

Empowering the Application of Machine Learning Techniques through Simulation Data Management

Marko Thiele – SCALE GmbH;

Akhil Pillai – TU Dresden; Prof. Dr.-Ing. habil. Uwe Reuter – TU Dresden

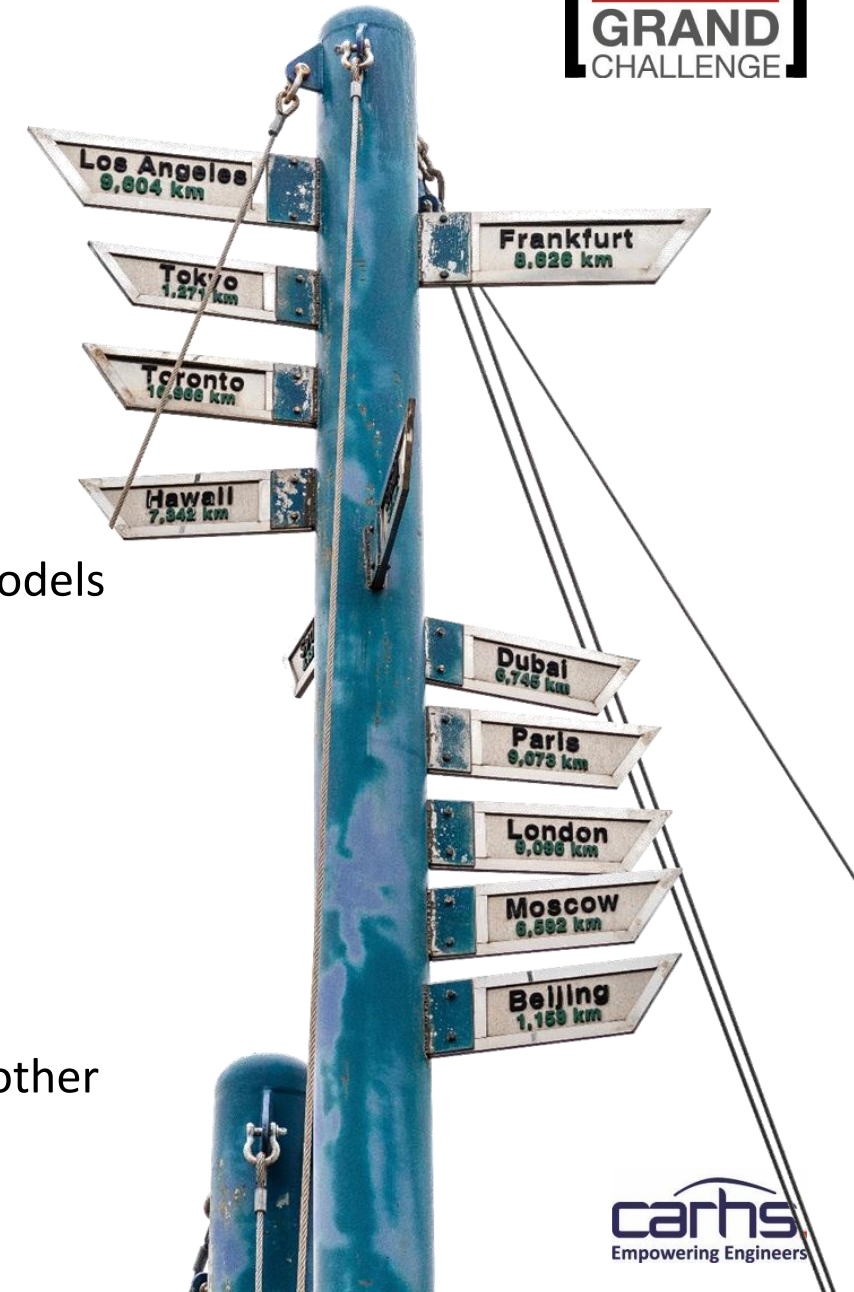
automotive CAE Grand Challenge 2019

April 16 - 17, 2019

Hanau, GERMANY

Agenda

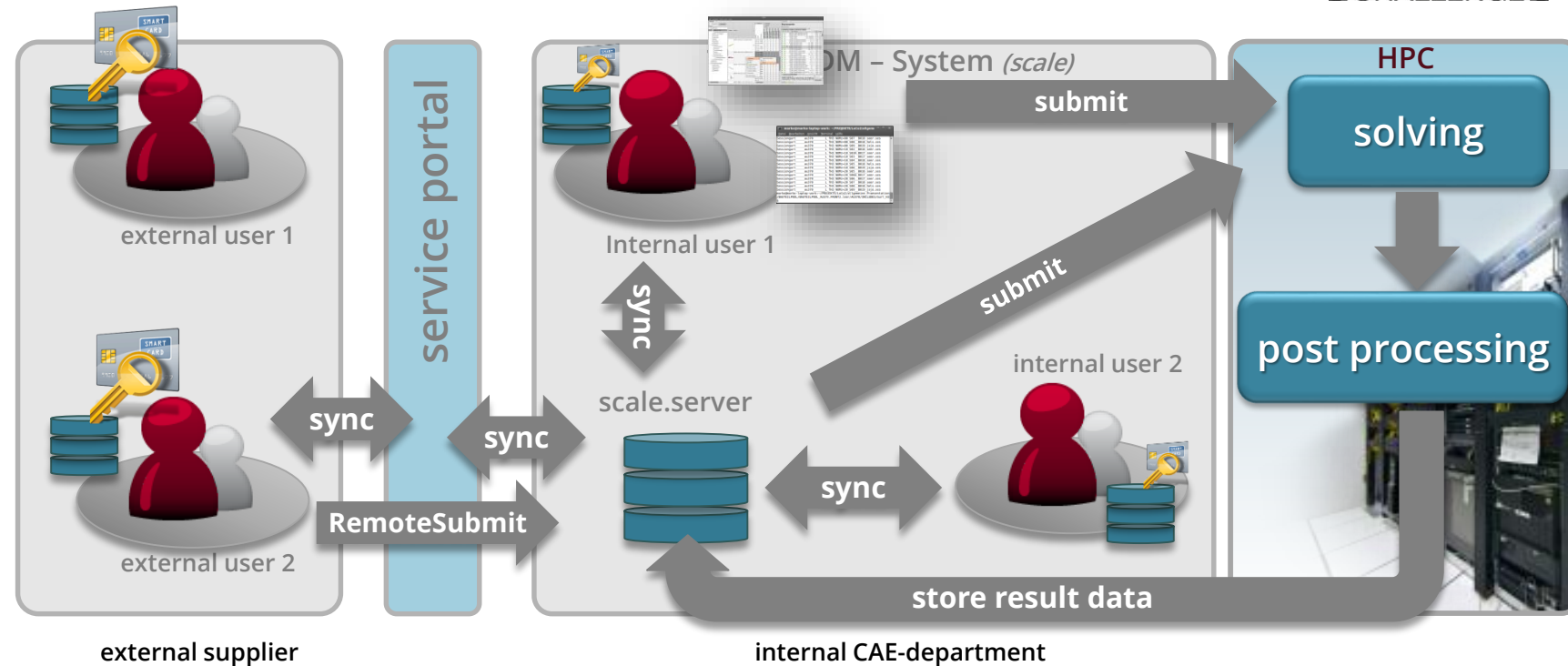
- *Challenge #1: Provide simple and fast data access*
Enabling simple and fast access to large amounts of Data
- *Challenge #2: Handle arbitrary simulation data with acceptable performance and robustness*
Handling of many and also sparse data with acceptable performance
- *Challenge #4: Ensure process chain reliability, stability and fault tolerance*
Implementing fault tolerant processes
- *Challenge #5: Educate engineers & students to get skills beyond scripting*
Empowering engineers to develop simulation processes and not just models
- *Challenge #3: Provide reliable applications understandable by non data-scientists*
Usability of applications that empower machine learning techniques
 - Usability of SDM systems
 - Example: automatic spot weld generation
- *Challenge #6: Provide benchmark data to enable algorithm development and comparison*
Development of algorithms with limited access to data
- *Challenge #7:*
Creating an environment where all the data is actually related to each other



