COLLABORATIVE MODEL DEVELOPMENT FOR SYSTEM SIMULATION

Andreas Erbes, Dirk Frerichs, Stefan Sinsel, Jochen Zäpf

Groupe PSA - Opel Automobile GmbH

XiL Simulation & Software Test Methods

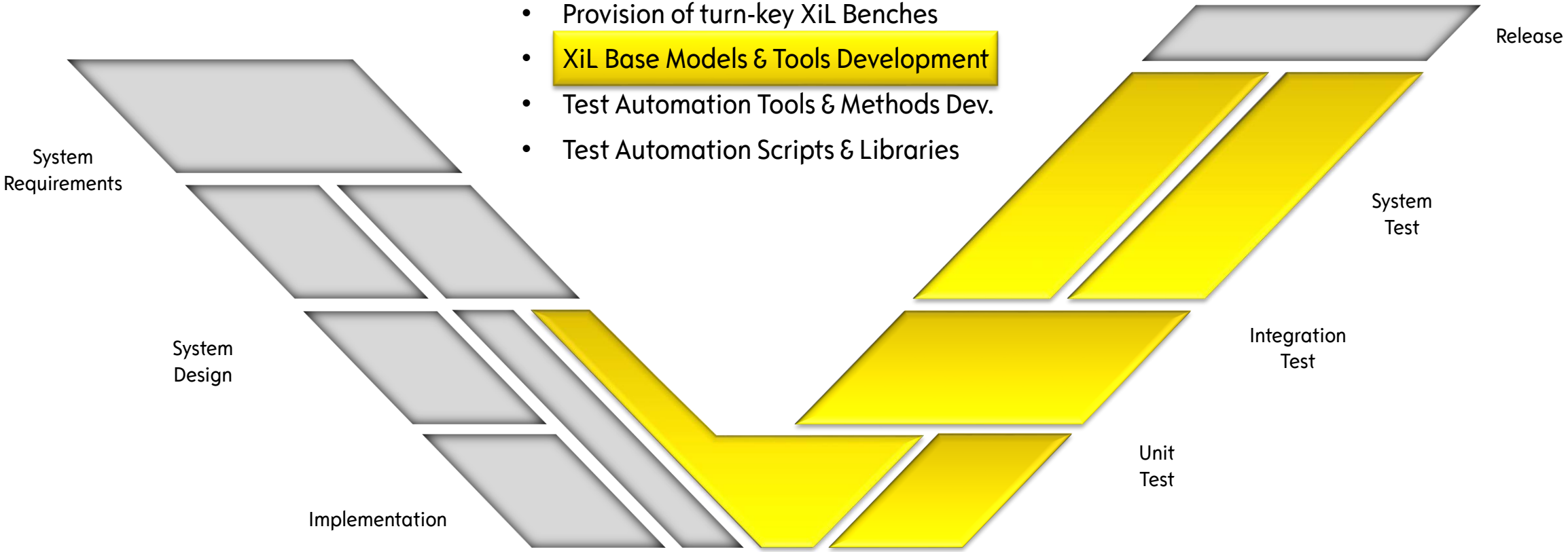
Stuttgart, MathWorks AUTOMOTIVE CONFERENCE 2019 EUROPE, 11. April 2019

SELF INTRODUCTION

Department: Controls Development & Validation

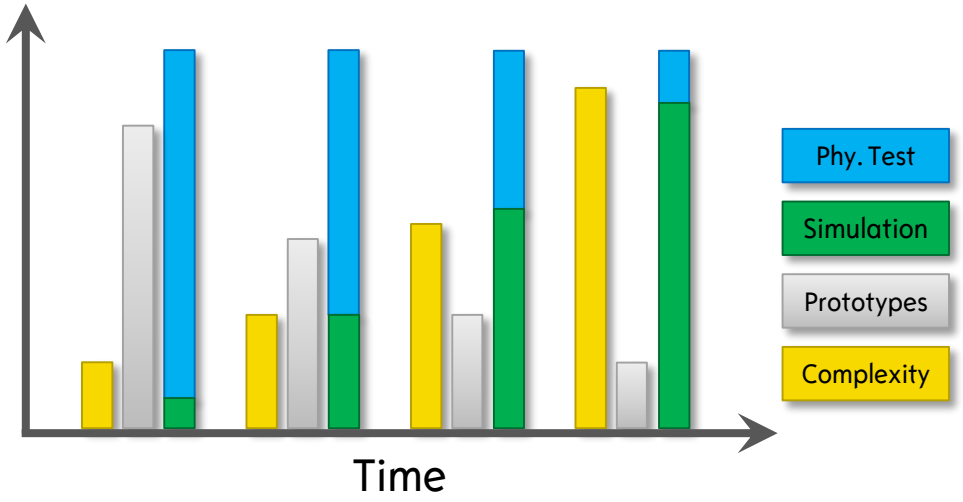
Team: XiL Simulation & Software Test Methods

- Provision of turn-key XiL Benches
- XiL Base Models & Tools Development
- Test Automation Tools & Methods Dev.
- Test Automation Scripts & Libraries



WHAT IS THE CHALLENGE TO SYSTEM SIMULATION?

- Dramatically increasing system complexity
- Reduction of development costs
- Strong move towards virtual development methods



→ Simulation based engineering is getting more and more important

→ Collaboration between departments becomes a prerequisite

→ Common fundament for model development, methods & tools

→ **Need of collaborative Simulation Framework**

AGENDA

- What is a Simulation Framework?
- Characteristics of a collaborative framework?
- Modular system modeling approach
- Practical examples for model integration
  - Model Interface Management
  - Model Configurator
- Summary

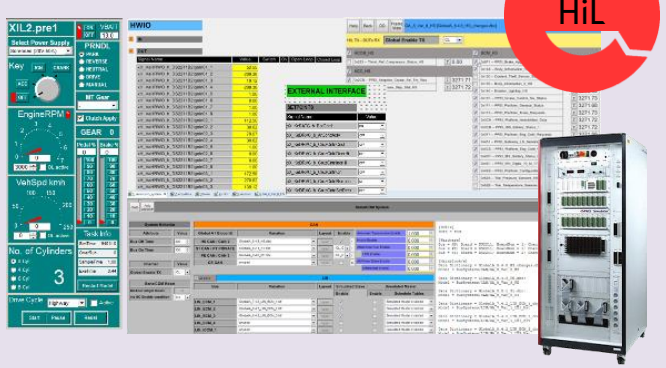
SYSTEM SIMULATION APPROACH

Turn-key Application Models

MIL Models

SIL Models

HIL Models



Which way to choose?

Base Software

MATLAB® SIMULINK®

dSPACE

LMS® ENGINEERING INNOVATION

IPG AUTOMOTIVE

git

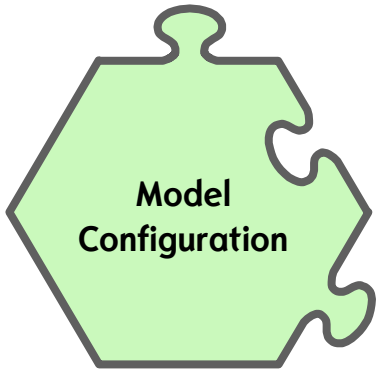
GT Gamma Technologies

ECU-TEST

GitHub

WHAT IS A SIMULATION FRAMEWORK?

