A Case Study on the Evolution of a Component-based Product Line

Wolfgang Heider[1], Michael Vierhauser[2], Daniela Lettner[1], Paul Grünbacher[1]

[1] Christian Doppler Lab. for ASE, Johannes Kepler University

[2] Siemens VAI Metals Technologies

Linz, Austria

August 23, 2012

Context, Motivation, Goal

- Maintenance and evolution of product lines is increasingly complex
- We developed a product line engineering tool suite, the DOPLER tool suite DOPLER Decision-Oriented Product Line Engineering for effective Reuse
- We observed the evolution of this industrial product line while it was refactored for industrial use cases by Siemens VAI

→ Analyze developer challenges and tool support requirements

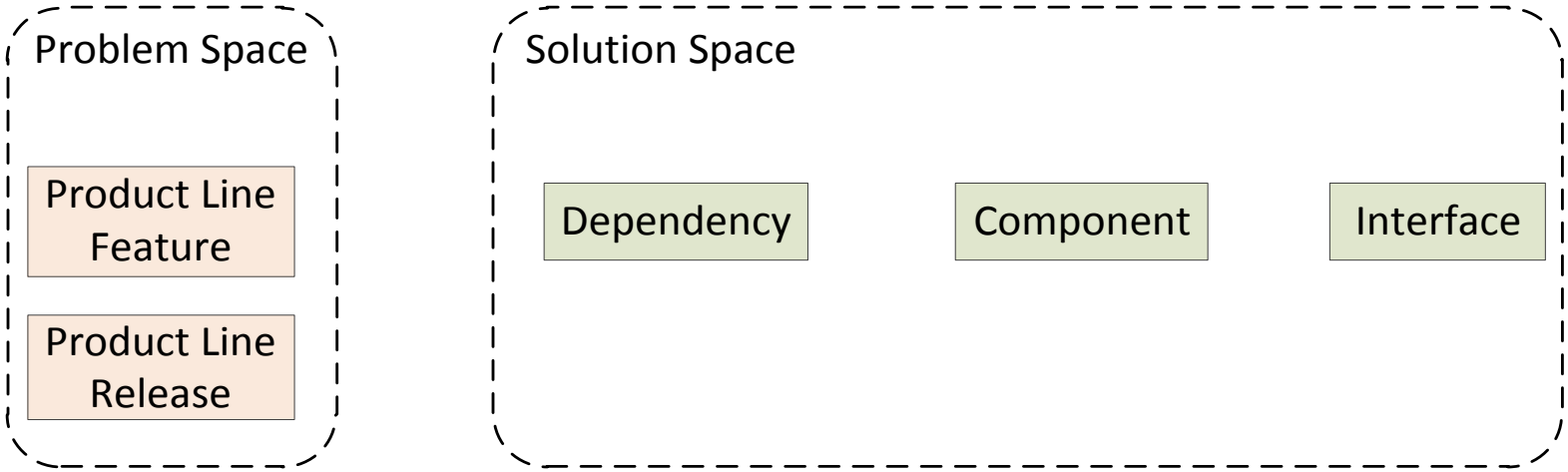