Challenge #1: Provide simple and fast data access

Enabling simple and fast access to large amounts of Data

- Requirements
 - Uncomplicated
 - Fast
- Challenges
 - Security
 - Data volume
 - Amount of data
 - Heterogeneous work environment
 - Collaboration



Sync
decentralized

Offline / Online
performance

- Central data storage, synchronization with local workstations (*cloud like infrastructure*)
- Encrypted transfer, encrypted storage (*two factor authentication and encryption*)
- Offline handling of data (*RichClient*)
 - Users/Teams are independent of servers and infrastructure
 - Users work with local data
 - Good performance while application of preprocessing tools

Challenge #2: Handle arbitrary simulation data with acceptable performance and robustness

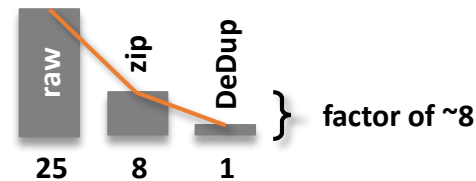
Handling of many and sparse data with acceptable performance

Requirements

- Open files fast
- Work with many files
- Solver independence

Reduction of data for transfer and storage

Inputs: Deduplication



Results: **SDMZiP** factor of ~3

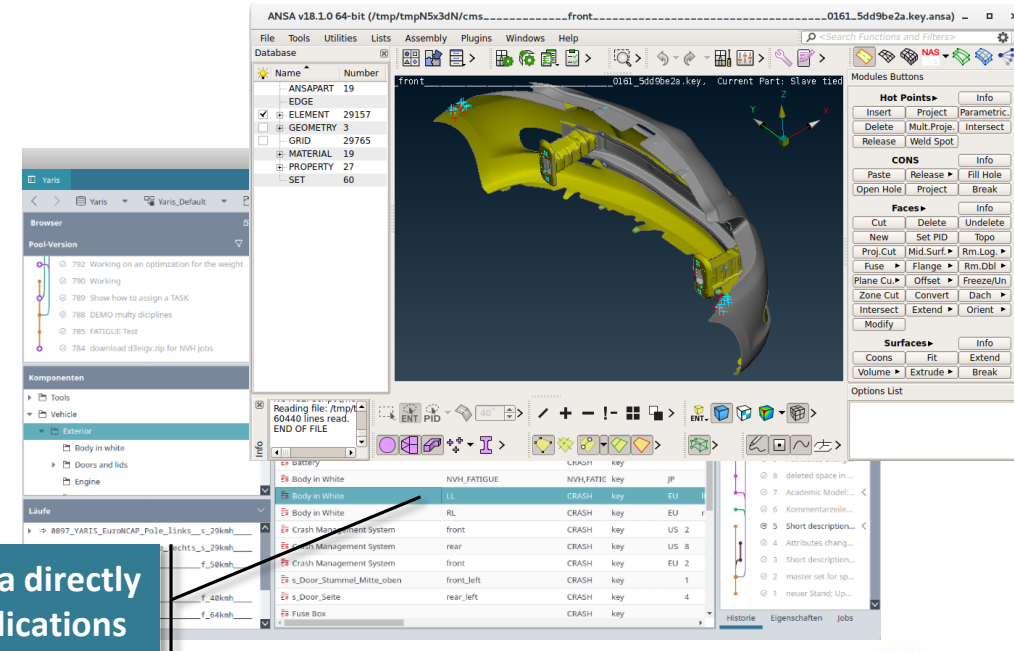
SIDACT GmbH

Challenges

- FEA files are sparse
- It's not known which files the user opens next
- Many simulation results
- Many different tools

Desktop integration

- Instant access to data
- Fast loading
- Tight integration of FEA tools



open data directly into applications

Implementing fault tolerant processes

Requirements

- Each simulations should return valuable results
- Model errors should be detected upfront
- Process errors should be easy to understand and fix

Challenges

- Complexity of simulation models
- Sophisticated FEA methods
- Complex HPC environments
- Number of people working on one model

Every time something is saved it will be checked

Name	Result	Line	Description
dynacheck	✓		
PRIMER	✓		
user_defined_warnings	ⓘ		
TABL_012	ⓘ		Table <values> not monotonically increasi 2150200
user_defined_errors	ⓘ		
LS-DYNA	✓		
user_defined_warnings	ⓘ		
Warning	ⓘ		Cross Section Labels # 11BUSYLORE00FO?B_X_Lower_Bumper_
Warning	ⓘ		The following ISO-Codes are used for multi
user_defined_errors	ⓘ		
check-hsp	ⓘ		
COMPONENT_MASS	ⓘ		1.40520647