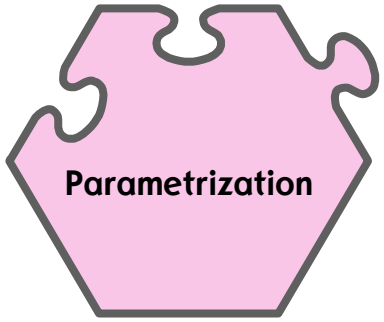
A framework is a puzzle of solutions for various disciplines



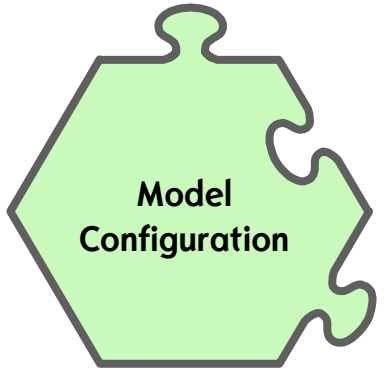
- Library concept
- Model integration
- Variant handling

WHAT IS A SIMULATION FRAMEWORK?

A framework is a puzzle of solutions for various disciplines

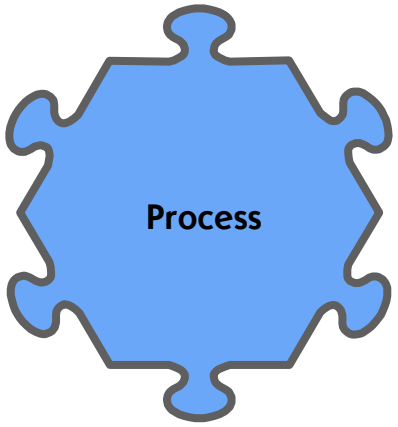


- Parameter initialization
- Definition of tunable parameters
- Parameter inheritance
- Maintenance of meta data

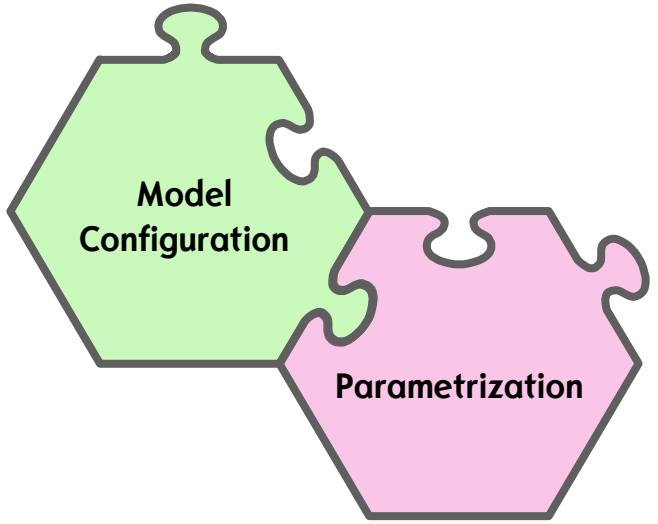


WHAT IS A SIMULATION FRAMEWORK?

A framework is a puzzle of solutions for various disciplines

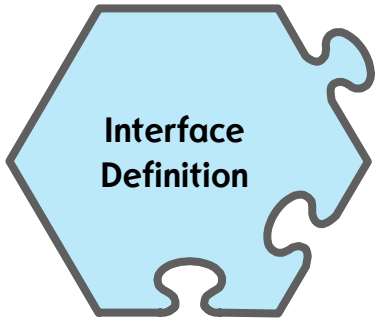


- Following agile principles
- Git for version control
- JIRA for planning & issue tracking
- Continuous Integration & Testing

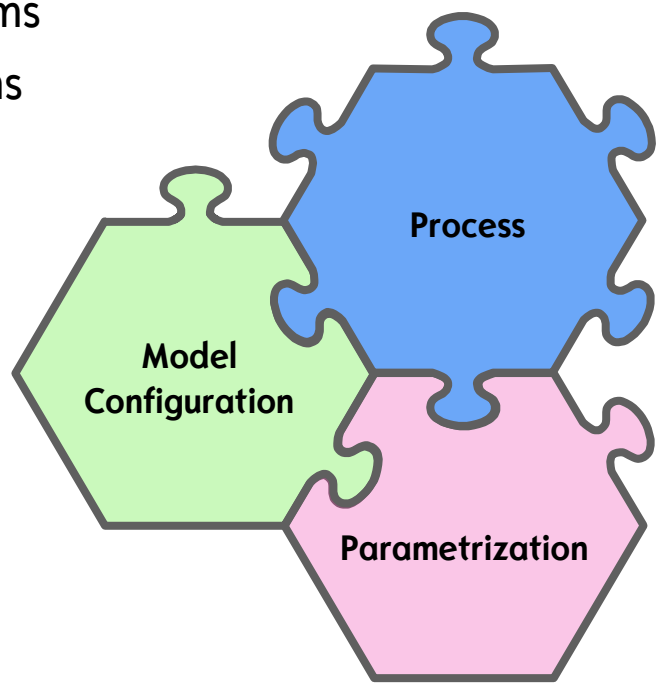


WHAT IS A SIMULATION FRAMEWORK?

A framework is a puzzle of solutions for various disciplines

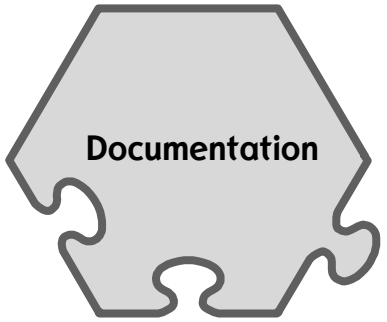


- Standards for model interface (e.g. FMU/FMI) and co-simulation methods
- Interface to external test automation tools
- Standard interfaces to RCP/HIL systems
- Data exchange with PLM/ALM systems

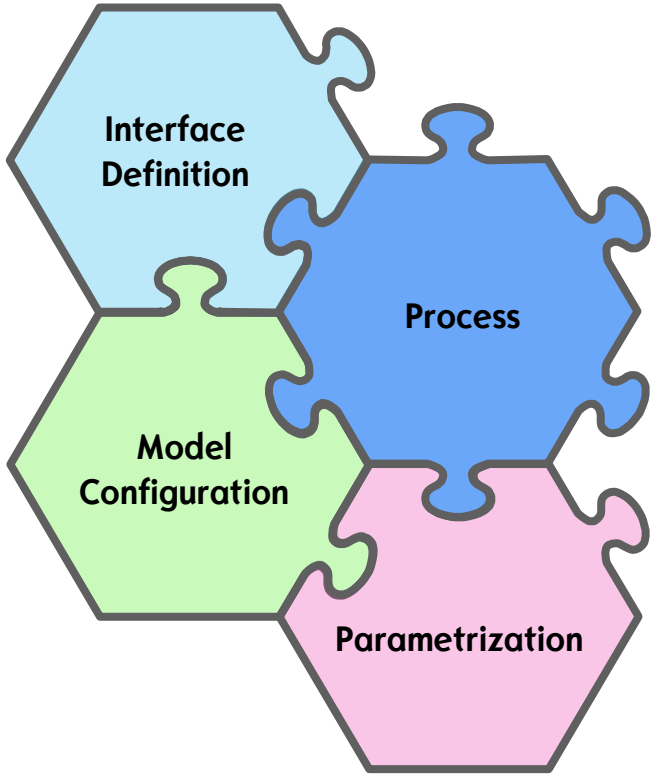


WHAT IS A SIMULATION FRAMEWORK?

