SysMLv2: Beyond the hype

by Jennifer Korent

TABLE OF CONTENTS

SODIUS WILLERT

From UML inheritance...

Fundament	Consistency	Model Exchange	Representation
• UML • an UML profile	• Inconsistencies between Definition and Usage • Inconsistent Semantics	• XMI • Open to different interpretations • Fragmented Tool Ecosystem • Hardly portable	• Diagram centricity • No formal textual language

From UML inheritance (SysML v1)

... to a native Systems Engineering Language

Fundament	Consistency	Model Exchange	Representation
• Not based on UML • well defined metamodel and language model • Based on KerML	• Consistent separation of definition and usage • more precise and more consistent semantics	• Standardized SysML v2 API • Portability between tool vendors • Machine-readable	• Formal graphical notation • Formal textual notation

To a native Systems Engineering Laguage (SysML v2)

From UML inheritance to a native Systems Engineering Language

SysML v2 addresses the pain points of v1

What matters is whether this substance also reaches everyday engineering practice

In principle **the hype got a basis**

But the **language alone** doesn‘t carry an engineering project

What needs to be added to **turn the hype into value**?

Three key factors for SysML v2 value

- Requirements are single source of truth
- Linked, versioned, and traceable across all V-levels

End-to-end requirements engineering

- SysML tool doesn‘t exist in isolation
- Integration in requirements management and testing via shared platform

Tool integration

- Tool must support the process itself not the other way
- Every day use depends on the extensibility of the tool

Methodology and process support

SODIUS WILLERT

Example project : Adaptive Cruise Control (ACC)

What the project represents

- Adaptive Cruise Control
  - driver assistance system that regulates distance and speed
- Limited scope
  - Complex enough for real traceability
  - Small enough for presentation
- Represented V-model levels
  - Stakeholder requirements
  - Logical architecture
  - Test cases

Tool environment

Jazz platform ✦ Global Configuration Management

IBM Doors Next
Requirement

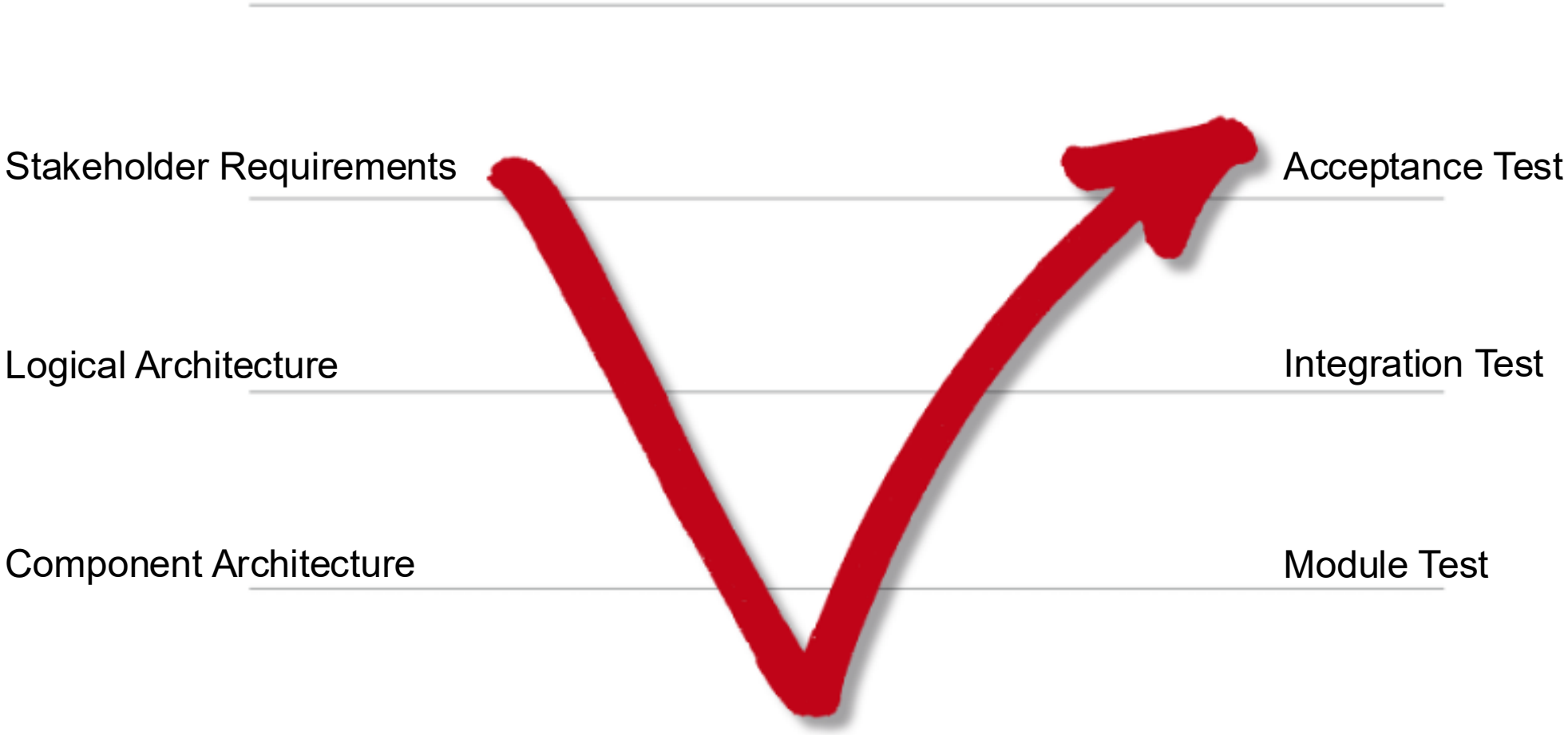
IBM Rhapsody Systems Engineering
SysML v2 model