Involved Artifacts

Dependency

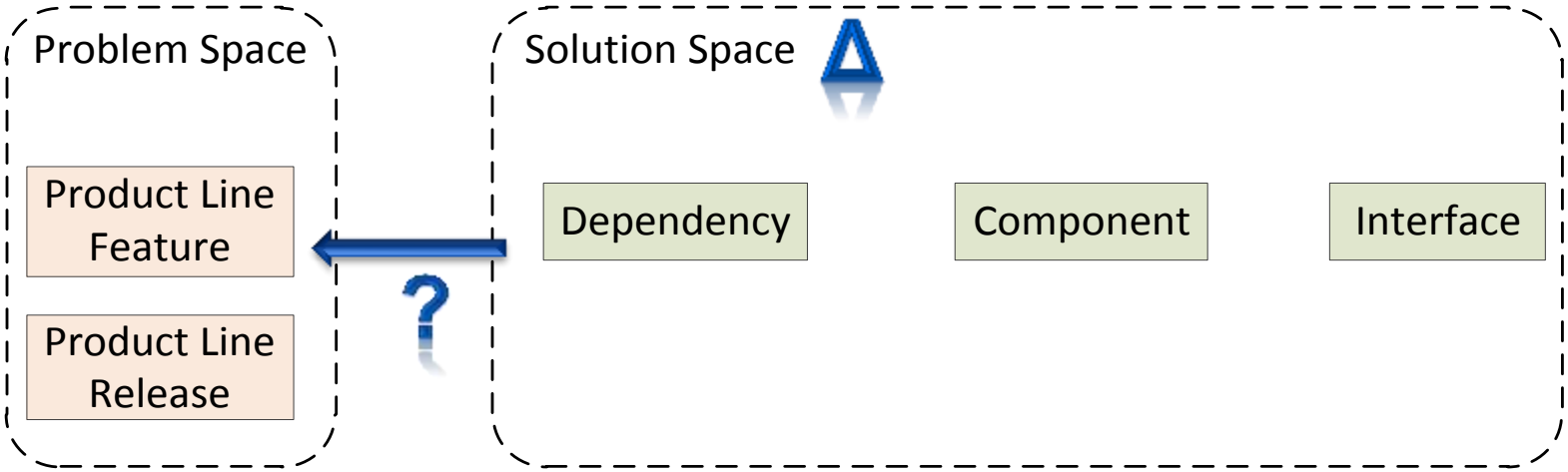
Component

Interface

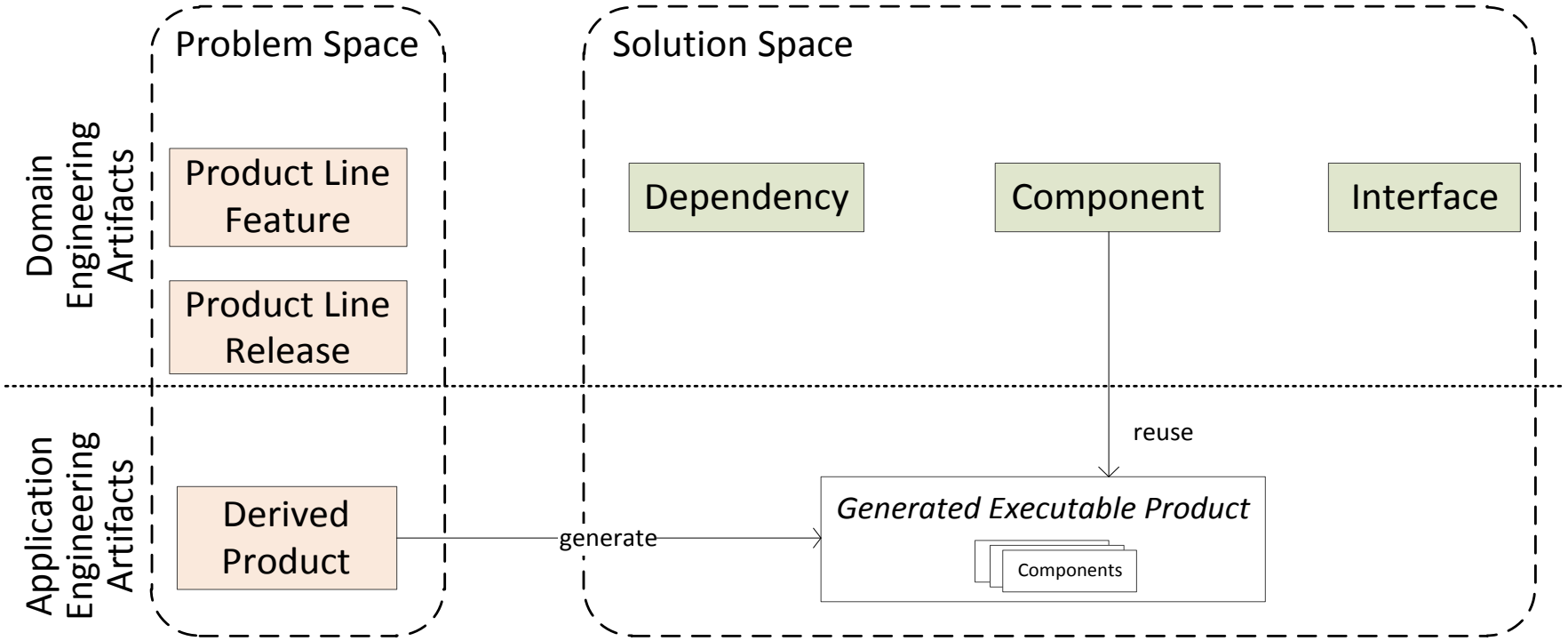
Involved Artifacts



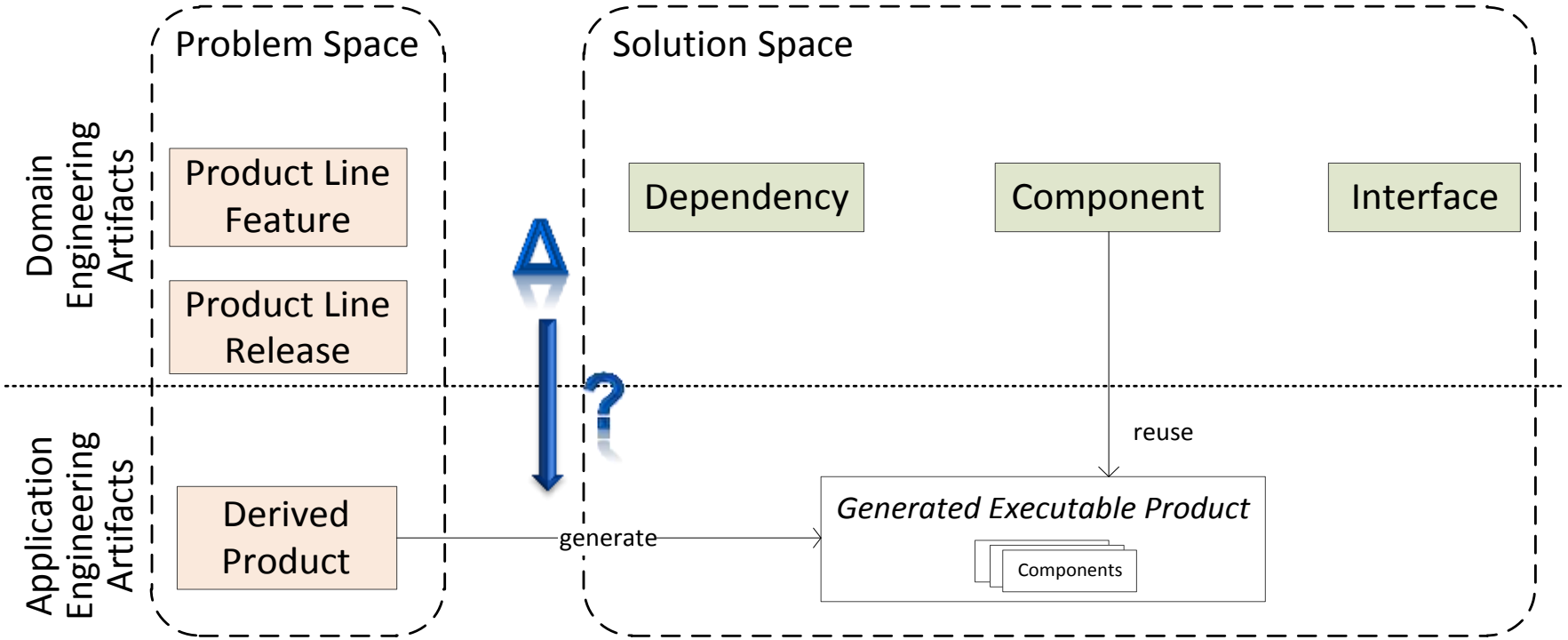
Involved Artifacts



Involved Artifacts



Involved Artifacts



Research Questions

Goal: Developer challenges and tool support requirements

RQ1: Which basic development activities are performed by developers to evolve the product line?

RQ2: Which types of change impact analyses are performed by developers?

RQ3: Which tool features are needed to perform these analyses?

The Case

SIEMENS VAI

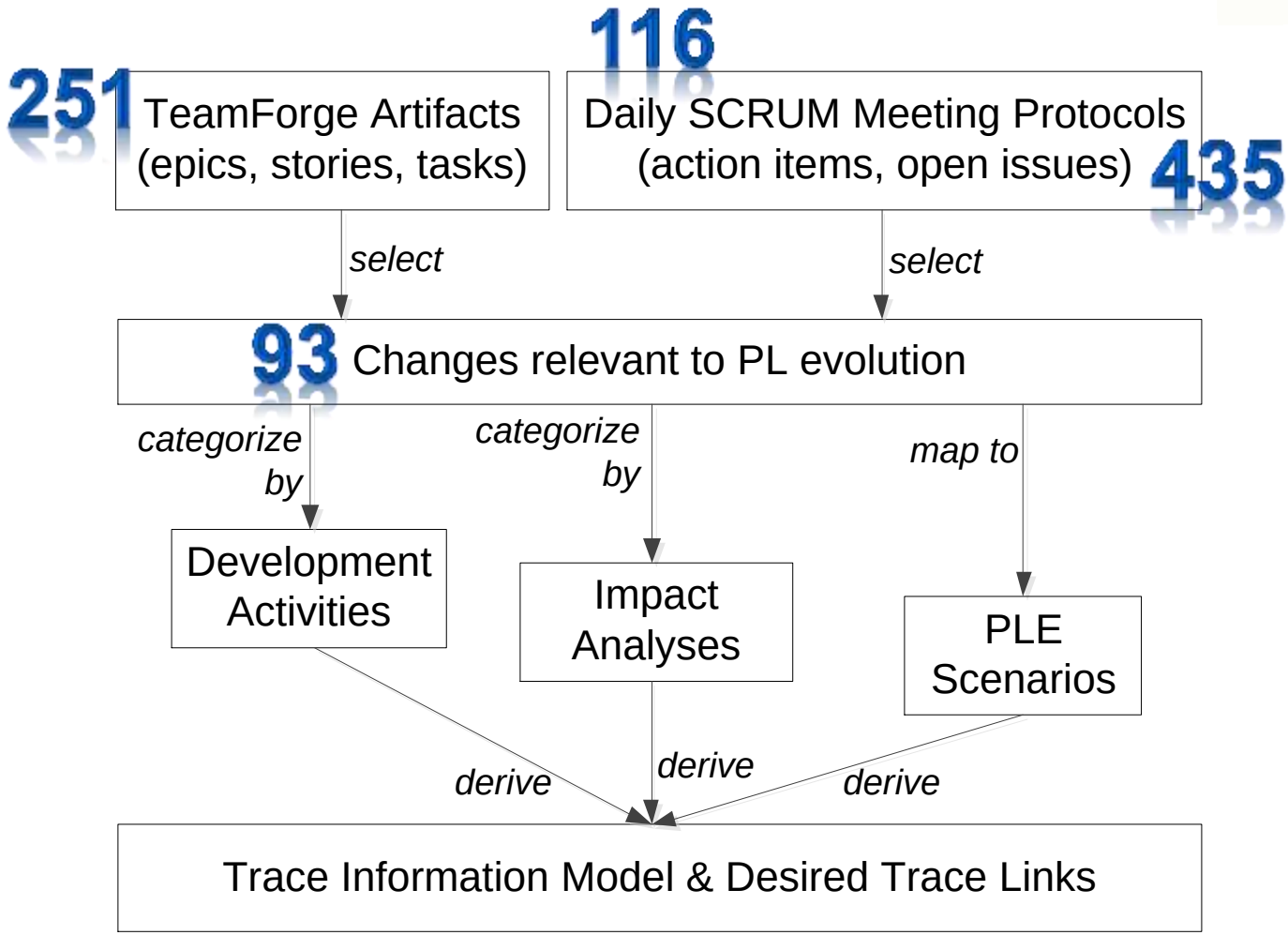
Project to Refactor the DOPLER Decision-Oriented Product Line Engineering for effective Reuse Tool Suite