A framework is a puzzle of solutions for various disciplines

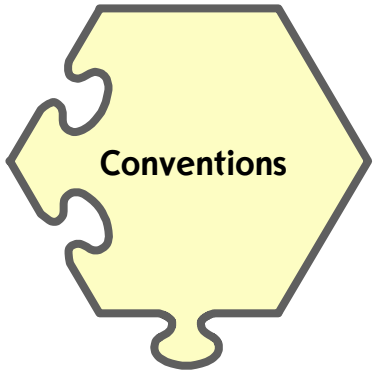


- Common documentation for tools & models

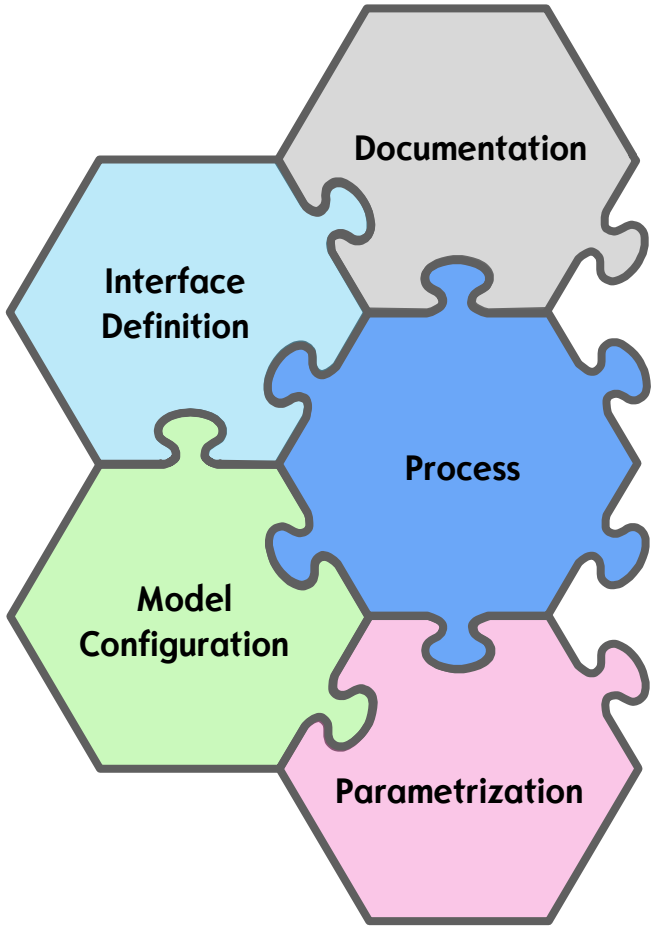


WHAT IS A SIMULATION FRAMEWORK?

A framework is a puzzle of solutions for various disciplines

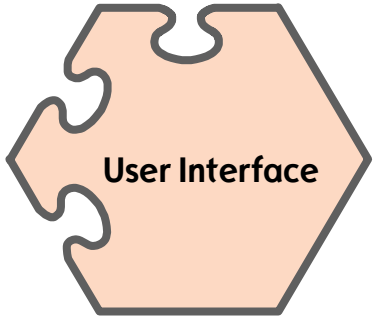


- Naming convention
- Modeling rules & style guides
- MAAB Standard

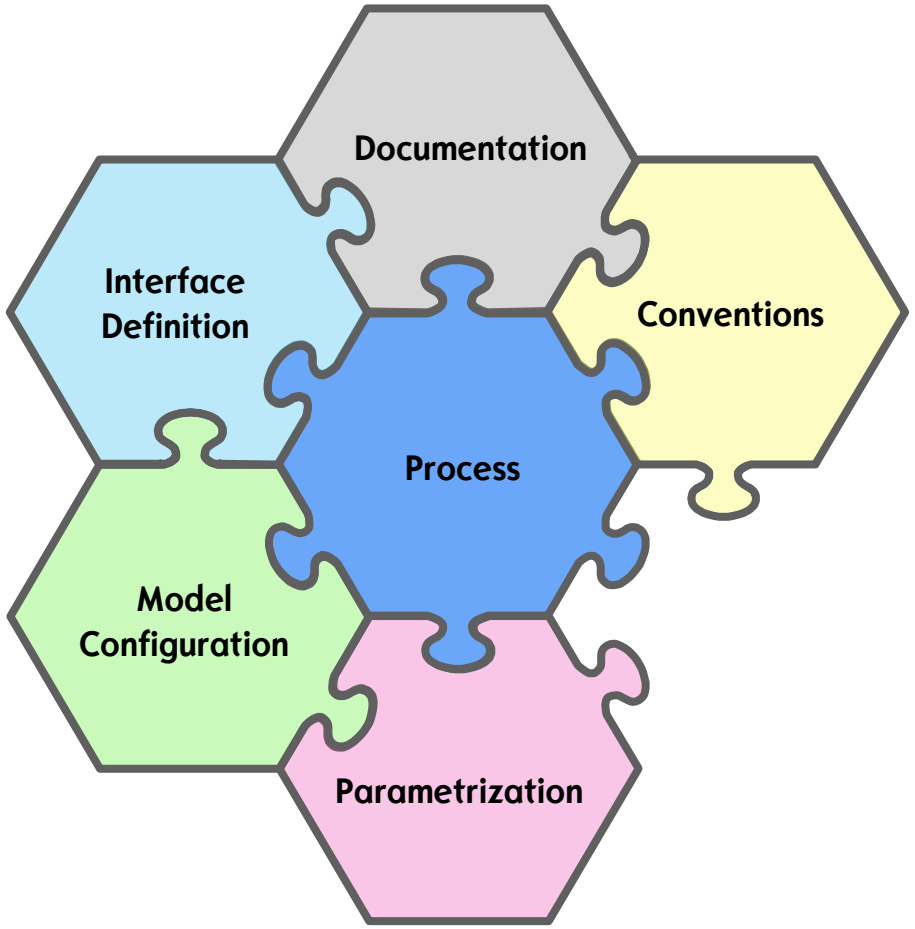


WHAT IS A SIMULATION FRAMEWORK?

A framework is a puzzle of solutions for various disciplines



- Common Look & Feel
- Automatic UI generation



WHAT IS A SIMULATION FRAMEWORK?

