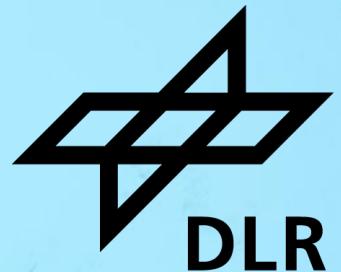


COMPLETE MODEL STORAGE – A SOLVER-AGNOSTIC HIERARCHICAL APPROACH

VMAP Working Group „Complete Model Storage“

19.09.2022

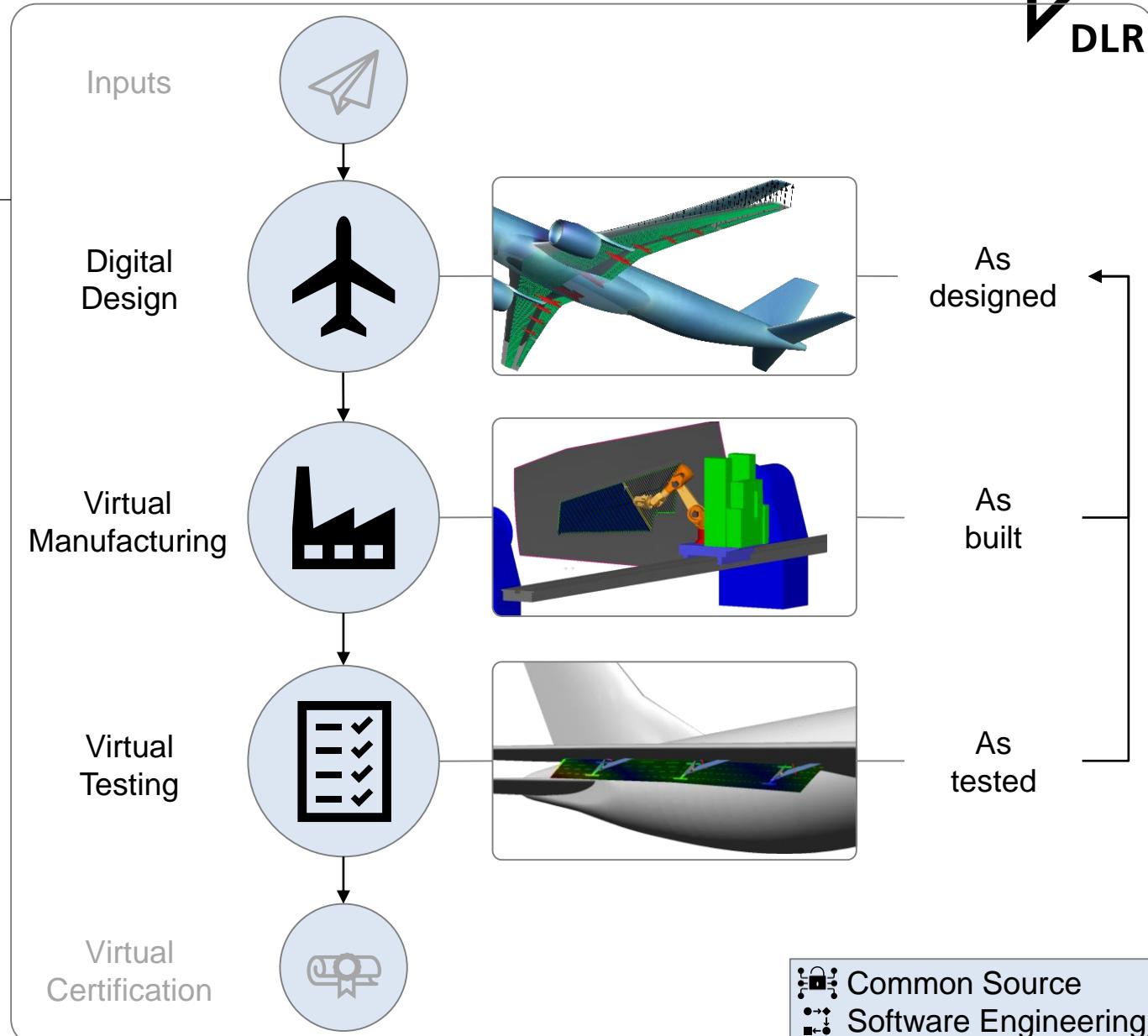
Martin Rädel



TASK/MOTIVATION

Task

- Virtual product development
- Multiple simulation steps
- Methods e.g.:
 - Numerical
 - FEM
 - Peridynamics
 - ...
 - Semi-analytical
 - Ritz
 - ...
 - Analytical
- Central for each method:
Model



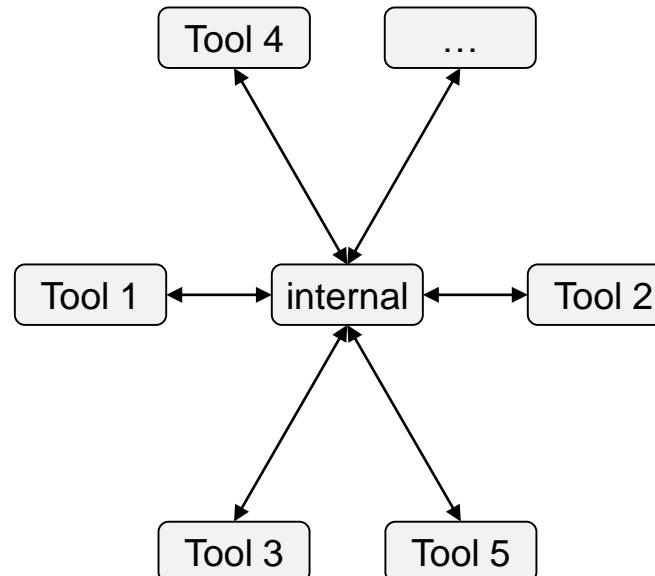
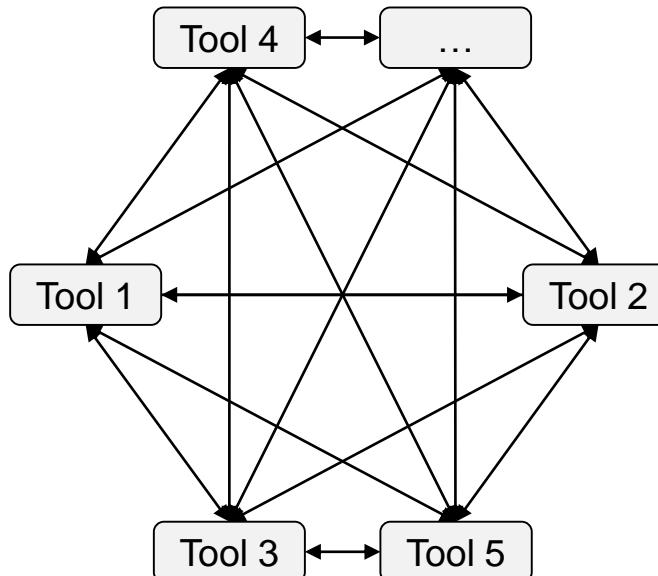
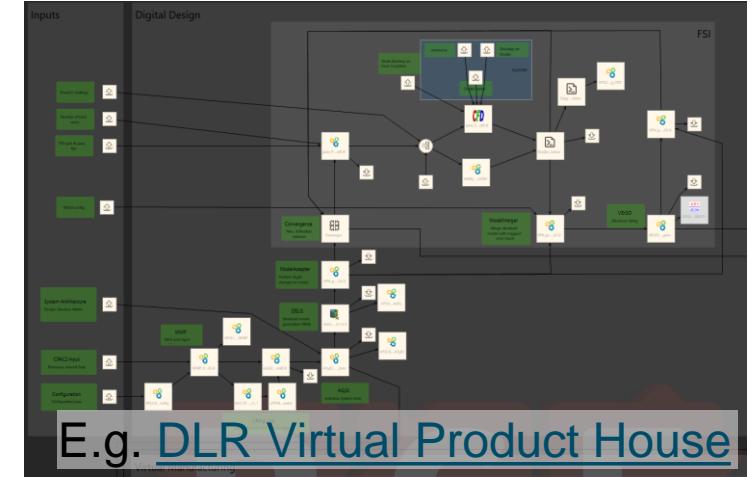
Task

Example: Digital design process aircraft moveable



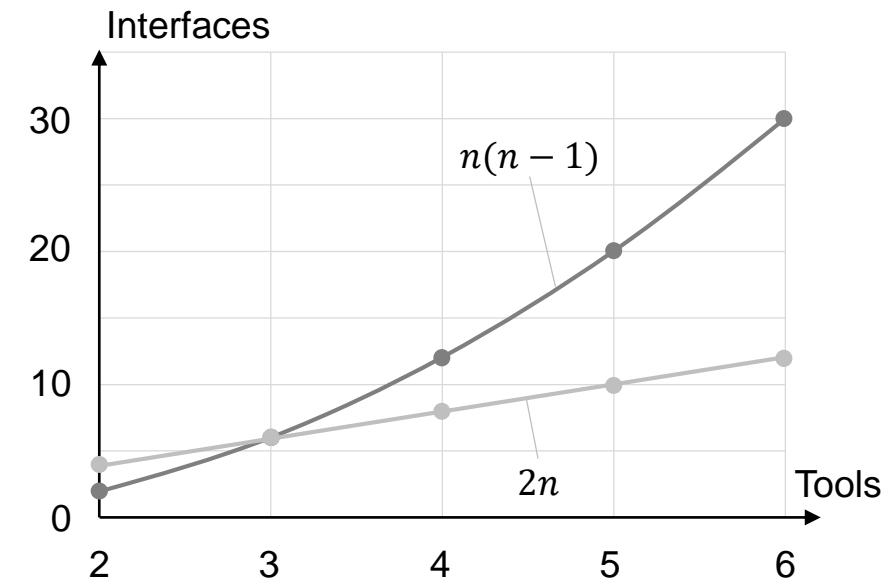
- Each tile
 - Assessment capability → Process
 - Requires input, delivers output → I & O
- Basically same information, vendor-specific formats

