), a new Configuration file format has been added
  - This file format can not only be used for initial values, but as well at any later time
  - This format has already been used with SMP2 models

- Justification: Initialising and setting values from file is essential.

Concepts added in the draft of E-ST-40-07
Link Registry Service and Linking Components

- SMP2 has no support to track links to a Component
  - A Link can e.g. be a Reference to a Model
  - A Link can e.g. be a connection to an Event Sink
  - A Link can e.g. be a pointer to an Entry Point

- In the draft of ECSS SMP (and in A-TM-40-07), a Link Registry Services has been added
  - Components can implement ILinkingComponent to support this

- Justification: E.g. for separation, all links to a component shall be removed.

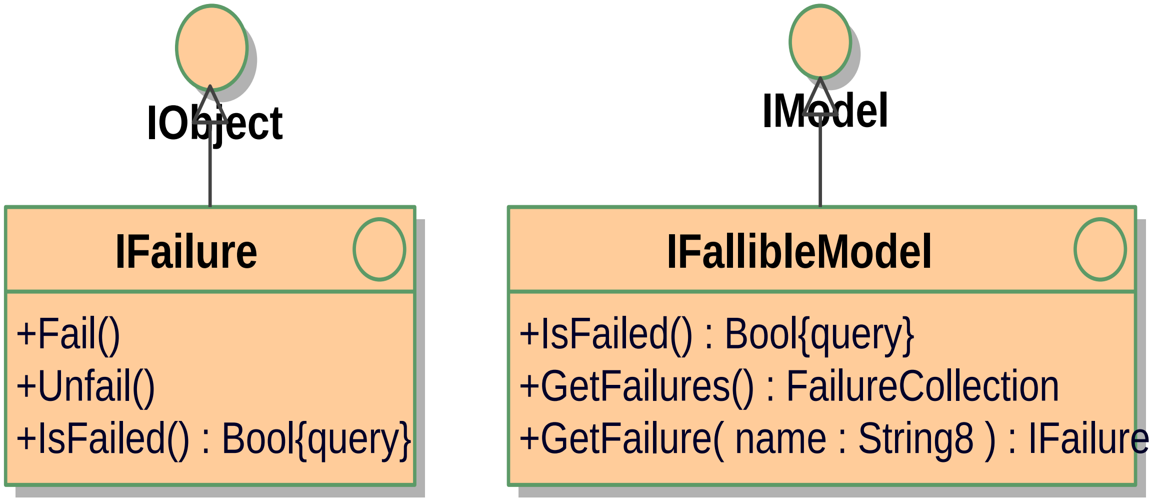
Concepts added in the draft of E-ST-40-07
Support for Failures

- SMP2 has no support for Failures

- In the draft of ECSS SMP (and in E-TM-40-07), two interfaces have been defined:
  - IFailure is an interface to a single Failure (object)
  - IFallibleModel extends IModel to provide access to Failures

- Justification: Modelling of failures shall be covered by the standard.

Concepts added in the draft of E-ST-40-07 Support for Failures (diagram)


IObject
IModel
IFailure
+Fail()
+Unfail()
+IsFailed() : Bool{query}
IFallibleModel
+IsFailed() : Bool{query}
+GetFailures() : FailureCollection
+GetFailure(name : String8) : IFailure

Concepts added in the draft of E-ST-40-07
String Support for Data Flow and Events

- SMP2 has partial support for strings: Only fixed-length strings are supported

- In the draft of ECSS SMP (and in E-TM-40-07), String8 has been added to the enumeration of primitive types, and to AnySimple
  - Use of Strings in Data Flow
  - Use of Strings in inter-component Events

- Justification: String support was found essential

Concepts added in the draft of E-ST-40-07
Improved Support for Data Flow

- SMP2 has basic support for data flow between fields of simple type, triggered externally e.g. via Assembly and Schedule files

- In the draft of ECSS SMP, this has been extended significantly to support fields of complex types (structures and arrays), and to allow for autonomous data transfer triggered by the models themselves

- Justification: Support for data flow was not found to be sufficient.

