NTNU | Norwegian University of Science and Technology

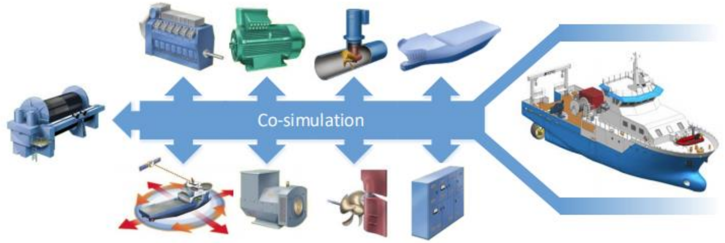
Enhancing SSP Creation using sspgen

American Modelica Conference 2022, Dallas, TX

Lars Ivar Hatledal
Associate Professor, NTNU Aalesund, Norway
26.10.2022

Background

- **Co-simulation** is used by NTNU campus Aalesund
  - Mostly to simulate maritime systems



- FMI $\longrightarrow$ Individual models
- SSP $\longrightarrow$ Complete systems

System Structure & Parameterization (SSP)

- The SSP is a tool independent standard to define complete systems consisting of one or more models.
  - including its parameterization that can be transferred between simulation tools.
- A model could be an FMU adhering to the FMI standard.
- An SSP is a zip archive that includes an XML document describing the system, connections and any initial values of the models together with any required data.

ssp System Structure & Parameterization

Enabling tools for co-simulation

- Lower-level (Individual models)

• FMI4j/FMU4j	(FMI import/export in Java)
• fmi4cpp/fmu4cpp	(FMI import/export in C++)
• PythonFMU	(FMI export in Python)
• FMU-proxy/proxy-fmu	(Distributed FMU access)

sspgen -> Defining simulations

- Higher-level (Orchestration)
    - libcosim (Open-simulation-platform, C++/CLI)
    - Vico (NTNU, Java/CLI)
    - Ecos (NTNU, C++/CLI)

sspgen

- *sspgen* is a Kotlin DSL for easing the creation of SSP 1.0 compatible systems.
  - Creates the SystemStructure.xml.
  - Handles packaging of the SystemStructure.xml and any additional resources (local files, URLs) into a SSP archive (.zip).
  - Checks the system for correctness.

- The DSL is available though Maven and can be referenced in scripts

- Allows SSP definitions to be distributed as a script.
  - Easy to modify and share.
  - Expressions as initial values!
  - Can reference files from both the file systems and URLs.

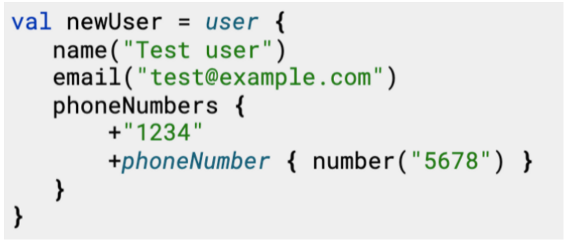
https://github.com/Ecos-platform/sspgen

Domain-specific languages (DSLs)

- A DSL is a computer language specialized to a particular application domain.
- Two main classes of DSLs:
  - External
    - LaTeX, CMake++
  - Embedded
    - Gradle++

Kotlin DSL

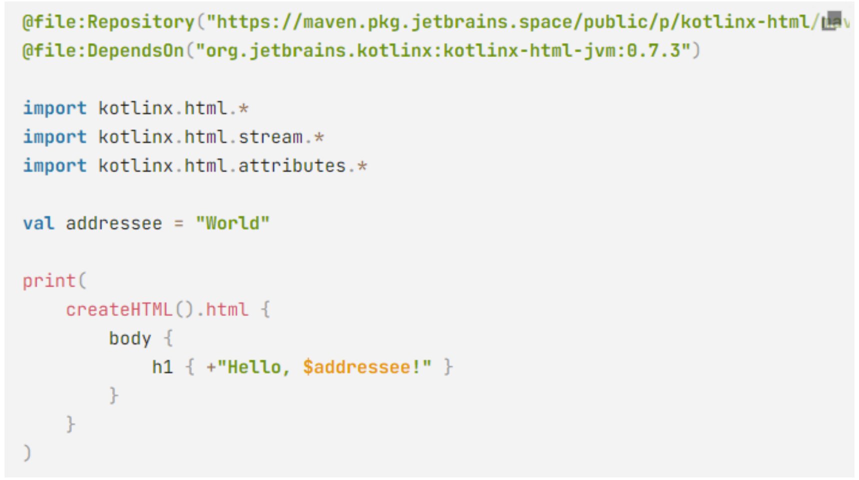
- Kotlin is a modern language known for its readable, clean, and concise syntax.
  - Default language for Android development.
  - Drop-in replacement for Java.
- With its advanced functional programming capabilities, we can create type-safe, statically typed builders that act as DSLs
  - which are suitable for expressing complex hierarchical data structures in a semi-declarative way.



```kotlin
val newUser = user {
   name("Test user")
   email("test@example.com")
   phoneNumbers {
        +"1234"
        +phoneNumber { number("5678") }
   }
}
```

