TERMINALS IN SSP

Dag Brück
Dymola R&D Technology Director



© Dassault Systèmes | Confidential Information | 2024

“TERMINALS IN SSP” DECONSTRUCTED

- SSP
  - What is System Structure and Parameterization (SSP)?
  - What SSP is not
- in
  - Why “in” and not “outside”
  - Different needs → multiple solutions → one solution
- Terminals
  - Why we need terminals and what is a terminal?
  - How we can define terminals? (Three times)





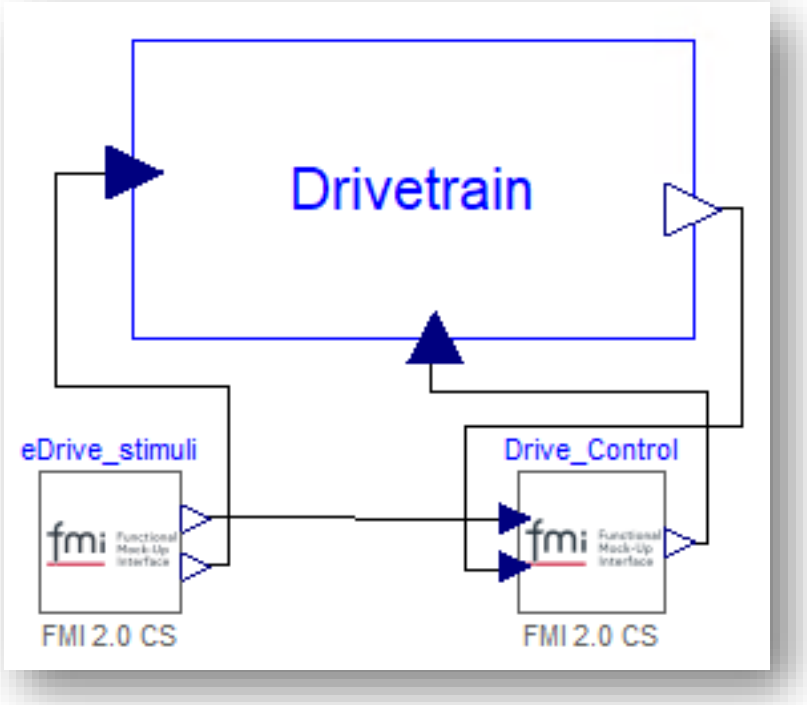
© Dassault Systèmes | Confidential Information | 2024

WHAT IS SSP?