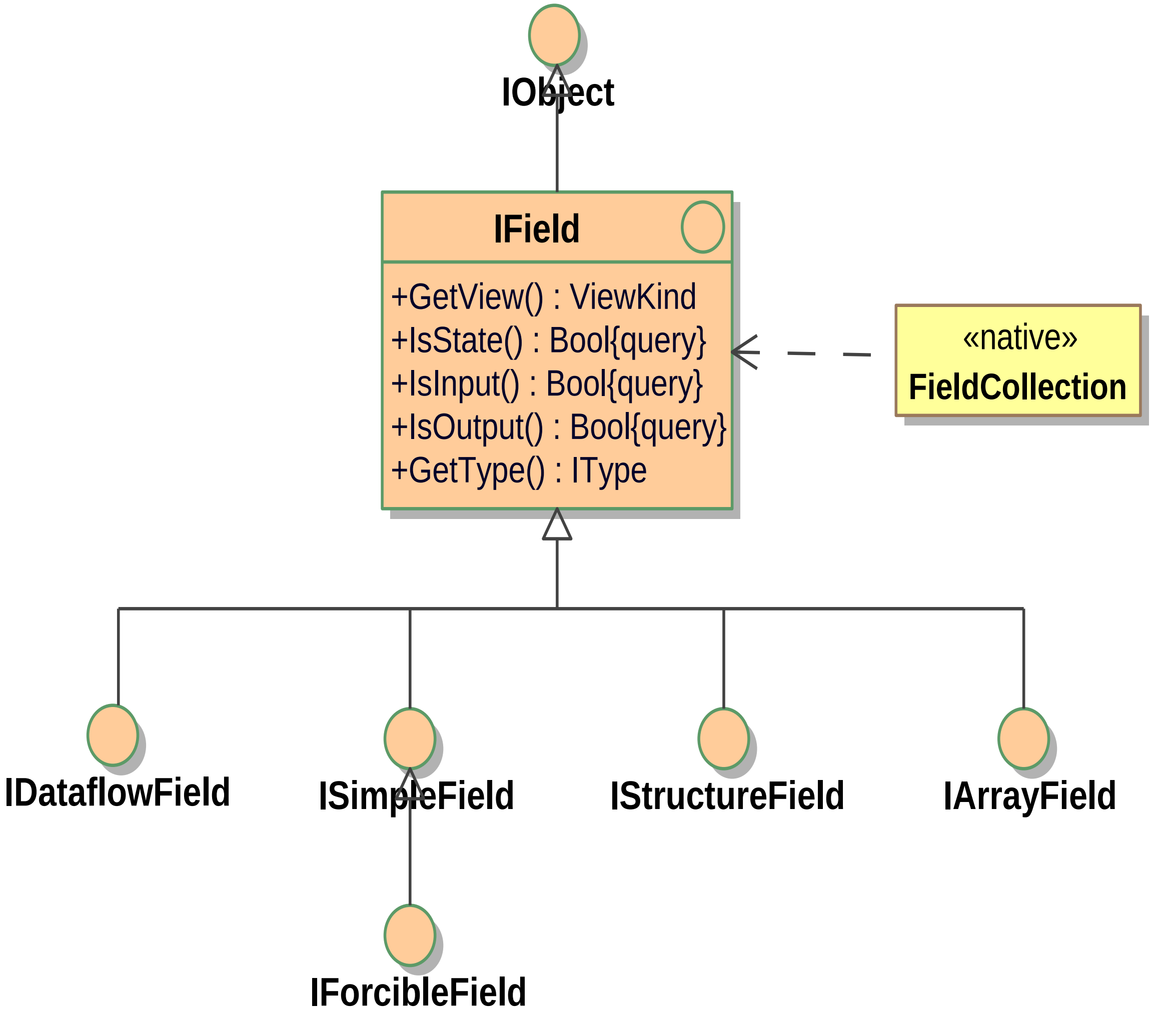
Concepts added in the draft of E-ST-40-07
Interfaces to Access Fields

- SMP2 allows publication of Fields, but has no interfaces for Fields

- In the draft of ECSS SMP (and partially in E-TM-40-07), various interfaces to access fields as objects have been defined
  - IField is the base interface for all fields
  - ISimpleField is the interface to a field of simple type
  - IStructureField is the interface to a field of structure type
  - IArrayField is the interface to a field of an array type
  - IDataflowField is the interface to a field that supports data flow
  - IForcibleField is the interface to a simple field allowing forcing

- Justification: Data flow and forcing require field objects (interfaces).

