dynardo | presented at the 14th Weimar Optimization and Stochastic Days 2017
Source: www.dynardo.de/en/library

AUTOMATED OPTIMIZATION WITH THE OPX INTERFACE USING THE EXAMPLE OF VEHICLE STABILIZERS

Sergej Schneider, Mubea Fahrwerksfedern GmbH

Weimar, 2nd of June 2017

Mubea
light.efficient.global.

Agenda

Mubea
light.efficient.global.

1. THE COMPANY “MUBEA”
2. CHASSIS PART STABILIZER BAR
3. (OPTIMIZATION) WORKFLOW
4. XML-BASED INTERFACES IN OPTISLANG

THE COMPANY „MUBEA“

Facts

- Global market leader in development and manufacture of automobile products
- Owner-operated family company since 1916
- Lightweight component design specialist
- Vertical integration from raw material to finished product
- Internal development of products and production processes


Represented worldwide

Mubea
light.efficient.global.

North and South America:

- 5 locations
- 2,400 employees

Europe:

- 22 locations
- 8,800 employees

Asia:

- 9 locations
- 1,800 employees

Product portfolio

Mubea
light.efficient.global.

Chassis

Body

Powertrain

Industry

Products

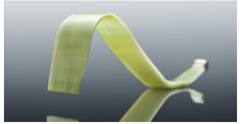
