Development of a Domain-specific Language for Modeling Component Versions and their Dependencies

Daniel Siegl

welcome@lieberlieber.com (email us for more info)

WHY?

FRESH MODEL VERSIONING

LemonTree©

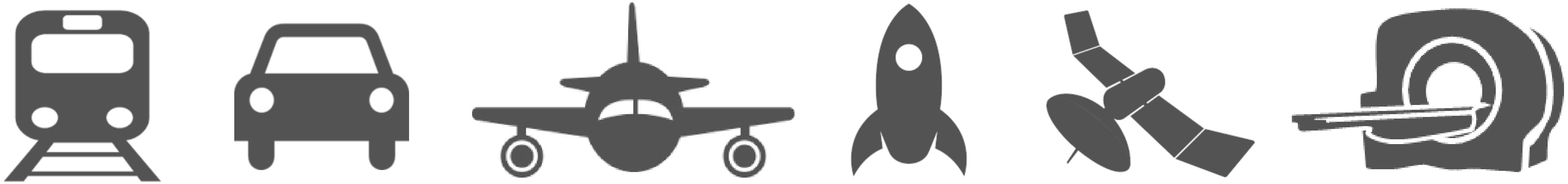
Fresh Model Versioning

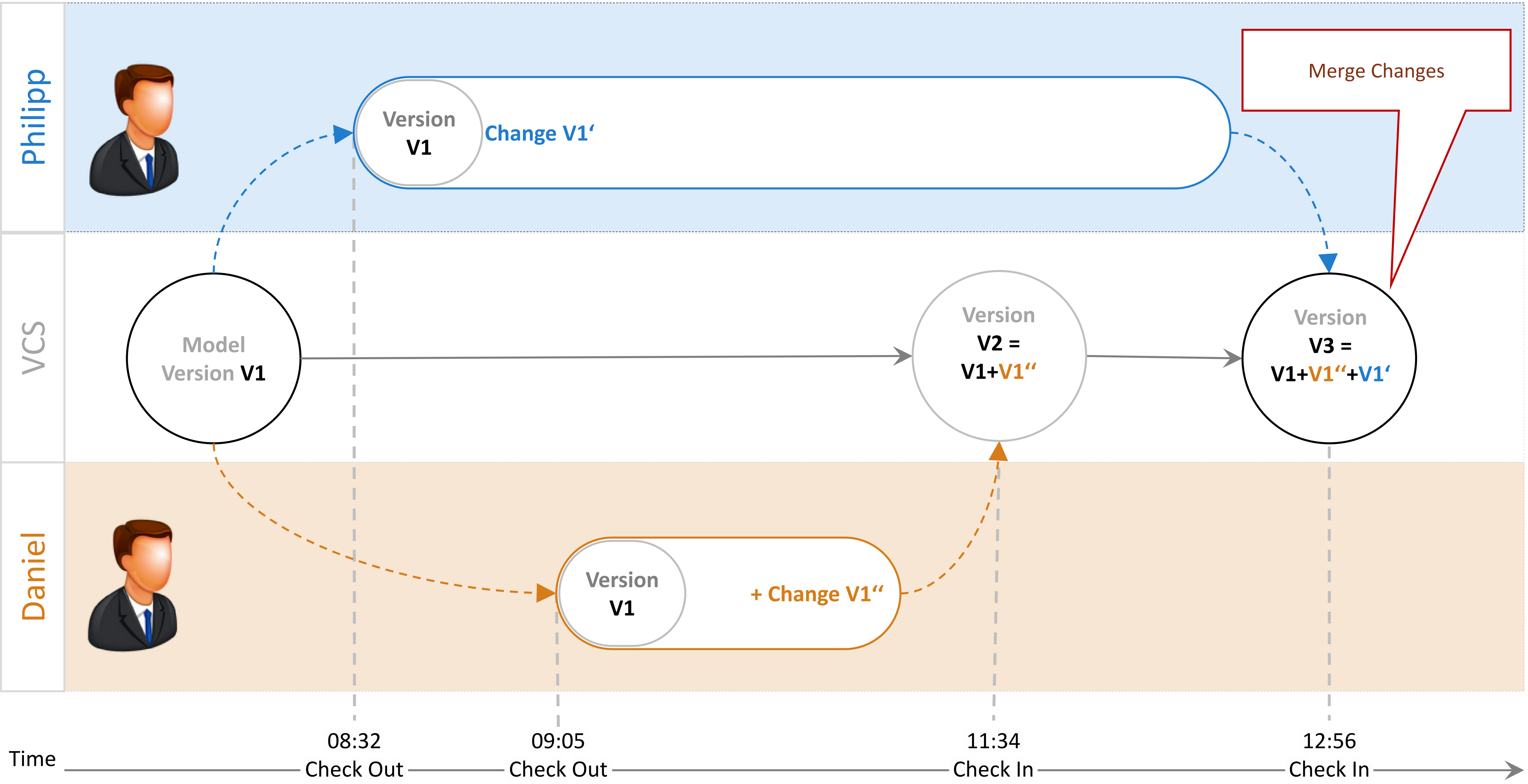
lemontree.lieberlieber.com

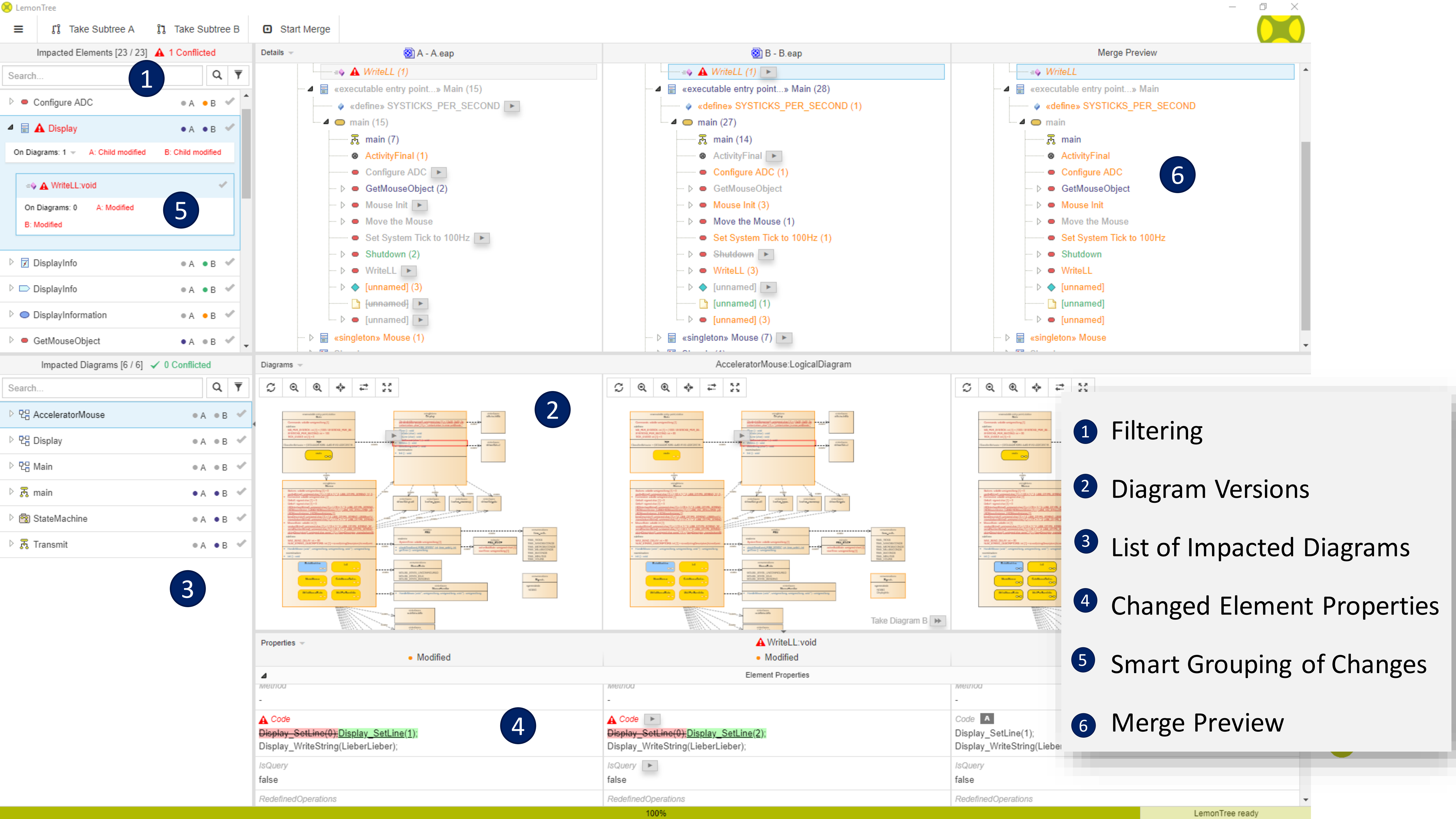
Who uses LemonTree?

We target companies manufacturing safety relevant Cyber Physical Systems!

But more and more Models from other Domains use it. (Banks, Insurances, …)




Philipp
VCS
Daniel
Version V1
Change V1‘
Merge Changes
Model Version V1
Version V2 = V1+V1“
Version V3 = V1+V1“+V1‘
Version V1
+ Change V1“
Time
08:32 Check Out
09:05 Check Out
11:34 Check In
12:56 Check In



