# Web based 3D Visualization for COMSOL Multiphysics®

Matthias Jüttner, Sebastian Grabmaier, and Wolfgang M. Rucker

Institute for Theory of Electrical Engineering, University of Stuttgart, Pfaffenwaldring 47,

70569 Stuttgart, Germany, http://www.ite.uni-stuttgart.de

# Introduction:

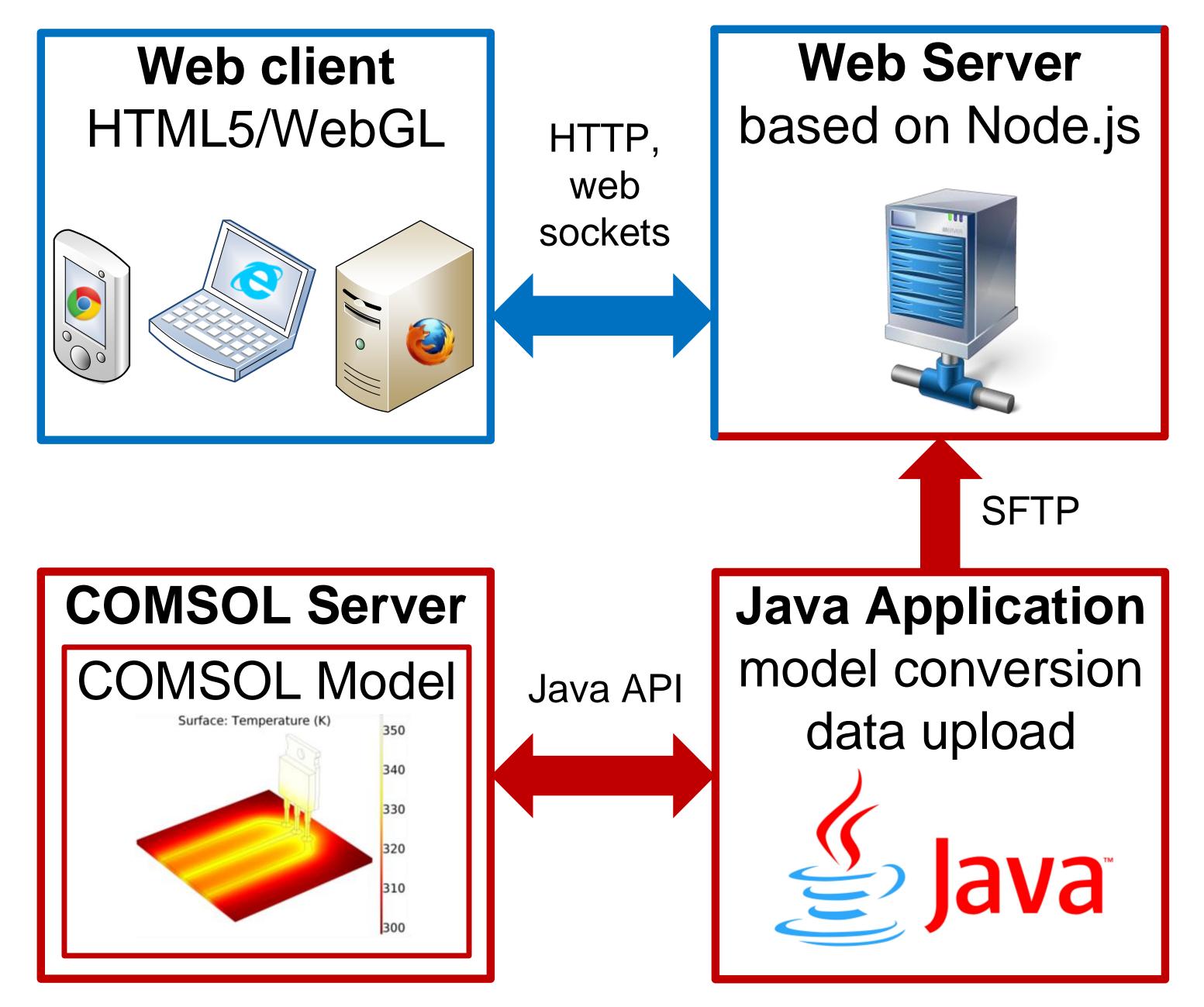
For teaching, product or research presentations and in future multiphysics problem solving environments an available and lightweight visualization solution is needed. Using standardized web technology a interactive web application for different plat-

# Web Application:

- WebGL is used to directly render the 3D data from COMSOL.
- Shading effects, arrow volumes, stream lines are calculated locally
- Optimized vertex and fragment shader
- Supports touch and common input

forms and its backend is presented.