… a project for transitioning the research tool to industrial settings

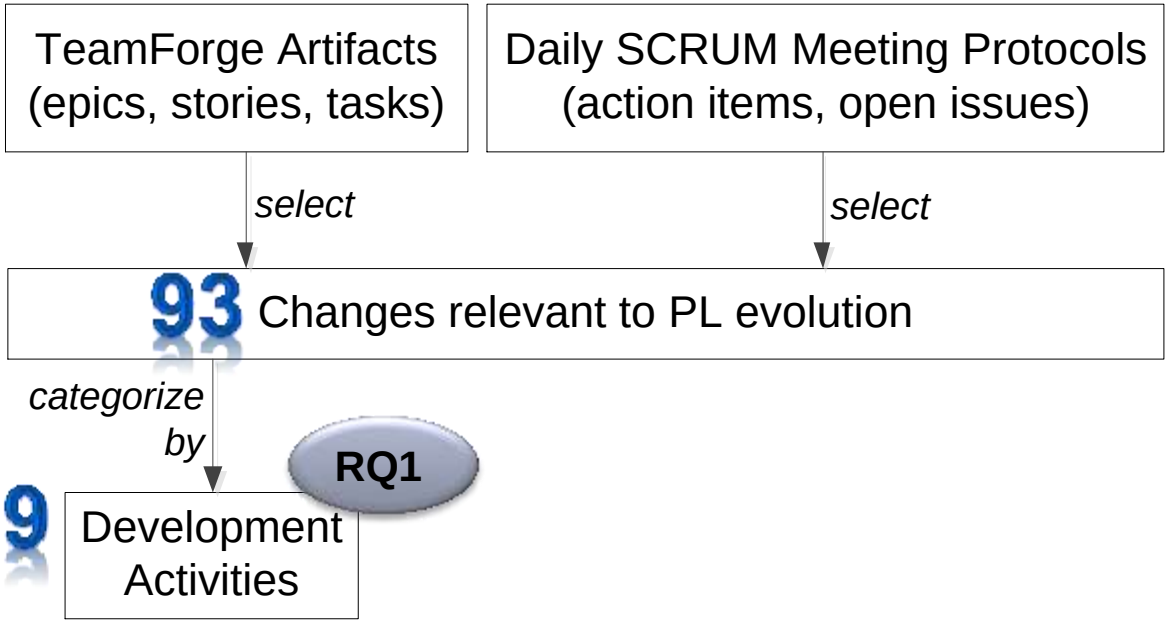
- Siemens VAI project
- Siemens managers and business users were involved as stakeholders
- 4 hired developers
- Researchers advised and observed development
- 30 person-months of professional development
- 6 derived products that need to be maintained

	v2.0	v2.10	Δ
LoC (measured with cloc 1.53)	192,427	154,924	-20%
Java Classes	1,902	1,776	-6%
Eclipse Plugins, Features	61	101	+66%

Research Process

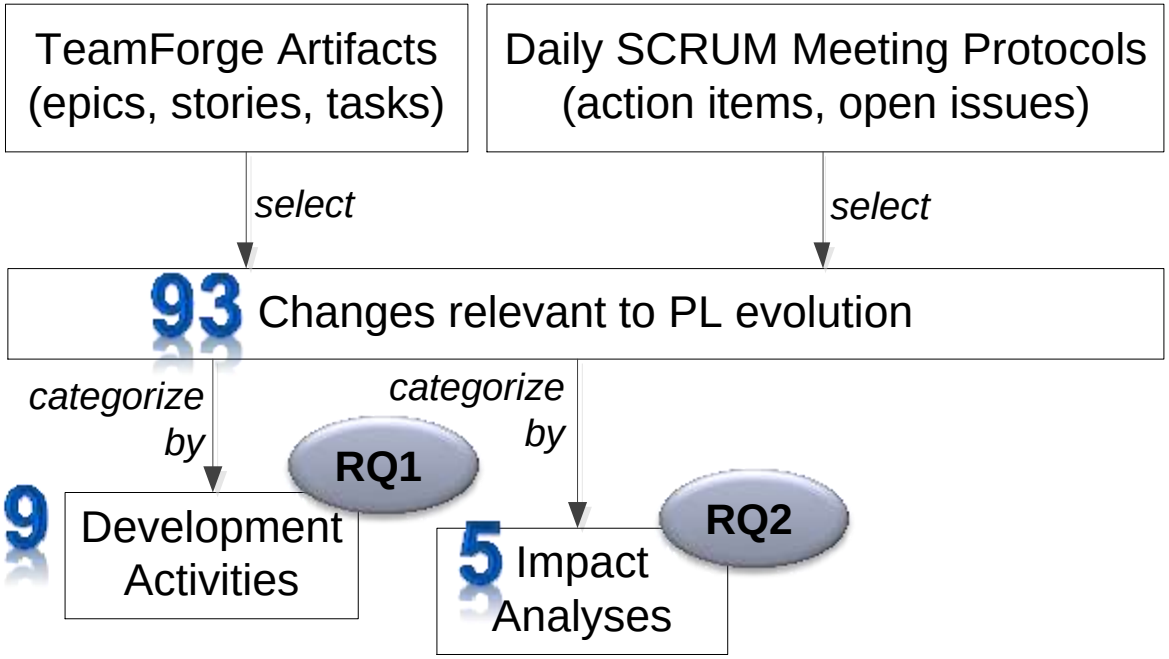


Research Process



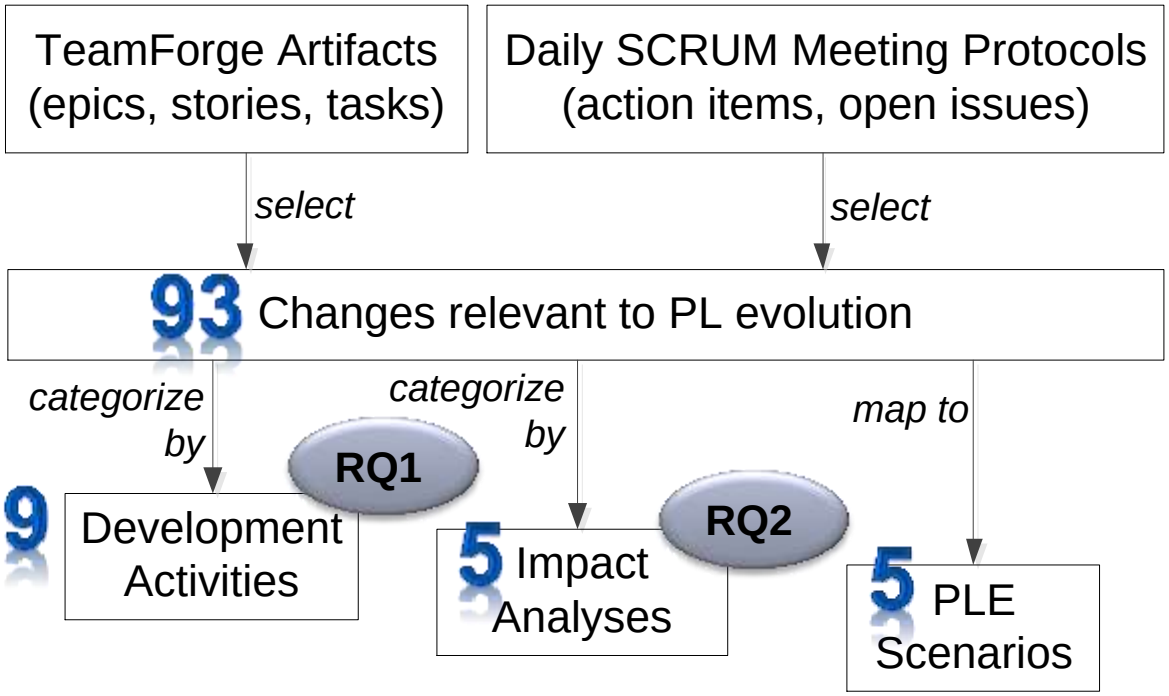
- Add / change / remove **component**
- Add / change / remove **interface**
- Add / remove **dependency**
- Define **release**