Chassis

Coil Springs

Stabilizer Bars

Stabilizer Bar add-on parts

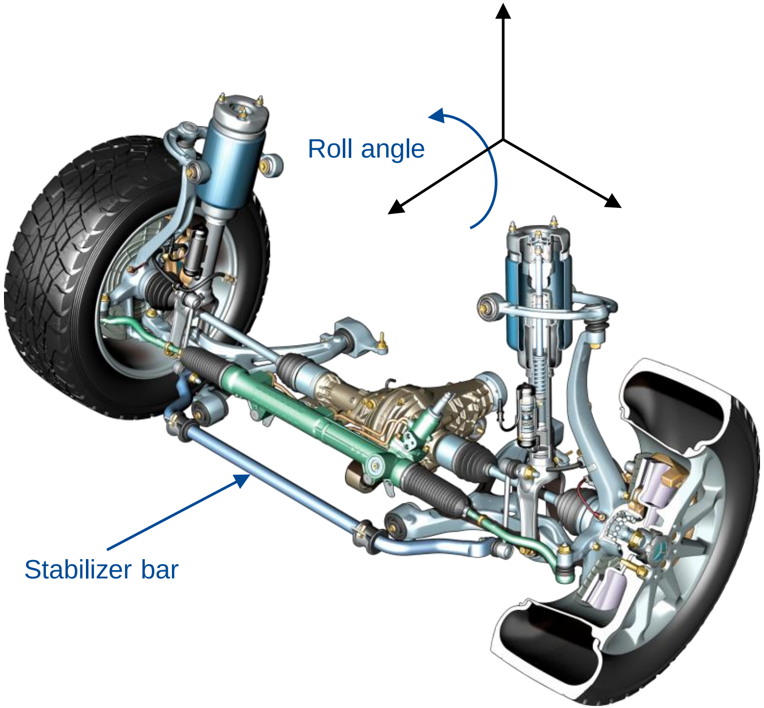
GFRP Leaf Springs



Wheels

CHASSIS PART STABILIZER BAR

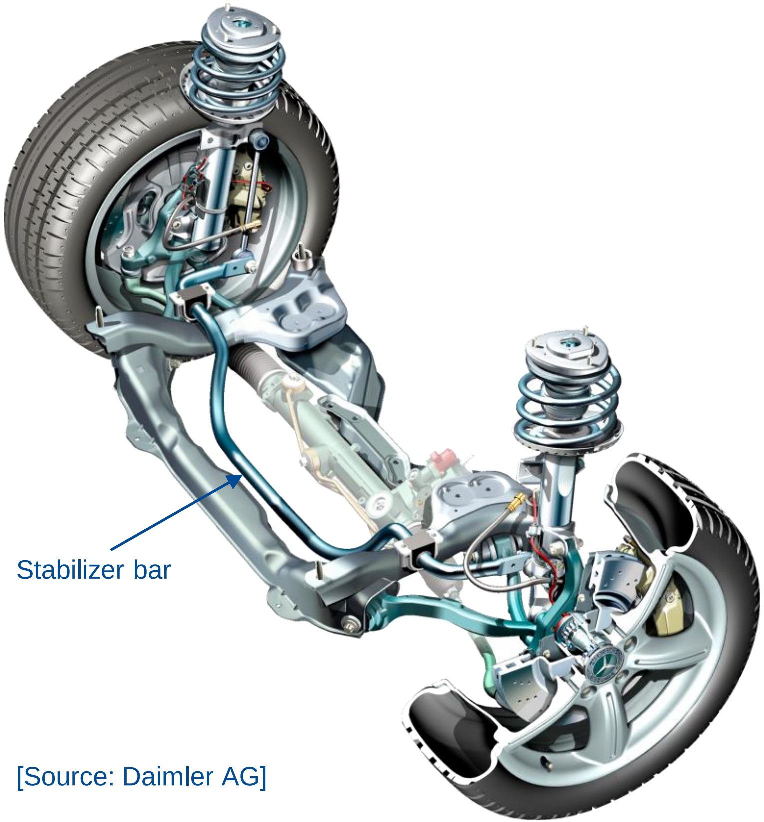
Stabilizer function



[Source: Daimler AG]

- Characteristics of stabilizer bar
  - Links right and left suspension
  - Compensates / decreases roll angle during turning manoeuvres wrt. local body frame
  - Loaded under torsion (and bending)
- Increasing spring rate results in higher compensation of roll moment
- Profile designs
  - Solid
  - Tubular
  - Hammered
  - Multiple wall thicknesses at const. outer diameter (MTT)

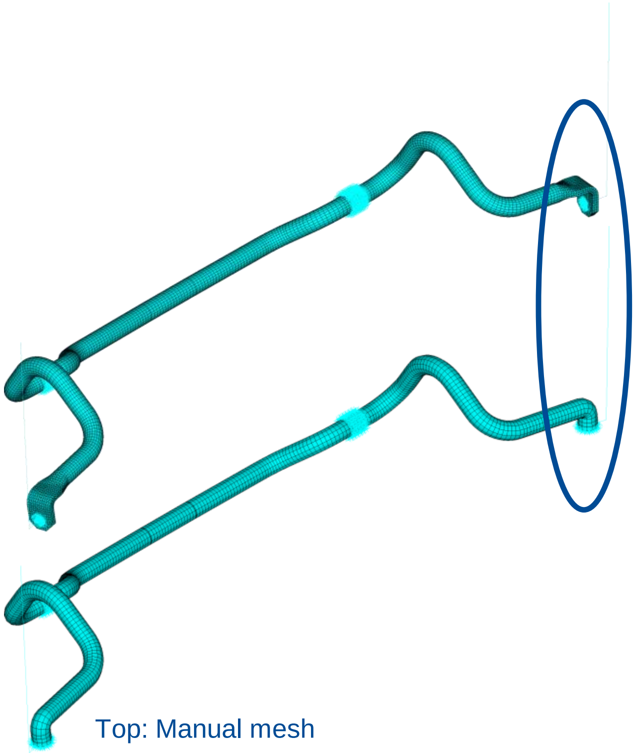
Stabilizer design



[Source: Daimler AG]

- Fixed routing due to limited design space
- Mechanical properties (spring rate) may only be adjusted through stabilizer bar profile
- Profile design is restricted by production-wise feasibility
  - Maximum wall thickness *t*
  - Minimum / maximum wall thickness ratio D_{out}/t
  - Form of crossing depending on raw material
    - Tailor Drawn Tube
    - Tailor Rolled Tube