Check results pop up immediately

Arbitrary checks can be implemented

Continuous testing of model data

- Each time a change is applied
- Before each simulation
- Customizable

Tight process monitoring

- Instant feedback to users
- Easy to understand error messages

Automated testing for processes

- Unit testing for process scripts

Tight monitoring of each process step

AU513_2xx_B_BF_EU_fzsl_16_2003_L_CD_AZT_links_ [✘]
SOLVING [SOLVING: FEM calculation: lsdyna: 0.09963ms of 6.0ms computed]
Preparation [create links for workdir: links for workdir created]
FEM calculation [lsdyna: 0.09963ms of 6.0ms computed]
preparing LS-DYNA Run [job prepared for execution]
lsdyna [0.09963ms of 6.0ms computed]
moving result files
Postprocessing

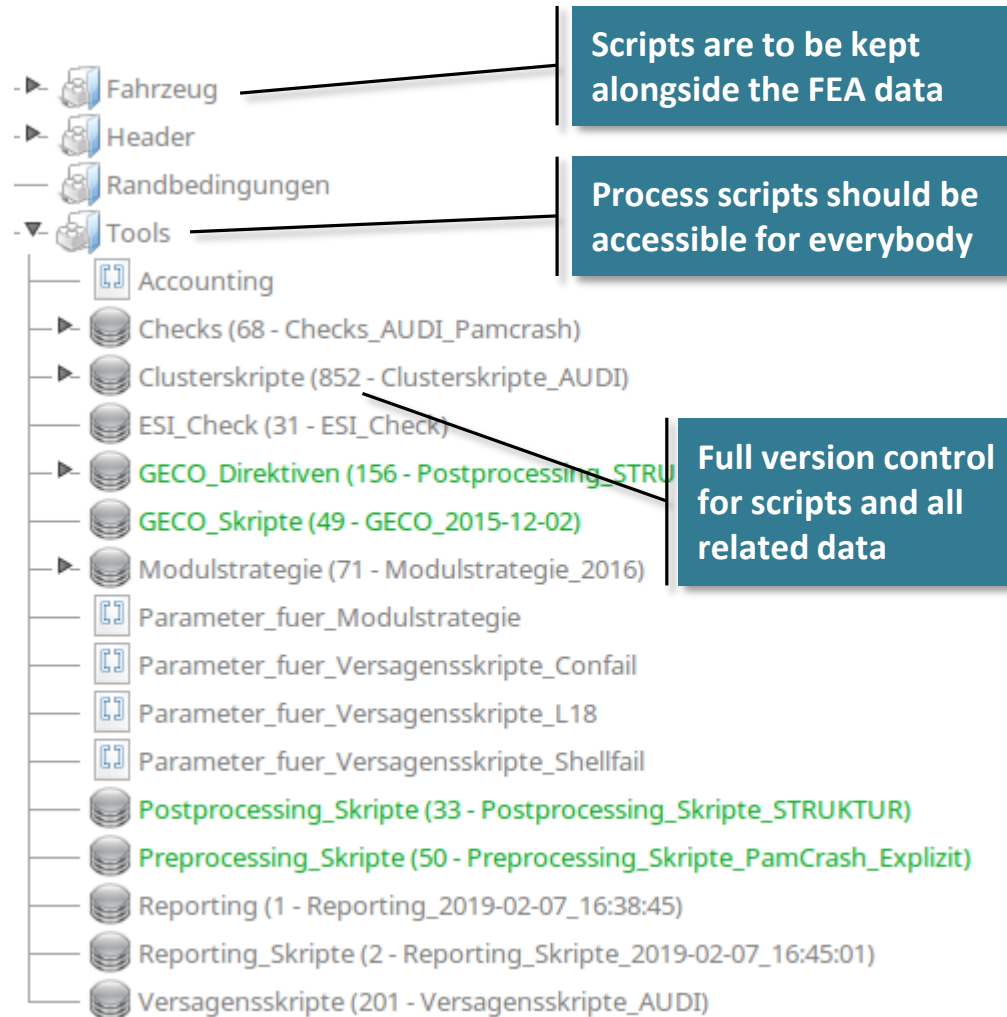
Empowering engineers to develop processes and not just models