IBM Engineering Test Management
Test Case ✦ Test Run

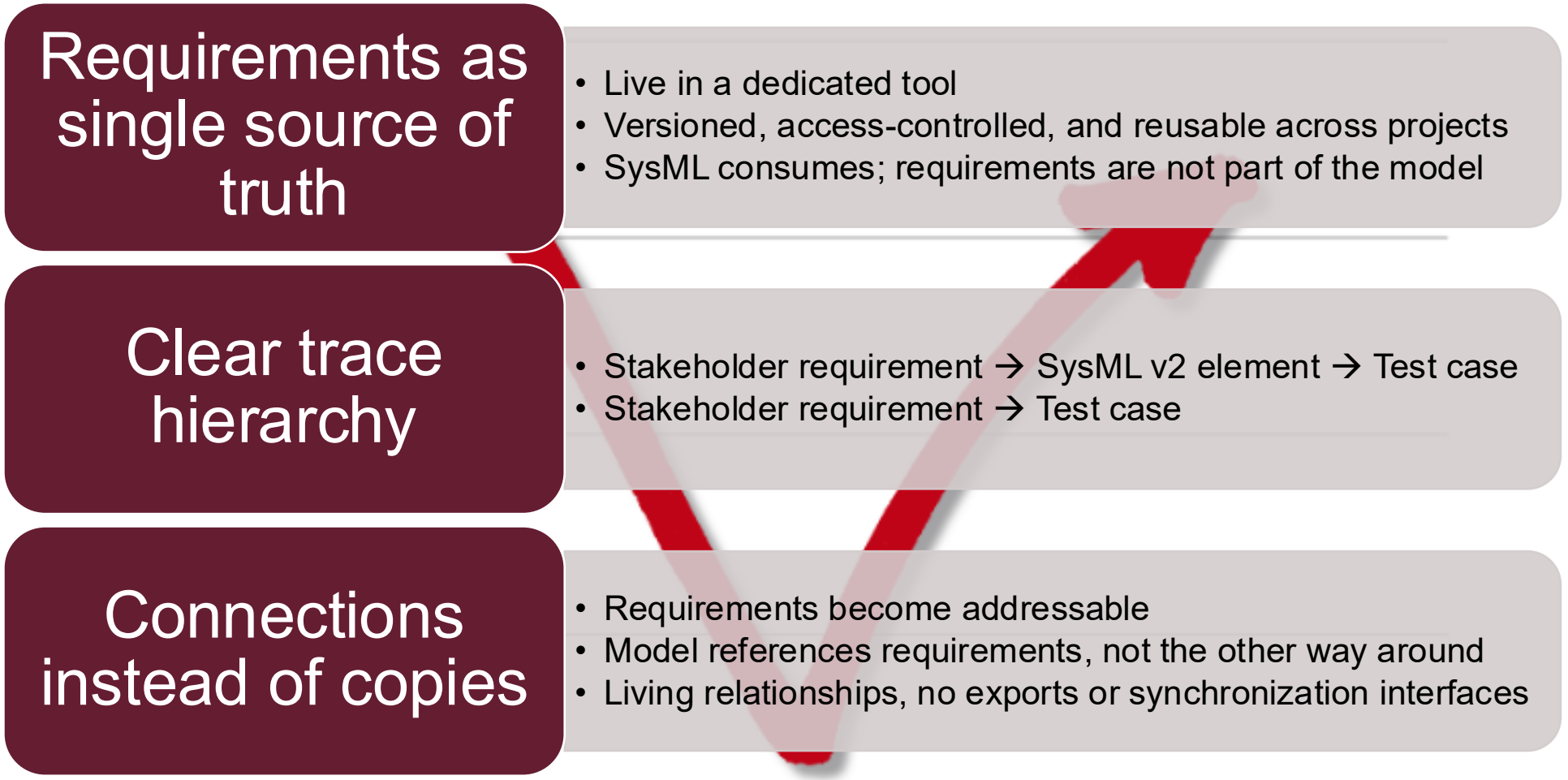
Connected via OSLC link ✦ global configuration baselines