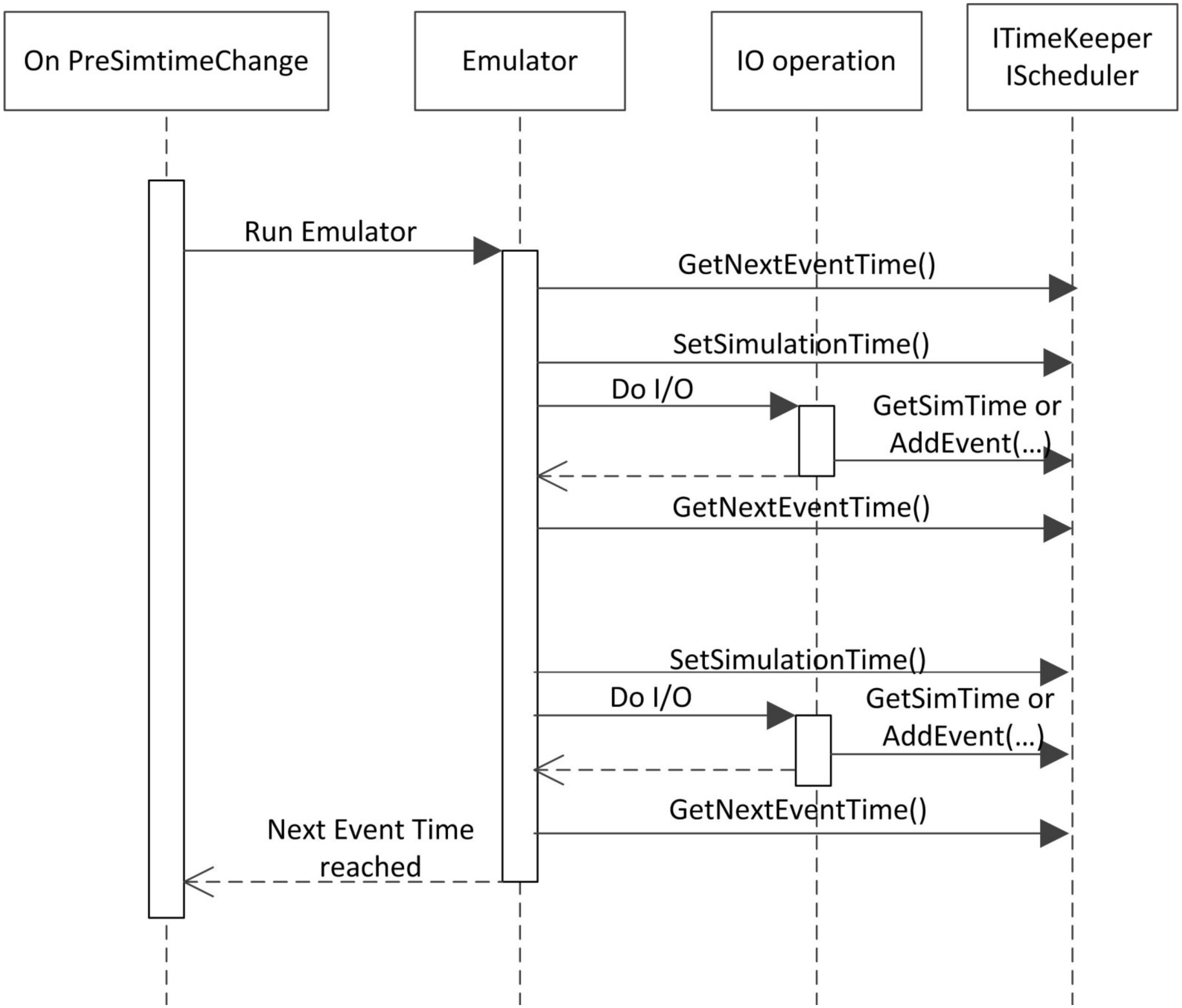
Concepts added in the draft of E-ST-40-07 Interfaces to Access Fields (diagram)