■ Requirements

- Programming should be made easy
- Integration into normal workflows

■ Challenges

- Education
- Multitude of programming tools
- Teamwork



- All data should be exposed by APIs

Usability of applications that empower machine learning techniques

■ Requirements

- Intuitive User interface
- Responsiveness

■ Challenges

- Complexity vs. intuitiveness
- Huge data amounts

product structure

model variants

model data

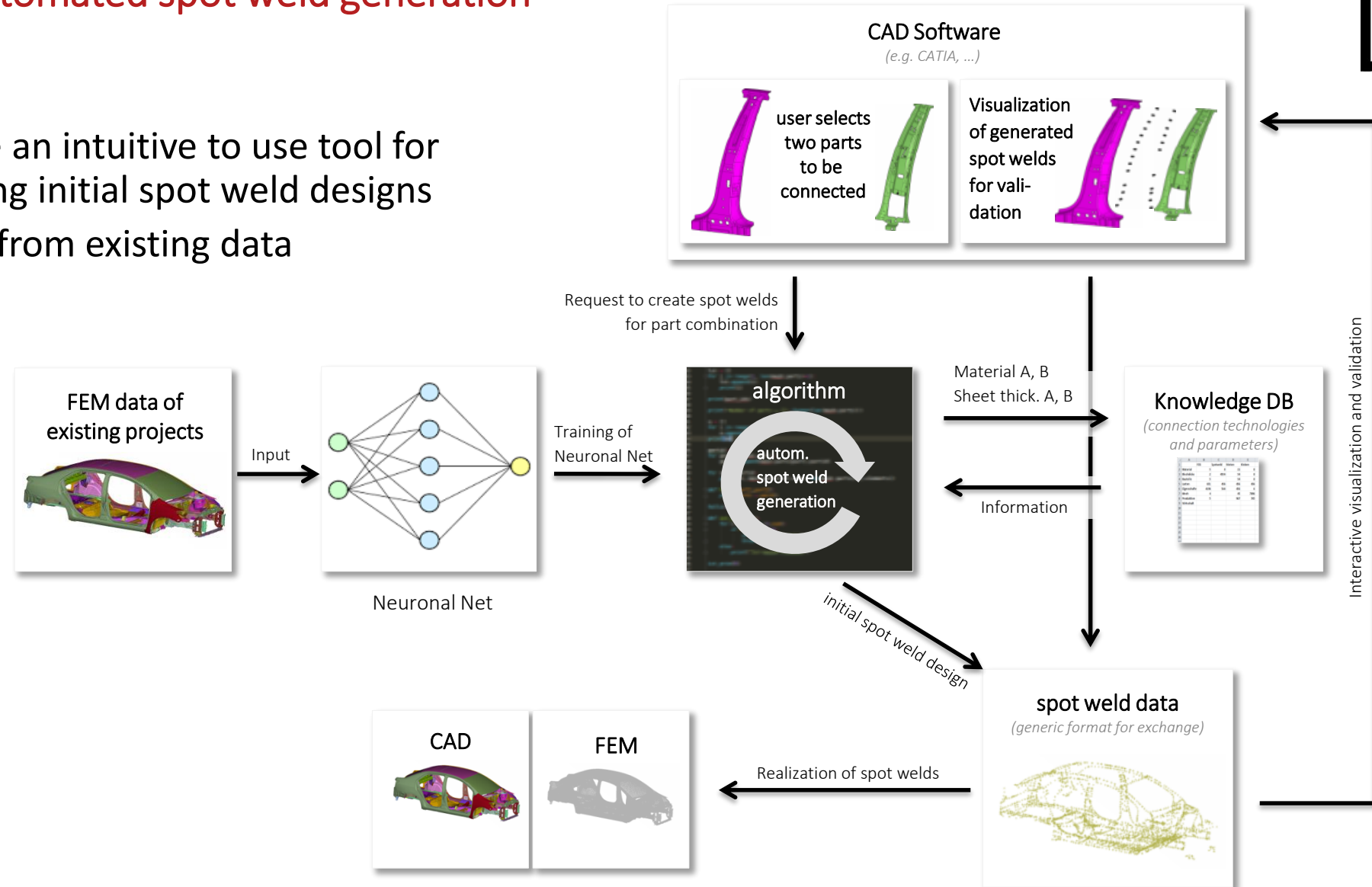
The screenshot shows the LoCoX software interface. On the left, there is a 'Browser' panel with a 'Pool-Version' list and a 'Komponenten' tree. The 'Komponenten' tree shows a hierarchy: Tools, Vehicle, Exterior (selected), Body in white, Doors and lids, Engine. Below it is a 'Läufe' list with various test runs. The main area is a table of model data with columns: Name, Short Description, Disziplin, FileType, Barrier, ImpactLocation, Region, Position, Hand Drive. The 'Body in White' row is highlighted. On the right, there is a 'Historie' panel showing a list of changes. Two blue callout boxes point to the version control features: 'version control for overall project' points to the 'Pool-Version' list, and 'version control for selected objects' points to the 'Historie' panel.

Name	Short Description	Disziplin	FileType	Barrier	ImpactLocation	Region	Position	Hand Drive
Air Filter		CRASH	key					
Axle	front_axle	CRASH	key				2	
Battery		CRASH	key					
Body in White	NVH_FATIGUE	NVH,FATIC	key			JP		
Body in White	LL	CRASH	key			EU		
Body in White	RL	CRASH	key			EU		
Crash Management System	front	CRASH	key			US	2	
Crash Management System	rear	CRASH	key			US	8	
Crash Management System	front	CRASH	key			EU	2	
s_Door_Stummel_Mitte_oben	front_left	CRASH	key				1	
s_Door_Seite	rear_left	CRASH	key				4	
Fuse Box		CRASH	key					

Example: automated spot weld generation

Goal

- Create an intuitive to use tool for creating initial spot weld designs
- Learn from existing data