© Dassault Systèmes | Confidential Information | 2024

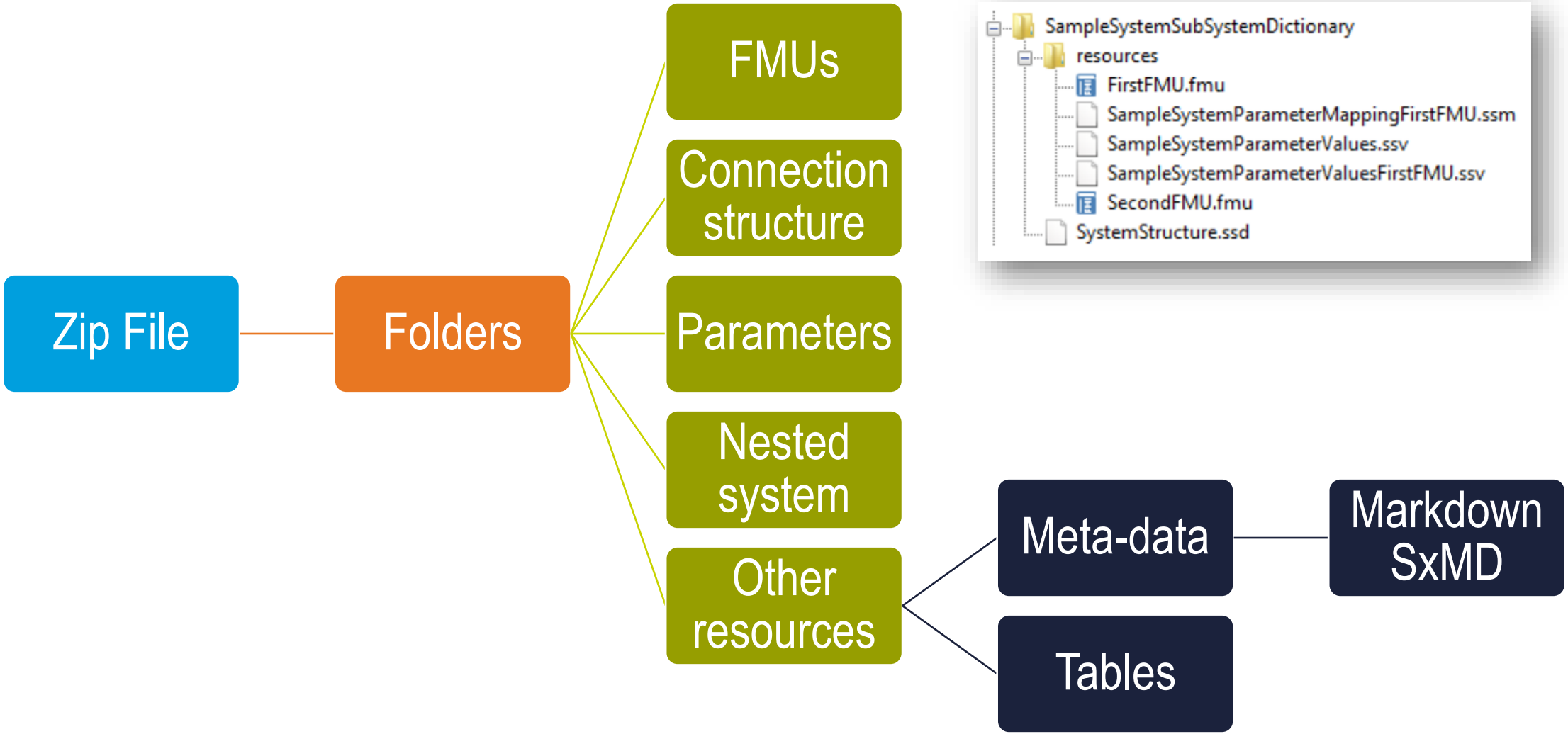
EXECUTIVE SUMMARY OF SSP

- Tool-independent format for the description and simulation of system structures
  - A network of component models with their signal flows
  - Typical: FMUs + connections
- Extends FMI with parameters and meta-data
  - Complete simulation package with documentation
- ZIP-based packaging
  - XML-based file formats for structure, parameters, etc.
  - Effective distribution of code and other resources
- Open format
  - Can be extended with additional resources and data
  - Can support all steps from architecture to simulation



© Dassault Systèmes | Confidential Information | 2024

SIMPLIFIED STRUCTURE OF AN SSP FILE



© Dassault Systèmes | Confidential Information | 2024

SSP 2.0 – THE CURRENT RELEASE

- Support for representing Modelica components and connectors in SSP
  - Only references to Modelica models, Modelica code is not part of SSP
  - Acausal (?) connector: `kind="undefined"`
- Support for FMI 3 extensions
  - Basic types, such as Float32
  - Scheduled execution (accepted in the SSP file, but not semantically defined)
  - Clocked signals
  - Array connectors and parameters
- Also
  - Local variable defined as connector with `kind="local"`
  - Initial meta-data support from “SSP LS Traceability” project
- Every SSP 1.0 file is compatible with SSP 2.0