SODIUS WILLERT

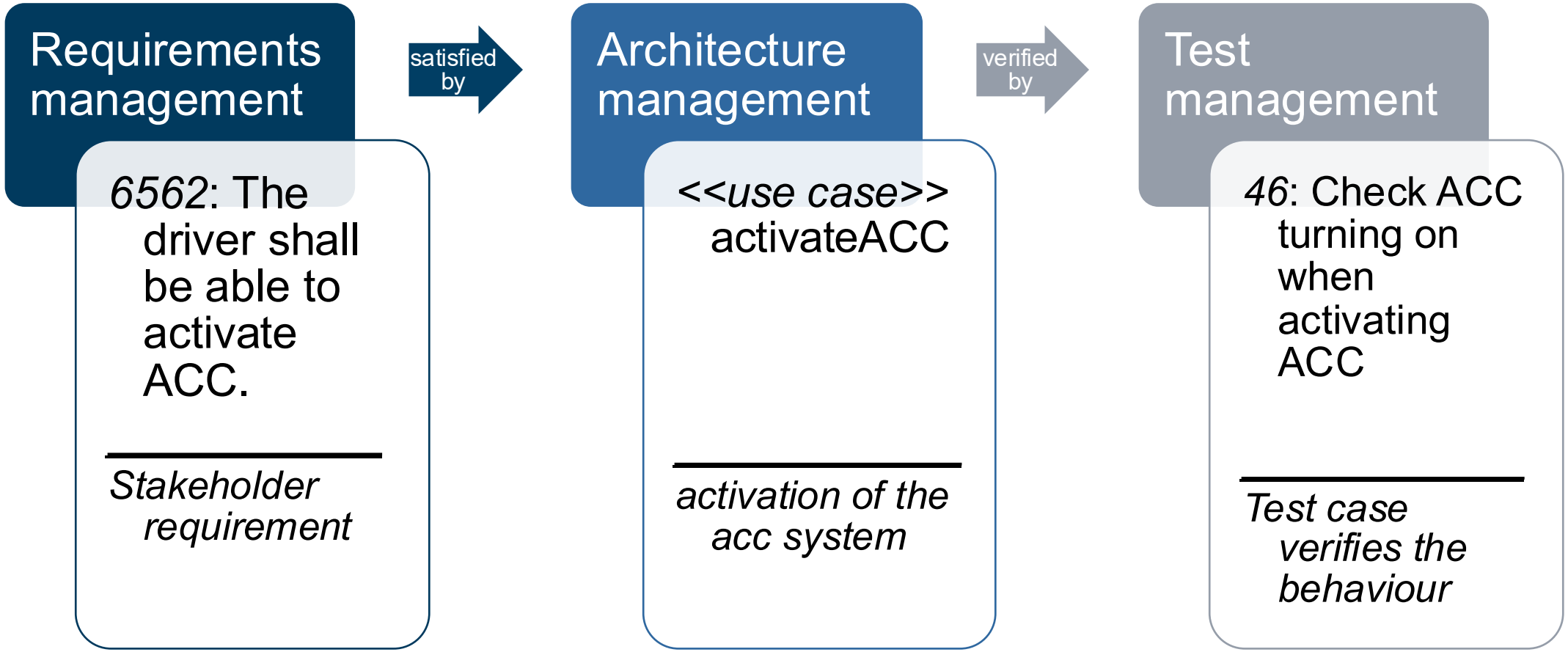
V-model as framework



Key factor: End-to-end requirements engineering

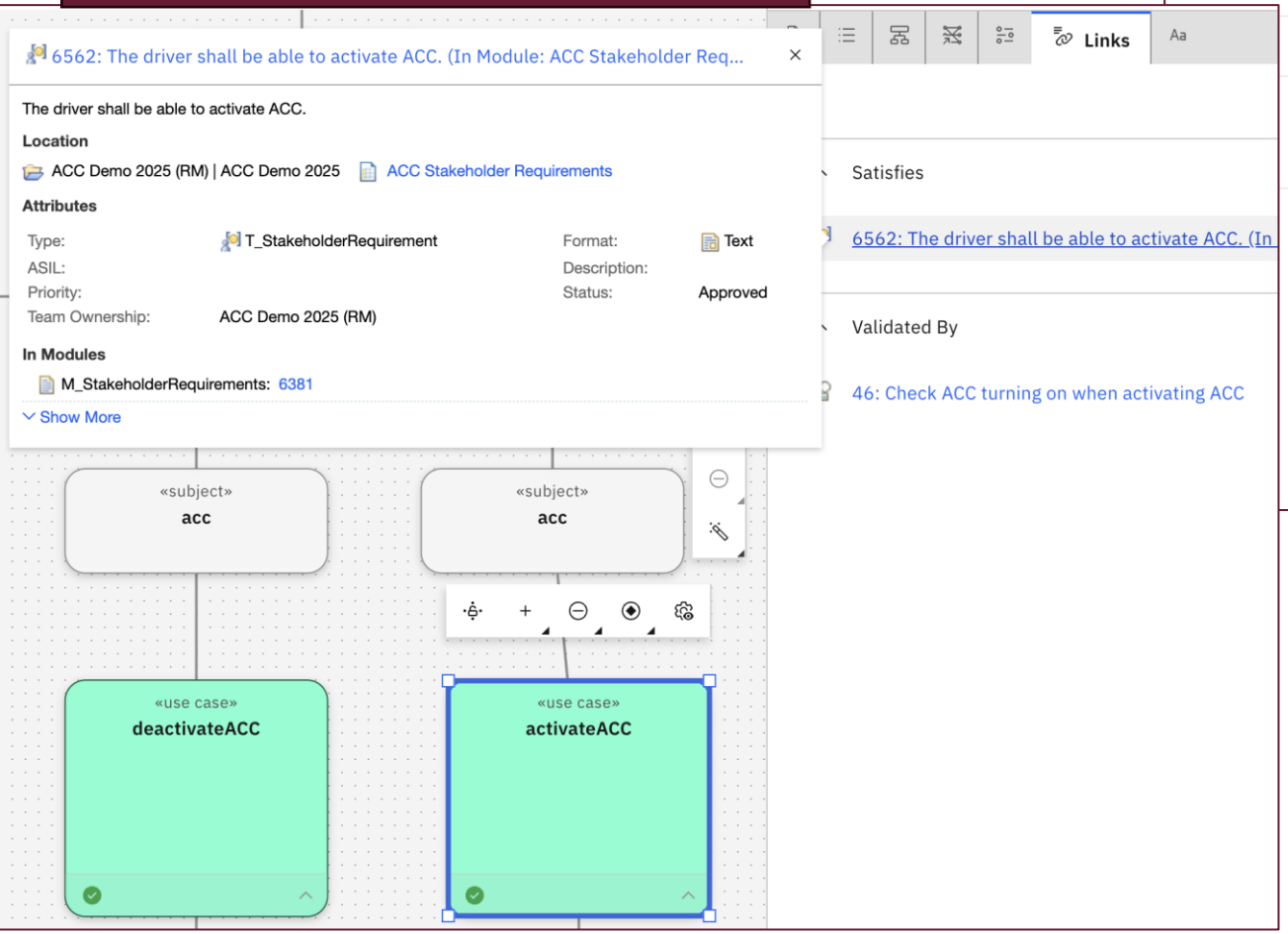


ACC : end-to-end traceability

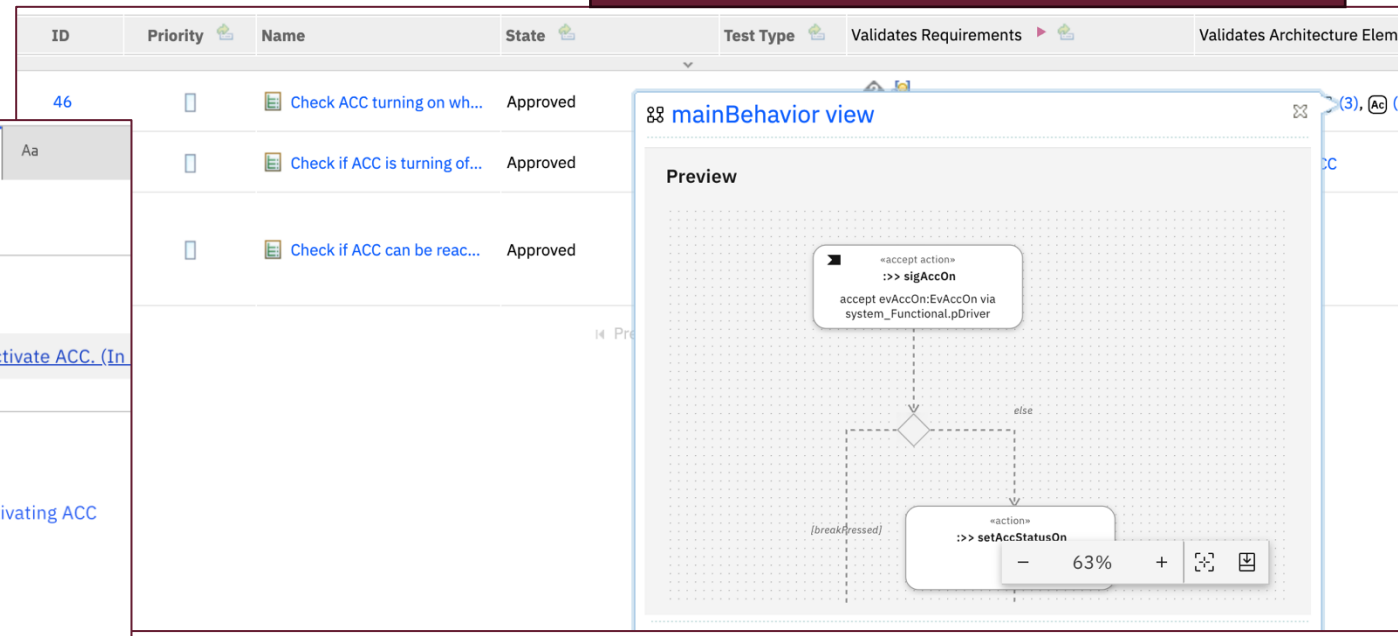


ACC : cross tool view in practice

Rhapsody Systems Engineering