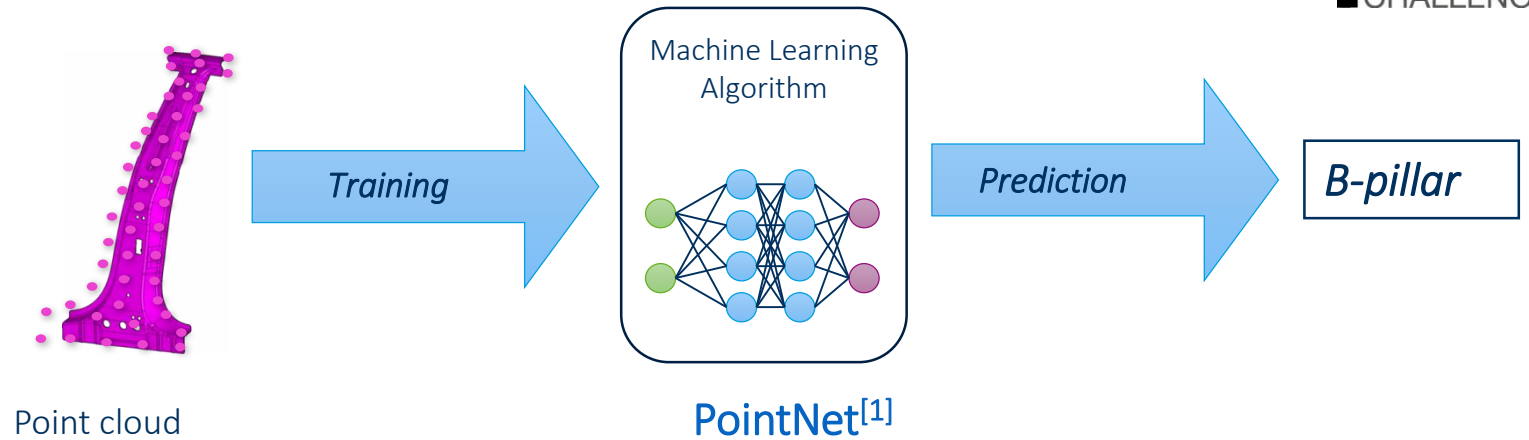


Example: automated spot weld generation

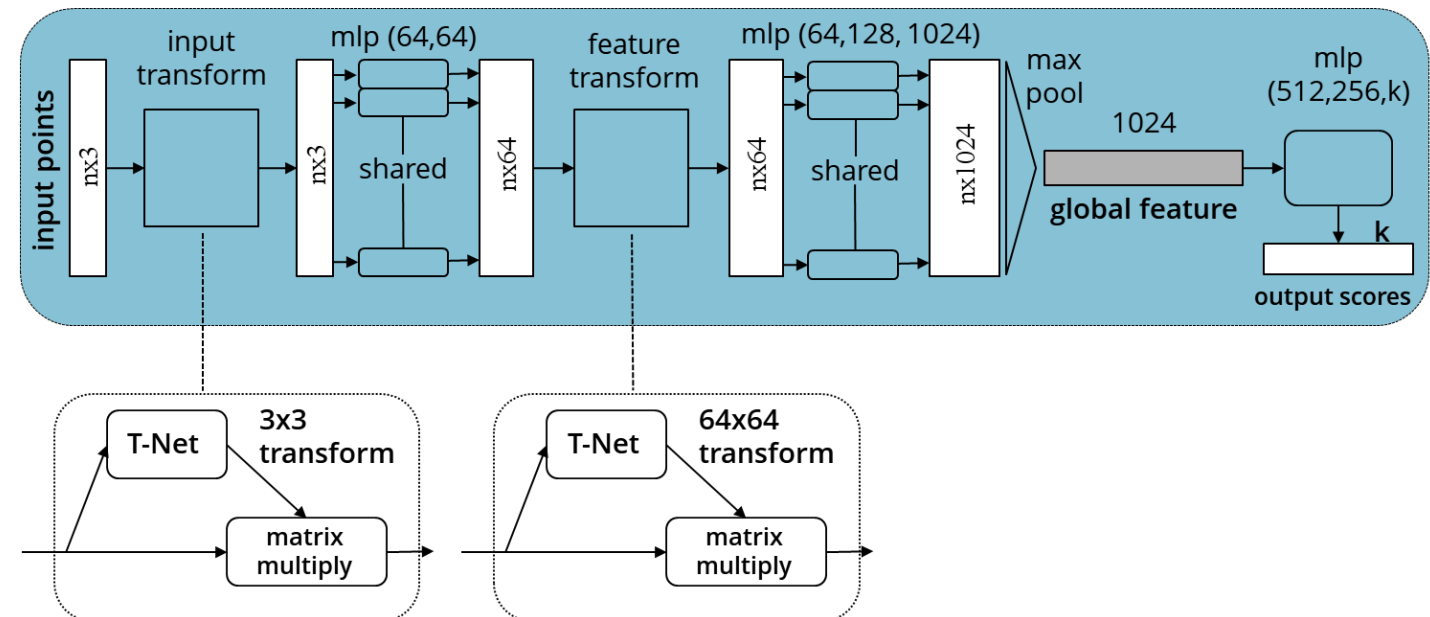
1st part recognition

(classification problem)

- Independent of geometric representation (CAD, FEM, ...)



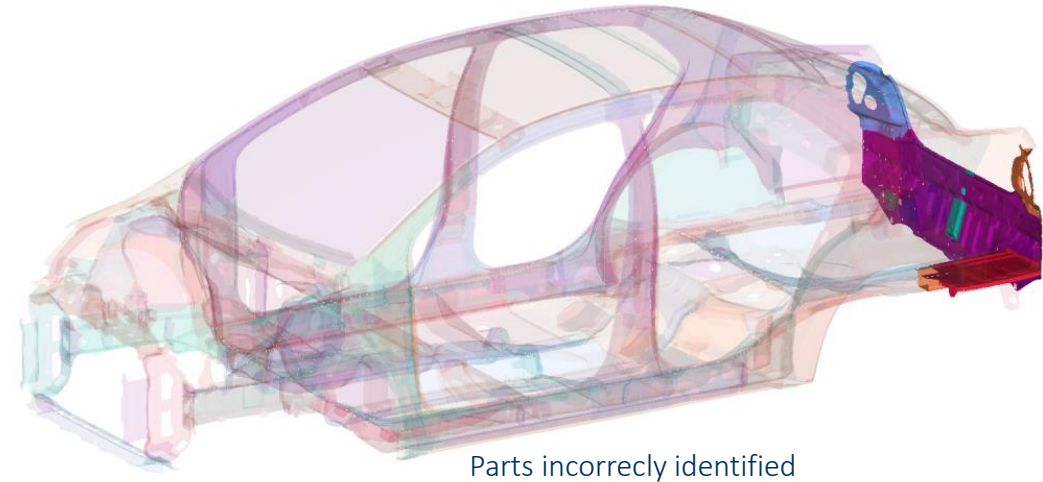
- Robust to
 - slight geometric differences
 - orientation
 - spatial location
- State of the art recognition accuracy



Example: automated spot weld generation

Part Identification of TOYOTA YARIS Model

Number of parts	250
Number of points per part	1024
Training samples	10,000
Test samples	2500
Epochs	300
Training time	~30 hours



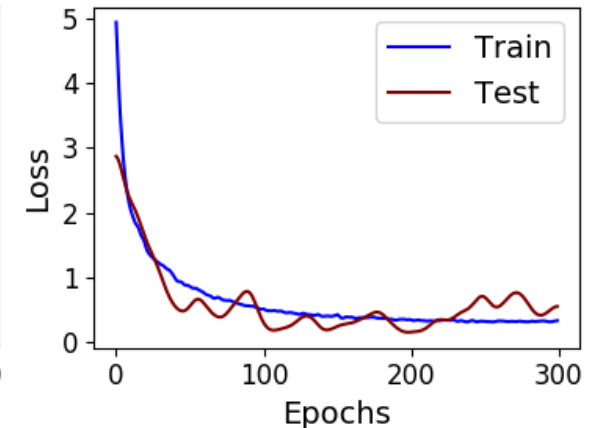
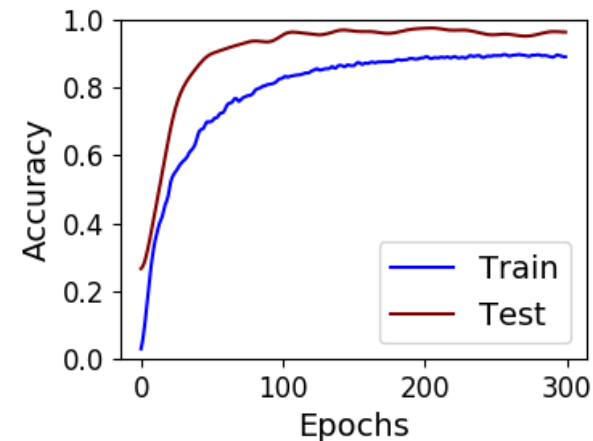
- 237 parts identified correctly
- 95 % identification accuracy



Part identification



one version of part only



Example: automated spot weld generation

Part Identification of AUDI Model

Number of parts	350
Number of points per part	1024
Training samples	17,500
Test samples	5000
Epochs	300
Training time	~48 hours