Concepts added in the draft of E-ST-40-07
Improved Support for Integration of Emulators

- SMP2 has limited support for an integration of an Emulator with the SMP Scheduler and Time Keeper services.
- In the draft of ECSS SMP, an Emulator can modify Simulation Time (via SetSimulationTime), so that during a call-out, other Models get the correct simulation time. For that, it can query for the simulation time of the next event on the scheduler (as it must not set a simulation time beyond the time of the next scheduled event).
- Justification: Accurate timing is essential for many simulators.

Integration of Emulators



Concepts added in the draft of E-ST-40-07
Visibility Kinds

- SMP2 has only two levels (visible/invisible) for Fields, and no way to control visibility of Operations or Properties.
- In the draft of ECSS SMP (and in E-TM-40-07), four visibility kinds are defined for Fields, Operations and Properties:
  - All
  - Expert
  - Debug
  - None
- Justification: Need to be able to control visibility in User Interface.

Major Changes done since SMP2 Issue 1.2

Clean-Up and Rationalisation

Clean-Up and Rationalisation
No Management namespace and Interfaces anymore

- In SMP2, several interfaces come in two flavours (e.g. `IObject` and `IManagedObject`), to minimise the effort for a minimal implementation.

- In the draft of ECSS SMP, no Management interfaces are included anymore, and the functionality has been merged with the "unmanaged" interfaces (or partially removed).

- Justification: It was found that all models implement Managed Interfaces anyway, as this can be fully automated via Code Generators.

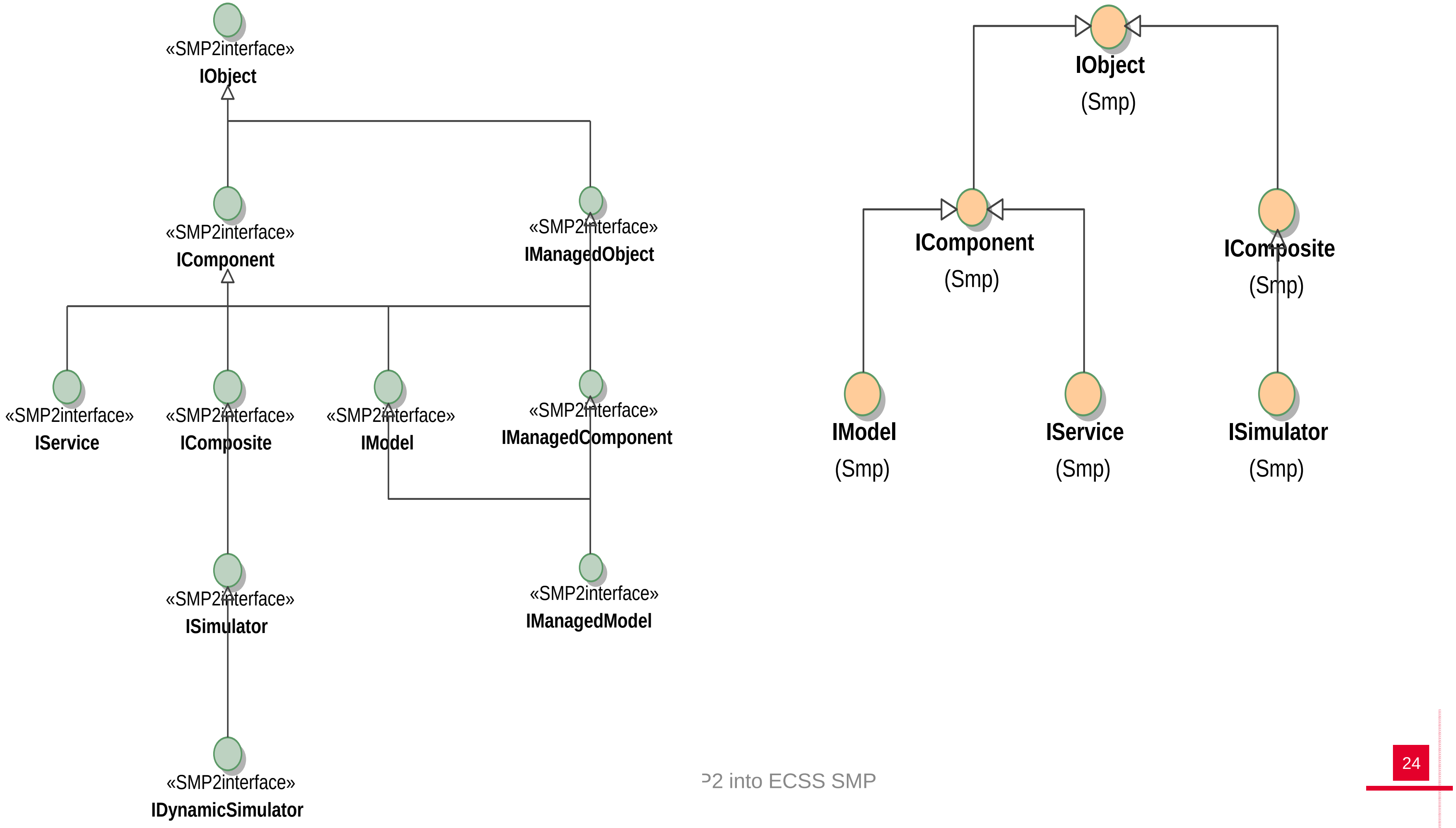
Clean-Up and Rationalisation
Services versus Models