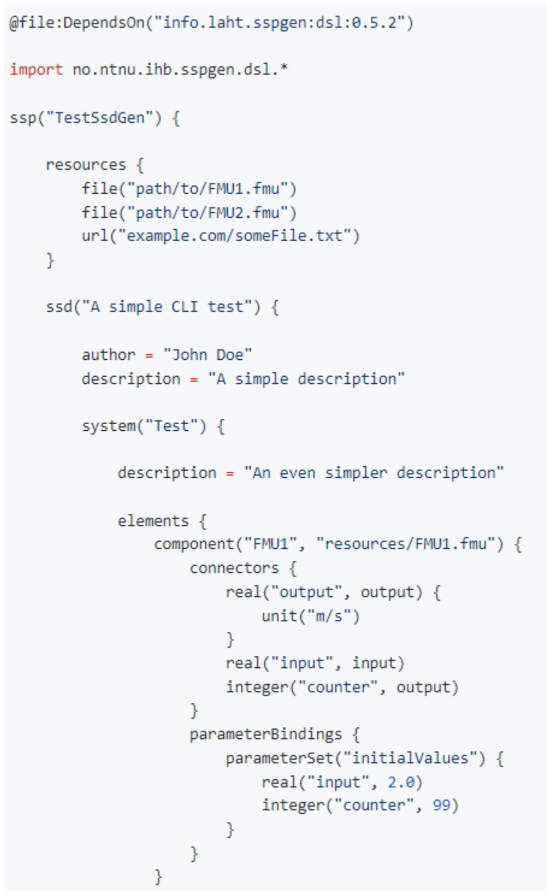
Kotlin scripting

- Kotlin code can execute as standalone scripts.
  - Runtime dependency resolution.

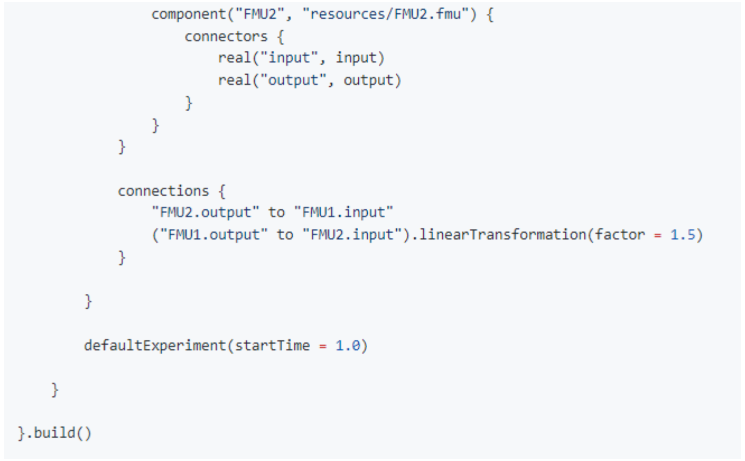


```
@file:Repository("https://maven.pkg.jetbrains.space/public/p/kotlinx-html/
@file:DependsOn("org.jetbrains.kotlinx:kotlinx-html-jvm:0.7.3")

import kotlinx.html.*
import kotlinx.html.stream.*
import kotlinx.html.attributes.*

val addressee = "World"

print(
    createHTML().html {
        body {
            h1 { +"Hello, $addressee!" }
        }
    }
)
```