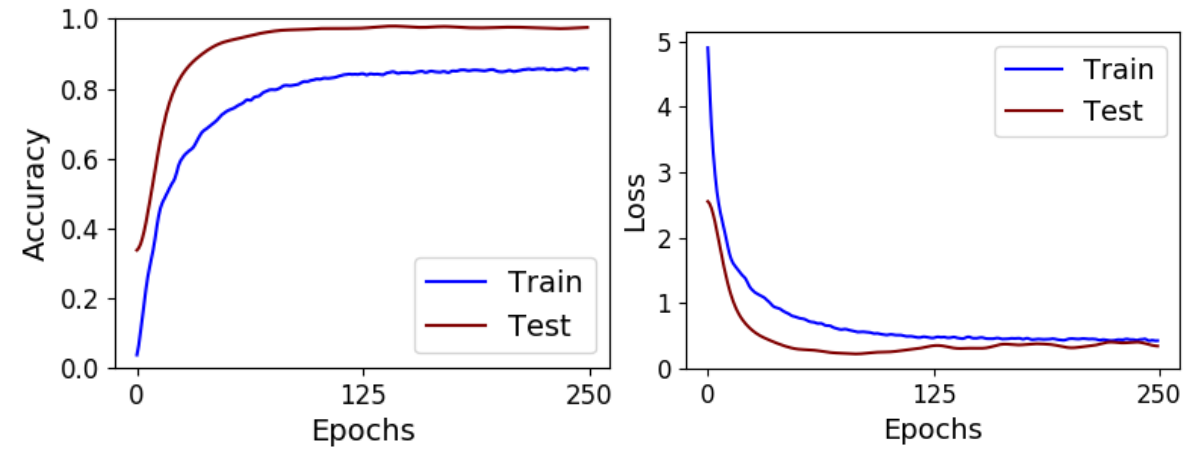
- 339 parts identified correctly
- 96 % identification accuracy
- New version of 13 parts used for evaluation



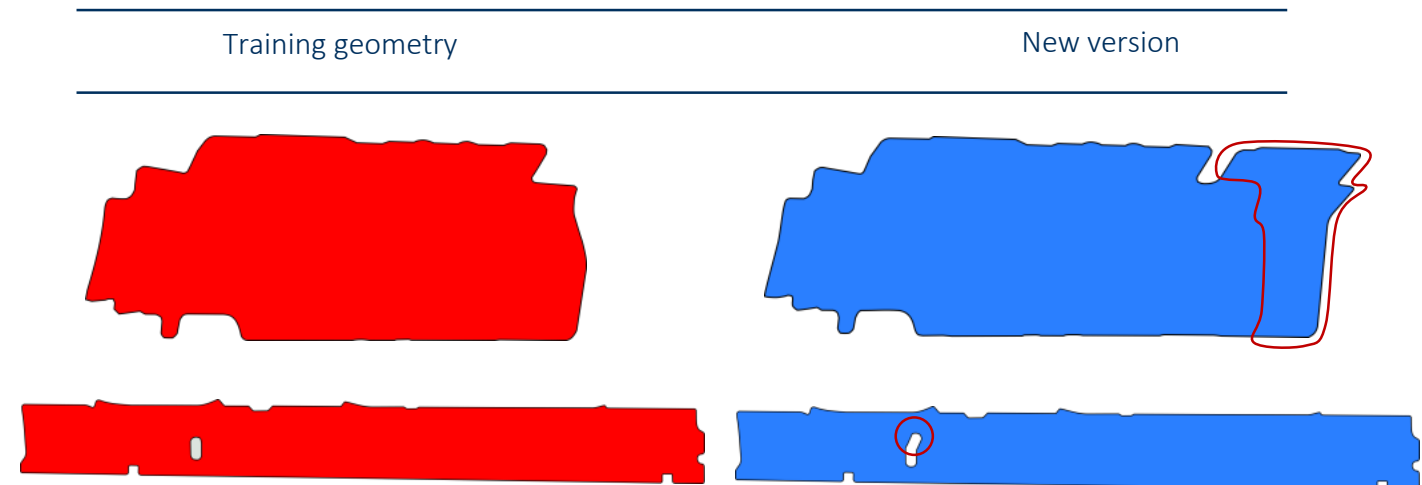
Part version identification



Parts of one model only



Examples of part versions

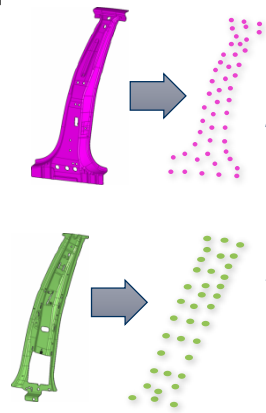


Example: automated spot weld generation

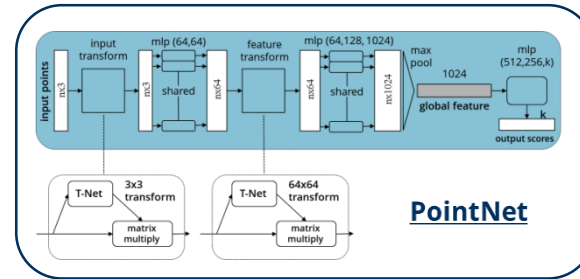
Estimation of spot weld design parameter

(prediction based approach)

- Support for geometric and non geometric information
- Extendable to more complex spot weld design parameters



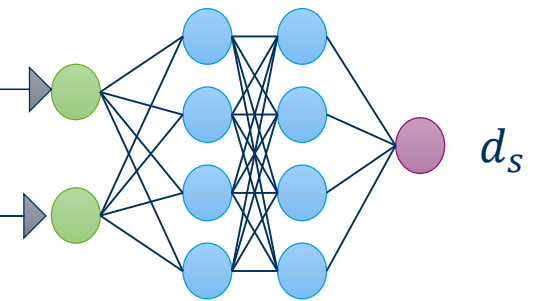
Part identification using PointNet^[1]