Research Process



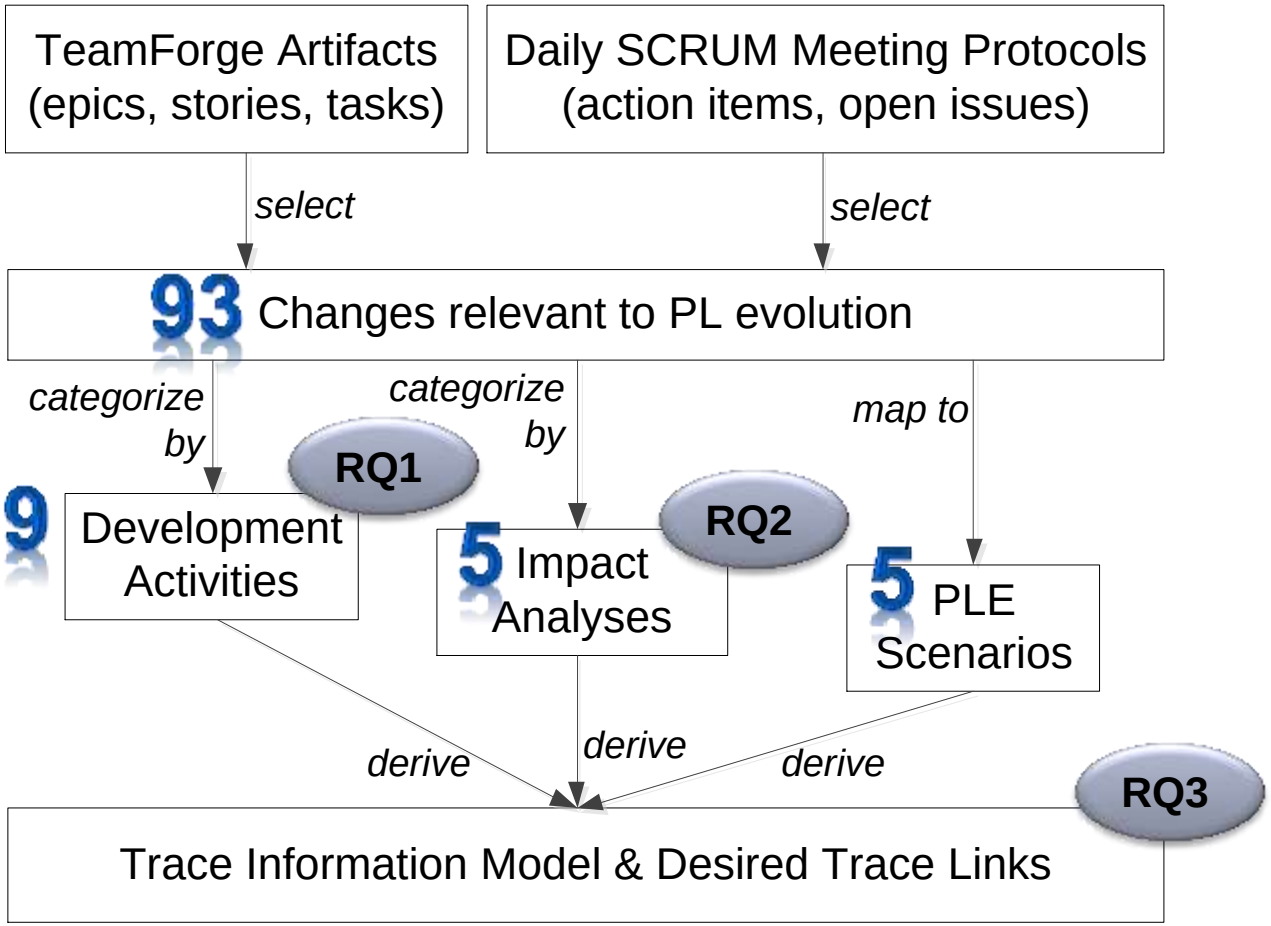
- Which **components** are affected?
- Which **interfaces** are changed?
- Which **dependencies** are added?
- Which **features** do change?
- Which **product** files need to be adapted?

Research Process



- Add feature
- Remove feature
- Define feature variability
- Maintain architecture
- Maintain assets

Research Process



Changed artifact type → Analyzed artifact type

Research Process

TeamForge Artifacts (epics, stories, tasks)	Daily SCRUM Meeting Protocols (action items, open issues)

Epic: Refactor Configuration Workflow

Story: Add additional authentication

Task: Define authentication extension point

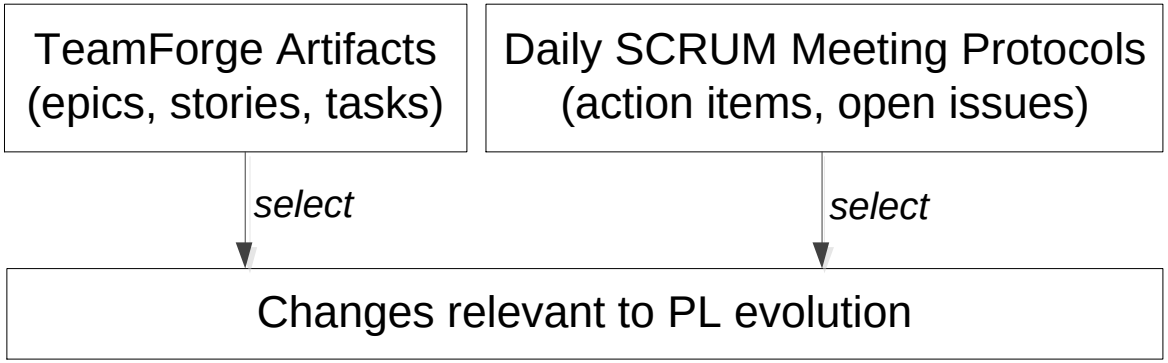
Action Item: Develop authentication interface

Open Issues: How to adapt old authentication?

Which products need which authentication?

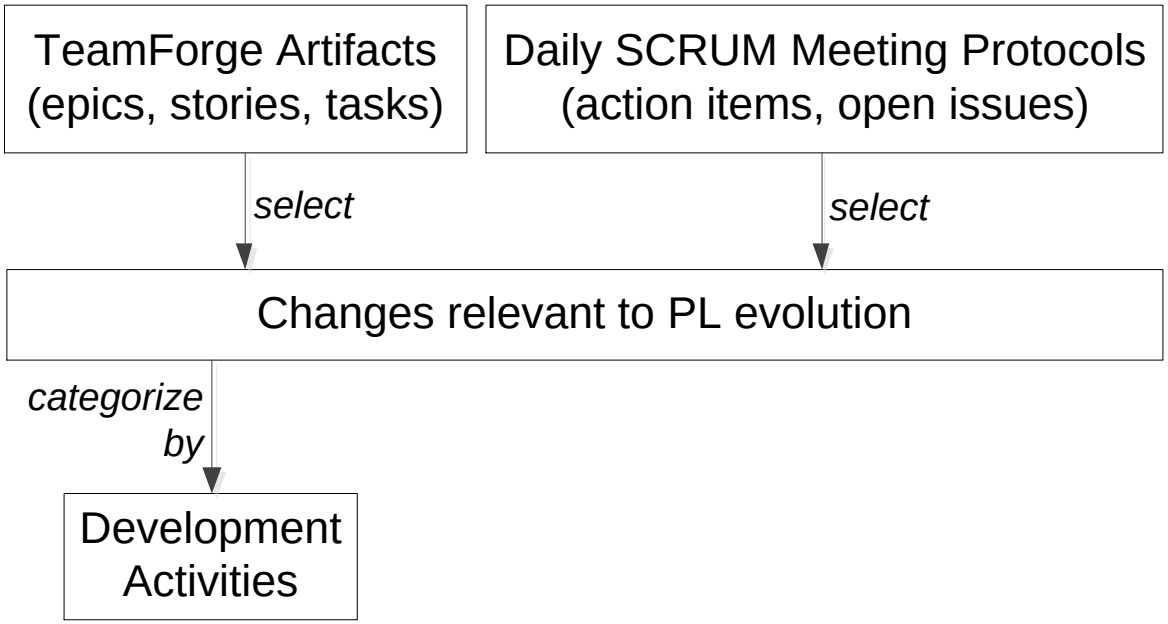
Impediment: The fu*beep* build server is offline!