IPO

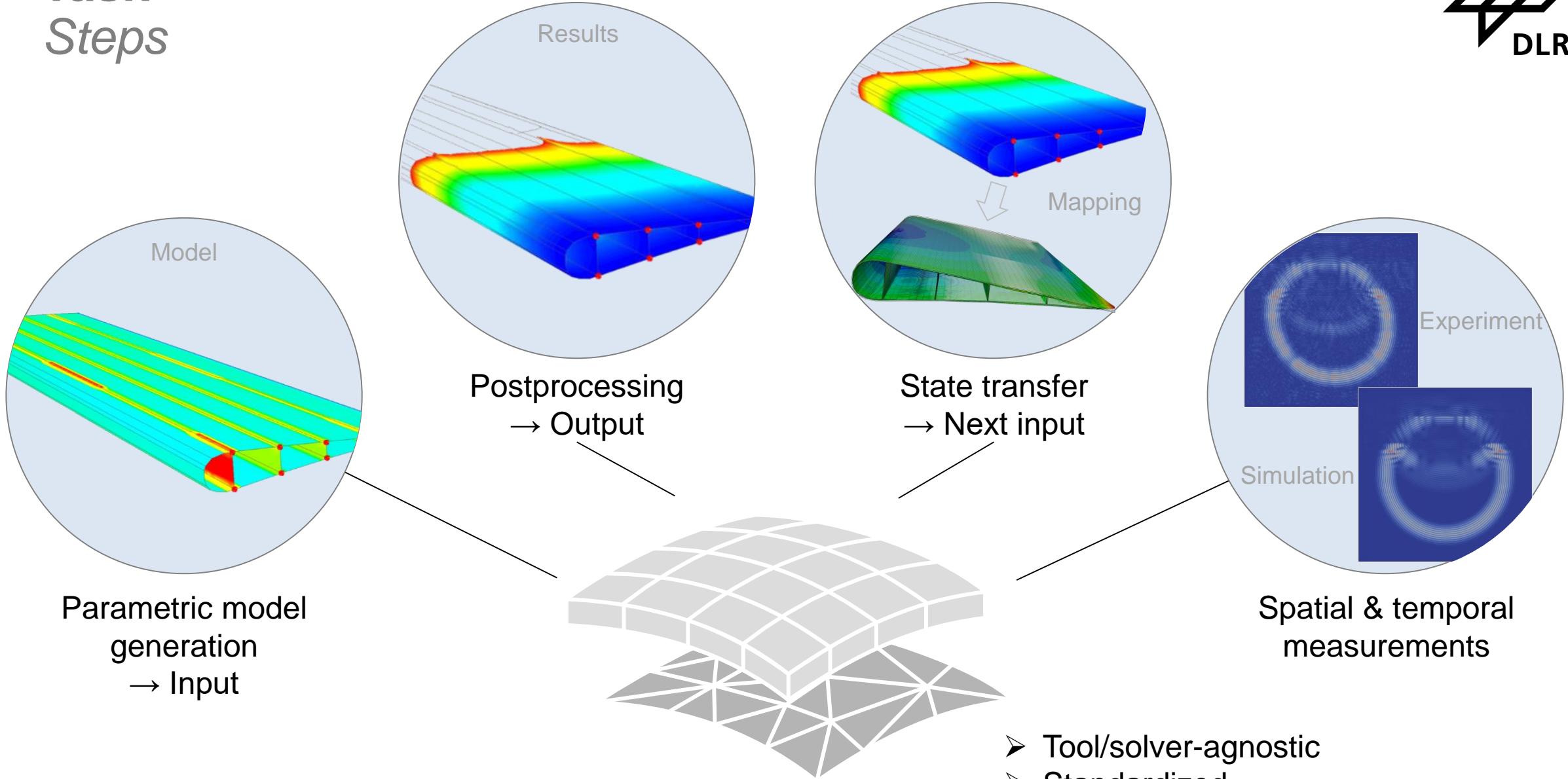


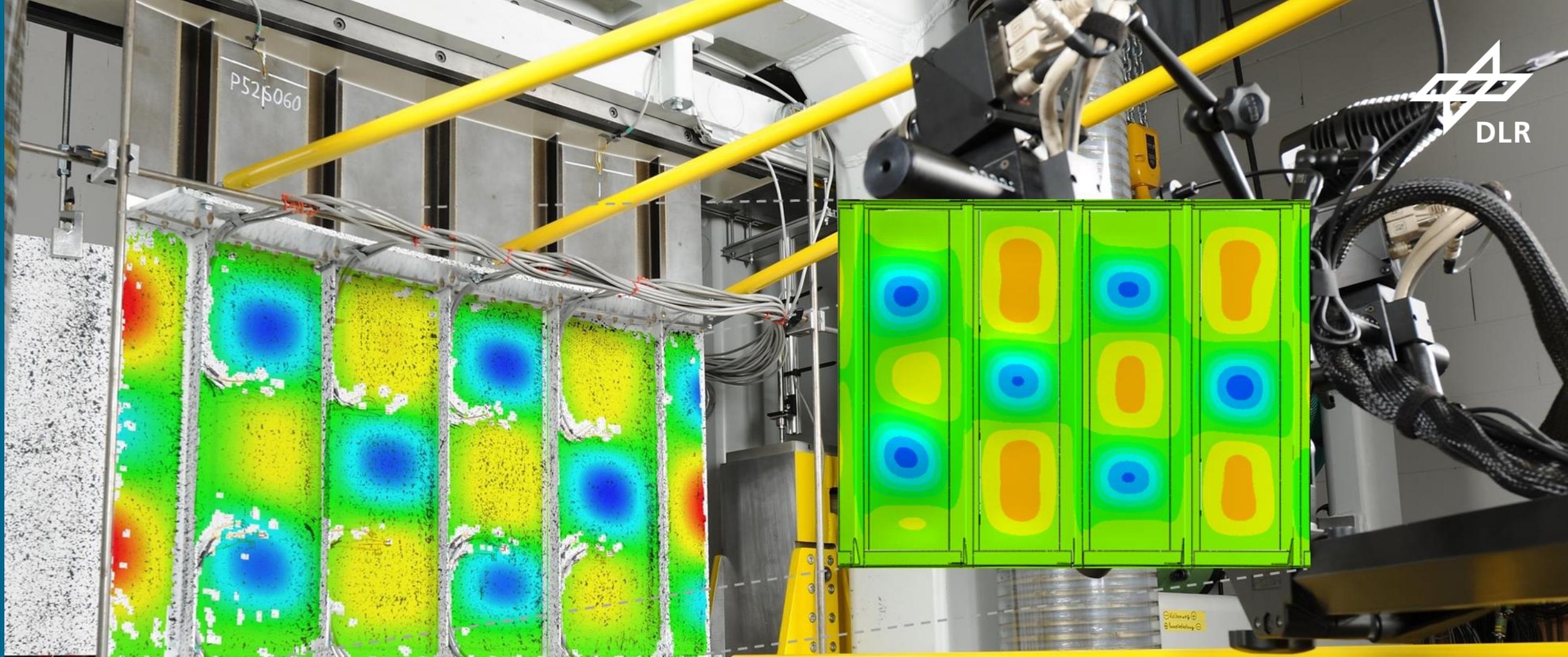
— Individual data standards

— Common data standard



Task Steps

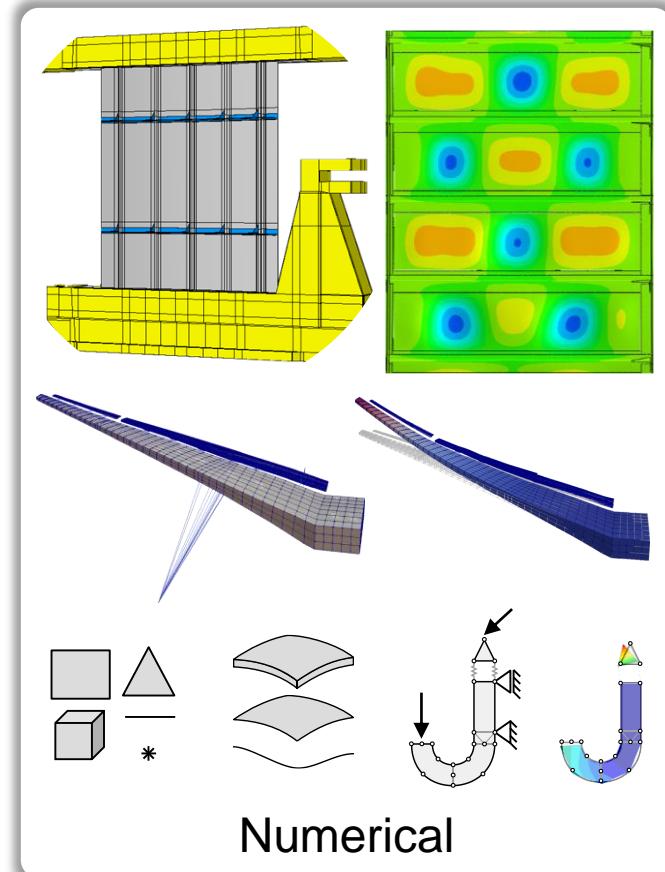
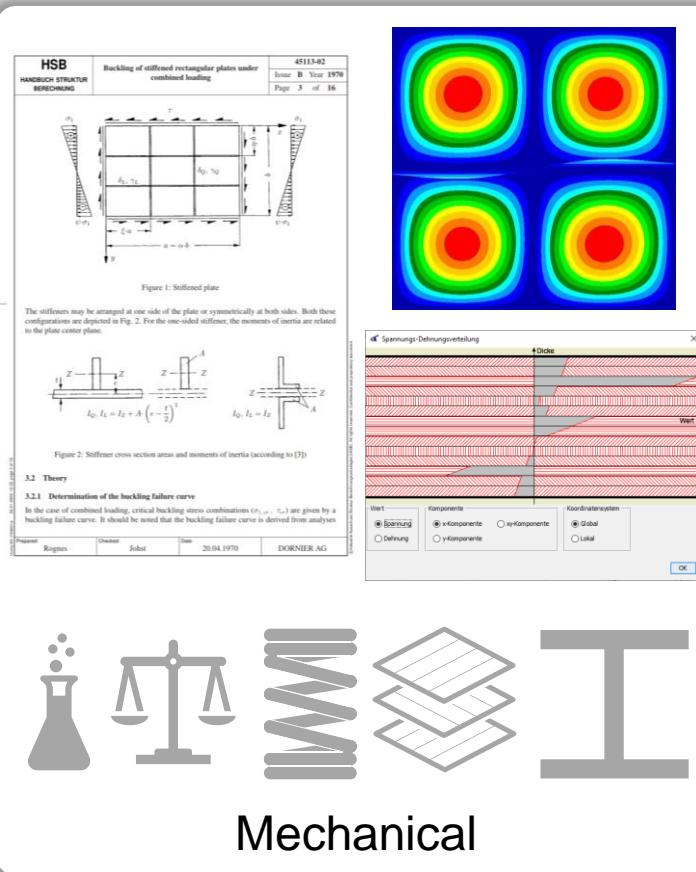
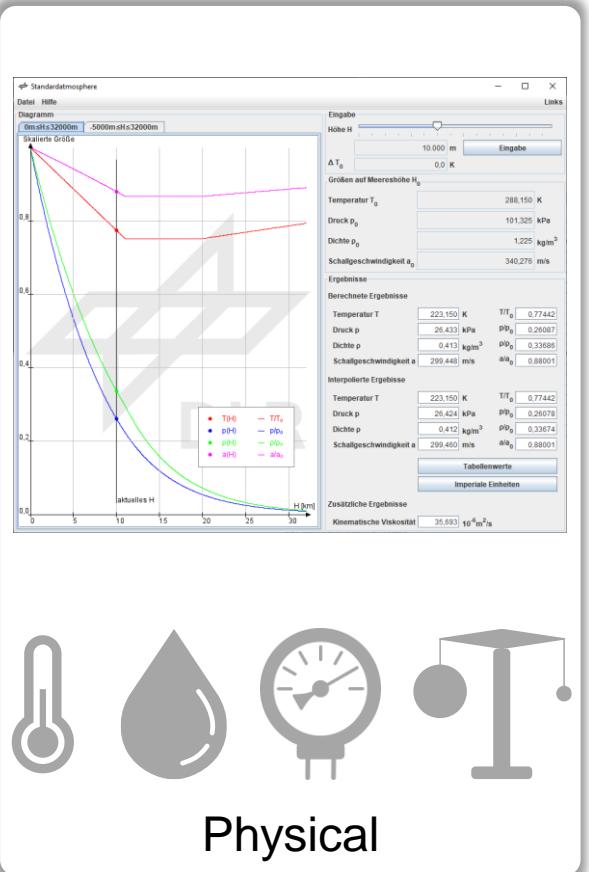
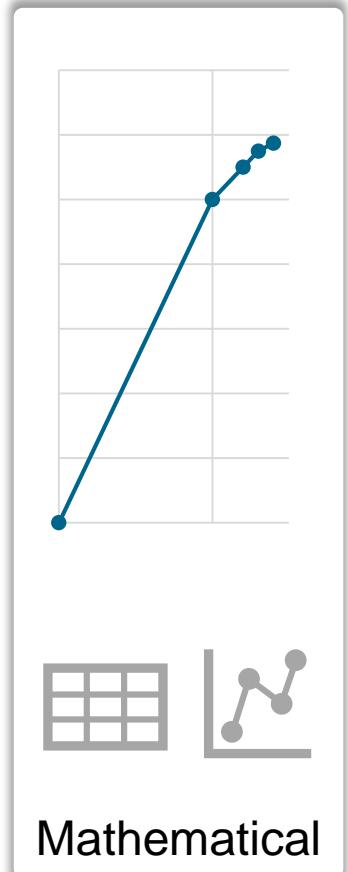