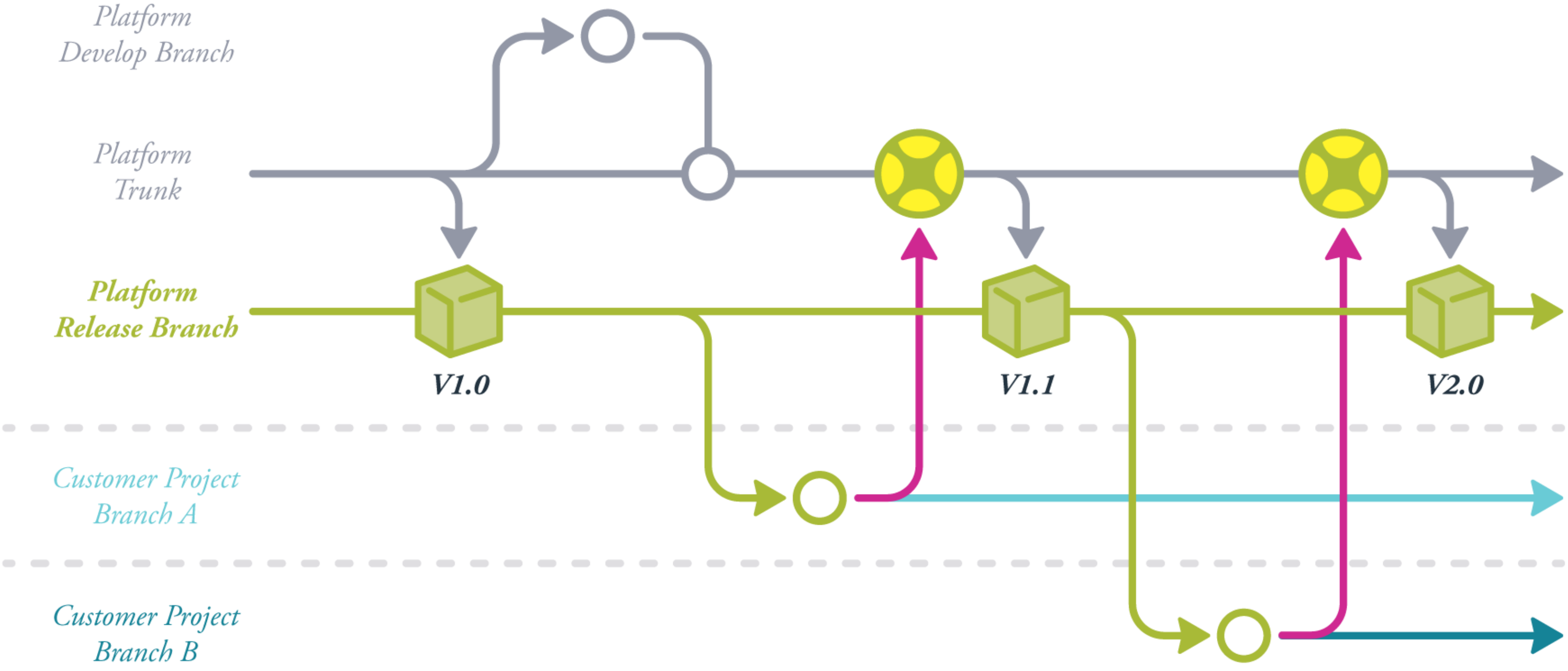
~~Display_SetLine(0);~~Display_SetLine(1);
Display_WriteString(LieberLieber);

~~Display_SetLine(0);~~Display_SetLine(2);
Display_WriteString(LieberLieber);

Display_SetLine(1);
Display_WriteString(LieberLieber

1. Filtering
2. Diagram Versions
3. List of Impacted Diagrams
4. Changed Element Properties
5. Smart Grouping of Changes
6. Merge Preview

Typical Scenario with Branching

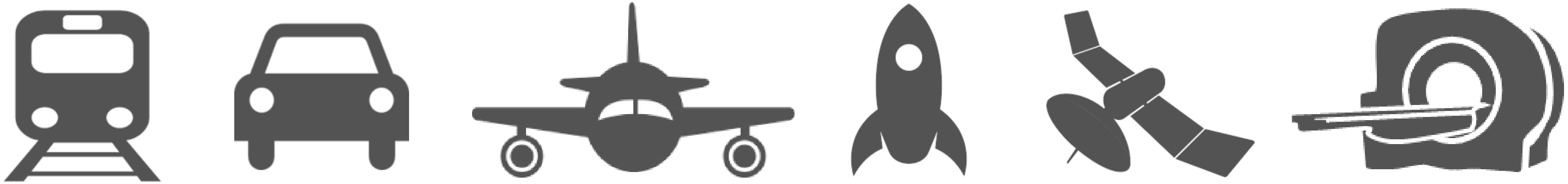


Users need model components!!

“Model Slicing”

“Model Components”

“Product Line Engineering”



Problem Statement

- Interaction of components/assets in systems is becoming more and more complex
- When new standards or directives appear, adaptations in the systems have to be made
  - Assets change over time
  - Different versions
  - Dependencies between different assets in specific versions

HOW?