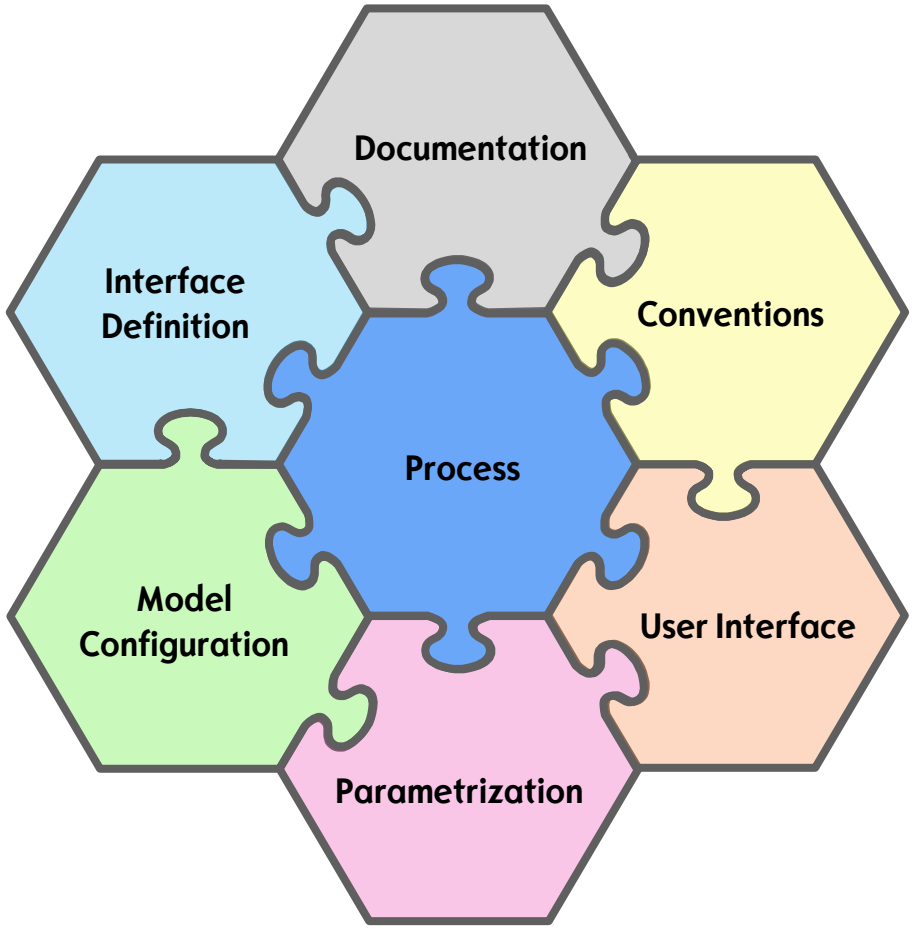
A framework is a puzzle of solutions for various disciplines

It's not the What

It's the How to ...

It's not the content

It's the method



COLLABORATIVE FRAMEWORK

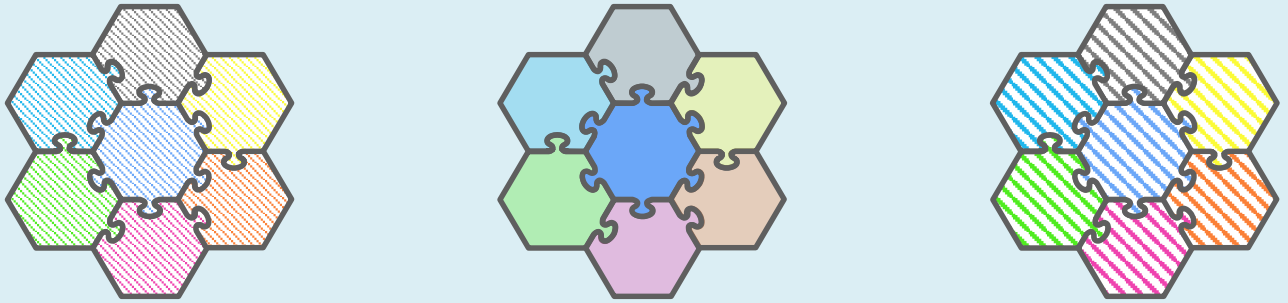
Turn-key Application Models
(with different purpose)

Powertrain | **ADAS** | **Alternative Propulsion**

MiL, SiL, HiL

Various Simulation Frameworks
(tailored to purpose)

Powertrain | ADAS | Alternative Propulsion



and more

Base Software

MATLAB® SIMULINK®

dSPACE | LMS® ENGINEERING INNOVATION | IPG AUTOMOTIVE | git

GT Gamma Technologies | ECU-TEST | GitHub

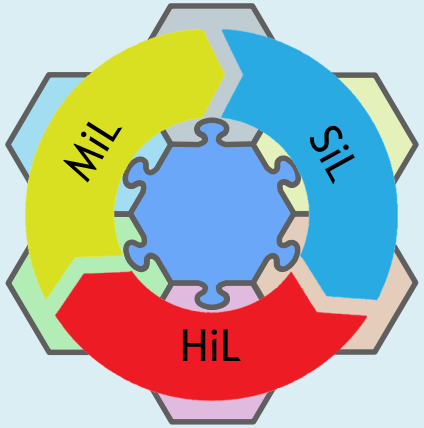
COLLABORATIVE FRAMEWORK

Turn-key Application Models
(with different purpose)

Powertrain **ADAS** **Alternative Propulsion**

Common Simulation Framework
(across various domains and departments)

Automotive
XIL
Objectoriented
Modelframework



Base Software

MATLAB SIMULINK
dSPACE
LMS ENGINEERING INNOVATION
IPG AUTOMOTIVE
git
GT Gamma Technologies
ECU-TEST
GitHub

WHAT ARE THE GOALS OF AXIOM?