APPROACH

Approach Hypothesis

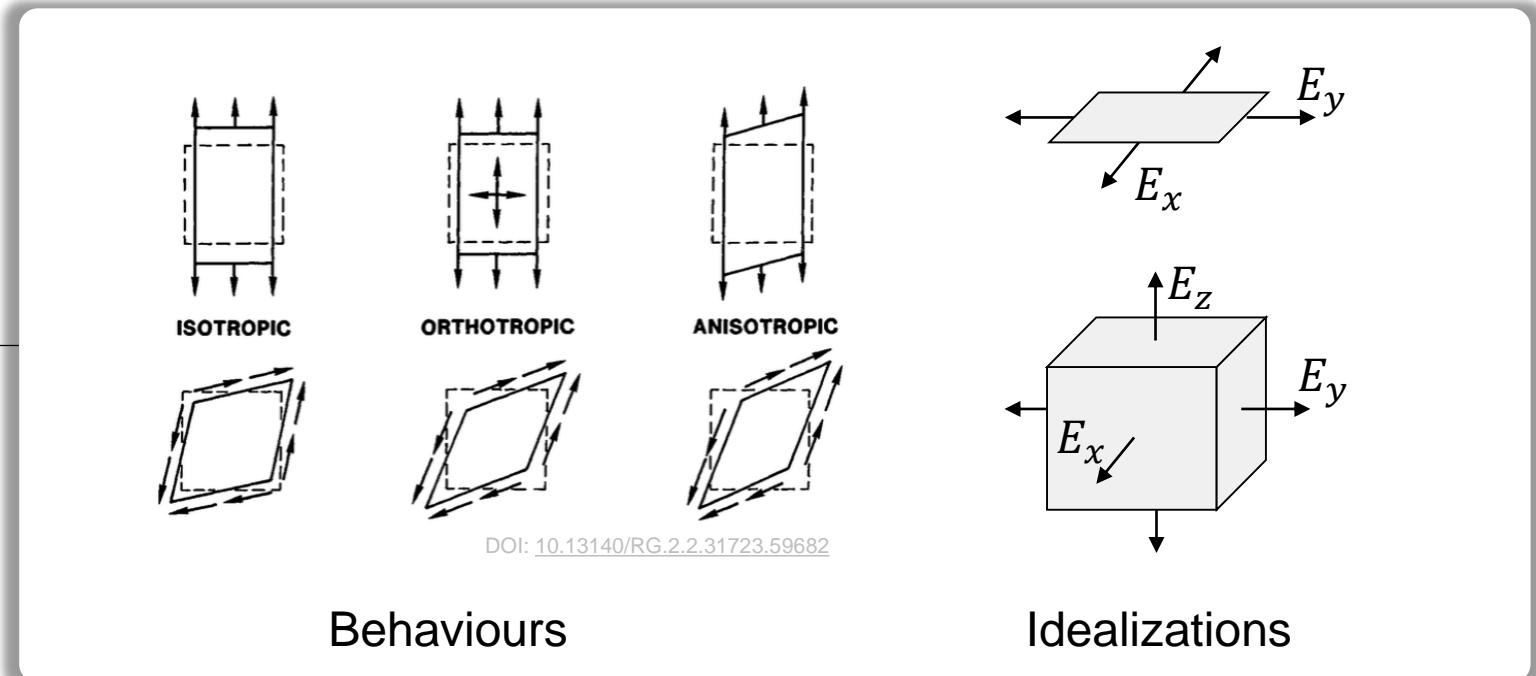
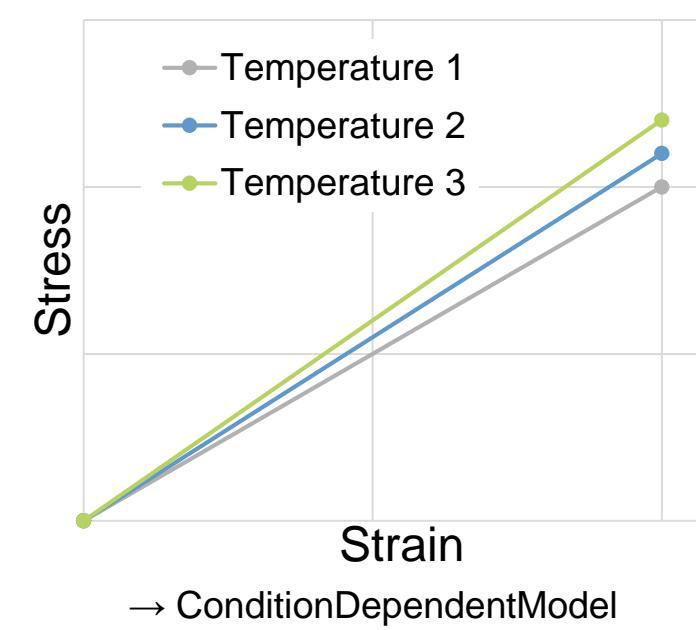
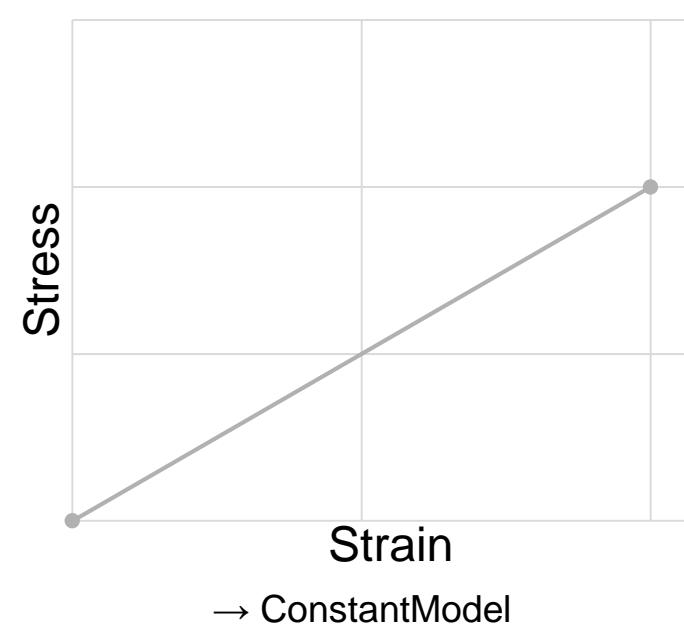
- Subdivision of information that makes up a model possible → based on level of assumptions & theories → object oriented approach



Approach

Base entity

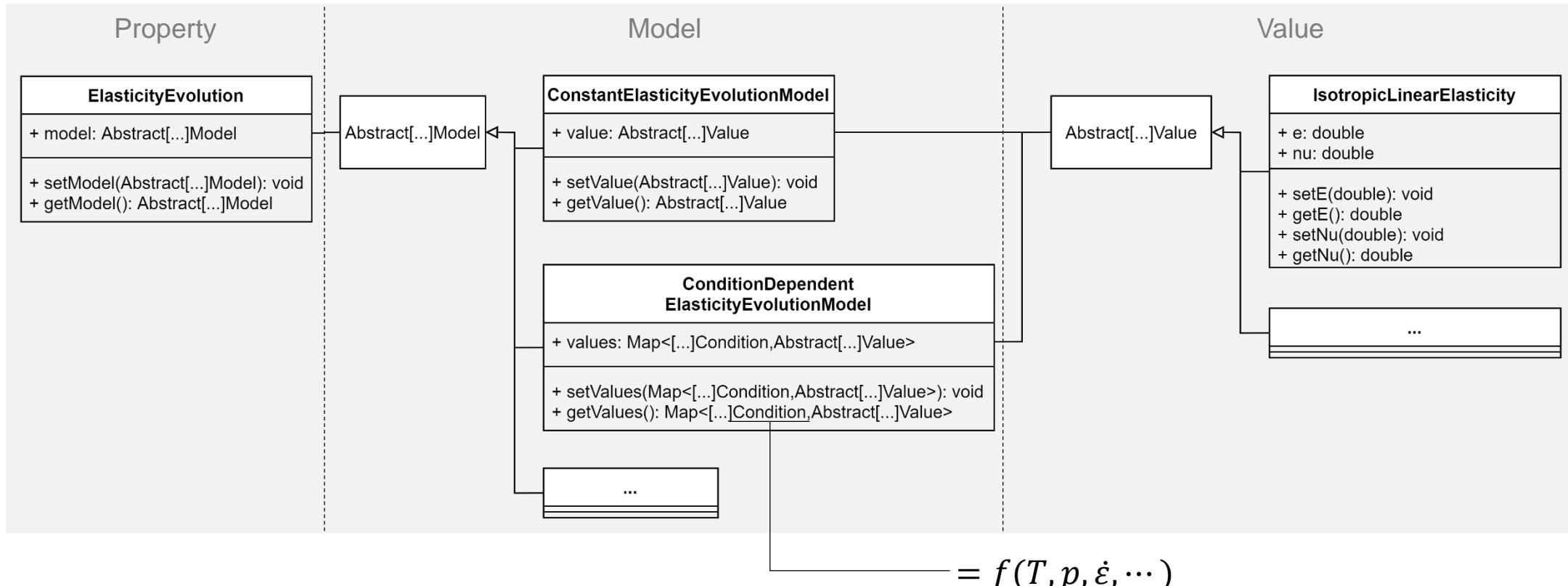
- Description:
 - Entity
 - Model
 - Value
- Example
 - ElasticityEvolution
 - Subtypes not complete
- Schema also applicable to nonlinear or non-cauchy value types