CDL-MINT Team – Modul Reactive Model Repositories

Univ.-Prof. Mag. Dr. **Manuel Wimmer**

Institute for Business Informatics – Software Engineering
JKU Linz

JOHANNES KEPLER
UNIVERSITY LINZ

DI Mag. Dr.techn.
Alexandra Mazak

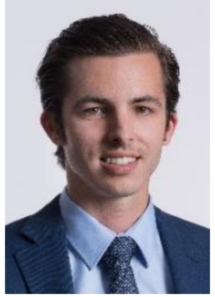
DI
Sabine Wolny

Daniel Lehner, BSc

Maximilian Medetz

LieberLieber

Daniel Siegl



DI Dr.rer.soc.oec.
Konrad Wieland

Challenge and Motivating Example

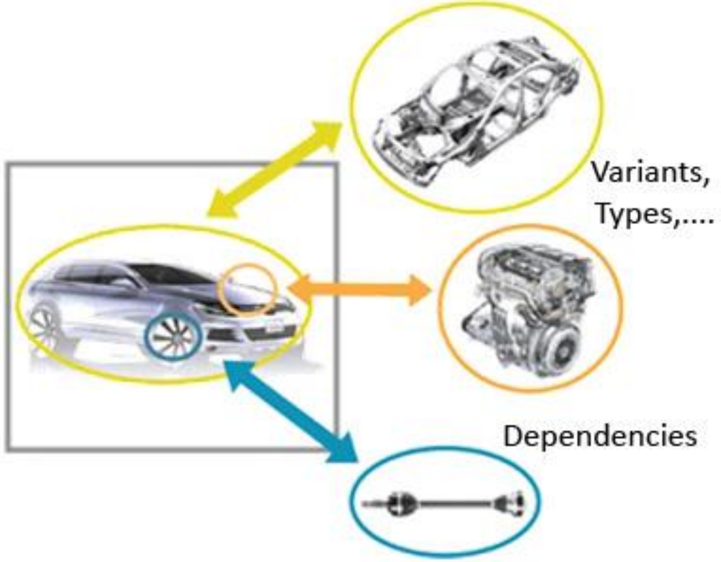
Open Issues

- Can the currently used system configuration be updated?
- Is a certain system configuration possible?
- Can the traceability of dependencies between assets be ensured across different versions?

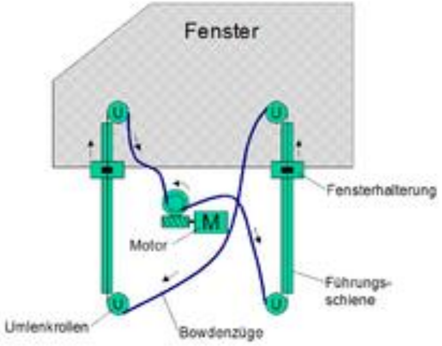
Challenge: Developing a concept for modeling version dependencies

Motivation

- Modeling version dependencies
- Example of an automatic window regulator of a car

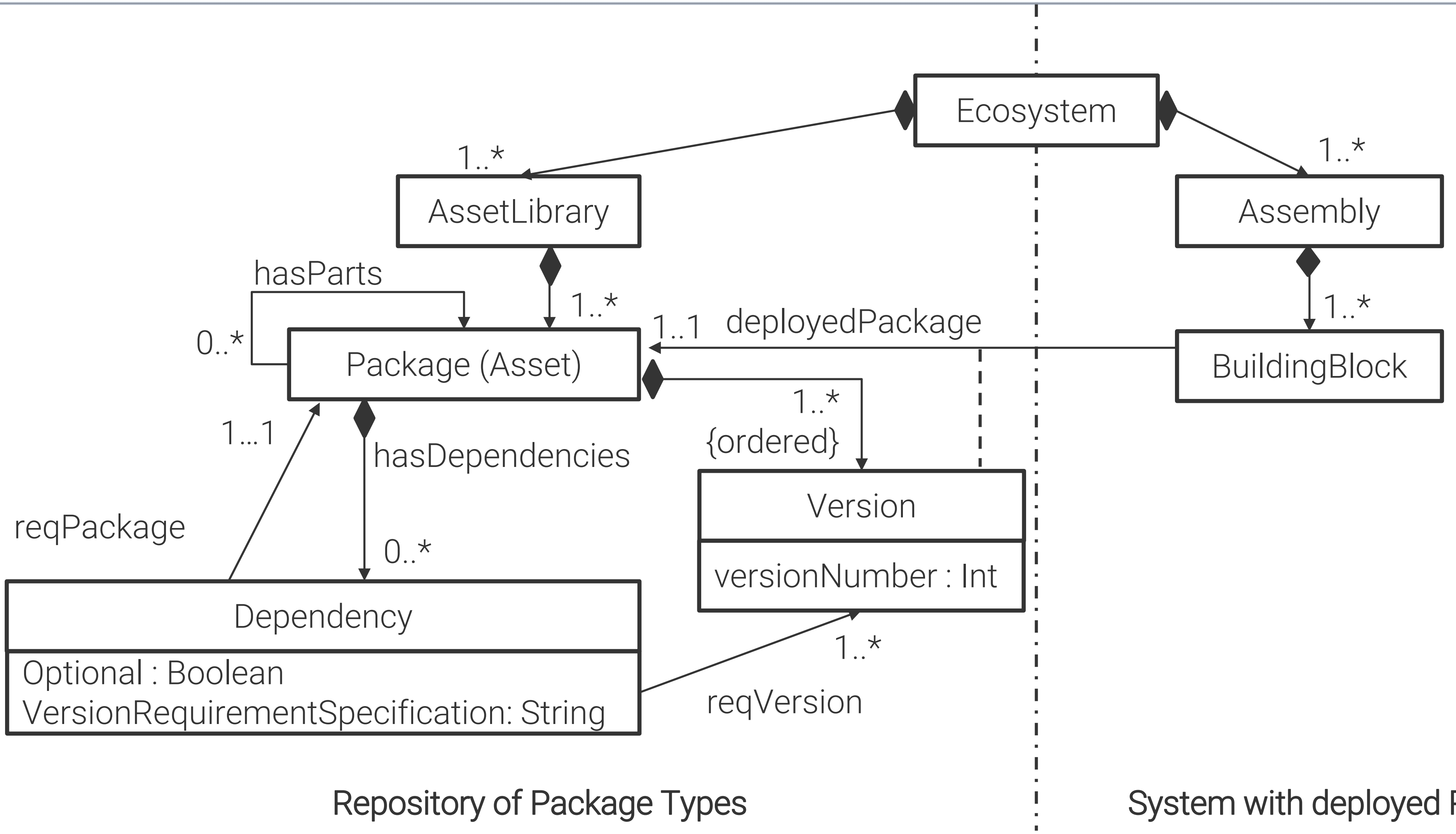


https://link.springer.com/chapter/10.1007/978-3-658-12295-9_5



https://de.wikipedia.org/wiki/Fensterheber

DSL for modeling Dependencies between Packages (Assets) across different Versions