Research Process



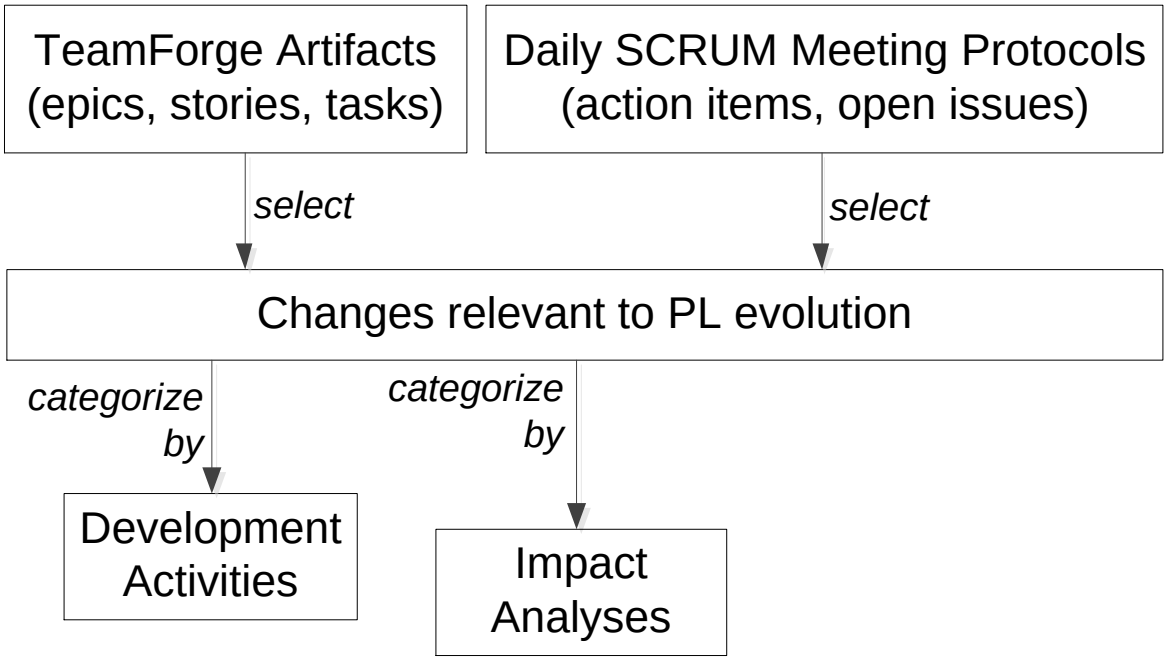
Add Smartcard Authentication

Research Process



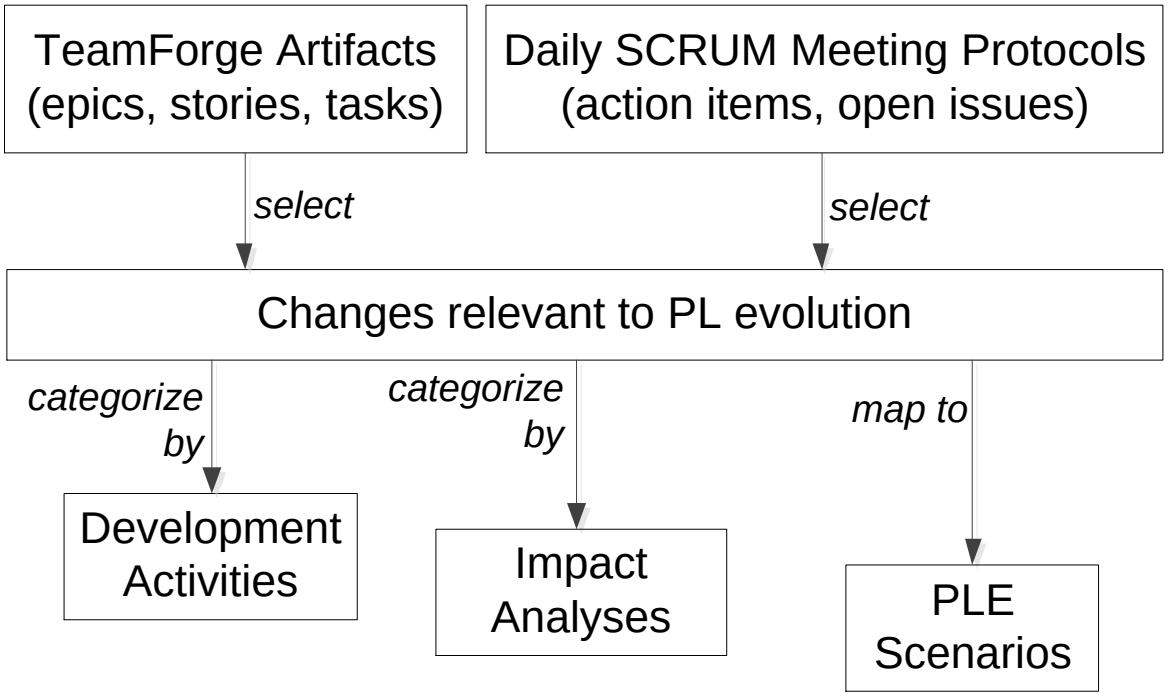
- **Change** configuration UI **plugin**
- **Add** authentication **interface**
- **Add** authentication **plugins**

Research Process



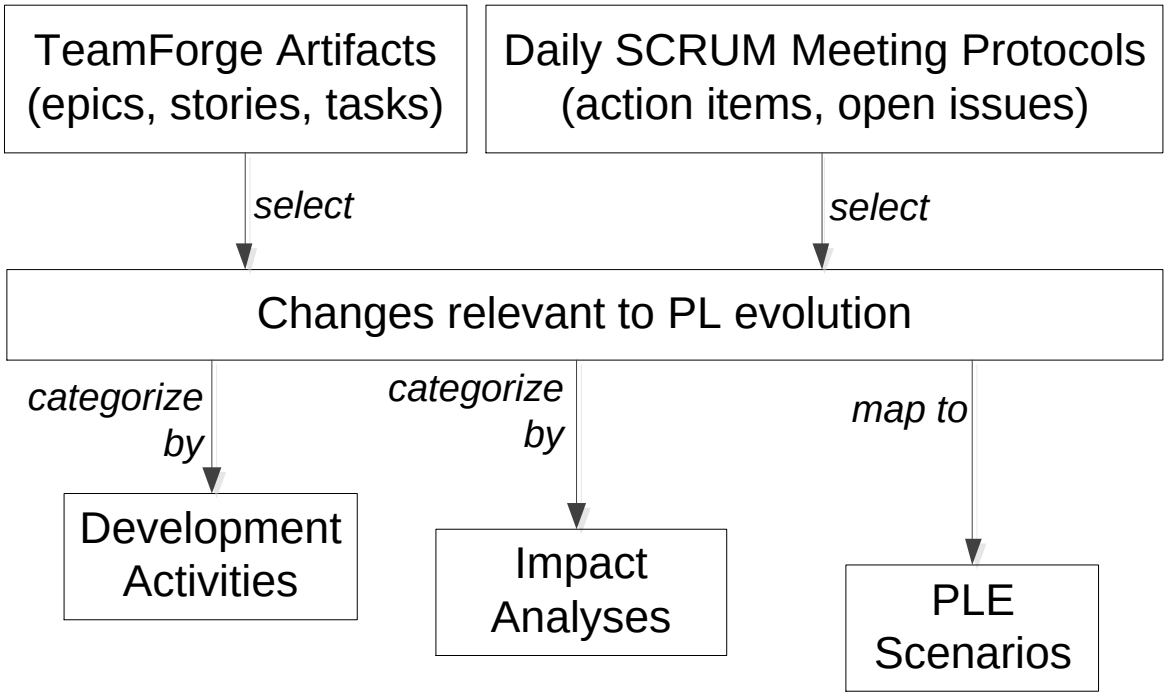
- Which **plugins** are affected by the new authentication?
- Which **product** files need to be adapted to include the new authentication?