Coordinated development of new features to avoid multiple decoupled solutions

Improved speed & robustness of model integration

Management & Traceability of Models

Reusability of models, tools and conventions

Efficient Maintainability of Models

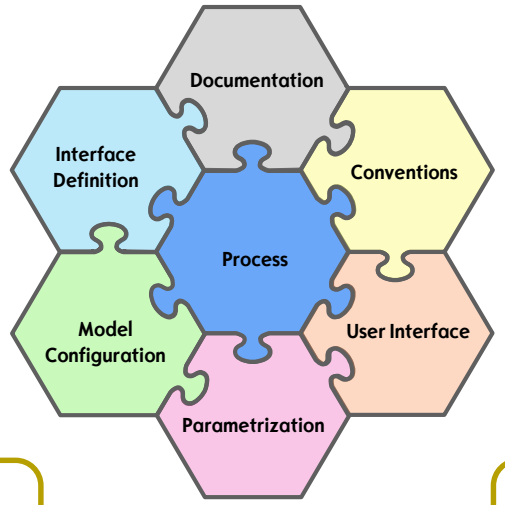
Quick deployment of new features

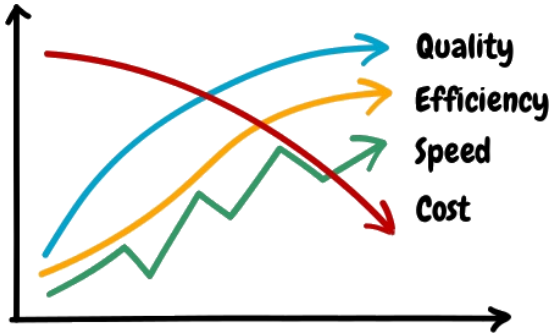
Common „Look & Feel“ for model users

Standardized model access for test automation

Share deliverables between different teams & regions

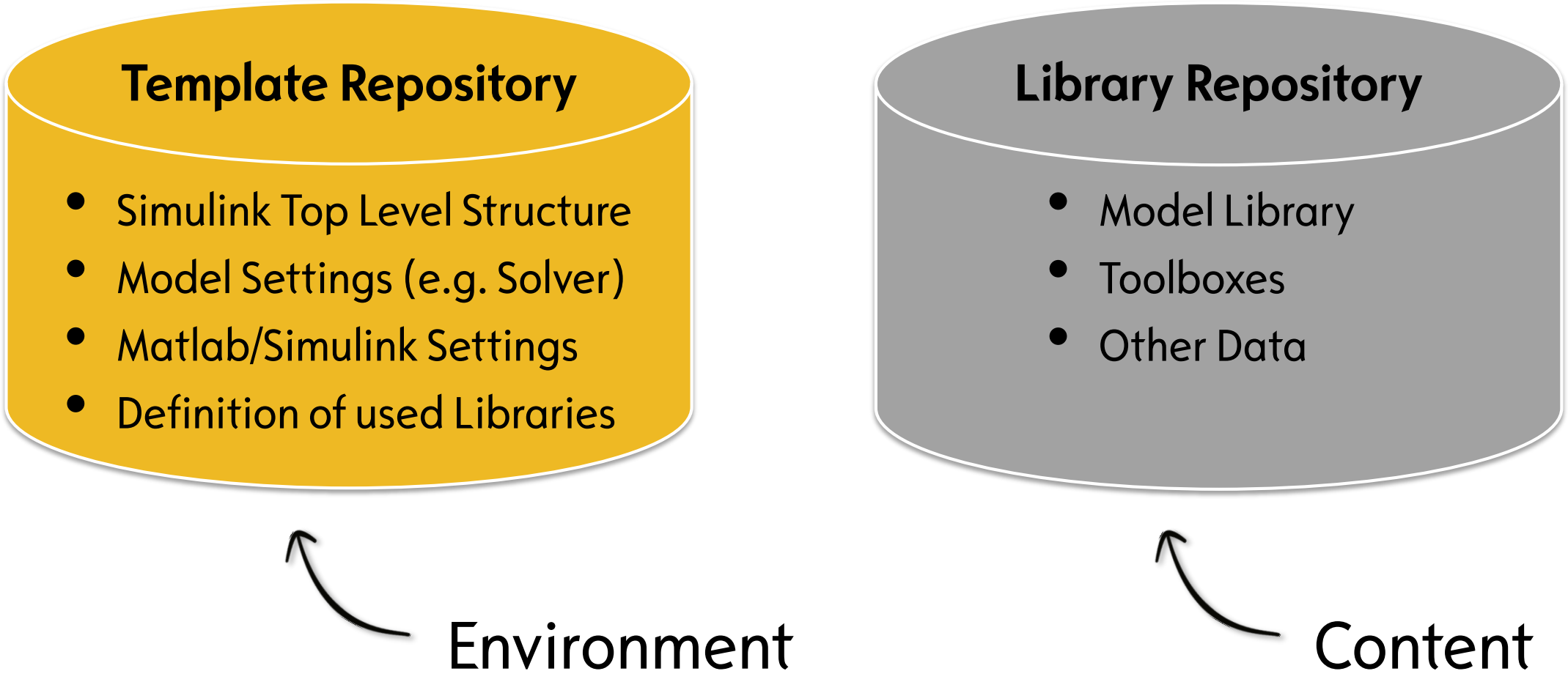
Follow Agile Principles to enable Continuous Integration & Testing





HOW TO SPECIFY AXIOM ENVIRONMENT?

Template vs. Library repository

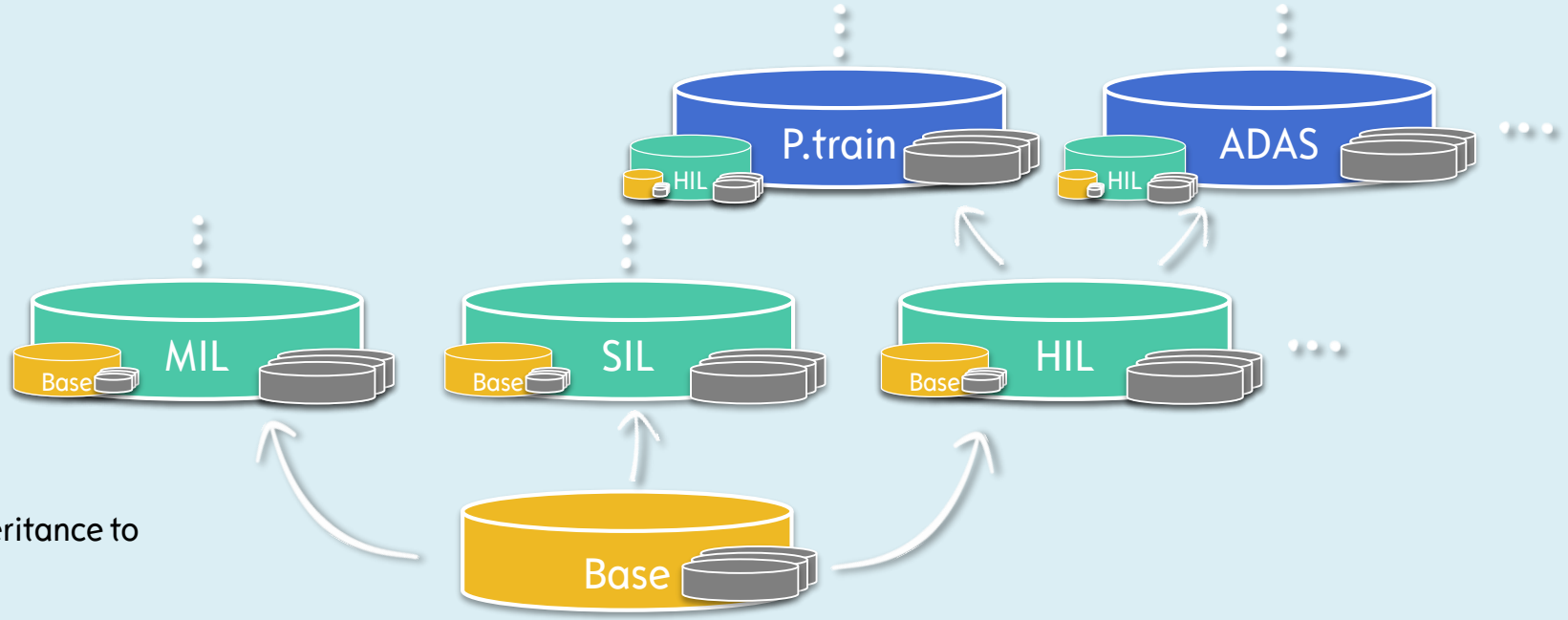


HOW TO SPECIFY AXIOM ENVIRONMENT?

Turn-key Application Models

Application 1 | Application 2 | Application 3 | ...