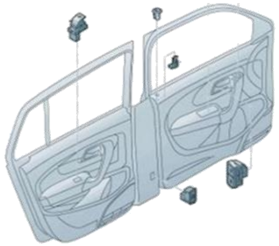


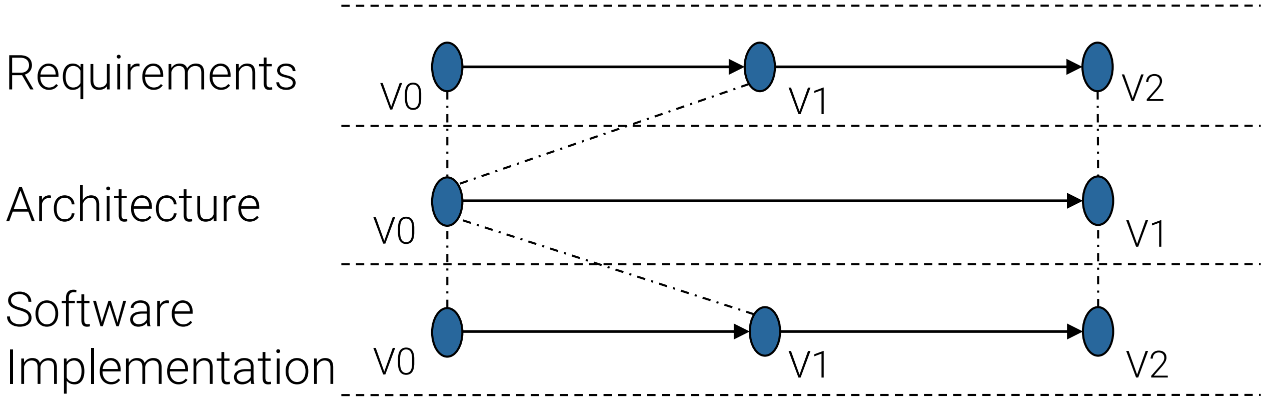
Repository of Package Types

System with deployed Packages

Definition of Dependencies - Example Window Regulator

ISO 26262 ASPICE





Syntax for specifying Version Dependencies

```
1.0:       x = 1.0
(,1.0]:    x <= 1.0
[1.2,1.3]: 1.2 <= x <= 1.3
[1.0,2.0): 1.0 <= x < 2.0
[1.5,):    x >= 1.5
```

Asset Library

- Package (Asset) **Requirements**
  - Version 0
  - Version 1
  - Version 2
- Package (Asset) **Architecture**
  - Version 0
    - Dependency: Requirements (,1]
    - Dependency: Software Implementation (,1]
  - Version 1

    …
- Package (Asset) **Software Implementation**

  …

Is the information in a given Asset Library consistent?

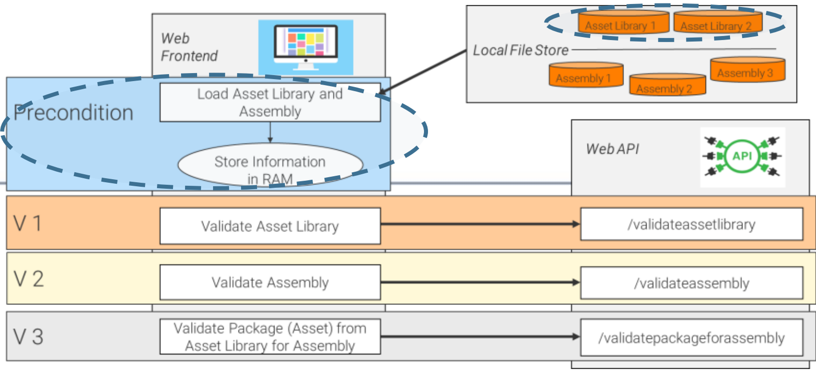
Are the packages (Assets) compatible in a concrete assembly?

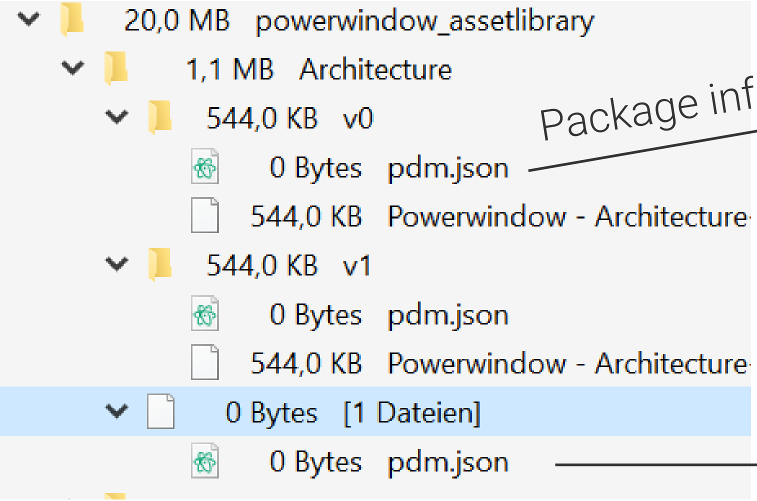
Can a particular package (asset) be installed from a given Asset Library into a concrete assembly?