© Dassault Systèmes | Confidential Information | 2024

WHAT SSP IS NOT

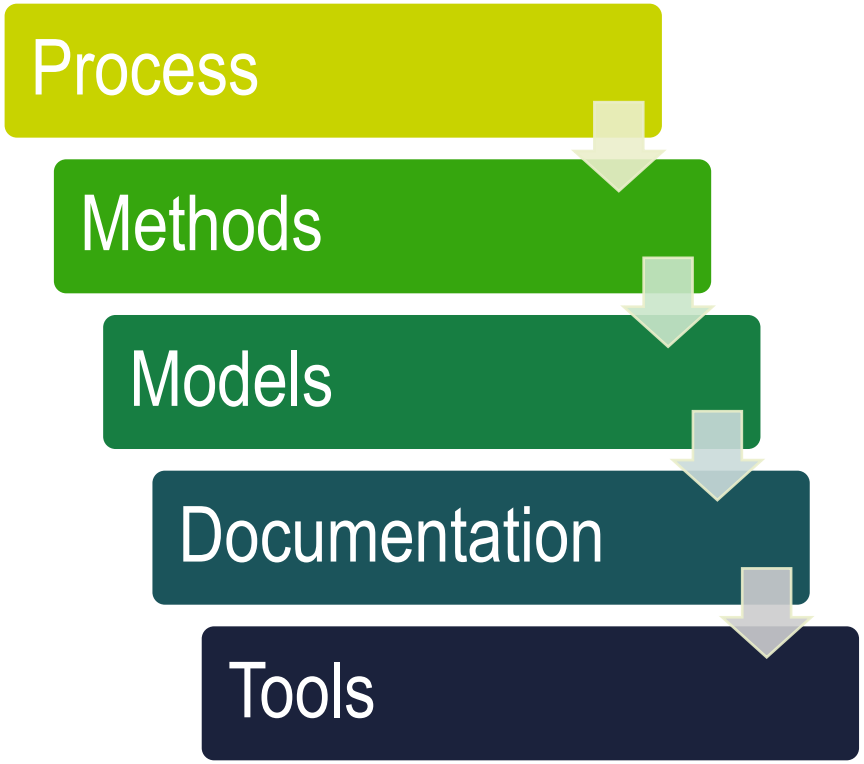
And may not have any intention of being

- SSP is not SysML
  - Focus has been simulation, not MBSE in general
  - Approximately Interconnection View – but no requirement, action, stakeholder, etc.
  - Still interest in SSP as a light-weight, tool-independent, architecture description
- SSP has almost no semantics
  - Describes structure and a few concepts carried over from FMI
  - Simulation semantics is implied by what is connected and the tool
- SSP is not limited to the plain vanilla connected-FMU scope

© Dassault Systèmes | Confidential Information | 2024

SSP LS TRACEABILITY

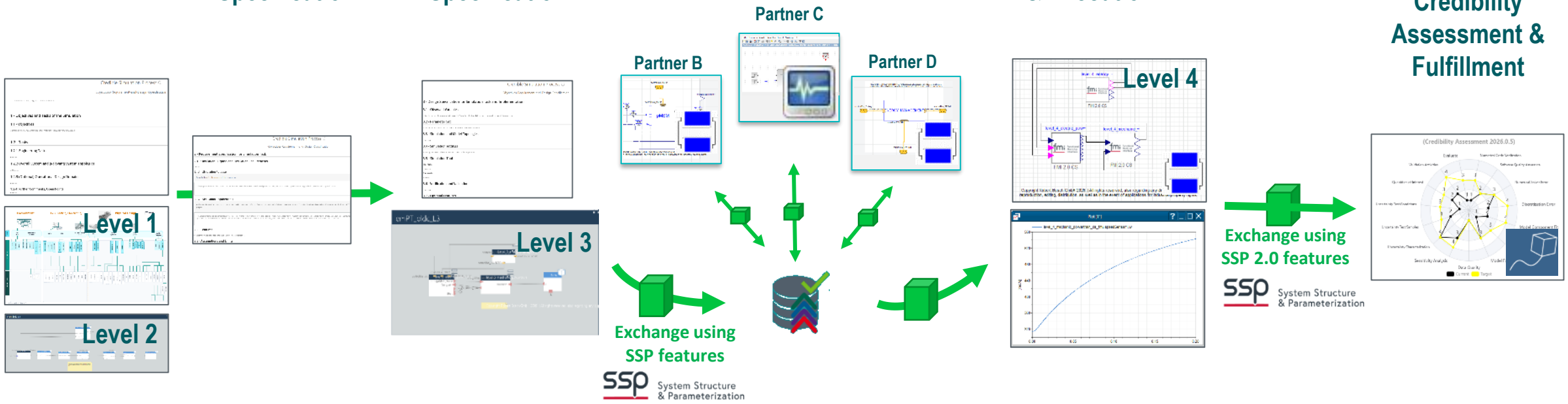
- Support traceability of decision and simulation tasks
  - Modeling + simulation → process + credible result
  - Standardize format and link into the SSP framework
  - Designed to be flexible – adaptable to (corporate) standards
- Building blocks for credibility
  - Human- and machine-readable meta data
  - Documentation
  - Supports the development process by incremental updates