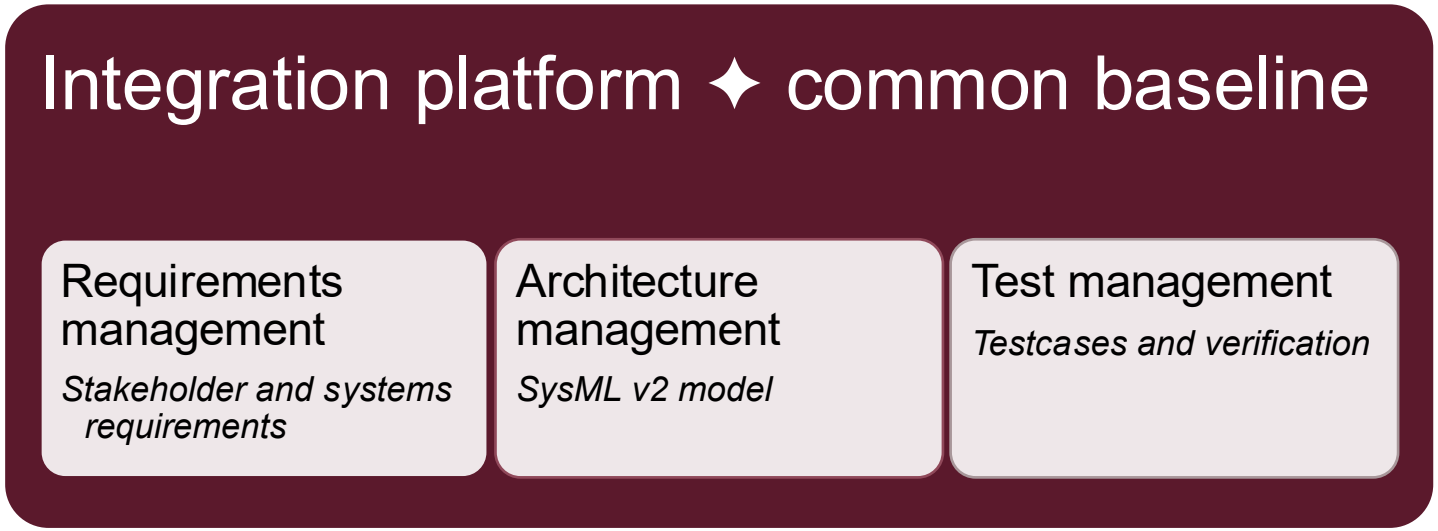


Engineering Test Management



Connections are bidirectional:
From the requirement into the model and from the model back again. Both views show the same trace, each from its own perspective. Cross-tool visibility without changing tools.

Key factor: tool integration via shared platform



Three effects

A baseline for everything
A platform baseline brings together requirements, models, and tests in a single, referenceable state.

Visible change impact
You can see a change in a requirement automatically in the related models and tests.

Traceability for audits
Who changed what, when? And how does it affect verified requirements?