Graphic Source:
- https://content.intland.com/blog/automotive/compliance-in-automotive-development-iso-26262-iec-61508-aspice-cmmi-and-more
- https://originalteilemarkt.de/schalter-fuer-elektrischen-fensterheber-fahrerseite-auto-down-2-fach-satinschwarz-6ru959858c-9b9-2822026
- https://de.aliexpress.com/item/32835519323.html

Version information – Asset Library

- Expected folder structure





```
20,0 MB   powerwindow_assetlibrary
    1,1 MB   Architecture
        544,0 KB   v0
            0 Bytes   pdm.json
            544,0 KB   Powerwindow - Architecture
        544,0 KB   v1
            0 Bytes   pdm.json
            544,0 KB   Powerwindow - Architecture
        0 Bytes   [1 Dateien]
            0 Bytes   pdm.json
```

Package information of a specific version

```
{
  "package name" : "Architecture",
  "dependencies" : [
    {
    "package name" : "Software_Implementation",
    "minimum version" : 0
    }
  ]
}
```

Top-level package information

```
{
  "package name" : "Architecture",
  "versions" : [
    {
      "version number" : 0,
      "relative path" : "/v0"
    },
    {
      "version number" : 1,
      "relative path" : "/v1"
    }
  ]
}
```

Szenario

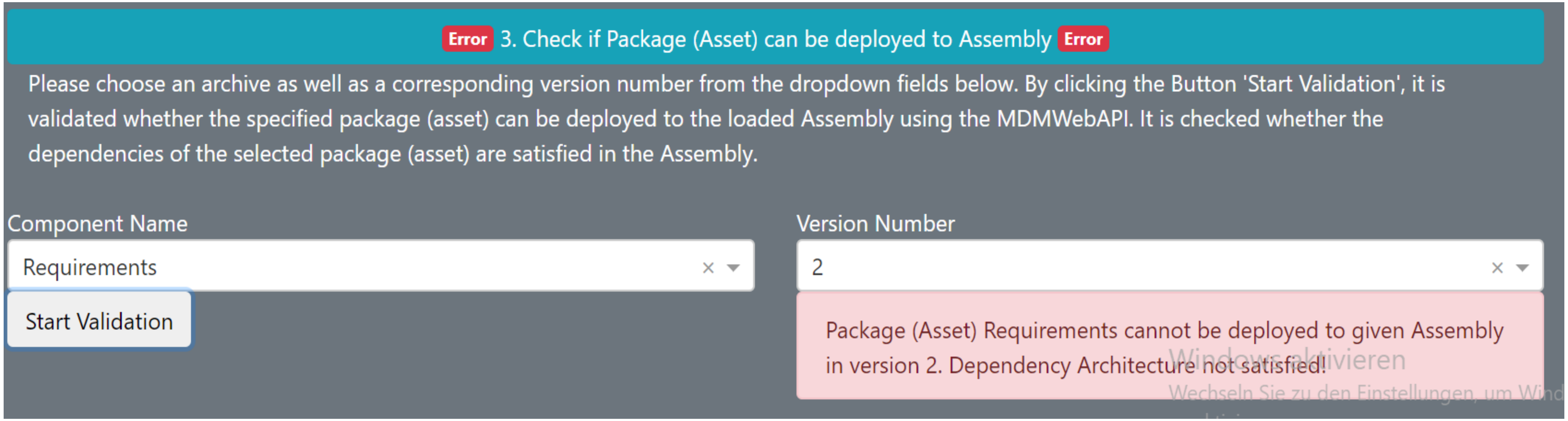
Existing Assembly has version 1 of asset „Requirements“ installed

We want to update to version 2 of asset „Requirements“

Version 2 of asset „Requirements“ has an additional dependency to asset „Architecture“, which is not available in the assembly

Can I update my assembly to version 2 of asset „Requirements“straight away?

No, the assembly validation correctly shows that the dependencies are not fulfilled



Error 3. Check if Package (Asset) can be deployed to Assembly Error

Please choose an archive as well as a corresponding version number from the dropdown fields below. By clicking the Button 'Start Validation', it is validated whether the specified package (asset) can be deployed to the loaded Assembly using the MDMWebAPI. It is checked whether the dependencies of the selected package (asset) are satisfied in the Assembly.