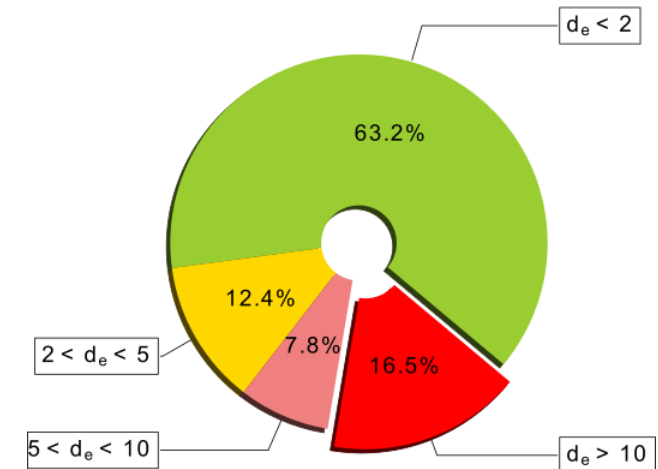
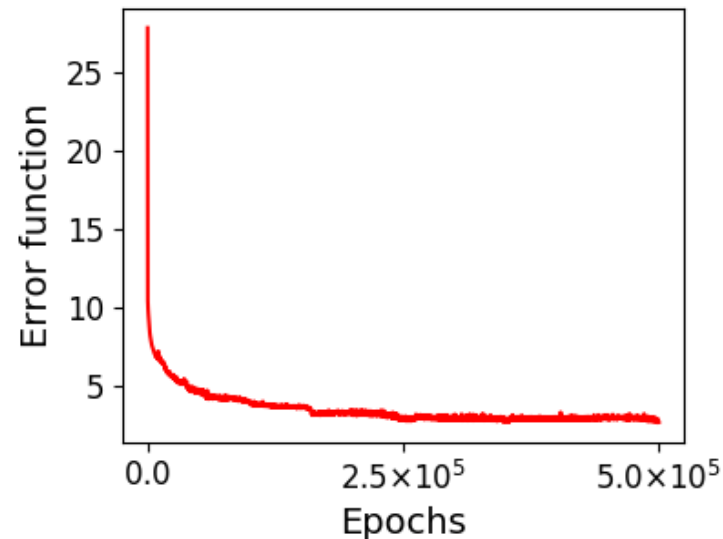
Part id

Part id

Spot weld parameter predictor



- Architecture : 2 | 150 | 50 | 1
- Error function: Mean Squared error
- Gradient descent optimization

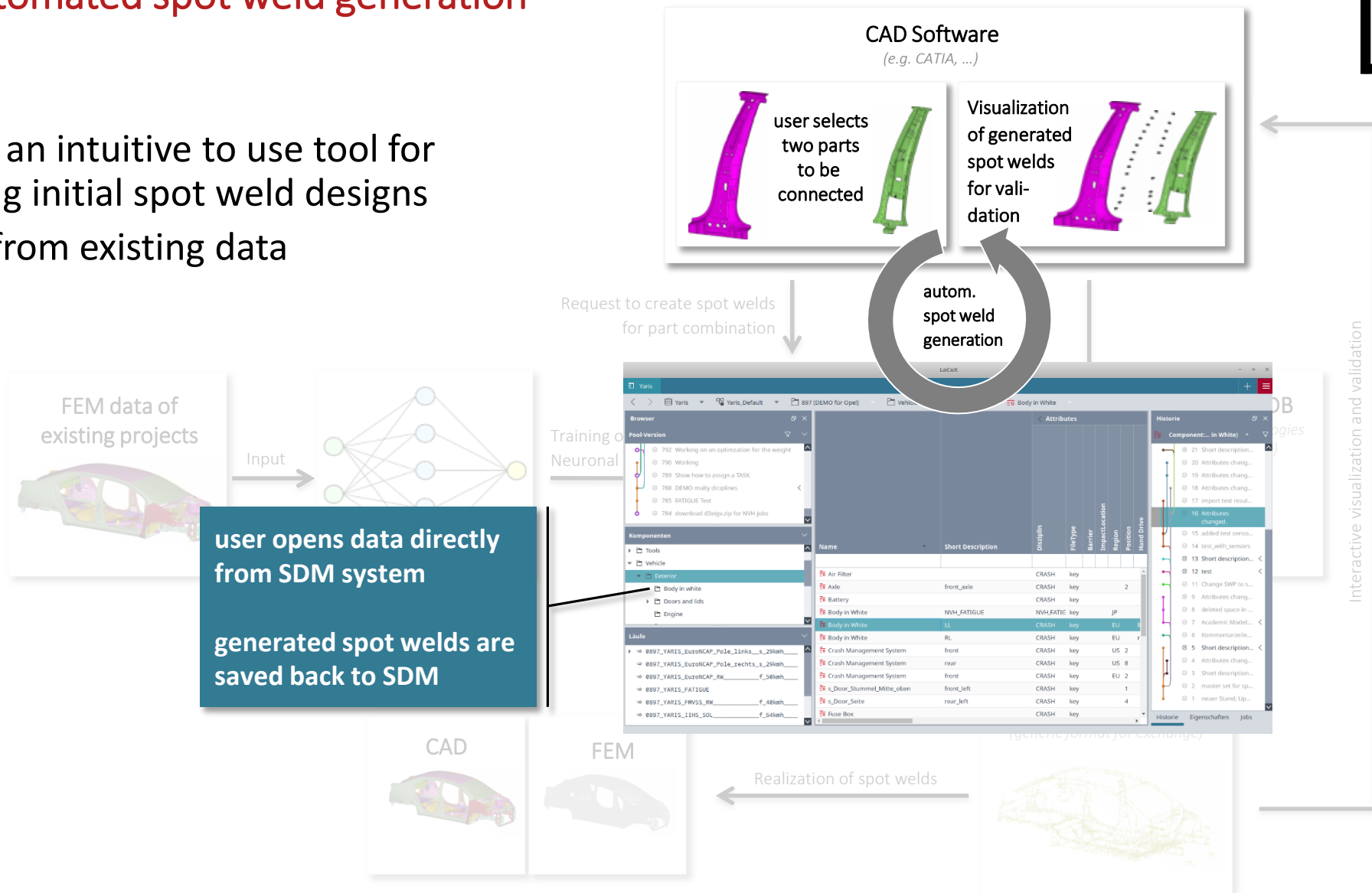


Challenge #3: Provide reliable applications understandable by non data-scientists

Example: automated spot weld generation

Goal

- Create an intuitive to use tool for creating initial spot weld designs
- Learn from existing data



Challenge #6: Provide benchmark data to enable algorithm development and comparison