Approach

Base entity

■ Property-Model-Value example



Approach

Compound entity

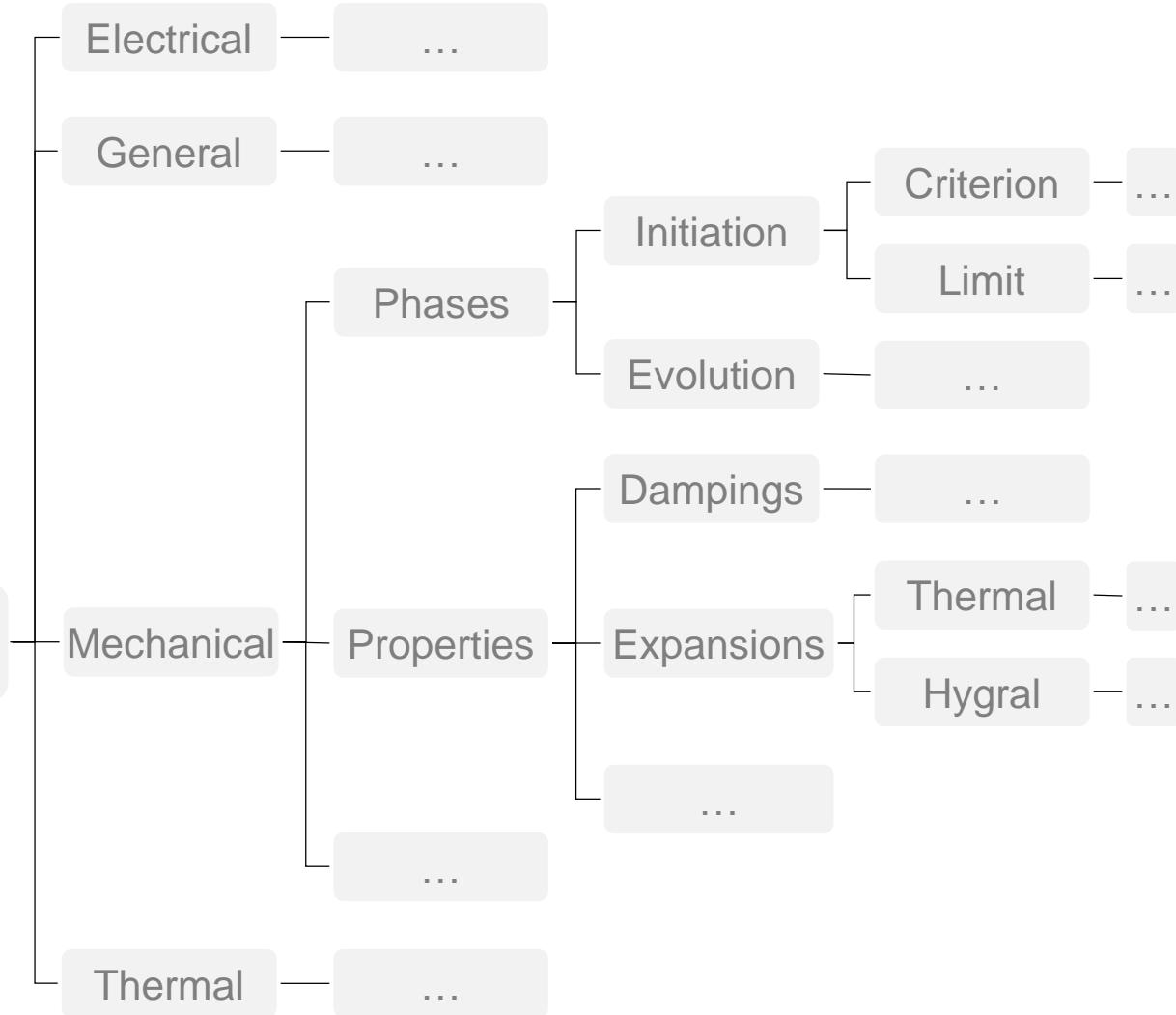
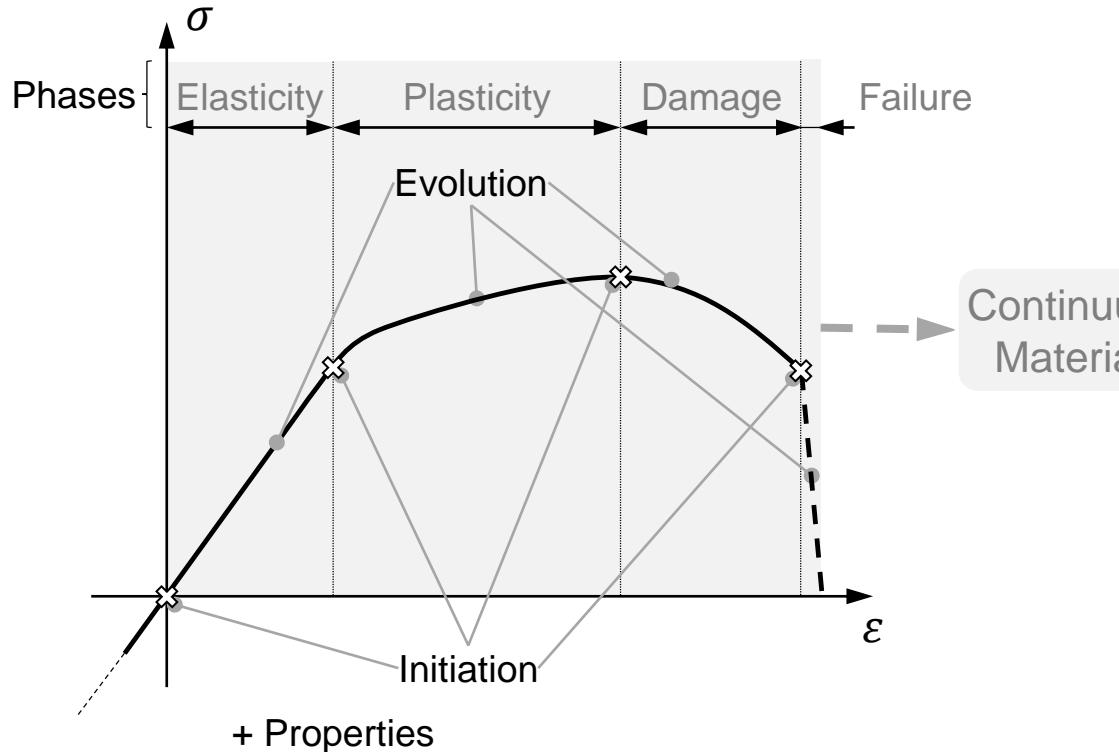


- Collect base entities
- Do not have models, just values, because models are contained in attributes
- Example: Material
 - Current value implementations:
 - CohesiveMaterial
 - ContinuumMaterial
 - DatabaseMaterial

Approach

Hierarchy development

- Search of model components
- Goal: expandable, modular
- Example: ContinuumMaterial