Research Process



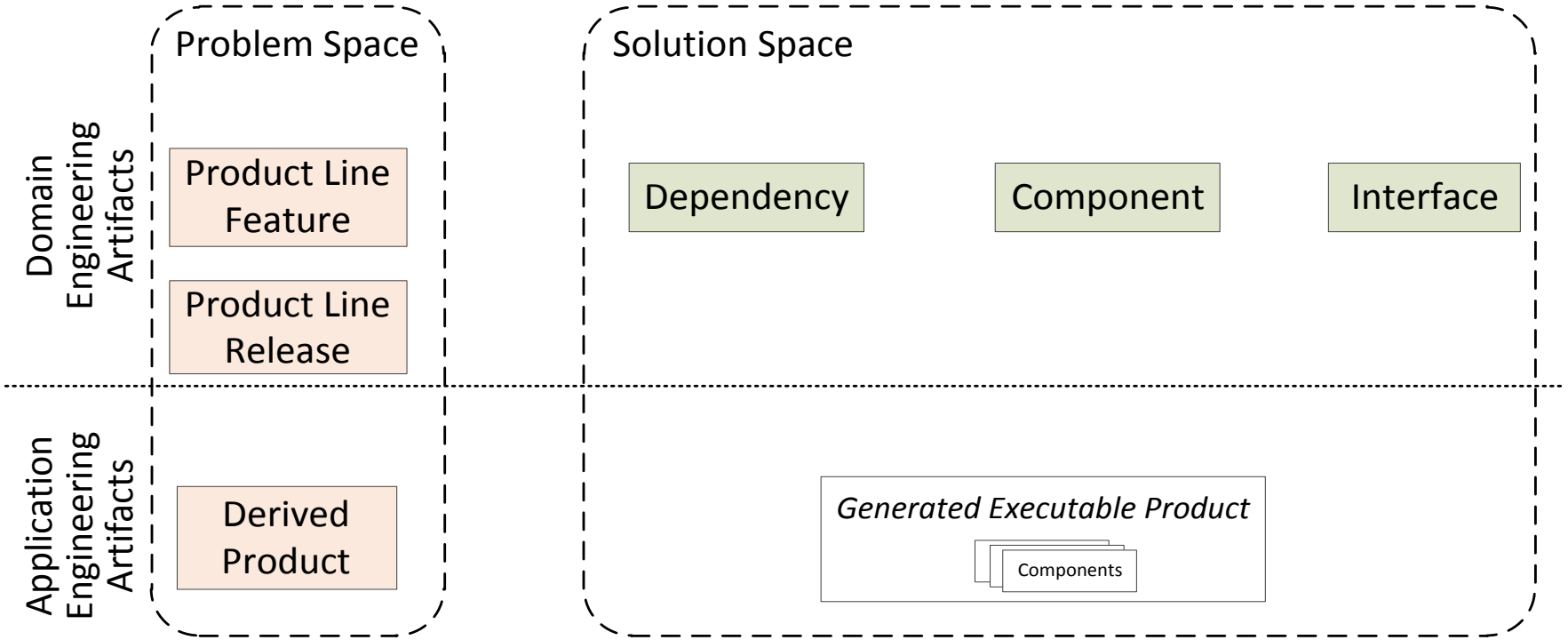
- Add authentication feature
- Define authentication variability

Research Process

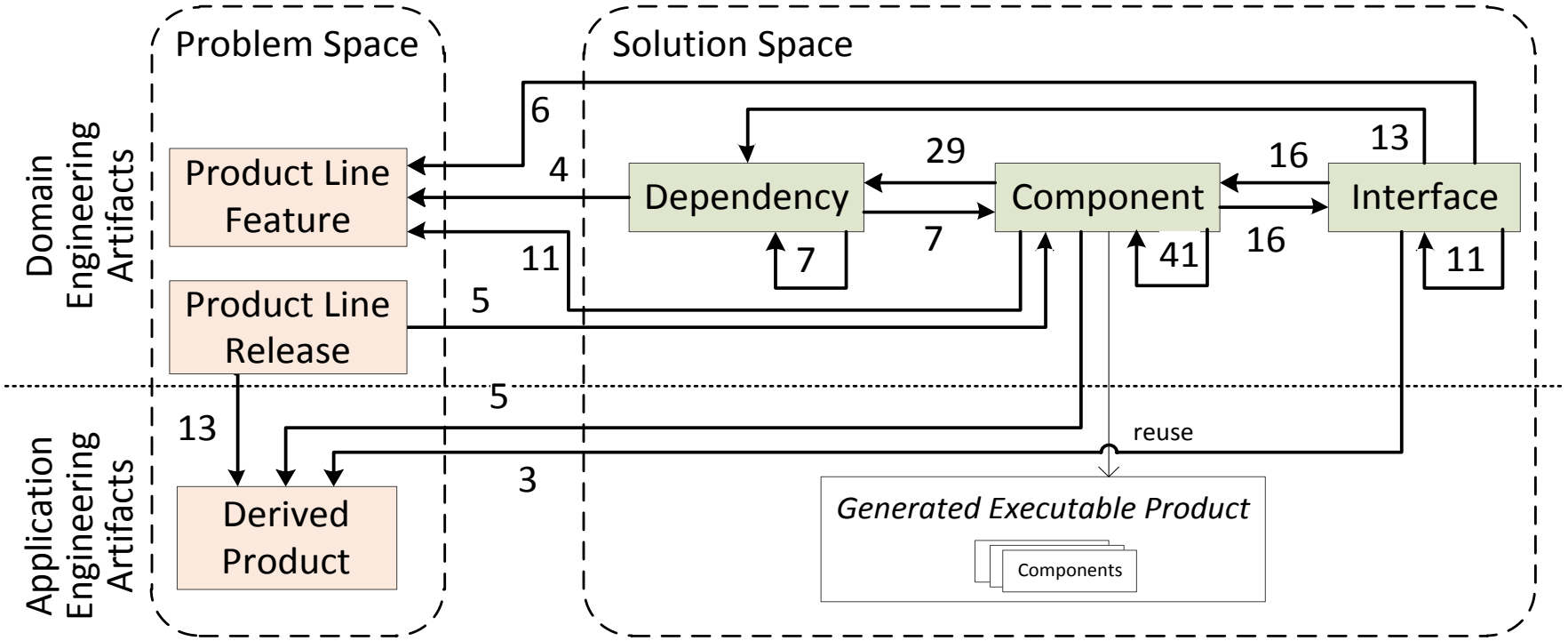


While defining feature variability,
the configuration UI **plugin** was **changed**
and developers analyzed which **product** files need to be adapted?