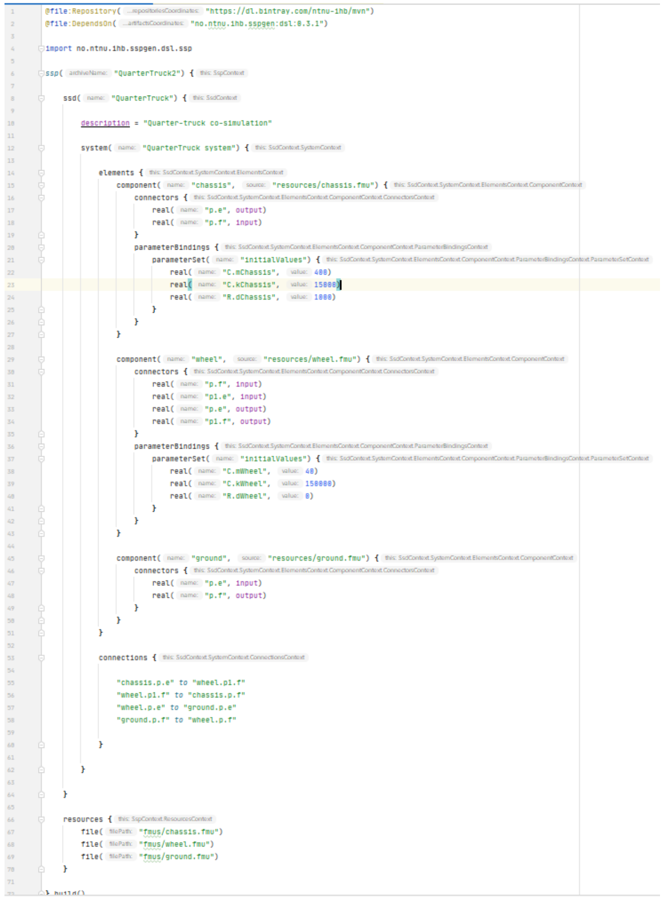
Anatomy of a sspgen script



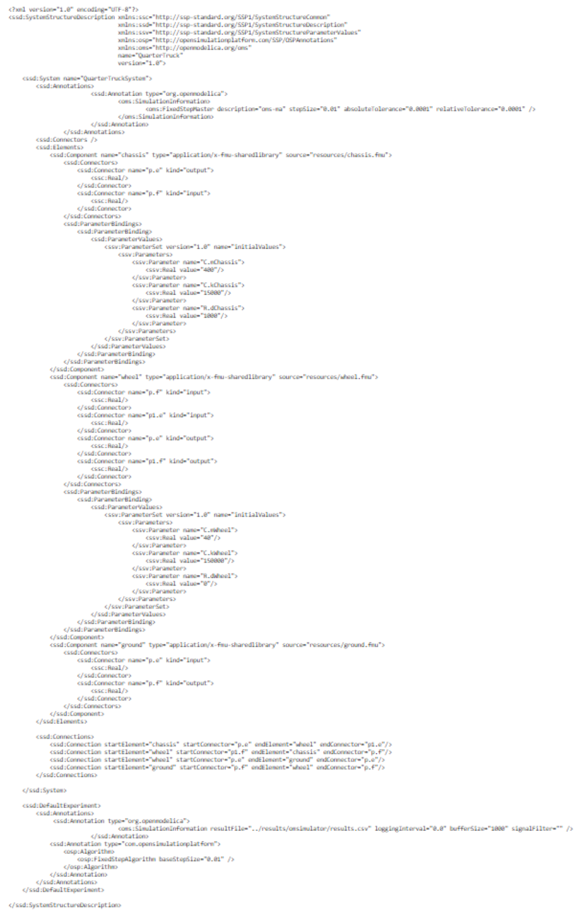
```
@file:DependsOn("info.laht.sspgen:dsl:0.5.2")

import no.ntnu.ihb.sspgen.dsl.*

ssp("TestSsdGen") {

    resources {
        file("path/to/FMU1.fmu")
        file("path/to/FMU2.fmu")
        url("example.com/someFile.txt")
    }

    ssd("A simple CLI test") {

        author = "John Doe"
        description = "A simple description"

        system("Test") {

            description = "An even simpler description"

            elements {
                component("FMU1", "resources/FMU1.fmu") {
                    connectors {
                        real("output", output) {
                            unit("m/s")
                        }
                        real("input", input)
                        integer("counter", output)
                    }
                    parameterBindings {
                        parameterSet("initialValues") {
                            real("input", 2.0)
                            integer("counter", 99)
                        }
                    }
                }
```



```
                component("FMU2", "resources/FMU2.fmu") {
                    connectors {
                        real("input", input)
                        real("output", output)
                    }
                }
            }

            connections {
                "FMU2.output" to "FMU1.input"
                ("FMU1.output" to "FMU2.input").linearTransformation(factor = 1.5)
            }

        }

        defaultExperiment(startTime = 1.0)

    }

}.build()
```