Stabilizer FEA

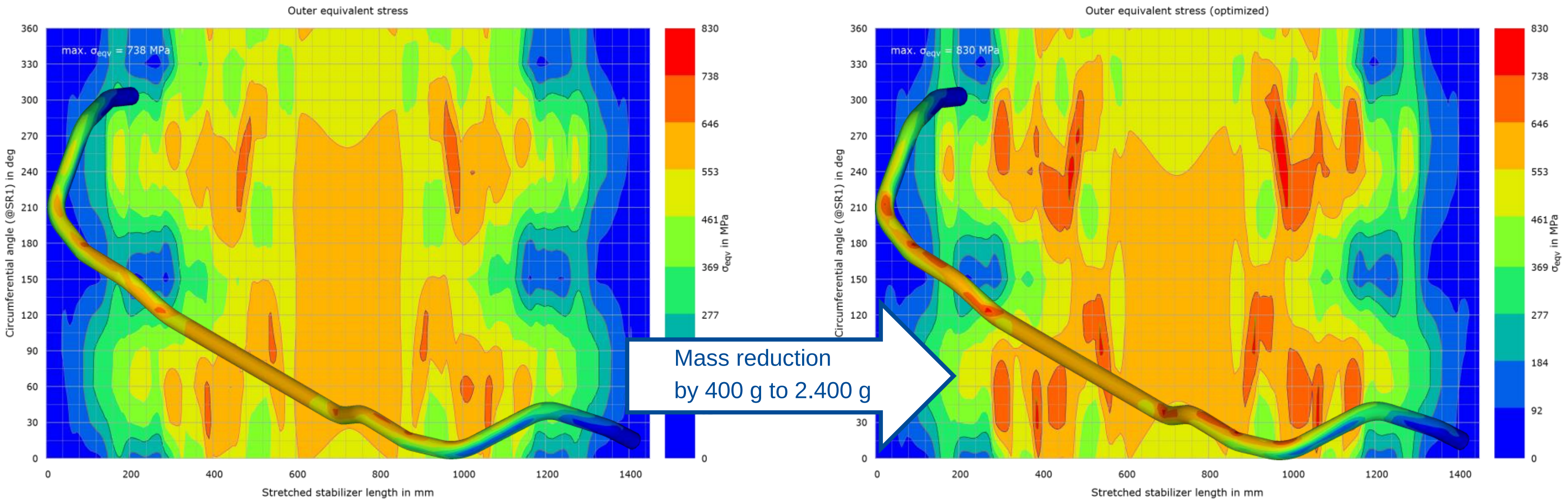


Top: Manual mesh

Bottom: Parametrized mesh (w/o paddles)

- Stabilizer is described via a polyline with circular cross sections
- Stabilizer profile may be separate input detached from routing
- Geometry and boundary conditions are parametrized in APDL
  - Neglecting paddle geometry
  - Using own meshing algorithm
- Static mechanical analysis using implicit solver in ANSYS
- Using quadratic elements
- Design objective is spring rate with minimum stabilizer mass below limit fatigue life

Stabilizer design optimization



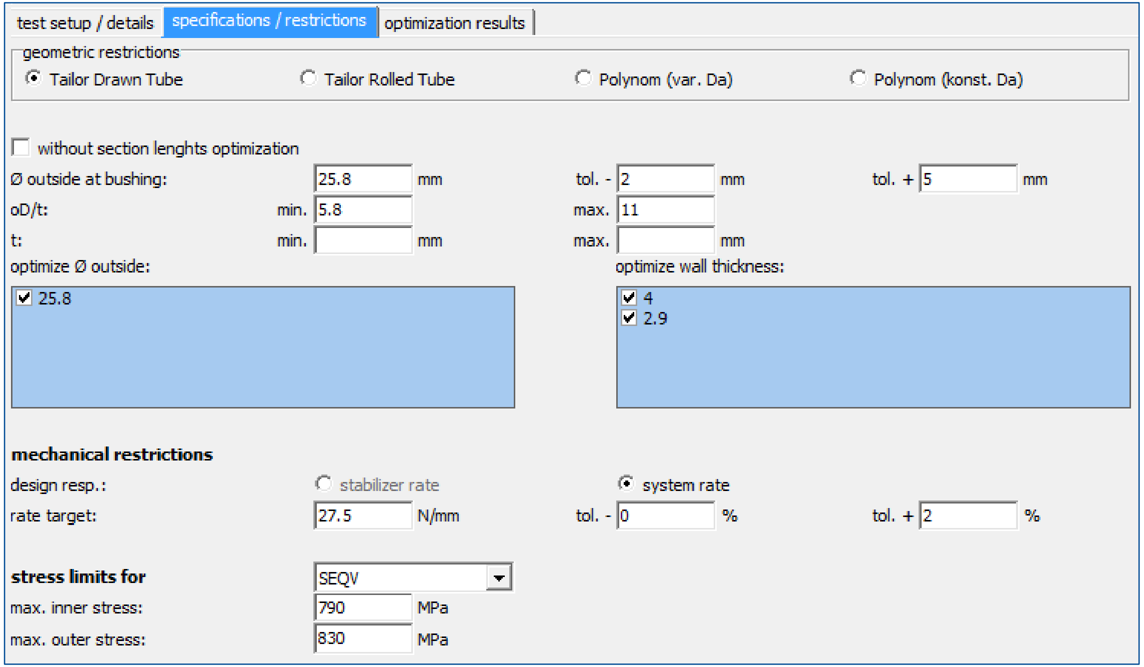
- Example of mass minimization for a stabilizer bar with multiple wall thicknesses
  - Displayed stabilizer bars are equal in spring rate (i.e. same function)
  - Utilizing the stress limits (von Mises) mass could be reduced by ca. 400 g to ca. 2.400 g