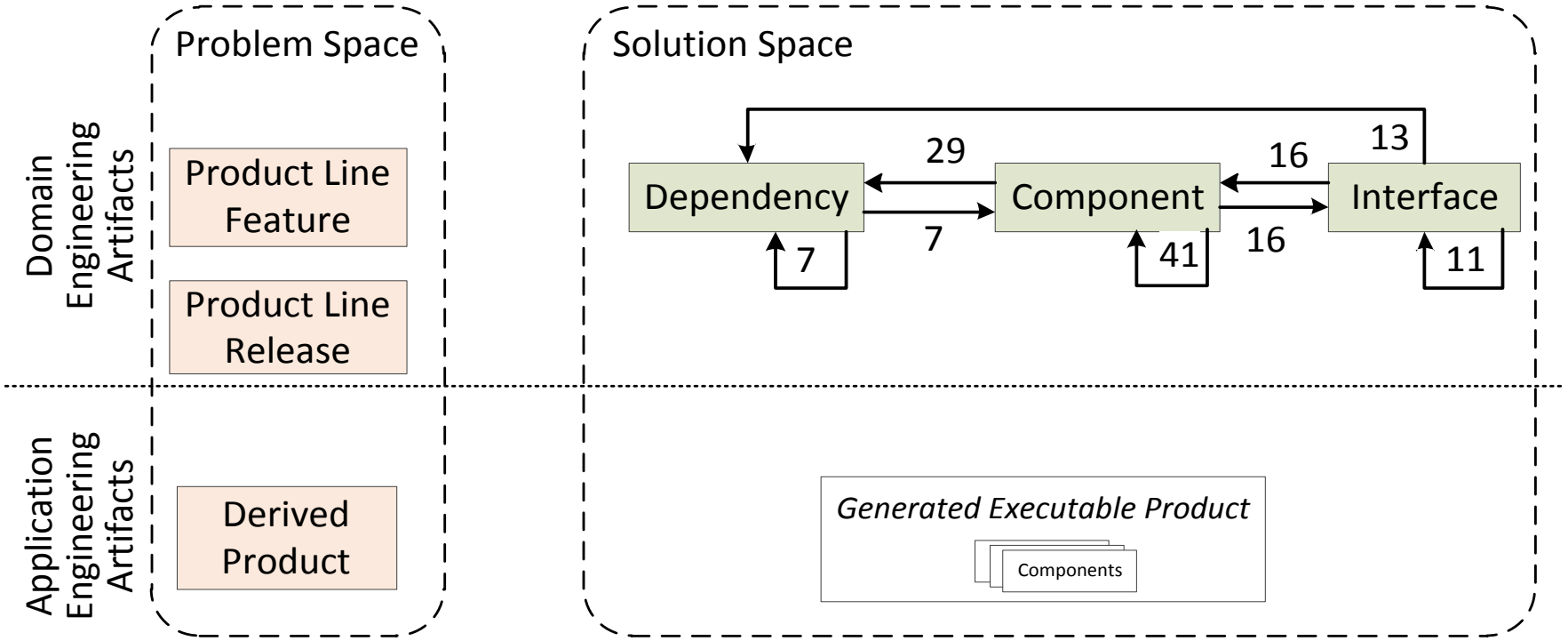
Resulting Trace Information Model



Resulting Trace Information Model

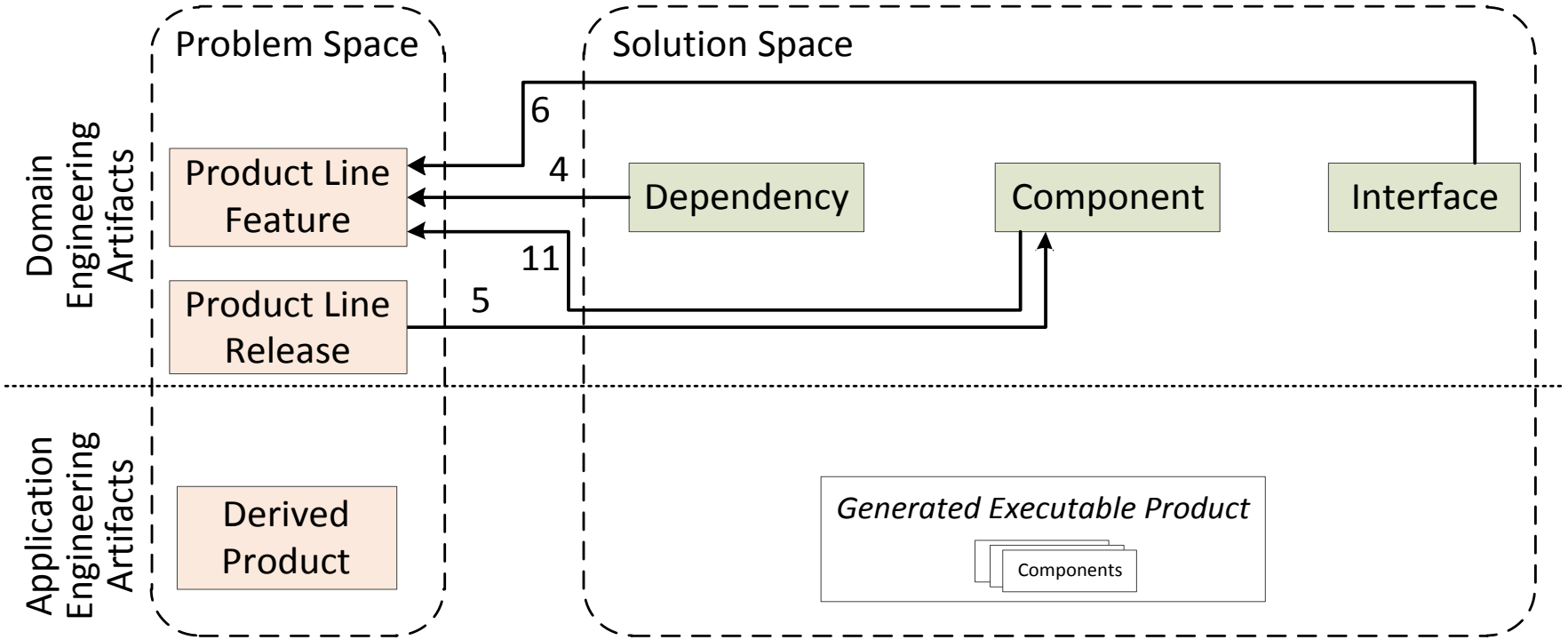


Resulting Trace Information Model



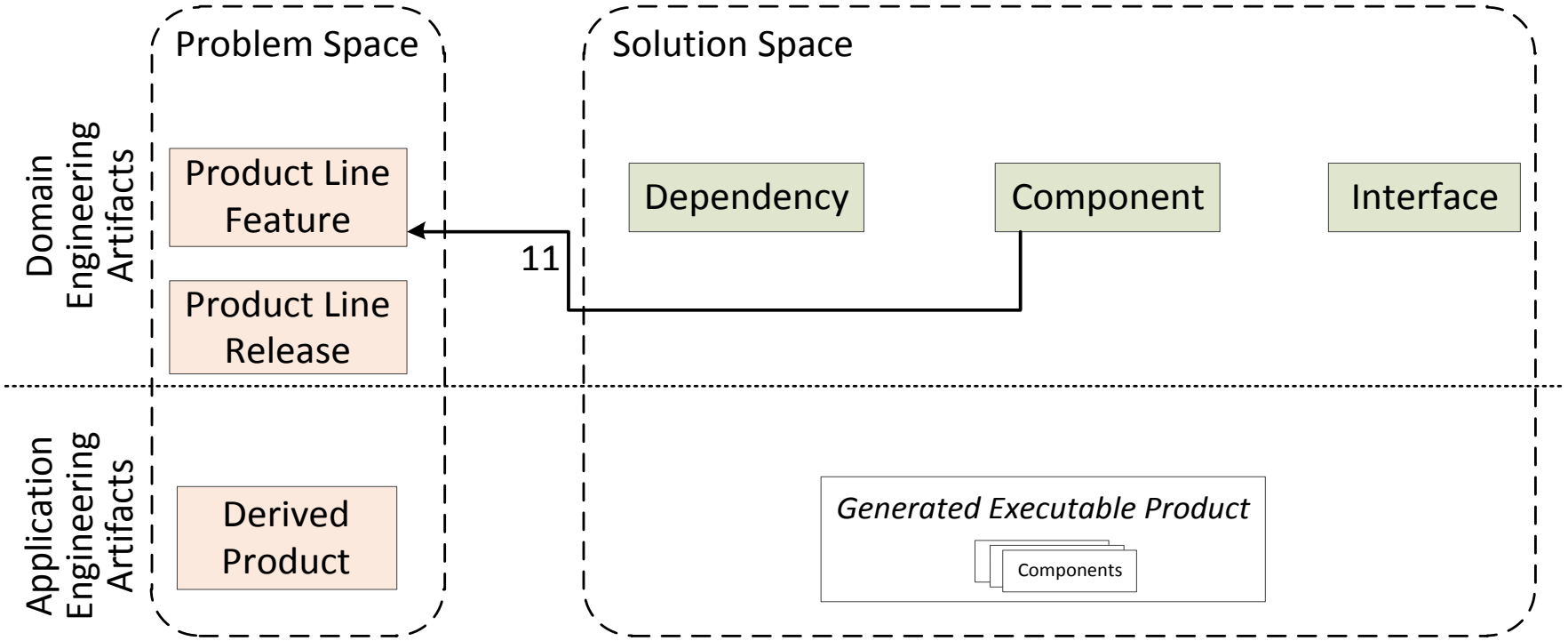
Resulting Trace Information Model

→ Developers need to see the effects of changes on features and feature variability