SODIUS WILLERT

ACC : change impact across the toolchain

Requirements management

Change on requirement 6562

Requirement is updated

Architecture management

Model elements are linked

Relationship between requirement and model element are known because of the OSLC link

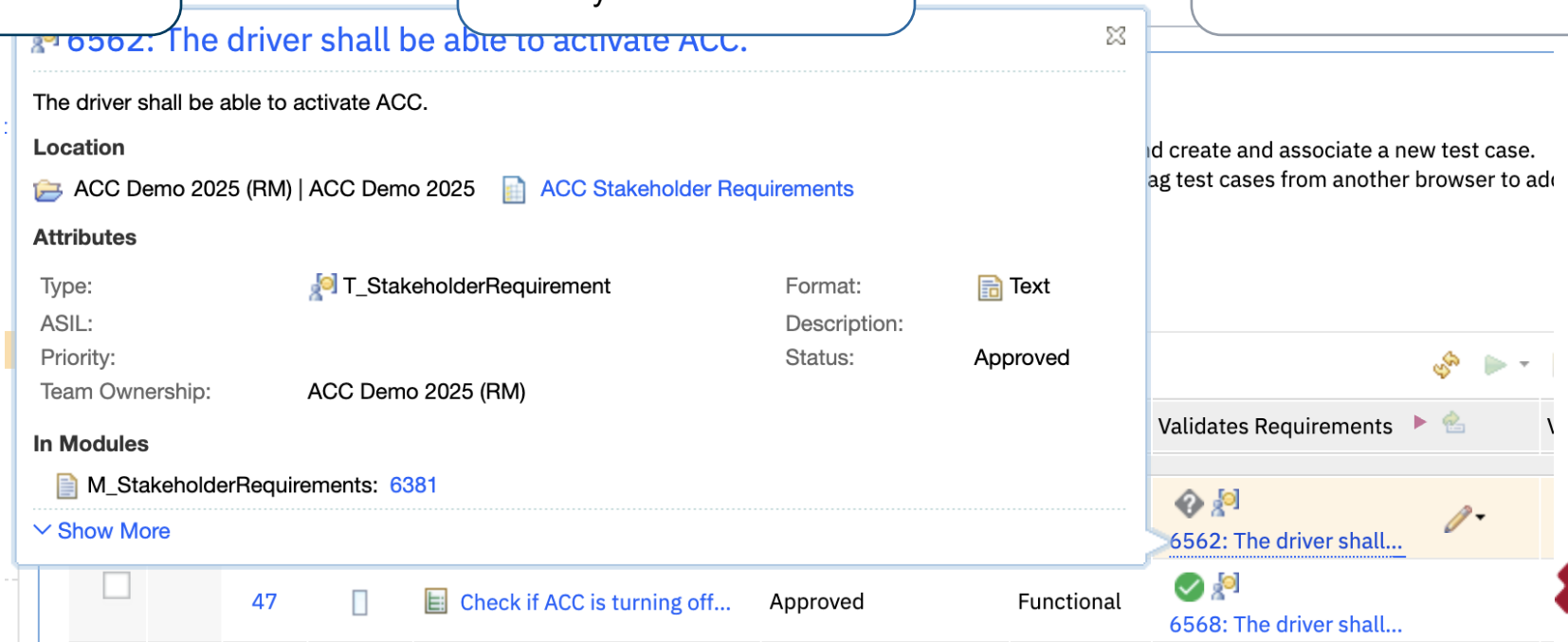
Consistency check manually or with link validity when activated

Test management

Test cases get a suspect link

Test cases which are linked to the requirement can be filtered directly for revalidation

Without tool integration this would stay a manual, risky search task



Key factor : Methodology and process support

Established methodologies

- Harmony SE
- OOSEM
- MagicGrid
- Domain specific processes
  - *e.g. ASPICE*

The key question

Does the tool support your process?
Or do you have to support the tool?

- No methodology fits your organization perfectly
- Evolved processes are not a weakness, they are the result of experience
- The tool must be able to use your processes

What needs to be customizable:
domain concepts as reusable language elements ✦ rules that the tool can verify ✦ process templates

ACC : methodology and process support

What Rhapsody SE offers

Standard Harmony Workflow

1.Requirements Analysis
2.Functional Analysis
3.Logical Analysis
4.Physical Analysis