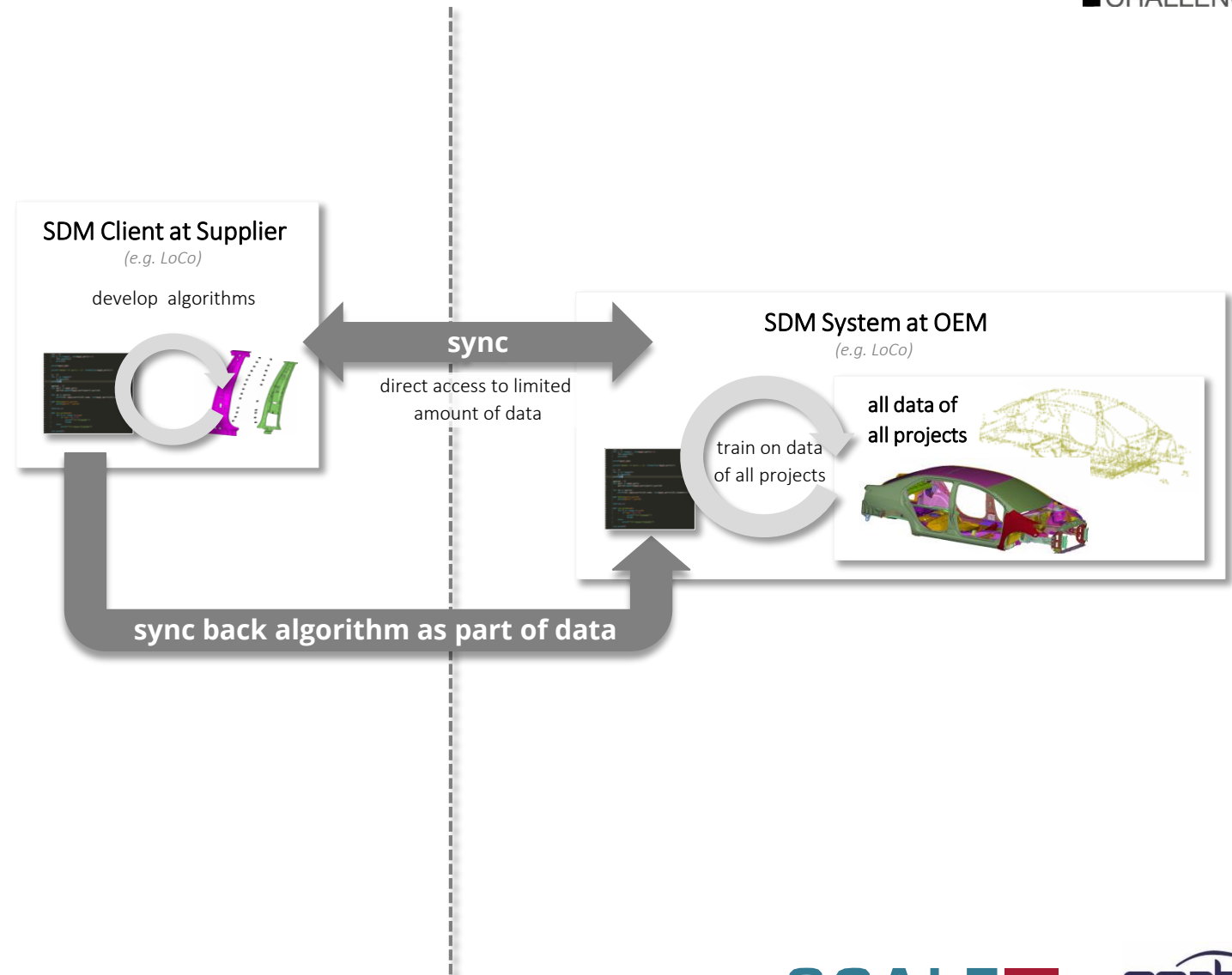
Development of algorithms with limited access to data

■ Requirements

- Data is needed to develop algorithms
- Data sets for training should contain as many data as available

■ Challenges

- OEMs want to protect their IP
- Amount of data that would have to be transferred
- Resources for training of data
- Security requirements



Challenge #7:

Create an environment where all the information is related to each other



Requirements

- All knowledge needed to develop machine learning algorithms has to be captured
- Relations between applied changes and observed effects are to be captured
- Assessments for applied changes

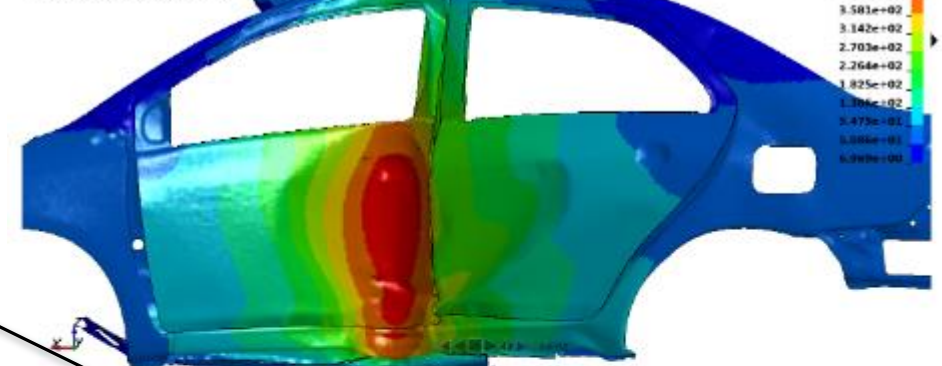
Challenges

- Implementation at OEMs is challenging
- Human users are needed for creating assessments
- “Chicken or egg” problem... there is currently no such related and/or assessed data

autom. detection of changed behavior

SDMZi SIDACT GmbH

0788 YARIS EuroNCAP Pole Inks s 29kmh
Time = 65
Contours of Relative Residual Displacement
min=6.96916, at node# 907576
max=445.88, at node# 1109605



interactive visualization

Typ: automatic detection
Tags: displacement
Evaluation: 9,5%
Event Name: ...
Event ID: 123

Description

increased intrusions

tags for measures related to previous similar events

Tag	Frequency	Rating (better, worse)
increase thickness	62%	63% 27%
remove part	42%	58% 42%
material	35%	43% 57%

suggestions based on previous assessments

Conclusions

- **Simulation Data Management** is the basis for gathering and providing the data needed for Machine Learning Applications
- Creating **truly related data** in the right format and with the correct amount of **data reduction** is essential
- Application of **machine learning** approaches will help to **solve many “small” problems** of the whole picture
- Current machine learning approaches can already be **very valuable** and **many things remain to be done** in order to harvests the full potential offered by the field of artificial intelligence. However we are still far away from strong artificial intelligence*.

*(*there won't be any time soon an automat that's able to take over the entire development of cars from us and be better in it than we humans)*

so long, and thanks for all the fish ...

SCALE