AXIOM



Template approach

Usage of object orientation and inheritance to create various stages of expansion

Base Software

MODULAR APPROACH

- Modular approach as main principle of Axiom
- Key enabler for collaborative work
- Module consists of a model together with it's parameters and interface definition
  - It is standalone capable and completely independent of other modules
  - Module interfaces are tunable parameters and signal ports

→ Powerful toolchain required to…

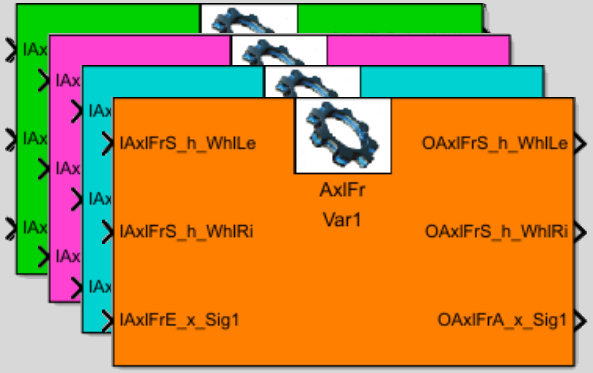
  - connect modules to each other
  - load application specific parametrization
  - maintain different configurations (variants)

EXAMPLE 1: CONNECTION MANAGER

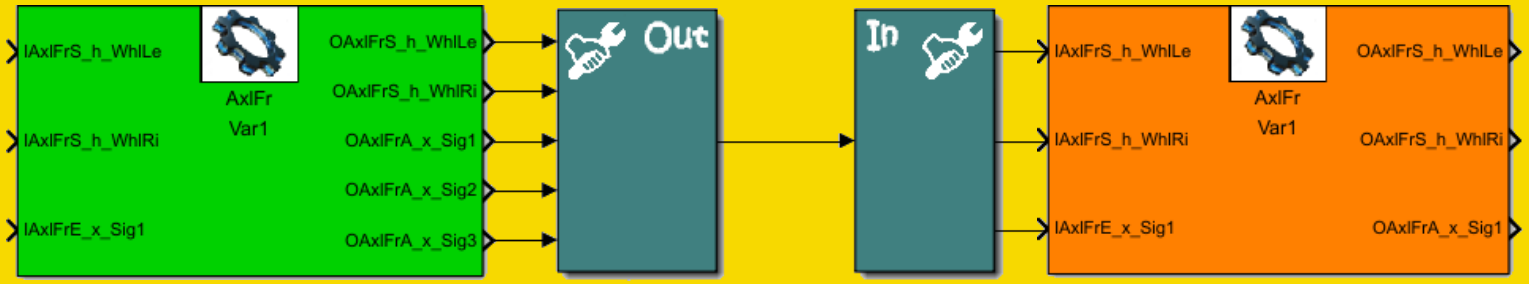
Adapter to connect multiple models

- Enabler for decoupled model development
- Well defined interface: prerequisite for model split
- Small busses realized by intelligent bus creation
- automatic satisfaction of open module interfaces