(OPTIMIZATION) WORKFLOW

Motivation

- The design process is iterative, where
  - the stabilizer profile is adjusted
  - the FEA is performed
  - the results are compared with target values
- The design process takes a lot of time and experience
- An automatization of this process would relieve the project engineer of this time consuming task and allow him/her to invest his/her time otherwise
- It is expected to find better designs using optimization algorithms

Optimization tasks using PDM



Snippet from the optimization order interface

- Optimization order is created in a PDM system
  - A reference design has to be attached
  - Geometric and FEA restrictions have to be provided with their respective tolerances
  - Input parameters may also be set as constant

Optimization workflow

- A service running on the calculation server checks regularly, if not-processed optimization orders are queued
- The optimization workflow is:
  - Order is loaded and the order‘s status is updated in the database
  - Calculation is performed based on the reference design in ANSYS
  - Input and output is text-based with a variable number of parameters and (geometric) restrictions
  - XML interface files are created for optiSLang and the optimization is performed
  - Best design is loaded and saved to the database
  - The order‘s status is updated in the database and the orderer is informed via email

XML-BASED INTERFACES IN OPTISLANG

XML usage in optiSLang

- The optimization process in optiSLang may be controlled via batch and an XML-file
- The XML-based interface in optiSLang 6.x.x is called OPX
- The XML file for problem description has to be recreated specifically for each project due to variable number of parameters
  - A DTD (Document Type Definition) file may be used to describe the XML file‘s structure
  - DTD files are also used to check XML/HTML files for structural integrity
  - Many computer languages support XML databinding to extract and create object from those DTD files
    - The XML object will contain all elements with their attributes defined in the DTD file
    - Using an XML object elements may be added and edited at run-time as often as needed before the XML (OPX) export is performed

Data Type Definition

```
<?xml version='1.0' encoding='ISO-8859-15' ?>

<!ELEMENT OPX (NODES, EDGES, NODE_ATTRIBUTES)>
  <!ELEMENT NODES (NODE)+>
    <!ELEMENT NODE (NODE)*>
    <!ATTLIST NODE
      ID  CDATA #REQUIRED
    >

  <!ELEMENT EDGES (EDGE)+>
    <!ELEMENT EDGE EMPTY>
    <!ATTLIST EDGE
      FROM      CDATA #REQUIRED
      FROMSLOT  CDATA #REQUIRED
      TO        CDATA #REQUIRED
      TOSLOT    CDATA #REQUIRED
    >

  <!ELEMENT NODE_ATTRIBUTES (ATTRIBUTES)+>
    <!ELEMENT ATTRIBUTES (ID, TYPE, (FILE, SEPARATOR, PARAMETERS, CRIT
      <!ELEMENT ID   (#PCDATA)>
      <!ELEMENT TYPE (#PCDATA)>
      <!ELEMENT FILE (#PCDATA)>
      <!ELEMENT SEPARATOR (#PCDATA)>
      <!ELEMENT PARAMETERS (PARAMETER+)>
        <!ELEMENT PARAMETER (OPTIMIZATIONATTS)>
        <!ATTLIST PARAMETER
          CONSTANT      (true|false)                      #REQUIRED
          NAME          CDATA                             #REQUIRED
          TYPE          (opti|mixed|robust|dependent)     #REQUIRED
          VALUE         CDATA                             #REQUIRED
          VALUETYPE     (real|integer|string|bool)        #REQUIRED
          FORMAT        CDATA                             #REQUIRED
          LINEOFFSET    CDATA                             #REQUIRED
          COLUMNOFFSET  CDATA                             #REQUIRED
          LENGTH        CDATA                             #REQUIRED
          EXPANDABLE    (true|false)                      #REQUIRED
        >
```