Projects jMeS × Files Services ...java MetaInfoTest.java × R2Property.java × AbstractMechObject.java × AbstractHyperElasticityEvolutionValue.java × ElasticityEvolution.java

```

32     @EqualsAndHashCode(callSuper=true)
33
34     // JAXB
35     @XmlRootElement
36     @XmlSeeAlso({
37         ConditionDependentElasticityEvolutionModel.class
38         ,ConstantElasticityEvolutionModel.class
39     })
40
41     // Jackson
42     @JsonSubTypes({
43         @JsonSubTypes.Type(value = ConditionDependentElasticityEvolutionModel.class)
44         ,@JsonSubTypes.Type(value = ConstantElasticityEvolutionModel.class)
45     })
46
47     // -----
48     // Class
49     // -----
50
51     public class ElasticityEvolution<
52         M extends AbstractElasticityEvolutionModel
53     >
54         extends
55             AbstractMechanicalPhaseEvolution<
56                 ElasticityEvolution<M>
57                 ,M
58             >
59
60             /**
61             * Create a new empty instance
62             */
63             public ElasticityEvolution() {}
64
65             /**
66             * Create a new instance
67             * @param model {@link M}
68             */
69             public ElasticityEvolution(
70                 M model
71             ...
72

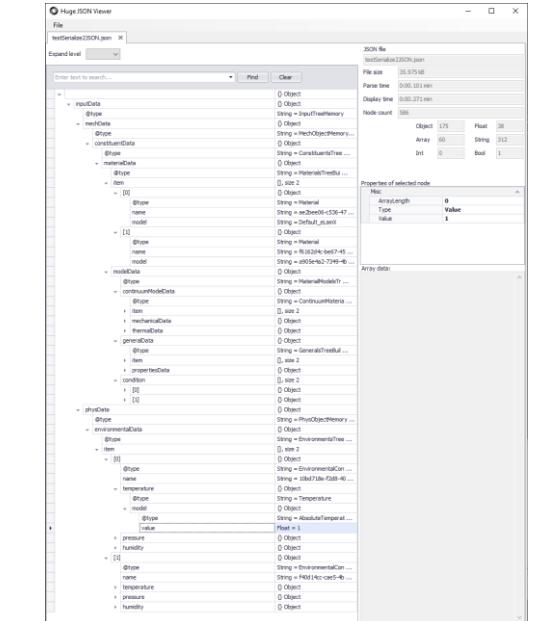
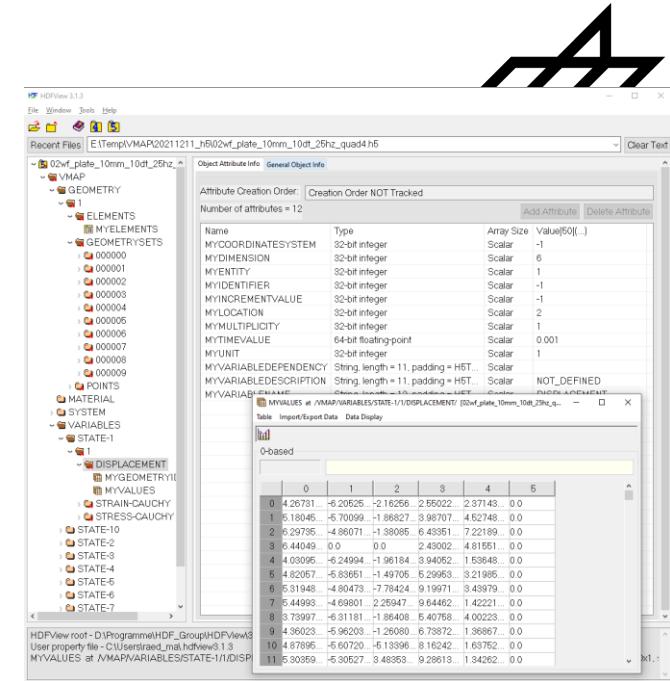
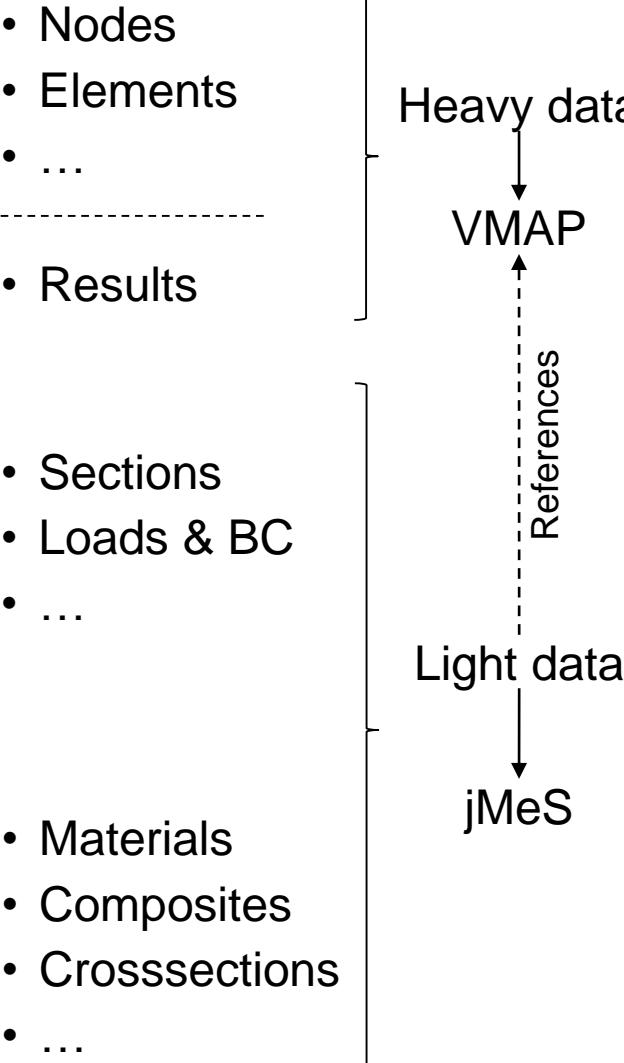
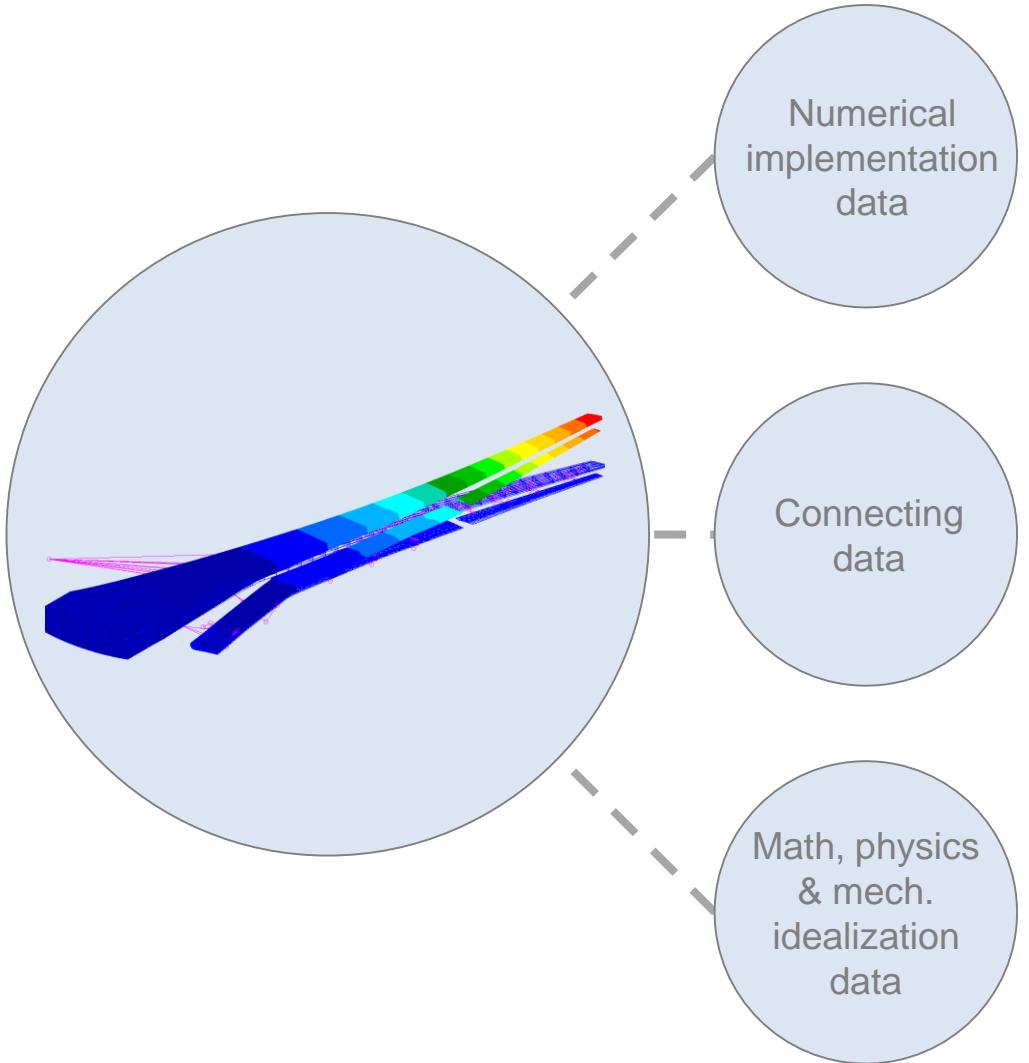
```



IMPLEMENTATION

Implementation

Data types



Implementation Goals



- Solver-agnostic CSM model & result data format
- Differentiation between
 - Numerical discretization-dependent information → „Heavy“ data
 - Model information → „Light“ data
- Heavy data:
 - Usage existing format
 - Preferably based on well-known format, supported by libraries
 - Visualization
- Light data:
 - Programming language independent
 - Machine readable
 - Human readable
 - Single-source of truth

}

Serialization format → JSON, XML, YAML, ...
→ REST ready

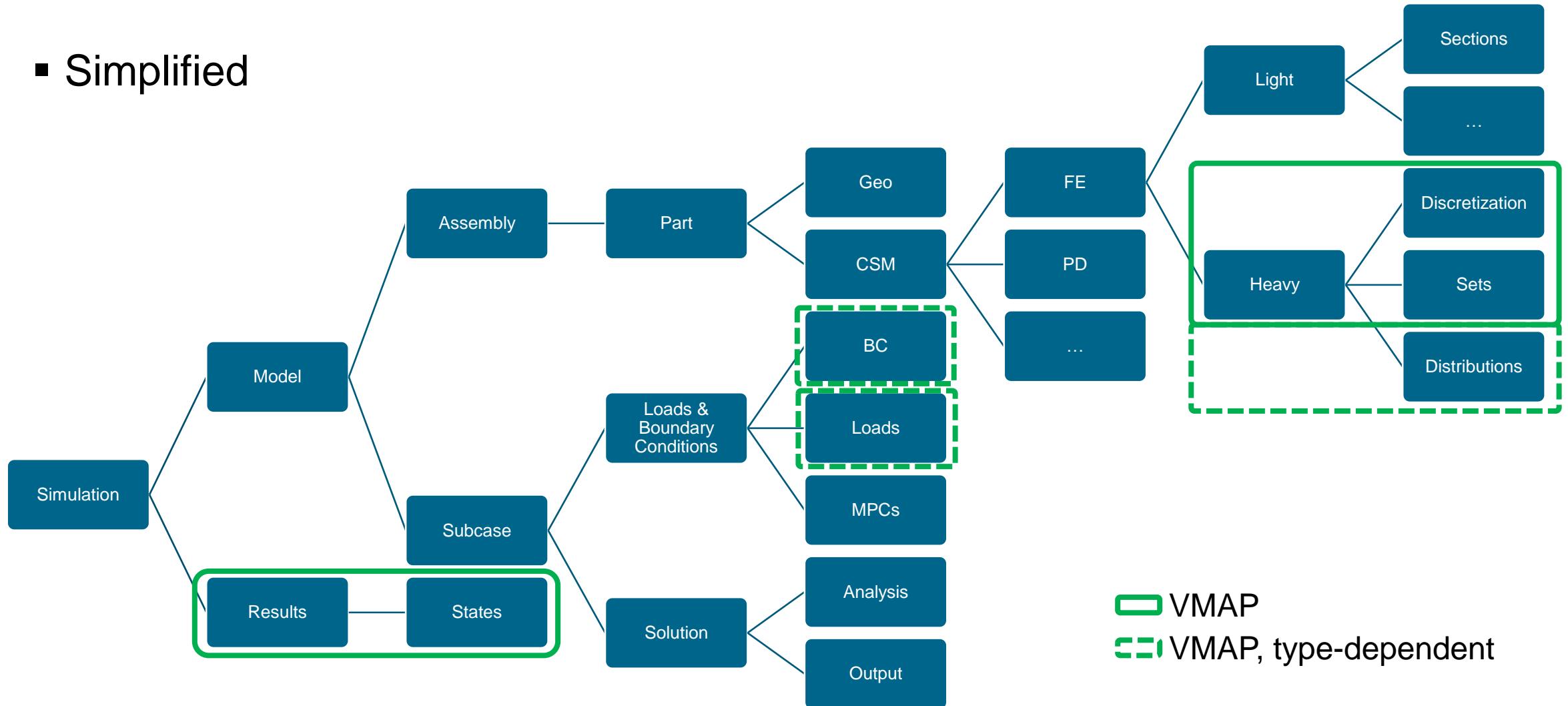
- FEM:
 - Exodus
 - MOAB
 - Silo
 - VMAP
 - XDMF
- Particles:
 - H5Part
- CFD:
 - CGNS