Model Component Libraries



Application Model

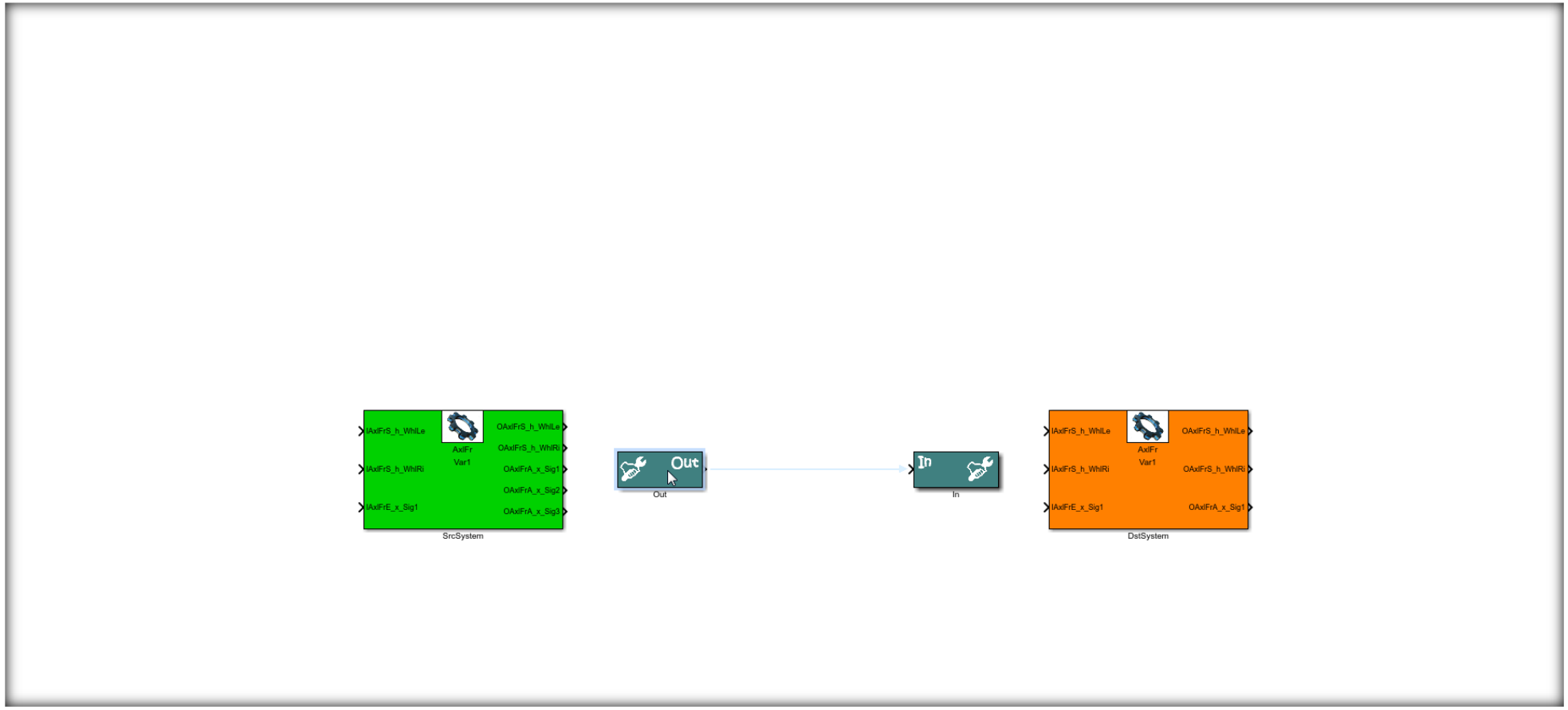


EXAMPLE 1: CONNECTION MANAGER

Demo

Step 1

Add Connection Manager Blockset



EXAMPLE 1: CONNECTION MANAGER

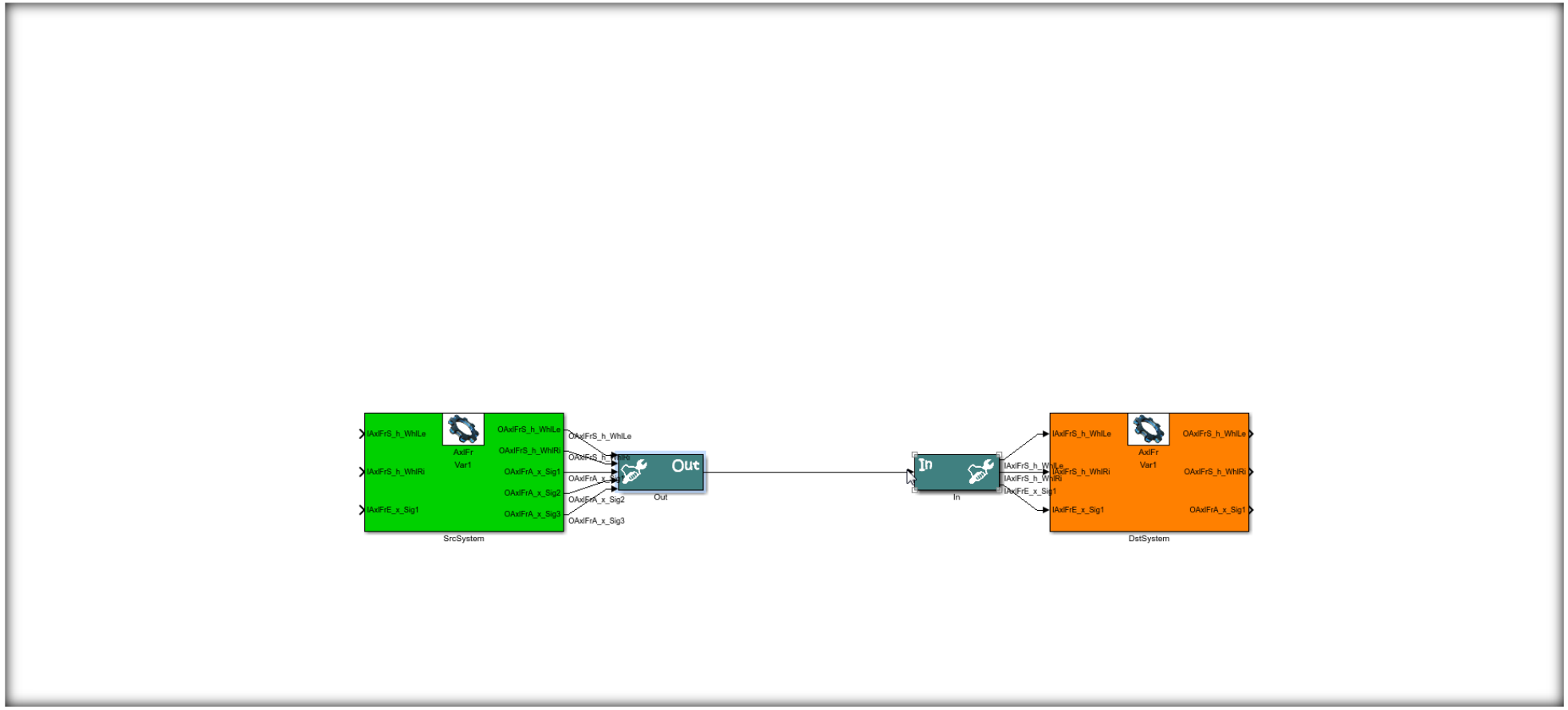
Demo

Step 1

Add Connection Manager Blockset

Step 2

Assign Connection Manager



EXAMPLE 1: CONNECTION MANAGER

Demo

Step 1

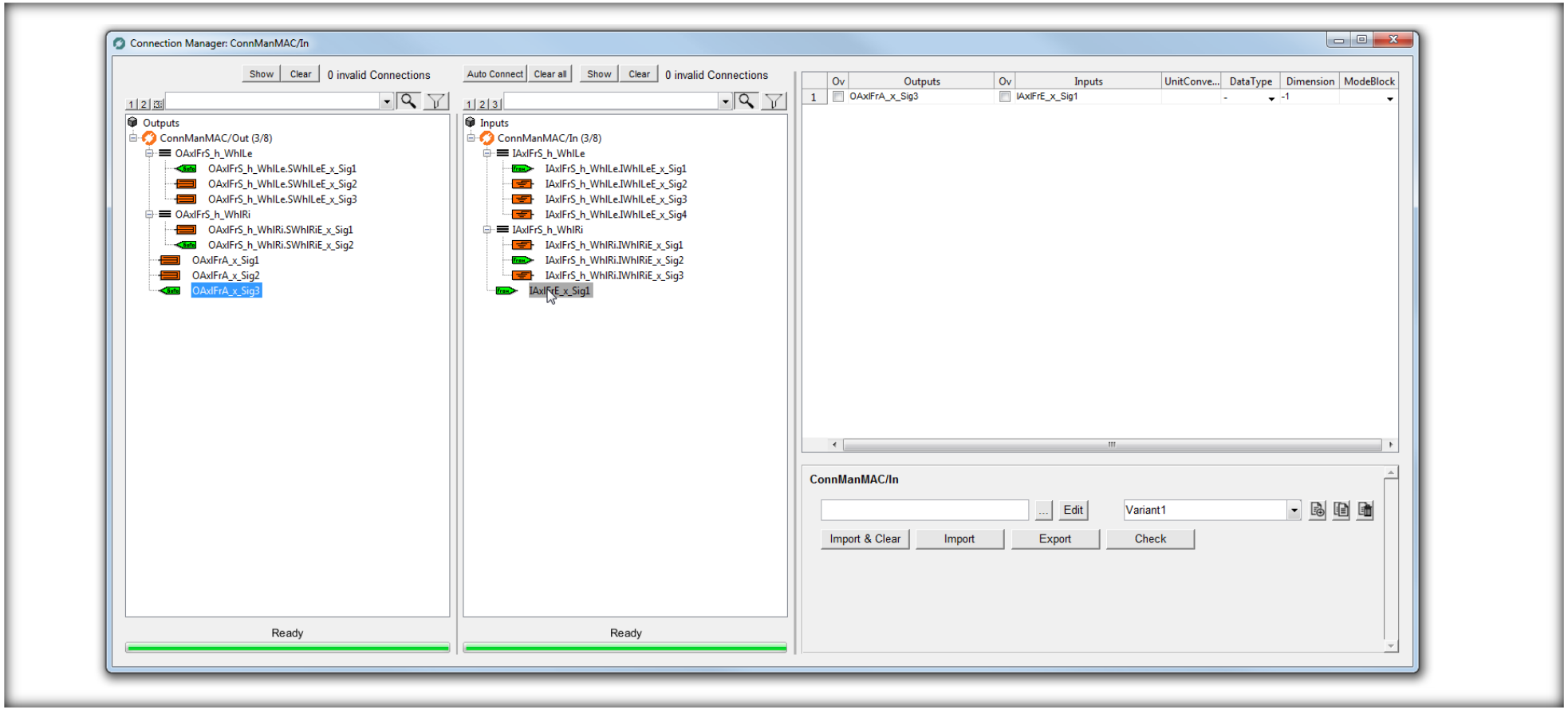
Add Connection Manager Blockset

Step 2

Assign Connection Manager

Step 3

Open Connection Manager



EXAMPLE 2: MODEL CONFIGURATOR

Management of parametrization

- Maintenance of parameter files (auto-generation, checks etc.)
- Apply specific parameterization by
  - Tunable parameter files
  - Overrides
  - References
- Automatic workspace initialization