- In SMP2, Models and Services differ fundamentally, both in their interface (IModel versus IService), and in their state machine (which is only defined for Models)

- In the draft of ECSS SMP, Services are almost identical to Models

- Justification: Services need all features that Models need:
  - They may publish information, and need to store their state
  - They may access other services (e.g. for Logging)

Clean-Up and Rationalisation
Changes in the Interface Hierarchy



Clean-Up and Rationalisation
Use of C++ 11

- In SMP2, several types (e.g. 64 bit integers) are defined in a header file “`Platform.h`” that needs tailoring for each platform.

- In the draft of ECSS SMP, no such tailoring is needed anymore, as C++ 11 provides all required data types as part of the language.
- Further, additional keywords like `nullptr` or `noexcept` are used.
- Finally, `throw` declarations have been removed (deprecated).

- Justification: Coming standard shall be based on C++ 11.

Clean-Up and Rationalisation
Removal of all implementation from the Standard

- In SMP2, some code is provided e.g. for the exceptions defined in the specification, or for the registration of types into the type registry.

- In the draft of ECSS SMP, no implementation code is included anymore.

- Justification: The standard shall only define interfaces.

First Feedback from a Migration of SIMULUS Models

Concepts added in the draft of E-ST-40-07

First Feedback from a Migration of SIMULUS Models Approach

- The SMP2 Generic Models (GENM) of SIMULUS cover a variety of SMP2 based interfaces, data types, classes and models
- In a first exercise, the existing source code was compiled against the new interfaces defined for ECSS SMP
- Where possible, `#define` was used to "translate" existing code
- Remaining changes were minor, and can be categorised:
  - Changes in code generated by a Code Generator
  - Changes in hand-written code
- The following slide provides some high-level statistics

- Note: A full migration would require additional changes, mainly due to the use of new features (e.g. forcing and failing).

First Feedback from a Migration of SIMULUS Models
Some Numbers

Model	Files	Lines	Modified	Lines
Common	353	42724	16	109
PEM	119	29350	2	2
SIMDYN	37	12073	1	4
TNET	69	14086	1	3
SENSE	91	18219	1	2
Generic	172	42724	3	8

- The SIMPACK Generic Model is missing, as it has not been migrated yet
  - It defines its own AnySimple with operator overloading
  - Problems are not with SMP interface, but with use of MDK

THANK **YOU** FOR YOUR ATTENTION

TELESPAZIO
a LEONARDO and THALES company