© Dassault Systèmes | Confidential Information | 2024

CREDIBLE SIMULATION PROCESS

System Structure & Parameterization



From Antoine Vandamme, Robert Bosch GmBH
ProSTEP ivip Smart System Engineering

© Dassault Systèmes | Confidential Information | 2024

CATIA

DASSAULT SYSTEMES

THE FUTURE OF SSP

- SSP will grow in scope
  - SSP LS Traceability
  - Improved FMI 3 compatibility
  - Extend capabilities from FMI to SSP systems, e.g. terminals
- SSP will be widely used
  - Fills a concrete need
  - Not overly complex – manageable to implement and deploy
  - Quite flexible and extensible
- Engage yourself
  - Start to use SSP
  - Request support from your vendor
  - Give feedback on SSP
  - Engage in the development of SSP
- The future of SSP includes *you*

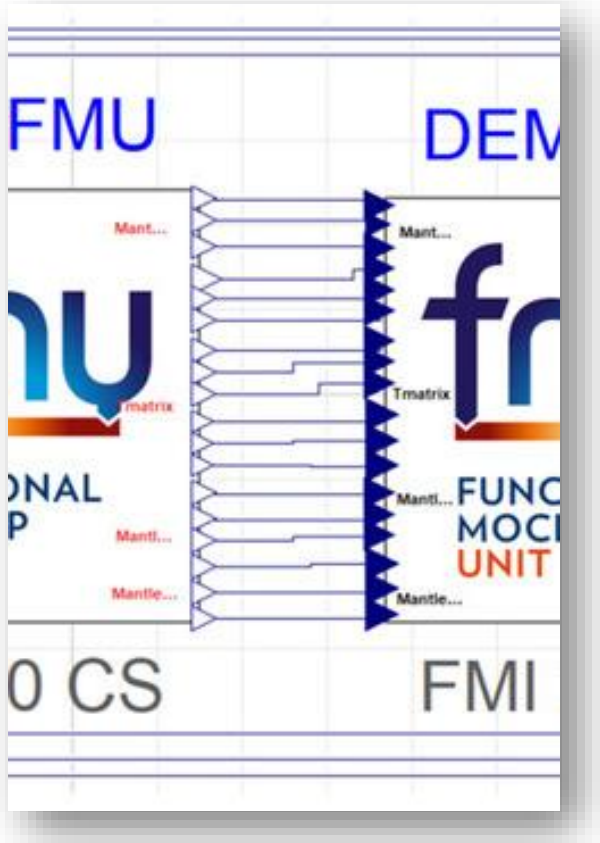
© Dassault Systèmes | Confidential Information | 2024

DEFINITION OF TERMINALS (THREE ANSWERS)

© Dassault Systèmes | Confidential Information | 2024

WHY STRUCTURED CONNECTORS/TERMINALS?

- Group single connectors
  - Avoid the maze of connection lines that is common
  - Input, output, acausal, nested terminals
  - (But array connectors are an orthogonal concept)
  - Can be managed as one unit, for example to make a connection
- Separate the terminal definition from its use
  - Terminal definitions can be reused
  - Terminal can be updated without affecting the system topology
  - Component implementation can be changed while keeping terminals
- Declutter the namespace
  - Members of the terminal should be local
- Have existed for 60+ years
  - In both programming and simulation languages

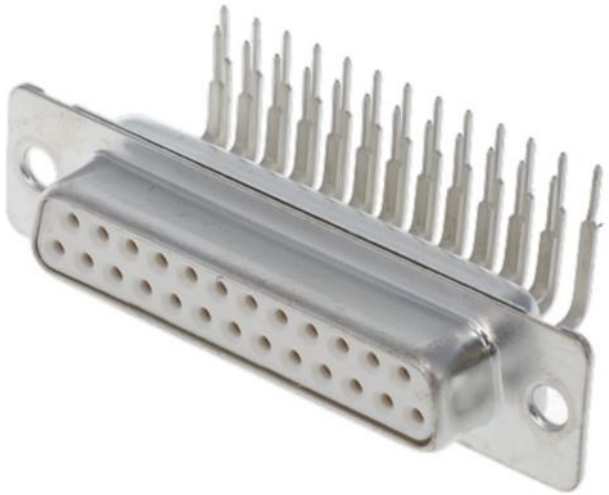


© Dassault Systèmes | Confidential Information | 2024

TERMINALS IN FMI 3

- Good news: FMI 3 already has terminals
  - But it is an add-on that you can ignore…
  - The single connectors still exist as independent ports
  - They can grouped and be referenced as a terminal (at the same hierarchical level)
- What the FMI 3 terminal does
  - Gives the group (terminal) a name
  - Enumerates the member variables and gives a local name
  - Binds the member variable to an existing connector variable
- Issues from an SSP view point
  - SSP deals with more than one component – FMI does not
  - Decluttering is not great, because the single connectors still exist
  - Limited reuse because terminals must be at the same level (no types)
  - Obviously no acausal connectors – issue because SSP has them