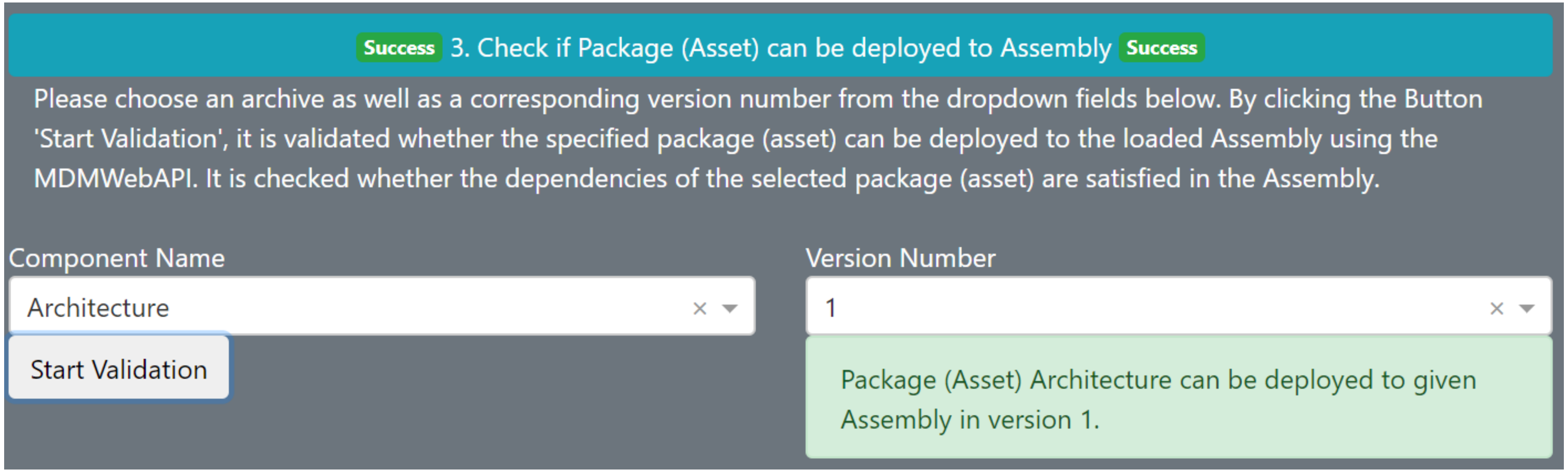
Component Name: Requirements

Version Number: 2

Package (Asset) Requirements cannot be deployed to given Assembly in version 2. Dependency Architecture not satisfied!

Can I add the required dependency „Architecture“ to my assembly?

Yes, version 1 of asset „Architecture“ can be added



Success 3. Check if Package (Asset) can be deployed to Assembly Success

Please choose an archive as well as a corresponding version number from the dropdown fields below. By clicking the Button 'Start Validation', it is validated whether the specified package (asset) can be deployed to the loaded Assembly using the MDMWebAPI. It is checked whether the dependencies of the selected package (asset) are satisfied in the Assembly.

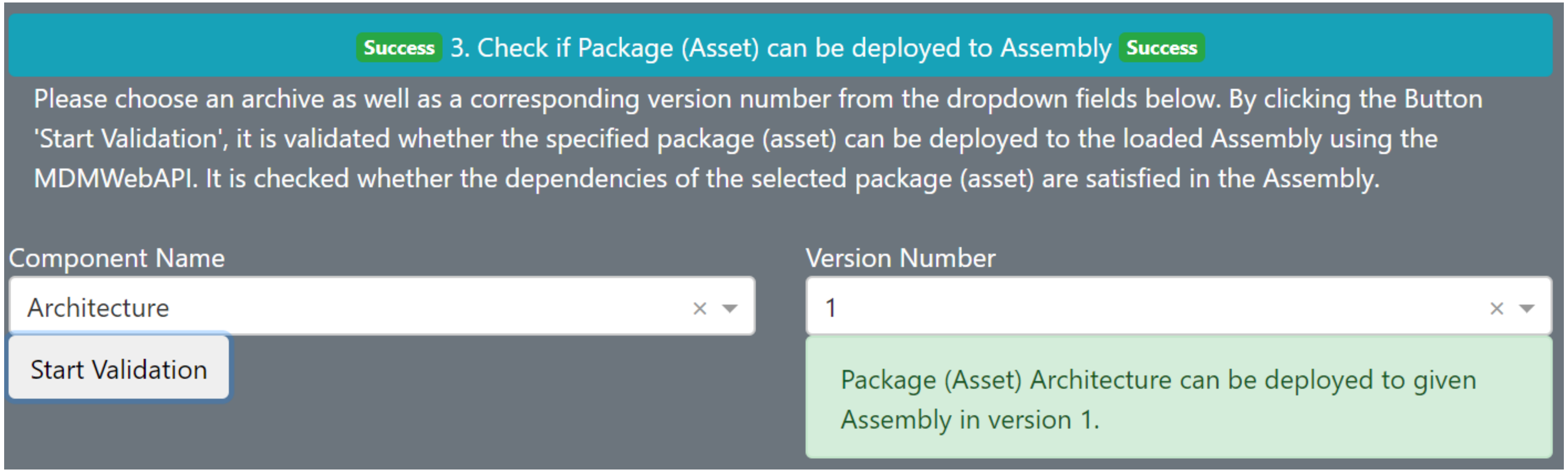
Component Name: Architecture

Version Number: 1

Package (Asset) Architecture can be deployed to given Assembly in version 1.

Can I update my assembly to version 2 of asset „Requirements“ now?