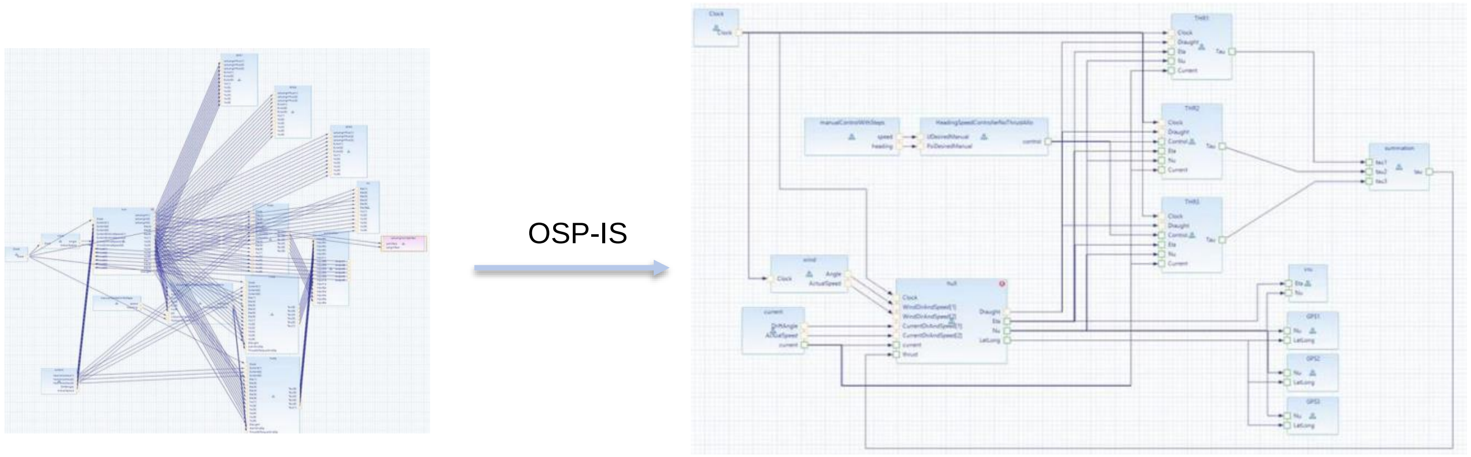
```kotlin
@file:Repository("https://dl.bintray.com/ntnu-ihb/mvn")
@file:DependsOn("no.ntnu.ihb.sspgen:dsl:0.3.1")

import no.ntnu.ihb.sspgen.dsl.ssp

ssp("QuarterTruck2") {

    ssd("QuarterTruck") {

        description = "Quarter-truck co-simulation"

        system("QuarterTruck system") {

            elements {
                component("chassis", "resources/chassis.fmu") {
                    connectors {
                        real("p.e", output)
                        real("p.f", input)
                    }
                    parameterBindings {
                        parameterSet("initialValues") {
                            real("C.mChassis", 400)
                            real("C.kChassis", 15000)
                            real("R.dChassis", 1000)
                        }
                    }
                }

                component("wheel", "resources/wheel.fmu") {
                    connectors {
                        real("p.f", input)
                        real("p1.e", input)
                        real("p.e", output)
                        real("p1.f", output)
                    }
                    parameterBindings {
                        parameterSet("initialValues") {
                            real("C.mWheel", 40)
                            real("C.kWheel", 150000)
                            real("R.dWheel", 0)
                        }
                    }
                }

                component("ground", "resources/ground.fmu") {
                    connectors {
                        real("p.e", input)
                        real("p.f", output)
                    }
                }
            }

            connections {

                "chassis.p.e" to "wheel.p1.f"
                "wheel.p1.f" to "chassis.p.f"
                "wheel.p.e" to "ground.p.e"
                "ground.p.f" to "wheel.p.f"

            }

        }

    }

    resources {
        file("fmus/chassis.fmu")
        file("fmus/wheel.fmu")
        file("fmus/ground.fmu")
    }

} build()
```



```xml
<?xml version="1.0" encoding="UTF-8"?>
<ssd:SystemStructureDescription xmlns:ssc="http://ssp-standard.org/SSP1/SystemStructureCommon"
                                xmlns:ssd="http://ssp-standard.org/SSP1/SystemStructureDescription"
                                xmlns:ssv="http://ssp-standard.org/SSP1/SystemStructureParameterValues"
                                xmlns:osp="http://opensimulationplatform.com/SSP/OSPAnnotations"
                                xmlns:oms="http://openmodelica.org/oms"
                                name="QuarterTruck"
                                version="1.0">

    <ssd:System name="QuarterTruckSystem">
        <ssd:Annotations>
                    <ssd:Annotation type="org.openmodelica">
                        <oms:SimulationInformation>
                            <oms:FixedStepMaster description="oms-ma" stepSize="0.01" absoluteTolerance="0.0001" relativeTolerance="0.0001" />
                        </oms:SimulationInformation>
                    </ssd:Annotation>
                </ssd:Annotations>
        <ssd:Connectors />
        <ssd:Elements>
            <ssd:Component name="chassis" type="application/x-fmu-sharedlibrary" source="resources/chassis.fmu">
                <ssd:Connectors>
                    <ssd:Connector name="p.e" kind="output">
                        <ssc:Real/>
                    </ssd:Connector>
                    <ssd:Connector name="p.f" kind="input">
                        <ssc:Real/>
                    </ssd:Connector>
                </ssd:Connectors>
                <ssd:ParameterBindings>
                    <ssd:ParameterBinding>
                        <ssd:ParameterValues>
                            <ssv:ParameterSet version="1.0" name="initialValues">
                                <ssv:Parameters>
                                    <ssv:Parameter name="C.mChassis">
                                        <ssv:Real value="400"/>
                                    </ssv:Parameter>