What we adopted

Domain specific Workflow

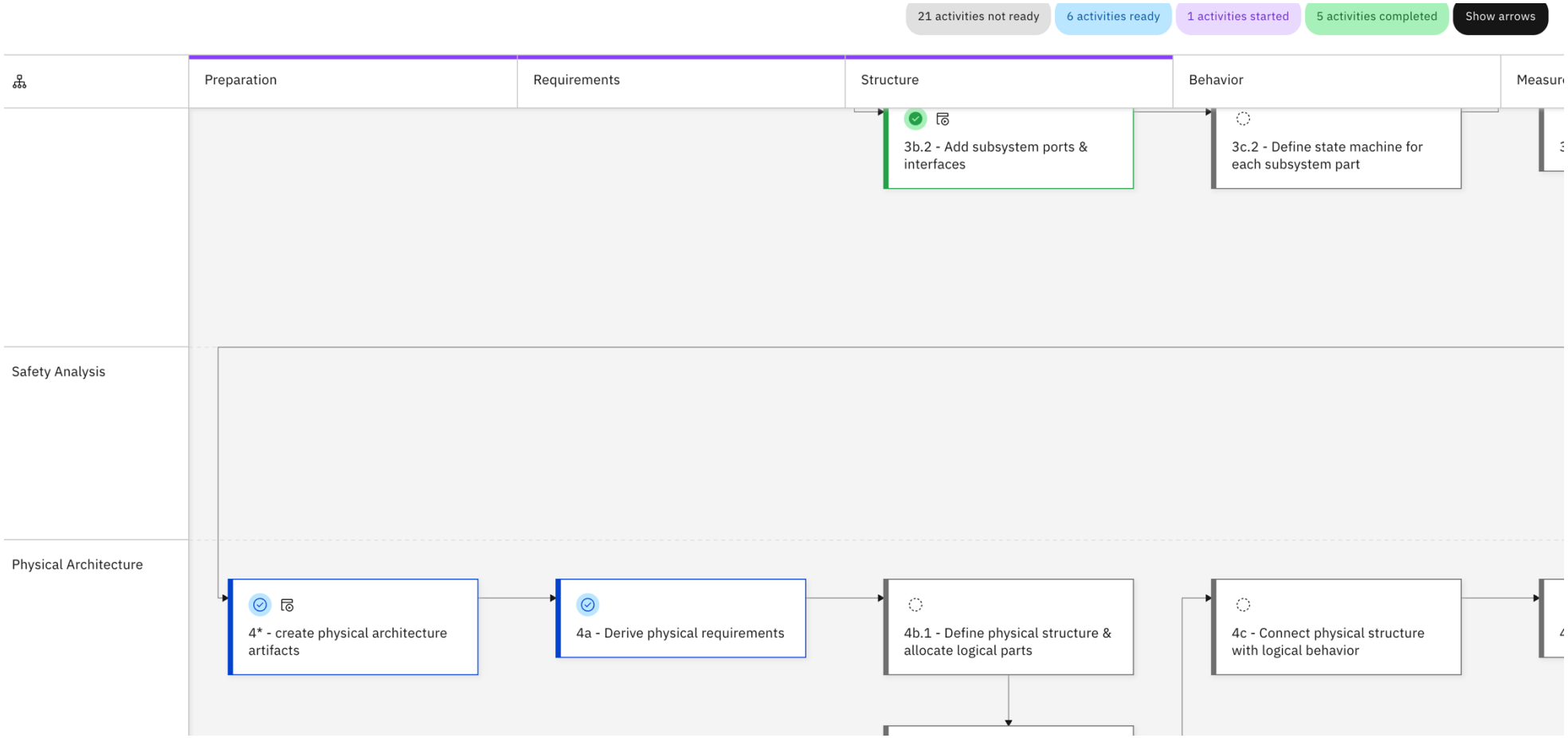
1.Requirements Analysis
2.Functional Analysis
3.Logical Analysis
4.Safety Analysis
5.Physical Analysis

SysML v2 concepts

Language extension ✦ Libraries ✦ Templates

SODIUS WILLERT

ACC : the process updated in the tool



Engineers follow the path the tool sets for them; consciously or unconsciously.
Adapt the workflow to your process.

Conclusion ✦ Take away

1

The language is necessary but not sufficient

- SysML v2 solves problems of v1
- The hype got a basis
- The fundament of a SE project is created from the interaction of requirements, tools and methodology

2

Established processes are strength not weakness

- The tool must be able to be adapted to your rules, not the other way around
- Libraries, extensions and workflow templates are the way to adopt your processes

3

Extensibility separates concept from reality

- if the tool knows your path, methodology is structurally supported not dependent on discipline

SODIUS WILLERT

What is final? What is still growing?