Snipped from the manually created DTD file for the OPX interface

- A DTD file consists of a root element:
  - (Sub)elements and attributes are defined top-down
  - Element and attribute names have to be unique
  - Frequency indicators may be used to enforce that certain elements are used (multiple times) or restrict an element to a single occurance
  - Atttributes may be either arbitrary strings or enumerations
- For older optiSLang versions 3.2.x DTD files are located in the installation directory
- For the OPX interface a DTD file may be create manually based on the interface description

OPX example

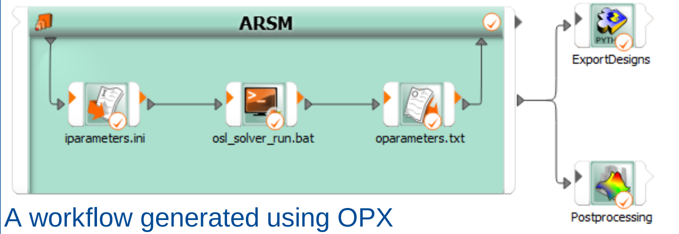
Mubea
light.efficient.global.

```
<!ELEMENT EDGES (EDGE)+>
  <!ELEMENT EDGE EMPTY>
  <!ATTLIST EDGE
    FROM      CDATA #REQUIRED
    FROMSLOT  CDATA #REQUIRED
    TO        CDATA #REQUIRED
    TOSLOT    CDATA #REQUIRED
  >
```

Snippet from DTD file for the OPX interface for the element EDGES

```
<?xml version="1.0"?>
<OPX>
  <NODES>
    <NODE ID="Scenery">
      <NODE ID="ARSM">
        <NODE ID="iparameters.ini"/>
        <NODE ID="osl_solver_run.bat"/>
        <NODE ID="oparameters.txt"/>
      </NODE>
      <NODE ID="Postprocessing"/>
      <NODE ID="ExportDesigns"/>
    </NODE>
  </NODES>
  <EDGES>
    <EDGE FROM="ARSM" FROMSLOT="IODesign" TO="iparameters.ini" TOSLOT="IDesign"/>
    <EDGE FROM="iparameters.ini" FROMSLOT="ODesign" TO="osl_solver_run.bat" TOSLOT="IDesign"/>
    <EDGE FROM="osl_solver_run.bat" FROMSLOT="ODesign" TO="oparameters.txt" TOSLOT="IDesign"/>
    <EDGE FROM="oparameters.txt" FROMSLOT="ODesign" TO="ARSM" TOSLOT="IIDesign"/>
    <EDGE FROM="ARSM" FROMSLOT="OMDBPath" TO="Postprocessing" TOSLOT="IMDBPath"/>
    <EDGE FROM="ARSM" FROMSLOT="OBestDesigns" TO="ExportDesigns" TOSLOT="IDesigns"/>
  </EDGES>
  <NODE_ATTRIBUTES>
```

Snippet from an OPX file



A workflow generated using OPX

- The OPX file consists of the following elements
  - NODES (input/output files, solver etc.)
  - EDGES (connections between NODES)
  - NODE_ATTRIBUTES (detailed information for all prior defined NODES)
- Example:
  - The element EDGES contains a set of the subelement EDGE
  - The element EDGE has to appear at least once (frequency indicator +)

Conclusion

- ***Optimization algorithms are powerful tools in the design process with regards to light weight products***
- ***Given a parametrized structural element a generally vaild optimization workflow can be defined***
- ***Data bases may be used to administrate optimization tasks and results***
- ***Project specific OPX files may be created automatically and optiSLang may be controlled via batch und OPX files***
- ***Automated processes are more performant and less sensitive to errors than manual processing of optimization tasks***
- ***Automated processes may free bound resources and allow focussing on other tasks***

Thank you very much for your attention!