Alternatives

Implementation Hierarchy



Simplified



Implementation Approach

- Hierarchical, object-oriented data structure
 - Implementation in Java-based backend jMeS (Java Mechanics Suite)
 - Usage of JAXB annotations across hierarchy to denote
 - Abstraction
 - Attribute types
- Annotations understood by de-/serialization libraries
 - E.g. Jackson, MOXy, ...
 - Allows creation of schema, e.g. XML
 - Creation of XML, JSON, YAML, ...

```
// JAXB
@XmlTransient
@XmlAccessorType(XmlAccessType.FIELD)

// Jackson
@JsonTypeInfo(use = JsonTypeInfo.Id.NAME, include = JsonTypeInfo.As.PROPERTY)

public abstract class AbstractMechObjectProperty<
    T extends AbstractMechObjectProperty<T,M>
    ,M extends AbstractMechObjectModel
    >

// JAXB
@XmlRootElement
@XmlSeeAlso({
    ConditionDependentElasticityEvolutionModel.class
    ,ConstantElasticityEvolutionModel.class
})

// Jackson
@JsonSubTypes ({
    @JsonSubTypes.Type(value = ConditionDependentElasticityEvolutionModel.class)
    ,@JsonSubTypes.Type(value = ConstantElasticityEvolutionModel.class)
})

public class ElasticityEvolution<

public abstract class AbstractMechObjectProperty<
    T extends AbstractMechObjectProperty<T,M>
    ,M extends AbstractMechObjectModel
    >
    @XmlAttribute
    @XmlID
    private String name = DEFAULT_NAME;
    @XmlIDREF
    protected M model;

    protected MetaInfo<? extends AbstractMetaInfoValue> metaInfo;
```

Abstract superclass

Implementation

Attributes

The screenshot shows a JSON viewer application interface. On the left, a tree view displays the structure of a JSON file named `testSerialize2JSON.json`. The root node is `model`, which contains `@type`, `name`, and `assemblyData`. `assemblyData` contains `@type`, `name`, and `part`. `part` is an array with three items, indexed [0], [1], and [2]. Each item has `@type`, `name`, and `geometry`. `geometry` contains `@type`, `r0Data`, and `item`. `r0Data` contains `@type`. `item` is an array with three items, indexed [0], [1], and [2]. Each item has `@type`, `name`, and `topology`. `topology` contains `@type`, `coordinate1`, `coordinate2`, `coordinate3`, and `preferences`. The right side of the interface provides detailed statistics for the file: File size (83.737 kB), Parse time (0:00.121 min), Display time (0:00.286 min), and Node count (1510). It also shows a breakdown of data types: Object (381), Array (130), String (751), Int (153), and Bool (37). Below these statistics, a table titled "Properties of selected node" shows the properties for the selected node, which is an array with `ArrayLength` 0, `Type` Object, and `Value` empty. A section titled "Array data:" is also present.

EXAMPLES

EXAMPLES

Examples



■ Implementation

The screenshot displays two windows illustrating a linked data structure:

- HDFView 3.1.3** window (top): Shows a hierarchical tree view of an HDF5 file named "discretization.vmap.h5". The tree includes nodes for VMAP, GEOMETRY, ELEMENTS, and POINTS. A sub-table for "MYELEMENTS" is shown, containing columns: myIdentifier, myElementType, myCoordinateSystem, myMaterialType, mySectionType, and myOrder. Data rows are present for identifiers 0, 1, and 2.
- Huge JSON Viewer** window (bottom): Shows a JSON tree for "model.jmes.json". The "model" object contains a "discretization" object with "type" (String = VMAP) and "path" (String = ..\discretization.vmap.h5). It also contains "modelData", "mechData", "physData", and "mathData" objects.

Annotations in the bottom-left corner point to specific parts of the JSON tree:

- Link to heavy data file** points to the "path" field under "discretization".
- Light data** points to the "modelData", "mechData", "physData", and "mathData" fields under "model".

■ Demonstration

Examples

■ Implementation

Geometry

CSM

Huge JSON Viewer

File

testSerialize2JSON2.json

Expand level

Enter text to search... Find Clear

model

- @type Object
- Object
- String = ModelTreeBuilder ...

assemblyData

- @type Object
- String = TreeBuilderAssem ...

name

- String = c7882879-97e2-49 ...

part

- [0]

- @type TreeBuilderPart
- String = b02093cc-f5bb-4a ...

- name

 - String = b02093cc-f5bb-4a ...

- geo

 - @type Object
 - String = GeoMemoryTreeBui ...

- r0Data

 - Object
 - Object

- r1Data

 - @type Object
 - String = R1GeometriesTree ...

- item

 - [0]
 - [1]
 - [2]
 - [3]
 - [4]
 - [5]
 - [6]
 - [7]
 - [8]
 - [9]
 - [10]
 - [11]

- preferencesData
- biasData
- r2Data
- r3Data
- csm

- @type Object
- String = CSMMemoryTreeBui ...

- feData

 - Object
 - String = FEDataMemoryTree ...

- lightData

 - Object
 - String = FELightDataMemor ...

- sectionData

 - Object
 - String = FESectionsTreeBu ...

- r0Data
- r1Data
- r2Data
- r3Data
- enrichmentData
- discretizationPreferences
- discretizationWriterPreferences
- discretizationPreferences
- discretizationWriterPreferences
- subcaseData
- discretizationPreferencesData
- discretizationWriterPreferencesData

JSON file
testSerialize2JSON2.json
File size 57.532 kB
Parse time 0:00.097 min
Display time 0:00.229 min
Node count 1073

Object 267 Float 49
Array 91 String 626
Int 9 Bool 31

Properties of selected node

Misc

- ArrayLength 0
- Type Value
- Value ModelTreeBuilderMemory

Array data:



Examples

■ Implementation

Load & boundary condition (LBC) state

Subcases

Solution

The screenshot shows the Huge JSON Viewer application interface. The main window displays a hierarchical JSON tree. A specific section of the tree is highlighted with a dashed blue rectangle, corresponding to the 'Subcases' and 'Solution' annotations. The highlighted section starts at the 'item' array under 'subcaseData' and includes several nested objects like 'LbcStateData', 'bcData', 'setData', 'loadData', 'mpcData', 'solutionData', 'analysisData', 'valueData', and 'outputData'. An annotation points to the 'LbcStateData' object. The 'File' menu is open, showing options like 'File', 'Expand level', 'Enter text to search...', 'Find', and 'Clear'. On the right side, there's a summary panel with statistics: File size 57.532 kB, Parse time 0:00.097 min, Display time 0:00.229 min, Node count 1073, and a breakdown of data types: Object 267, Float 49, Array 91, String 626, Int 9, Bool 31. Below the summary is a 'Properties of selected node' table and an 'Array data' section.



Examples

Machine readable formats



JSON

Huge JSON Viewer
File
model.jmes.json
Expand level
Enter text to search... Find Clear
JSON file
model.jmes.json
File size: 60,956 kB
Parse time: 0:00.109 min
Display time: 0:00.268 min
Node count: 1117
Object: 293 Float: 102
Array: 93 String: 616
Int: 2 Bool: 10
Properties of selected node
Msc
ArrayLength: 0
Type: Value
Value: JMeSConverterModel
Array data:
model
@type
discretization
type
path
assemblyData
@type
name
parts
subcaseData
@type
item
lbcStateData
solutionData
discretizationPreferencesData
discretizationWriterPreferencesData
mechData
@type
constituentData
@type
materialData
springData
crosssectionData
offsetData
@type
r1Data
propertyData
@type
r1Data
r2Data
physData
@type
baseData
@type
densityData

XML

XML Notepad - E:\Temp\jMeS\CompleteModelStorage\testR2LogoXML\model.jmes.xml
File Edit View Insert Window Help
E:\Temp\jMeS\CompleteModelStorage\testR2LogoXML\model.jmes.xml
Tree View XSL Output
model
type
discretization
path
assemblyData
modelData
subcaseData
mechData
physData
Error List Dynamic Help
Description File Line Column

```
!<JMeSConverterModel>
  discretization:
    type: "VMAP"
    path: "..\discretization.vmap.h5"
  modelData: !<ModelTreeBuilderMemory>
    &80a23da6-3cf5-4f14-b77c-bb6207f39ca6 name: "80a23da6-3cf5-4f14-b77c-bb6207f39ca6"
  assemblyData: !<TreeBuilderAssembly>
    &d5ac25fb-0279-4f97-9a0a-204cf3bcc080 name: "d5ac25fb-0279-4f97-9a0a-204cf3bcc080"
  part:
    - !<TreeBuilderPart>
      discretizationPreferences: "355baec0-7fd0-450e-a79f-dda39e9afe6b"
      discretizationWriterPreferences: "3c15c20f-68e6-4832-84be-368a5caf4c7"
  subcaseData: !<SubcasesTreeBuilderMemory>
  discretizationPreferencesData: !<DiscretizationPreferencesTreeBuilderMemory>
  discretizationWriterPreferencesData: !<DiscretizationWriterPreferencesTreeBuilderMemory>
  mechData: !<MechObjectMemoryTreeBuilderMemory>
    constituentData: !<ConstituentsTreeBuilderMemory>
      materialData: !<MaterialsTreeBuilderMemory>
        springData: !<SpringsTreeBuilderMemory>
        crosssectionData: !<CrosssectionsTreeBuilderMemory>
        offsetData: !<OffsetsTreeBuilderMemory>
        r1Data: !<R1OffsetTreeBuilderMemory>
        propertyData: !<PropertiesTreeBuilderMemory>
        r1Data: !<R1PropertiesTreeBuilderMemory>
        r2Data: !<R2PropertiesTreeBuilderMemory>
        physData: !<PhysObjectTreeBuilderMemory>
        baseData: !<PhysBaseObjectTreeBuilderMemory>
        densityData: !<DensitiesTreeBuilderMemory>
```