Specification is final

SysML v2 & KerML are formal adopted

API specification is formal

Specification defines the standard
First tools use it

Textual Syntax is usable

Models as code is possible, models are reviewable like code

Commercial Tools are using v2

e.g. IBM Rhapsody SE and Dassault CatiaMagic offer SysML v2 solutions

Migration v1 → v2

No auto converter available yet

Methodology changes

The common methodologies are v1-based and the change is in progress

Team competence

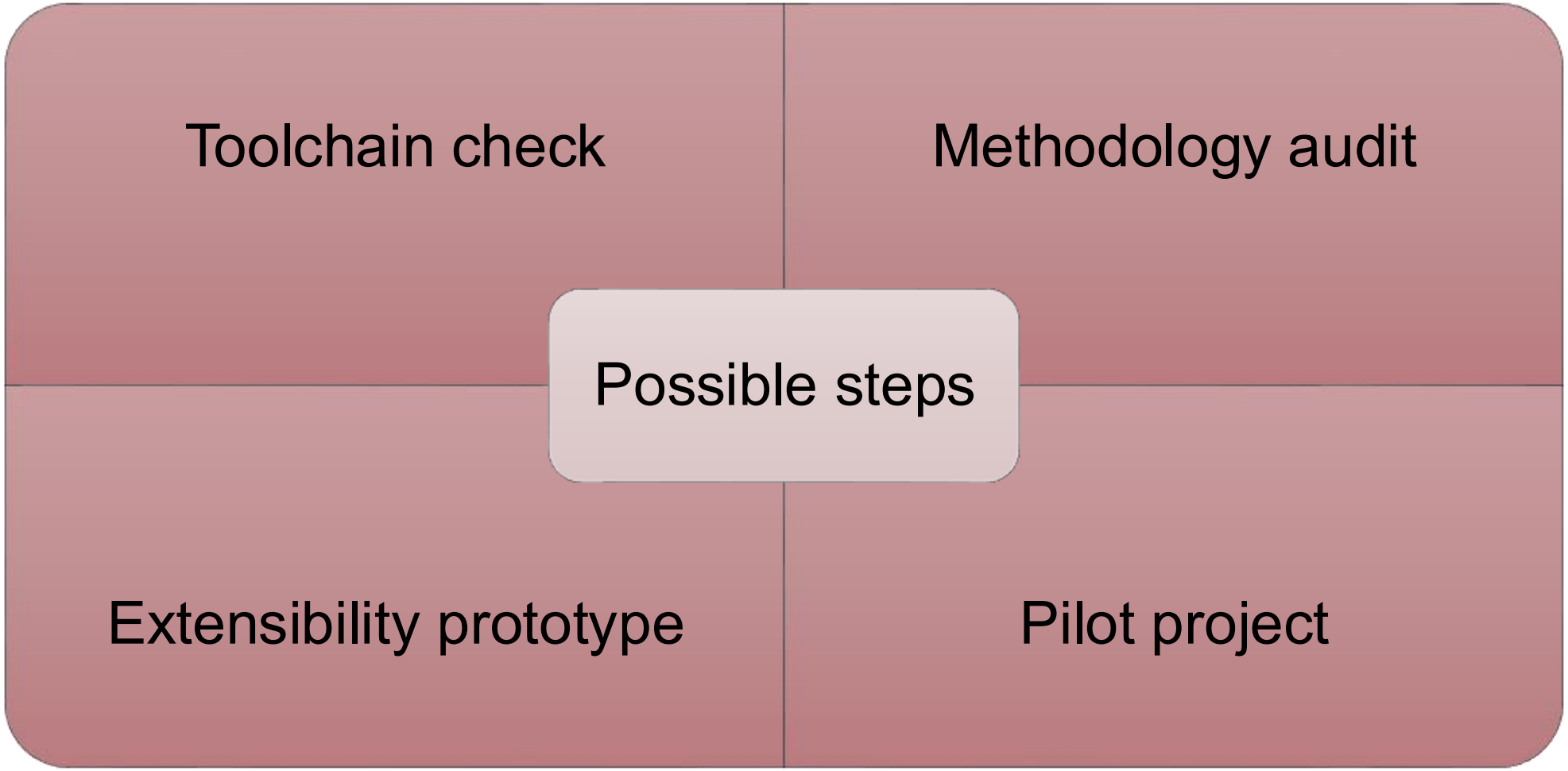
New language and concepts need to be learned

Feature gaps in tools

Not every tool supports the whole spec yet

SODIUS WILLERT

What could be your next steps?



Start small ✦ Make it measurable ✦ Then scale up

SODIUS WILLERT

Q & A

What are your take aways?

What remains open?

Test it! Use our Sandbox

Willert Software Tools offers access to Rhapsody SE Sandbox

Get your SysML v2 Cheat Sheet

You can get it also as multifunctional laptop cloth on our booth

Thank you

SODIUS SAS
34 Boulevard du Maréchal A. Juin
44100 Nantes, France
+33 (0)228 236 060

SODIUS CORP
418 N. Main Street 2nd Floor
Royal Oak, MI 48067, USA
+1 (248) 270-2950

WILLERT SOFTWARE TOOLS GmbH
Hannoversche Str. 21,
31675 Bückeburg, Germany
+49 5722 9678 60

For more information visit sodiuswillert.com