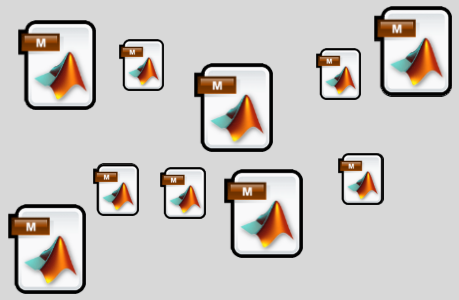
Management of model variants

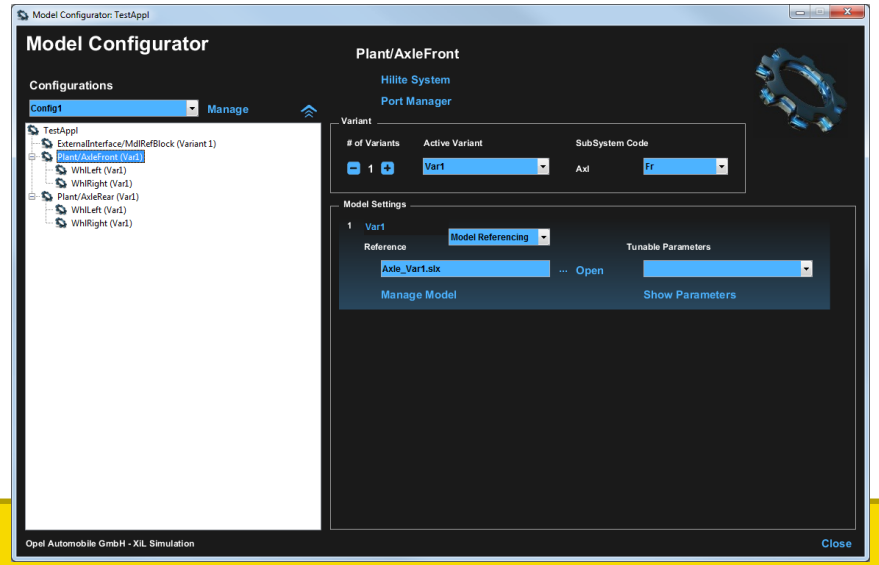
- Runtime switchable
- Provide functionality to store configurations → traceability, reuse
- Support of “Model Referencing”

EXAMPLE 2: MODEL CONFIGURATOR

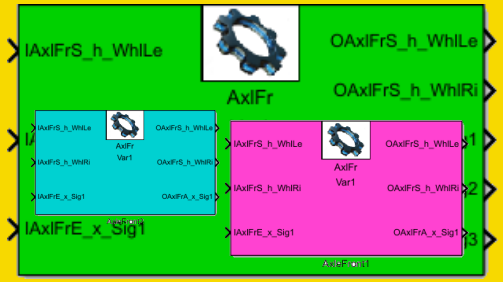
Parametrization via GUI

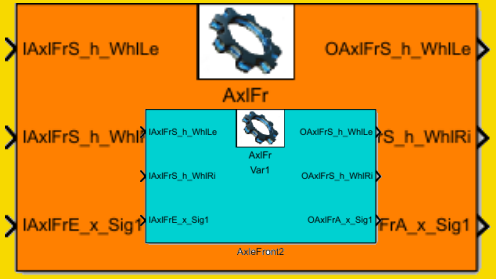
Parameter Files



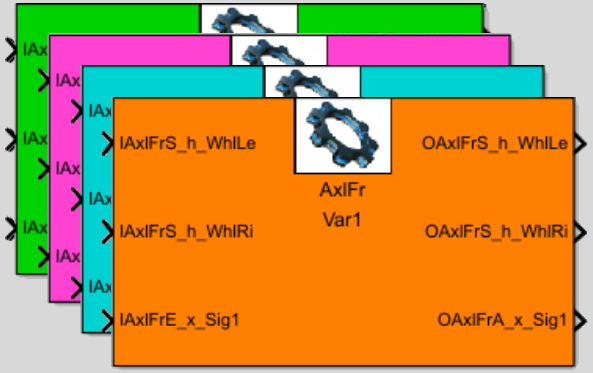


Application Model



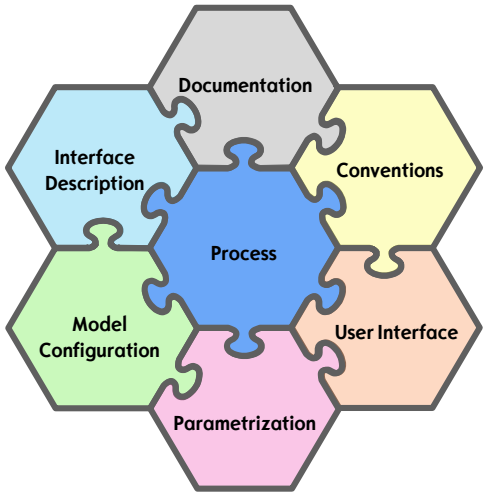


Model Component Libraries With base Parametrization



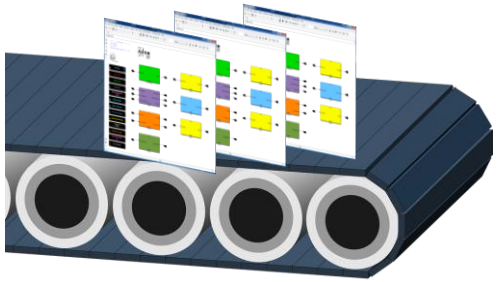
SUMMARY

AXIOM



Work across domains TEAMWORK	Compatibility *Test Automation* *Simulation*
Reusability	Powerful Toolchain
Agile principles	Continuous Integration CODE PLAN BUILD TEST RELEASE DEPLOY OPERATE MEASURE

Model assembly line



Efficiency
Quality

Cost

SUMMARY

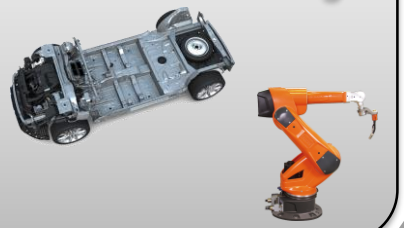
Car Factory

Processes

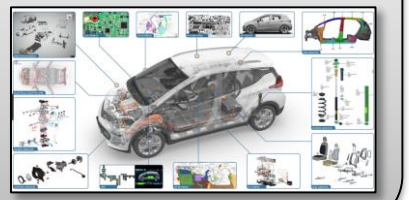
Logistics

Global Warehouse

Platform & Tooling



Car Configurations



Parts Warehouse

Assembly Line

Product

SUMMARY

Model Factory

Model Development Process

Change and Release Management

JIRA

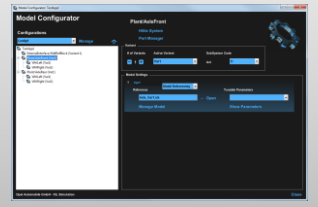
Version Control

git

Framework

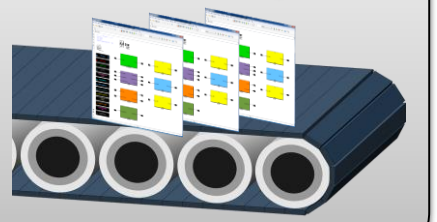
AXIOM

Model Configurations



Model Libraries

Model Assembly Line



Product

Turn-key Application Models

Collaborative Model Development for System Simulation

THANK YOU

Q &A