                                    <ssv:Parameter name="C.kChassis">
                                        <ssv:Real value="15000"/>
                                    </ssv:Parameter>
                                    <ssv:Parameter name="R.dChassis">
                                        <ssv:Real value="1000"/>
                                    </ssv:Parameter>
                                </ssv:Parameters>
                            </ssv:ParameterSet>
                        </ssd:ParameterValues>
                    </ssd:ParameterBinding>
                </ssd:ParameterBindings>
            </ssd:Component>
            <ssd:Component name="wheel" type="application/x-fmu-sharedlibrary" source="resources/wheel.fmu">
                <ssd:Connectors>
                    <ssd:Connector name="p.f" kind="input">
                        <ssc:Real/>
                    </ssd:Connector>
                    <ssd:Connector name="p1.e" kind="input">
                        <ssc:Real/>
                    </ssd:Connector>
                    <ssd:Connector name="p.e" kind="output">
                        <ssc:Real/>
                    </ssd:Connector>
                    <ssd:Connector name="p1.f" kind="output">
                        <ssc:Real/>
                    </ssd:Connector>
                </ssd:Connectors>
                <ssd:ParameterBindings>
                    <ssd:ParameterBinding>
                        <ssd:ParameterValues>
                            <ssv:ParameterSet version="1.0" name="initialValues">
                                <ssv:Parameters>
                                    <ssv:Parameter name="C.mWheel">
                                        <ssv:Real value="40"/>
                                    </ssv:Parameter>
                                    <ssv:Parameter name="C.kWheel">
                                        <ssv:Real value="150000"/>
                                    </ssv:Parameter>
                                    <ssv:Parameter name="R.dWheel">
                                        <ssv:Real value="0"/>
                                    </ssv:Parameter>
                                </ssv:Parameters>
                            </ssv:ParameterSet>
                        </ssd:ParameterValues>
                    </ssd:ParameterBinding>
                </ssd:ParameterBindings>
            </ssd:Component>
            <ssd:Component name="ground" type="application/x-fmu-sharedlibrary" source="resources/ground.fmu">
                <ssd:Connectors>
                    <ssd:Connector name="p.e" kind="input">
                        <ssc:Real/>
                    </ssd:Connector>
                    <ssd:Connector name="p.f" kind="output">
                        <ssc:Real/>
                    </ssd:Connector>
                </ssd:Connectors>
            </ssd:Component>
        </ssd:Elements>

        <ssd:Connections>
            <ssd:Connection startElement="chassis" startConnector="p.e" endElement="wheel" endConnector="p1.e"/>
            <ssd:Connection startElement="wheel" startConnector="p1.f" endElement="chassis" endConnector="p.f"/>
            <ssd:Connection startElement="wheel" startConnector="p.e" endElement="ground" endConnector="p.e"/>
            <ssd:Connection startElement="ground" startConnector="p.f" endElement="wheel" endConnector="p.f"/>
        </ssd:Connections>

    </ssd:System>

    <ssd:DefaultExperiment>
        <ssd:Annotations>
             <ssd:Annotation type="org.openmodelica">
                                <oms:SimulationInformation resultFile="../results/omsimulator/results.csv" loggingInterval="0.0" bufferSize="1000" signalFilter="" />
                        </ssd:Annotation>
            <ssd:Annotation type="com.opensimulationplatform">
                <osp:Algorithm>
                    <osp:FixedStepAlgorithm baseStepSize="0.01" />
                </osp:Algorithm>
            </ssd:Annotation>
        </ssd:Annotations>
    </ssd:DefaultExperiment>

</ssd:SystemStructureDescription>
```