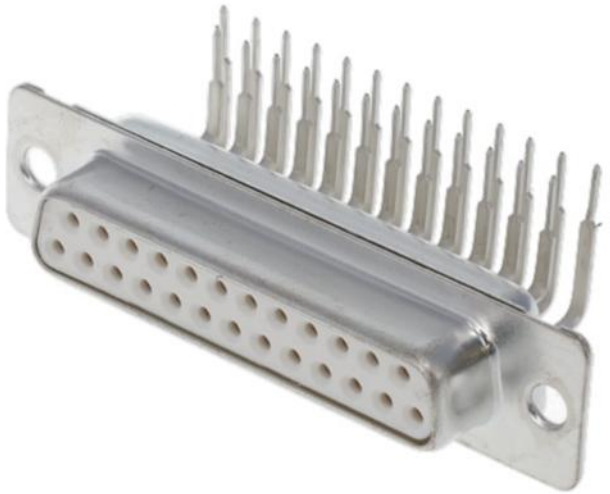


© Dassault Systèmes | Confidential Information | 2024

TERMINALS IN MODELICA

- Good news: Modelica has connectors (=terminals)
  - Variables can grouped and be referenced as a connector
  - Acausal variables and connections
- What the Modelica terminal does
  - Gives the group (terminal) a name
  - Enumerates the member variables and gives a local name
  - No connector member variable on the outside
  - Reuse is common → libraries of “interfaces”
- Issues from an SSP view point
  - Too much semantics if your focus is on FMI components
  - Backward compatibility (with FMI scalar connectors) is lost
  - SSP connection semantics is thin – you get what the tool gives you

```
connector Pin
  Voltage v;
  flow Current i;
end Pin;
```



© Dassault Systèmes | Confidential Information | 2024

TERMINALS IN SSP – AS PROPOSED

- Key design goals
  - Reusable terminal definitions, separate from their use
  - No redundant terminal definitions
  - Terminal definitions are semi-global (like enumerations)
  - Good enough for both FMI and Modelica semantics
- What has been sacrificed?
  - No 1-to-1 compatibility with FMI or Modelica
  - Have to chose if we want to use the FMI terminal connector or the single connectors (not both)
  - No composition of definitions in SSP (no extends)
  - Connection semantics is vague in good SSP tradition
- Tool and specification potential
  - FMI single connectors can be re-created from the SSP terminal definition
  - Complement terminals (input ←→ output)

```
<ssd:Component
  name="Source_component">
  <ssd:Connectors>
    <ssd:Connector
      kind="undefined"
      name="tSource">
      <ssd:Terminal name="tSourceType"/>
    </ssd:Connector>
  </ssd:Connectors>
</ssd:Component>
```

```
<ssd:Terminals>
<ssd:TerminalDefinition name="tSourceType">
  <ssd:Connector kind="output" name="pos">
    <ssc:Real/>
  </ssd:Connector>
  <ssd:Connector kind="input" name="feedback">
    <ssc:Real/>
  </ssd:Connector>
</ssd:TerminalDefinition>
</ssd:Terminals>
```

© Dassault Systèmes | Confidential Information | 2024

RE-CONSTRUCTION

© Dassault Systèmes | Confidential Information | 2024

“TERMINALS IN SSP” RE-CONSTRUCTED

- SSP needs terminals
  - Because the components we use have terminals/connectors
  - Manage a complex connection structure
- One terminal to unite them all
  - There are several terminal concepts to chose from
  - Different needs/history → several solutions
  - The goal has been to find a unified definition for SSP
  - Maintain the freedom to change component implementation
- The future
  - This is an ongoing effort in the SSP design group
  - We greatly appreciate your input (or acausal interaction)
  - We encourage implementation in as many tools as possible
  - If everything goes really well, part of SSP 2.1





© Dassault Systèmes | Confidential Information | 2024

ACKNOWLEDGEMENTS

- The SSP design group
  - Pierre Mai
  - Nikolai Fast

- The Modelica community
  - Sven Erik Mattsson
  - Hilding Elmqvist
  - My colleagues in Lund

© Dassault Systèmes | Confidential Information | 2024

© Dassault Systèmes | Confidential Information | 2024



DASSAULT SYSTEMES