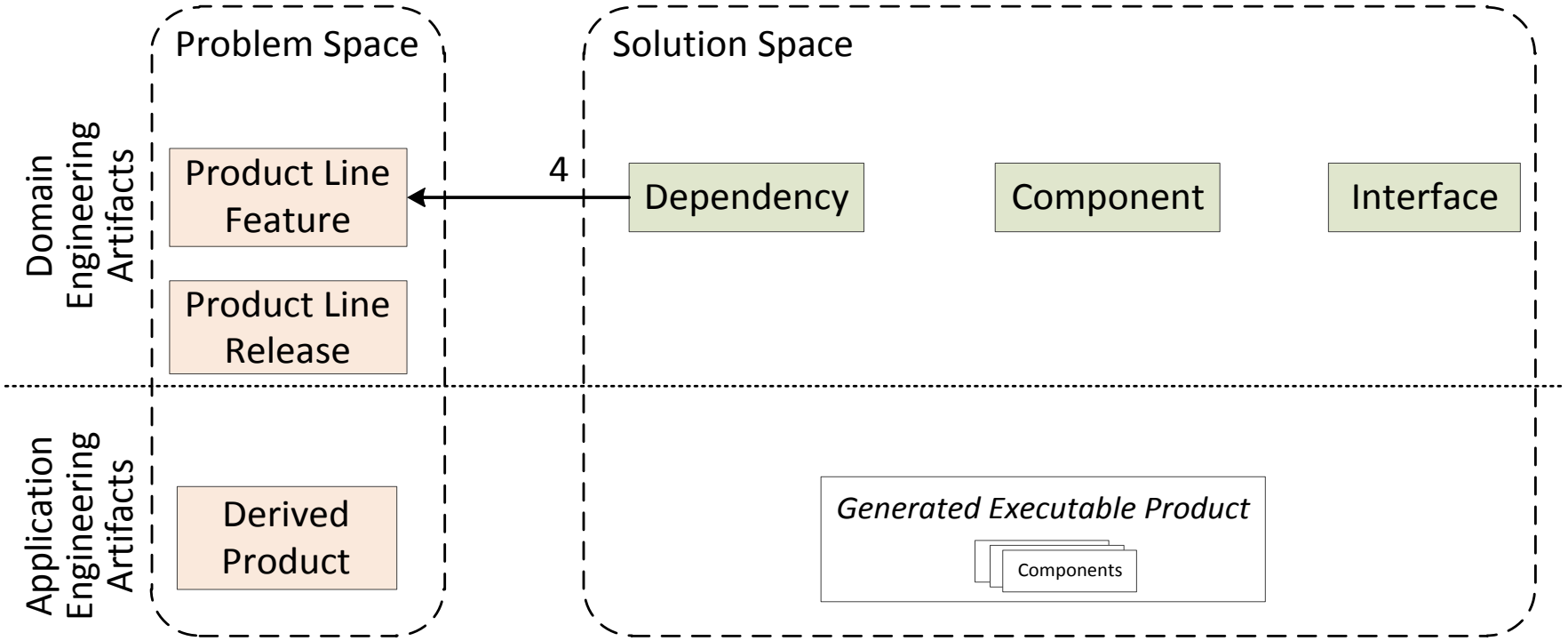
Resulting Trace Information Model

The tool presents features related to a component under change.



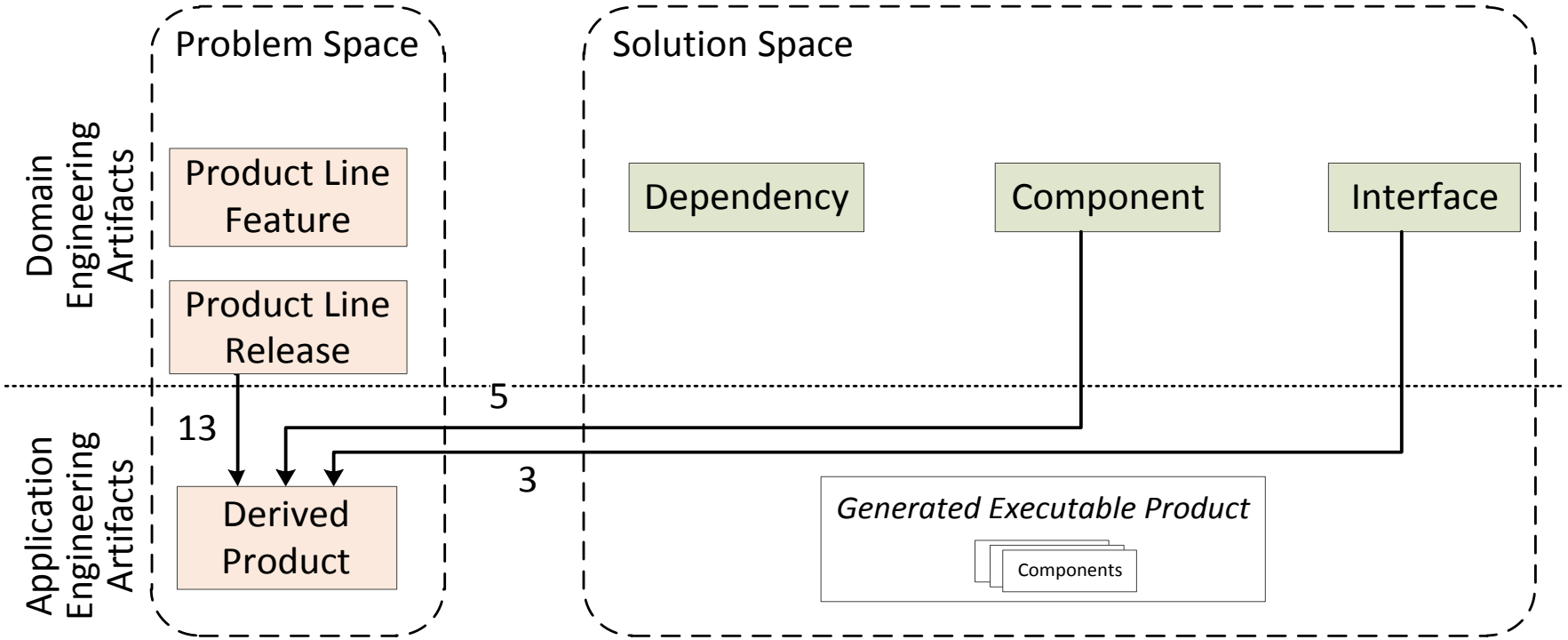
Resulting Trace Information Model

The tool presents affected features and violated feature variability (if dependencies between components are added or removed).



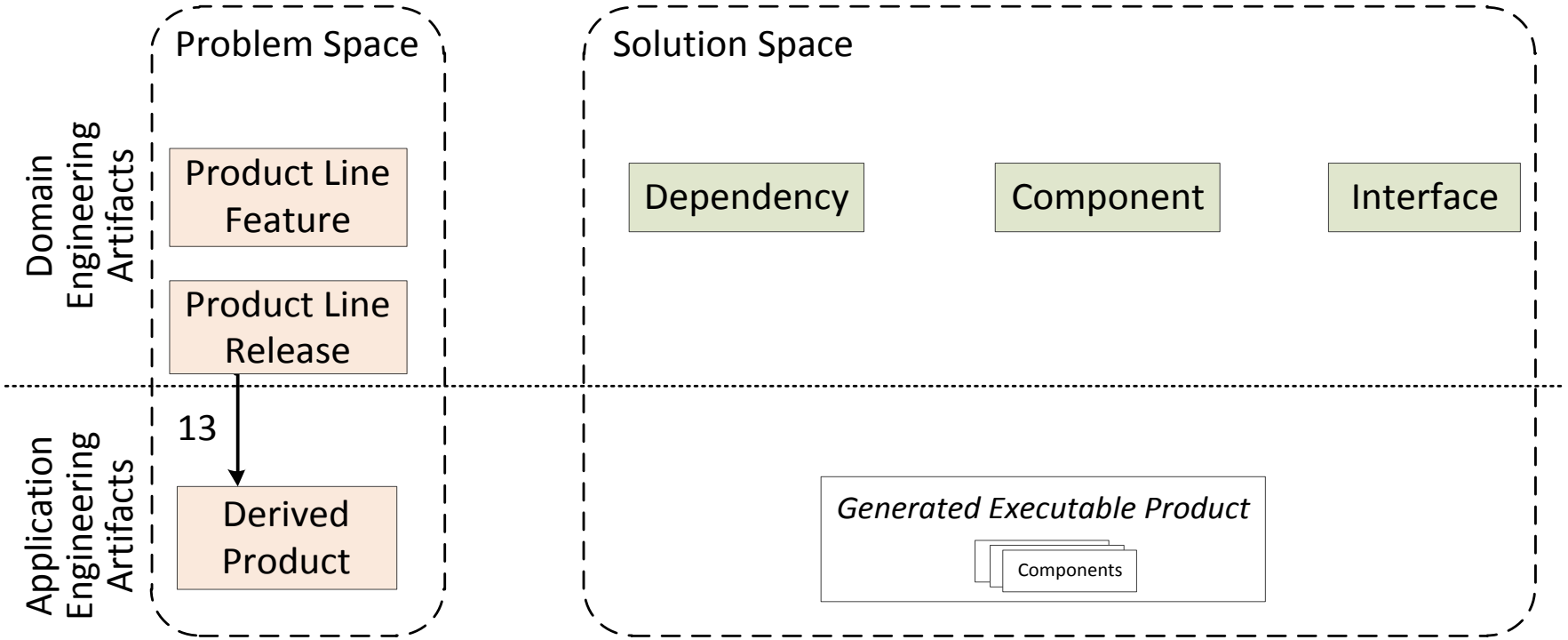
Resulting Trace Information Model

→ Developers need to see the impact of changes on derived products



Resulting Trace Information Model

The tool presents already derived products or ongoing product derivations based on a previous PL release to analyze the possibility of product updates.



Summary / Future work

Before we developed approaches and solutions …

- We conducted a case study to find industrial challenges
- We observed the evolution of an industrial product line
- We derived scenarios of impact analysis as requirements

- Our approach to provide impact analysis: Regression testing with variability models [Heider et al. @SPLC2012]

THANK YOU!