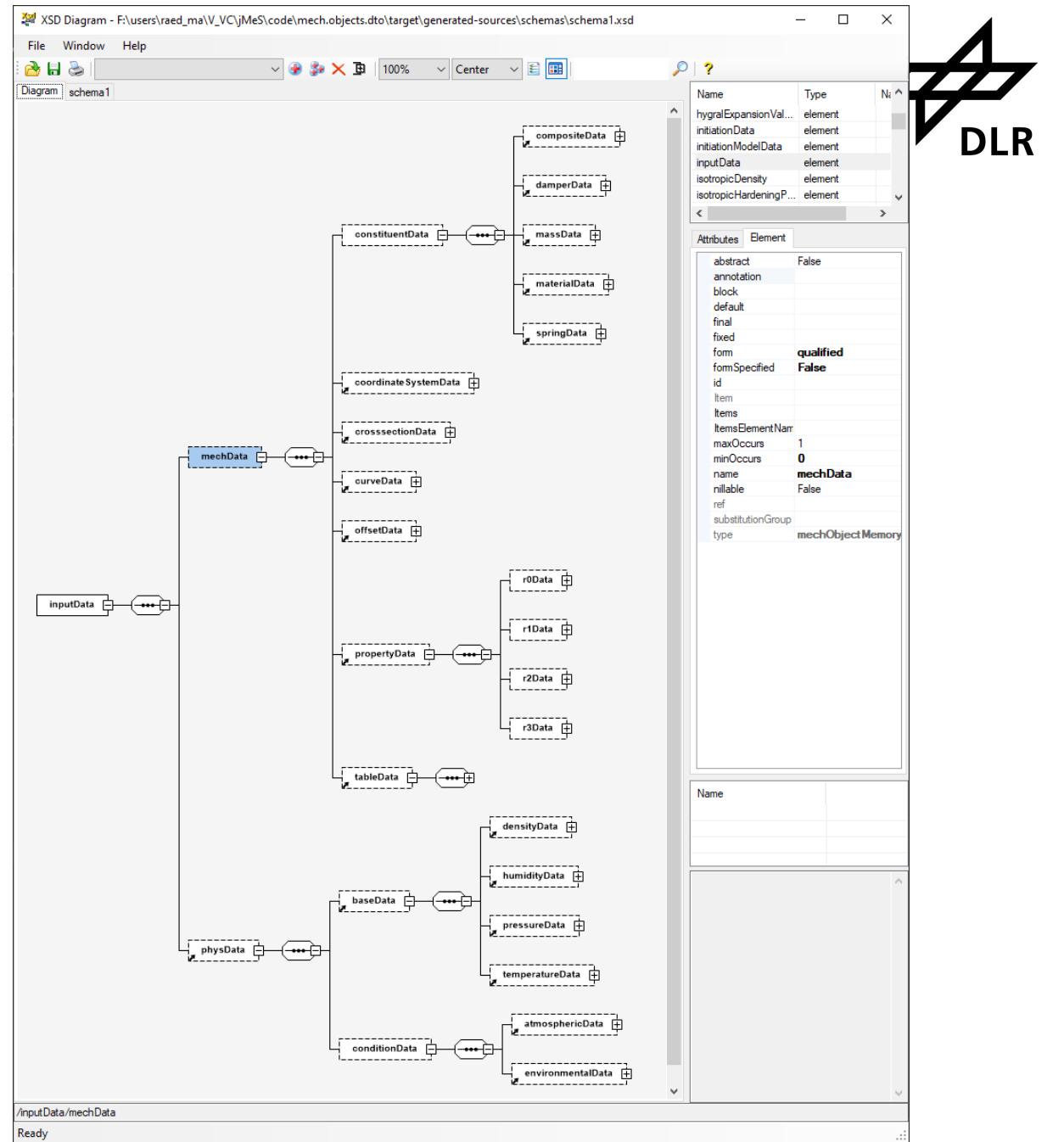
YAML

- Fully automatic generation by selection of serialization goal

Examples

Machine readable formats

- Automatic XSD schema generation during code compilation
- Automatic translation to code:
 - C++: codesynthesis
 - Python: generateDS
 - Java: Jackson
 - ...



Examples

Meta information



- E.g. reference to ID's from number-based formats (e.g. Nastran, LS-Dyna) for internal String UUID approach
- Currently:
 - Description
 - Reference
 - Source
- Modularly expandable
- Applicable to most DTO's

Huge JSON Viewer

File testSerialize2JSON3.json

Expand level

Enter text to search... Find Clear

{} Object	{} Object
{} Object	String = R2Property
{} Object	String = 18287231-9c5e-40...
{} Object	String = MetaInfo
{} Object	String = Description
{} Object	String = SimpleDescrip...
{} Object	String = This was NASTRAN...
{} Object	String = SimpleMetaInfo
{} Object	String = Reference
{} Object	String = IntegerReference
{} Object	Integer = 232
{} Object	String = eb755f59-15cb-44...

The screenshot shows a JSON viewer interface with a hierarchical tree view on the left and a detailed table view on the right. The tree view shows nested objects like R2Property, metaInfo, and value. The table view lists properties and their corresponding values, such as string representations of objects and integer values.

Examples

Solver-dependent information

- Will never achieve complete solver-independence
- Solver-independent information addable
- Example: ElementTypeManagers
- Fully expandable w.r.t.
 - Solution method preferences
 - ...

Huge JSON Viewer

File

testSerialize2JSON.json

Expand level

Enter text to search... Find Clear

Properties of selected node

Misc

- ArrayLength 0
- Type Value
- Value DiscretizationWriterPref...

Array data:

JSON file

testSerialize2JSON.json

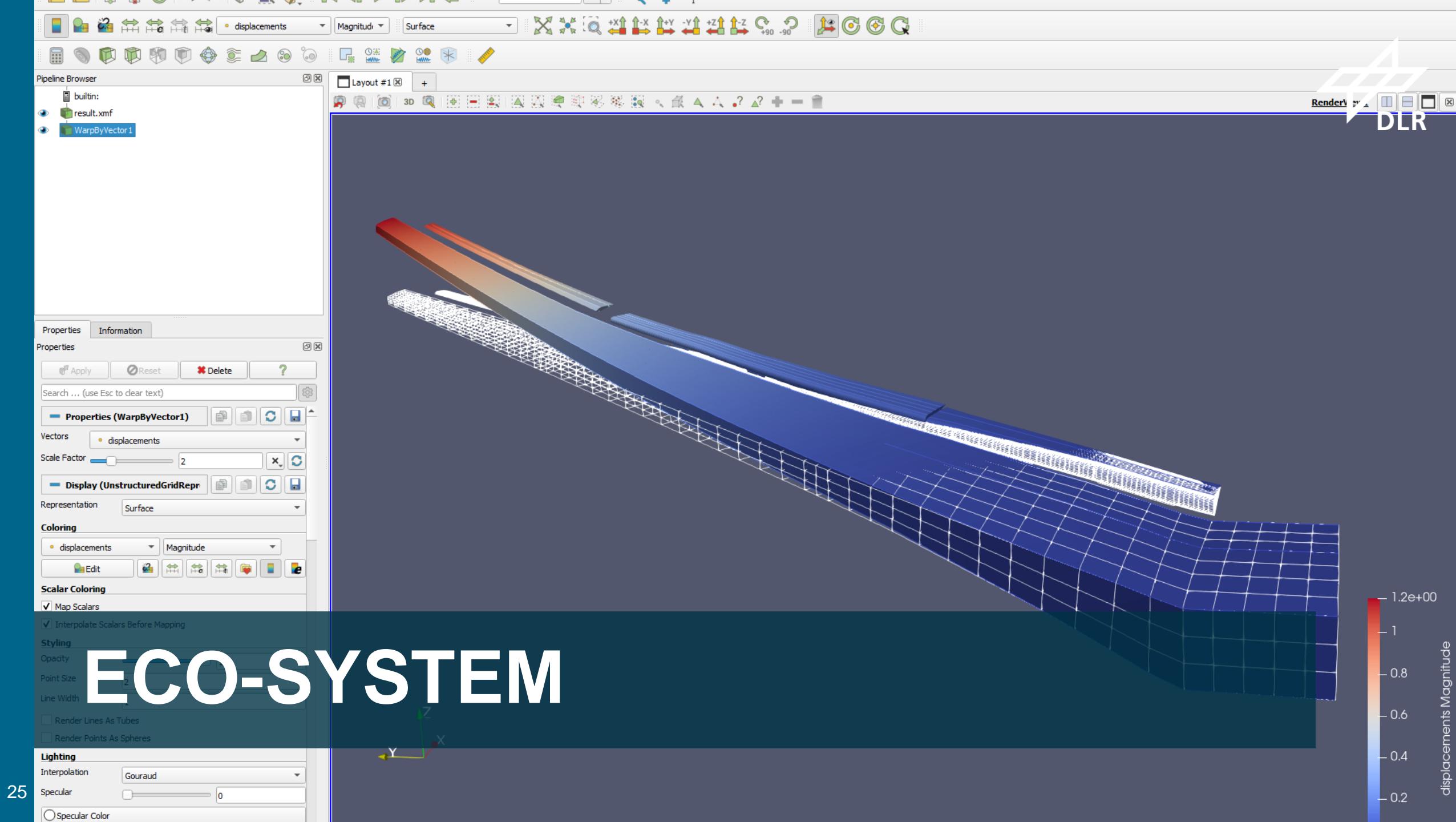
File size 6.715 kB

Parse time 0:00.095 min

Display time 0:00.259 min

Node count 161

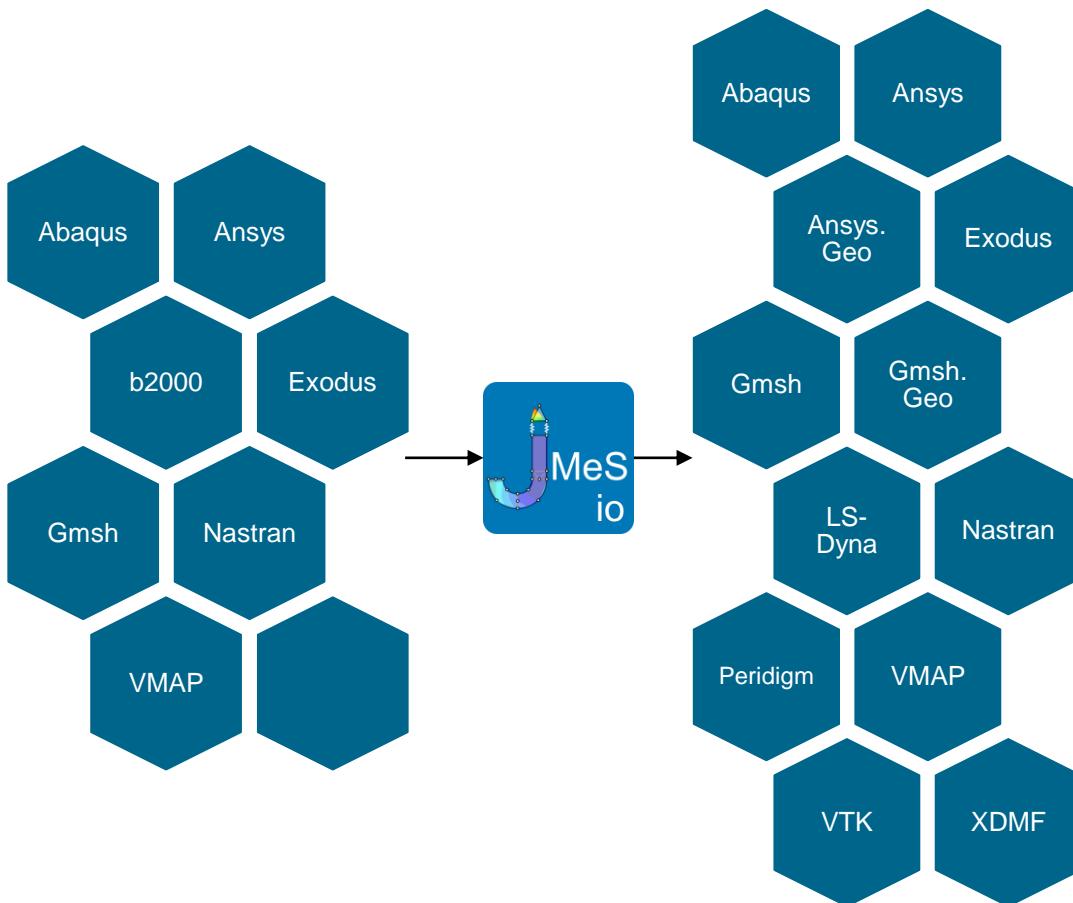
Object	25	Float	0
Array	0	String	135
Int	1	Bool	0



Eco-system *IO plugins*



- Conversion from & to solver-specific formats using io-plugins



- Usage from python with Py4J interface

- Software engineering:

- Testing

- Documentation

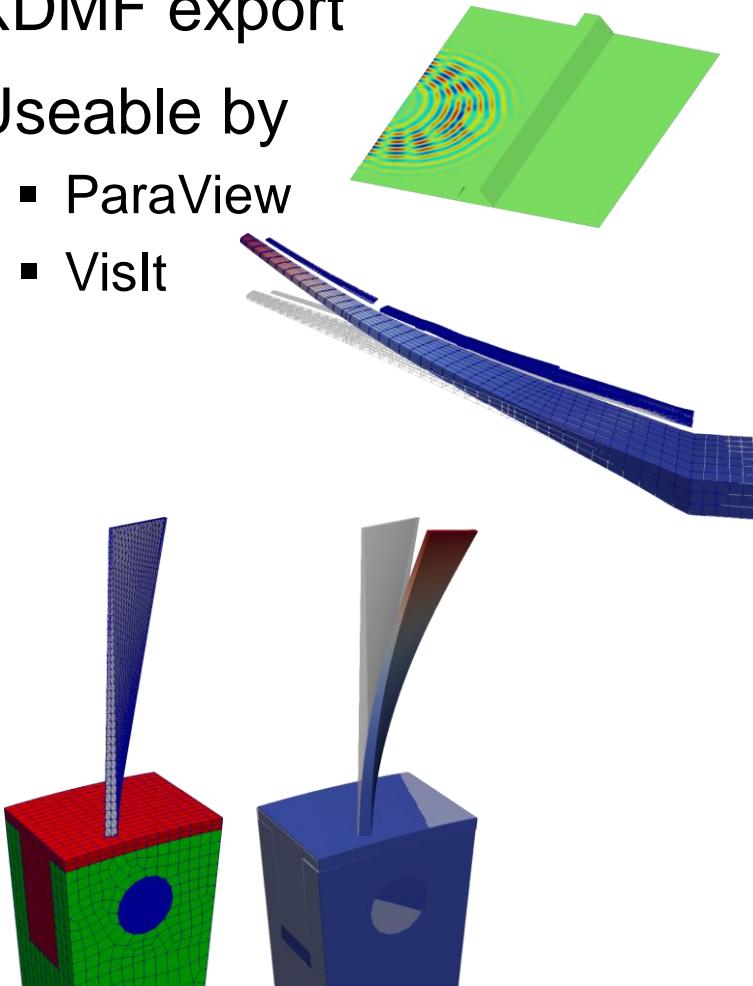
- CI/CD

		jmes.mech.numerx.func.io
		jmes.mech.numerx.func.io.in.core
		jmes.mech.numerx.func.io.in.impl.abaqus
		jmes.mech.numerx.func.io.in.impl.ansys
		jmes.mech.numerx.func.io.in.impl.b2000
		jmes.mech.numerx.func.io.in.impl.exodus
		jmes.mech.numerx.func.io.in.impl.gmsh
		jmes.mech.numerx.func.io.in.impl.nastran
		jmes.mech.numerx.func.io.in.impl.vmap
		jmes.mech.numerx.func.io.in.pom

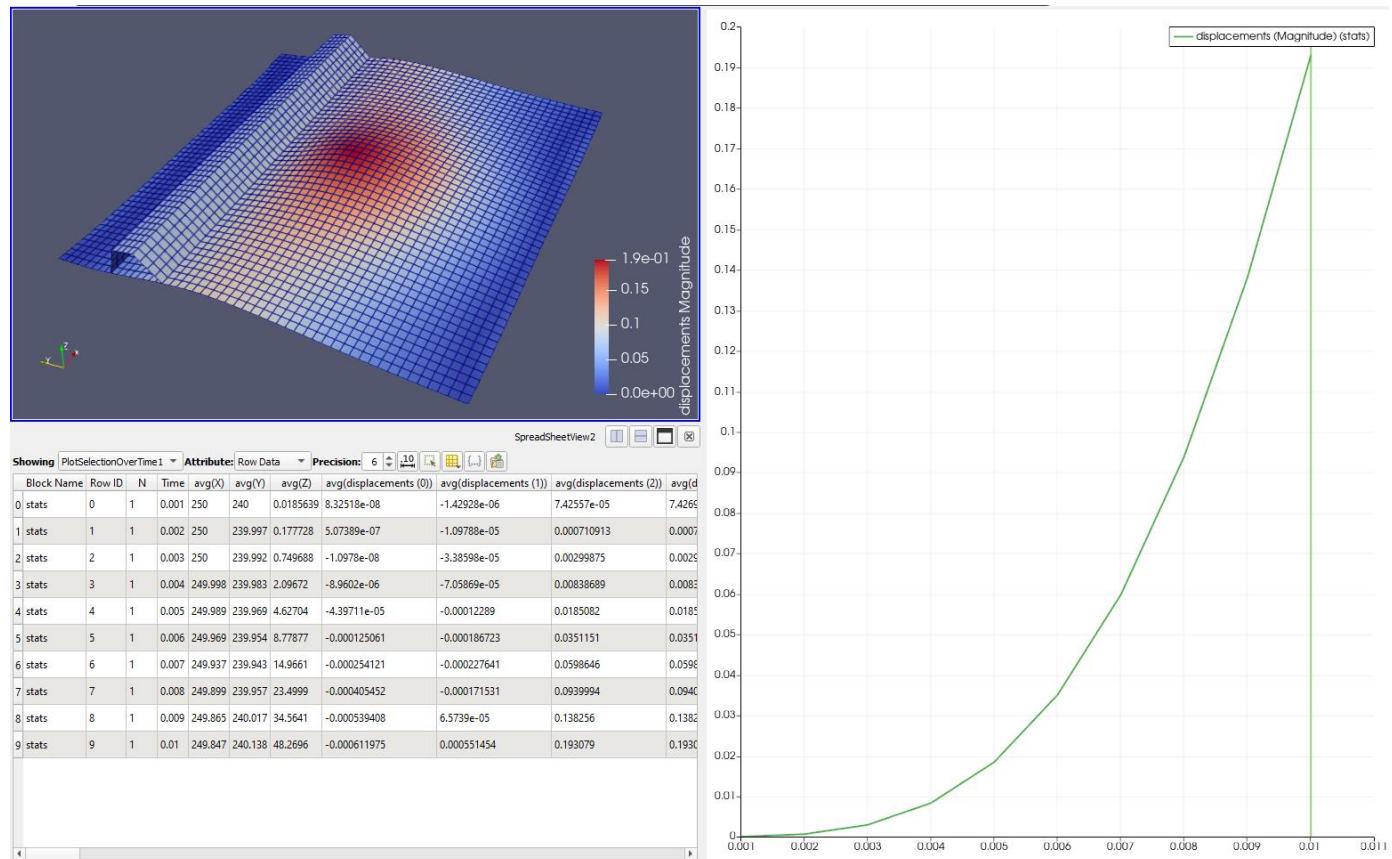
Eco-system Visualization



- XDMF export
- Useable by
 - ParaView
 - VisIt



- Automated postprocessing enabled,
e.g. by pvpthon



Block Name	Row ID	N	Time	avg(X)	avg(Y)	avg(Z)	avg(displacements (0))	avg(displacements (1))	avg(displacements (2))	avg(d
0 stats	0	1	0.001	250	240	0.0185639	8.32518e-08	-1.42928e-06	7.42557e-05	7.4265
1 stats	1	1	0.002	250	239.997	0.177728	5.07389e-07	-1.09788e-05	0.00010913	0.0007
2 stats	2	1	0.003	250	239.992	0.749688	-1.0978e-08	-3.38598e-05	0.00299875	0.0025
3 stats	3	1	0.004	249.998	239.983	2.09672	-8.9602e-06	-7.05869e-05	0.00838689	0.0083
4 stats	4	1	0.005	249.989	239.969	4.62704	-4.39711e-05	-0.00012289	0.0185082	0.0185
5 stats	5	1	0.006	249.969	239.954	8.77877	-0.000125061	-0.000186723	0.0351151	0.0351
6 stats	6	1	0.007	249.937	239.943	14.9661	-0.000254121	-0.000227641	0.0598646	0.0598
7 stats	7	1	0.008	249.899	239.957	23.4999	-0.000405452	-0.000171531	0.0939994	0.0940
8 stats	8	1	0.009	249.865	240.017	34.5641	-0.000539408	6.5739e-05	0.138256	0.1382
9 stats	9	1	0.01	249.847	240.138	48.2696	-0.000611975	0.000551454	0.193079	0.1930

Expand level 10

Enter text to search...

Find

Clear

model

- @type
- name
- assemblyData

 - @type
 - name
 - part

 - [0]
 - @type
 - name
 - geometry

 - @type
 - r0Data

 - @type
 - item

 - [0]
 - @type
 - name
 - topology

 - @type
 - coordinate1
 - coordinate2
 - coordinate3
 - preferences

- [1]
- @type
- name
- topology

 - @type
 - coordinate1
 - coordinate2
 - coordinate3
 - preferences

- [2]
- @type
- name
- topology

 - @type
 - coordinate1
 - coordinate2
 - coordinate3
 - preferences

- [3]
- @type
- name
- topology

 - @type
 - coordinate1
 - coordinate2
 - coordinate3
 - preferences

JSON file

testSerialize2JSON.json

File size 83.737 kB

Parse time 0:00.121 min

Display time 0:00.286 min

Node count 1510

Object	381	Float	51
Array	130	String	751
Int	153	Bool	37

Properties of selected node

Misc	
ArrayLength	0
Type	Object
Value	

Array data:

CONCLUSION

Conclusion

Current



- Hierarchical information storage based on level of assumptions possible
 - Grind: E.g. mech.object.dto project → 2891 classes, interfaces & enums
- Used as input/output for several DLR tools
 - Analytical methods
 - Model generators
 - FEM data handling
- Use of common solver-agnostic internal data format with solver-dependent extensions to convert from & to solver-specific formats possible
 - Proof of concept in io plugins
- State:
 - By no means feature-complete
 - Hierarchy converging, yet not free of refactoring

Conclusion

Next steps



- On our way to open source
- Goal:
 - Hierarchy development & standardization
 - jMeS backend directly linked to hierarchy creation, but is just one reference implementation
 - Further hierarchy development possible independent of backend, e.g. similar to [CPACS](#)
- Looking forward to feedback



THANK YOU FOR YOUR ATTENTION

Contact

Martin Rädel

German Aerospace Center
Institute of Composite Structures and Adaptive Systems
Department of Structural Mechanics

Lilienthalplatz 7
38108 Braunschweig

-  +49 (0)531 295-2048
-  +49 (0)531 295-2232
-  martin.raedel@dlr.de
-  www.dlr.de/fa



Virtual Product House

Cornelius-Edzard-Straße 15
28199 Bremen

 www.dlr.de/vph



Impressum



Thema: Complete Model Storage – A solver-agnostic hierarchical approach

Datum: 19.09.2022

Autor: Martin Rädel

Institut: [DLR-FA-STM](#) & [DLR VPH](#)

Bildcredits: -