After adding version 1 of asset „Architecture“, asset Requirements can be updated to version 2



Success 3. Check if Package (Asset) can be deployed to Assembly Success

Please choose an archive as well as a corresponding version number from the dropdown fields below. By clicking the Button 'Start Validation', it is validated whether the specified package (asset) can be deployed to the loaded Assembly using the MDMWebAPI. It is checked whether the dependencies of the selected package (asset) are satisfied in the Assembly.

Component Name: Architecture

Version Number: 1

Package (Asset) Architecture can be deployed to given Assembly in version 1.

Conclusion and Future Work

- DSL for modeling version dependencies

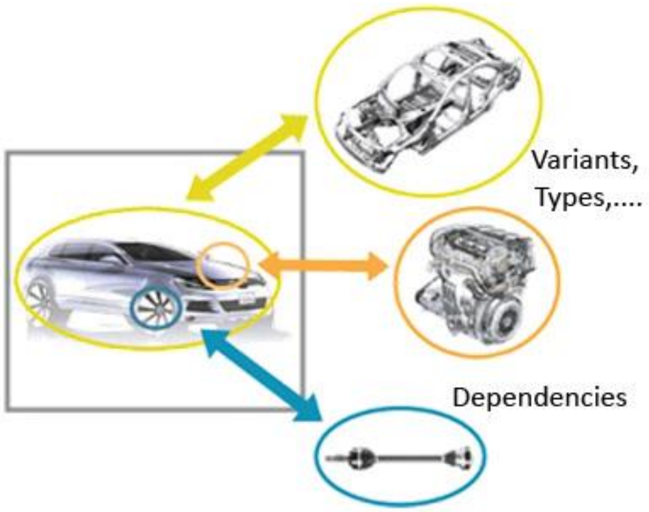
- First implementation as Web Frontend

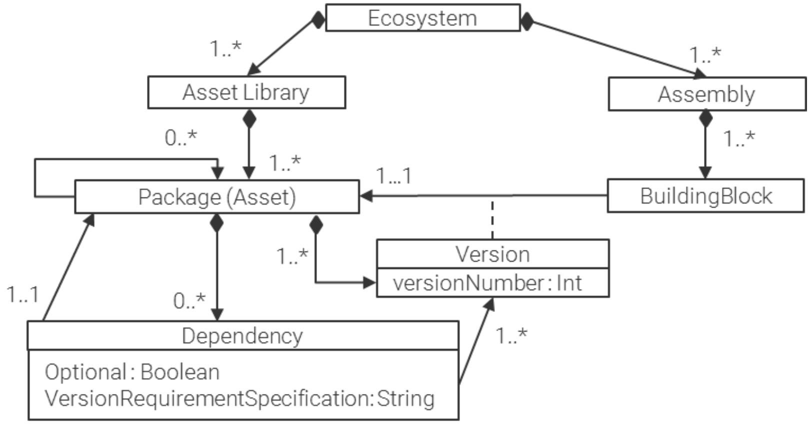
Open Challenge

- PLM/ALM/VCS – Where should the version dependency be represented?

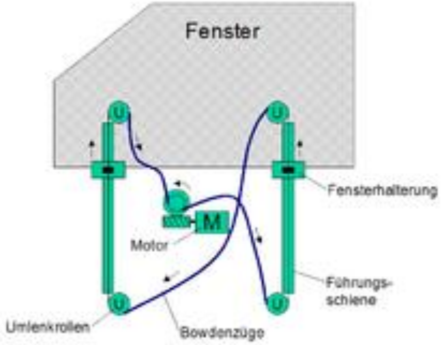
Next steps

- Web interface for automated configuration of an asset library
  - Add, update and delete packages (assets) and their dependencies
  - Abstraction from actual storage to increase user acceptance
- Automatic Derivation of an Assembly
  - From a Git Repository/Enterprise Architect project
- **First Commercial Rollout at an OEM Q4 2020**



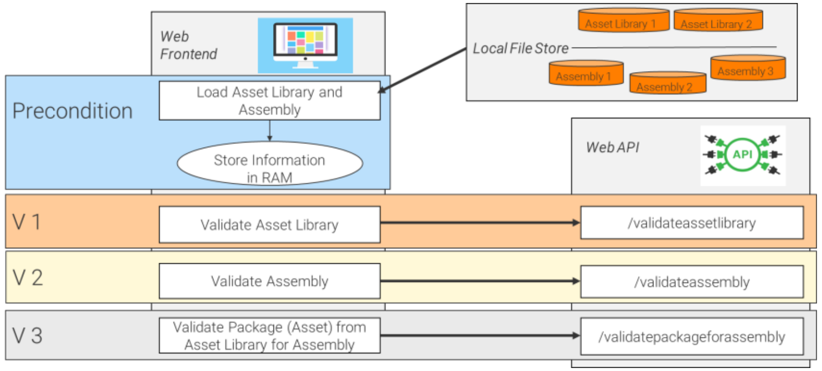


Questions?



Daniel Siegl

welcome@lieberlieber.com
(email us for more info)


Web
Frontend
Local File Store
Asset Library 1
Asset Library 2
Assembly 1
Assembly 2
Assembly 3
Precondition
Load Asset Library and Assembly
Store Information in RAM
Web API
API
V 1
Validate Asset Library
/validateassetlibrary
V 2
Validate Assembly
/validateassembly
V 3
Validate Package (Asset) from Asset Library for Assembly
/validatepackageforassembly