OSP-IS

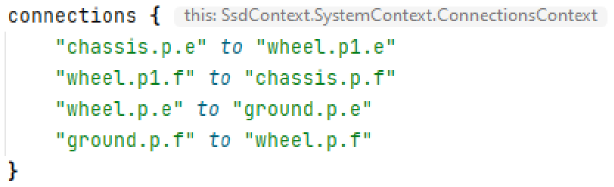
- The OSP interface specification (OSP-IS) is an addition to the FMI standard which provide:
  - A method for adding semantic meaning to model interface variables.
  - A simpler model connection process.
  - Validation of semantically correct simulations.



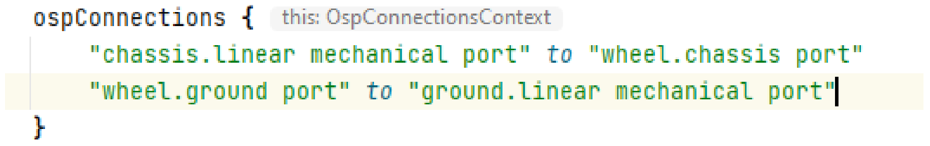
https://opensimulationplatform.com/specification/

sspgen + OSP-IS = True

- *sspgen* can transpile compound OSP-IS connections to single scalar connections supported by SSP.
- Additionally, *sspgen* can verify the connections according to the standard (both SSP and OSP-IS).



```
connections {  this: SsdContext.SystemContext.ConnectionsContext
    "chassis.p.e" to "wheel.p1.e"
    "wheel.p1.f" to "chassis.p.f"
    "wheel.p.e" to "ground.p.e"
    "ground.p.f" to "wheel.p.f"
}
```



```
ospConnections {  this: OspConnectionsContext
    "chassis.linear mechanical port" to "wheel.chassis port"
    "wheel.ground port" to "ground.linear mechanical port"
}
```

Other sspgen features

- proxy-fmu support
  - Distributed FMUs
  - https://github.com/open-simulation-platform/proxy-fmu

- PythonFMU integration
  - Build FMUs on demand from Python code
  - https://github.com/NTNU-IHB/PythonFMU

- FMI-VDM-Model integration
  - Optional static analysis of included FMUs
  - https://github.com/INTO-CPS-Association/FMI-VDM-Model

Conclusion

- Utilizing the SSP, simulations can be defined in a standardized way and *sspgen:*
    - Makes them easier to create, modify and share.
    - Enables non-trivial parameters to be defined.
    - Enables the OSP-IS to be used in this context, making it usable by a larger audience.

Future work

- Add more tests and general polishing of the code.
- Improve current SSP 1.0 support.
- Support FMI 3.0 and future SSP versions.

Q & A