### Visualisation System Setup



GPU	Frames per Second	Model
Nvidia	58.5	Power Transistor,
Quadro 600		Surface &
PowerVR	44.5	Arrow Volume
SGX544		ca. 600,000 Triangles

#### Table 2. Performance of the render

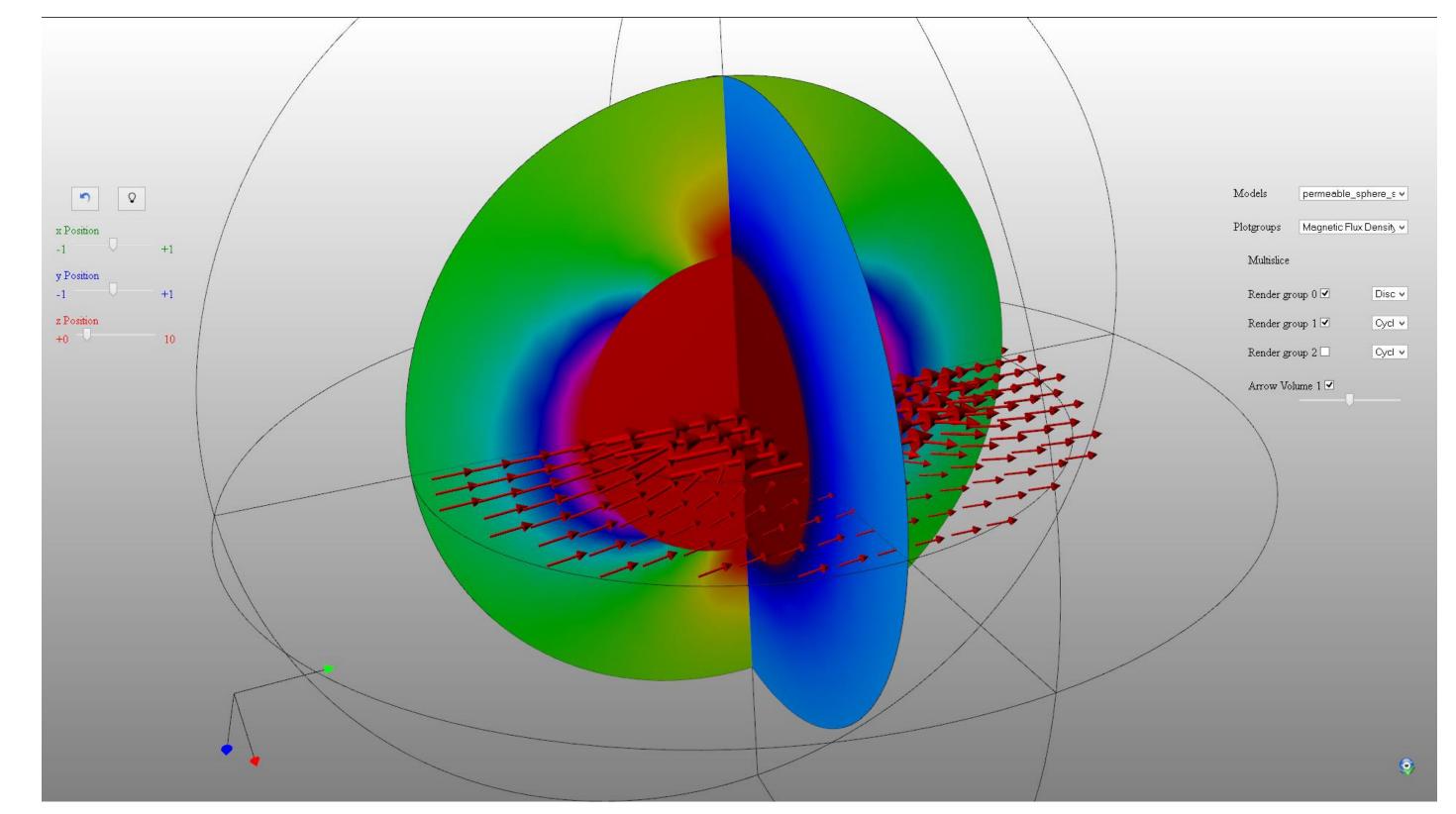


Figure 1. System architecture

### Web Server:

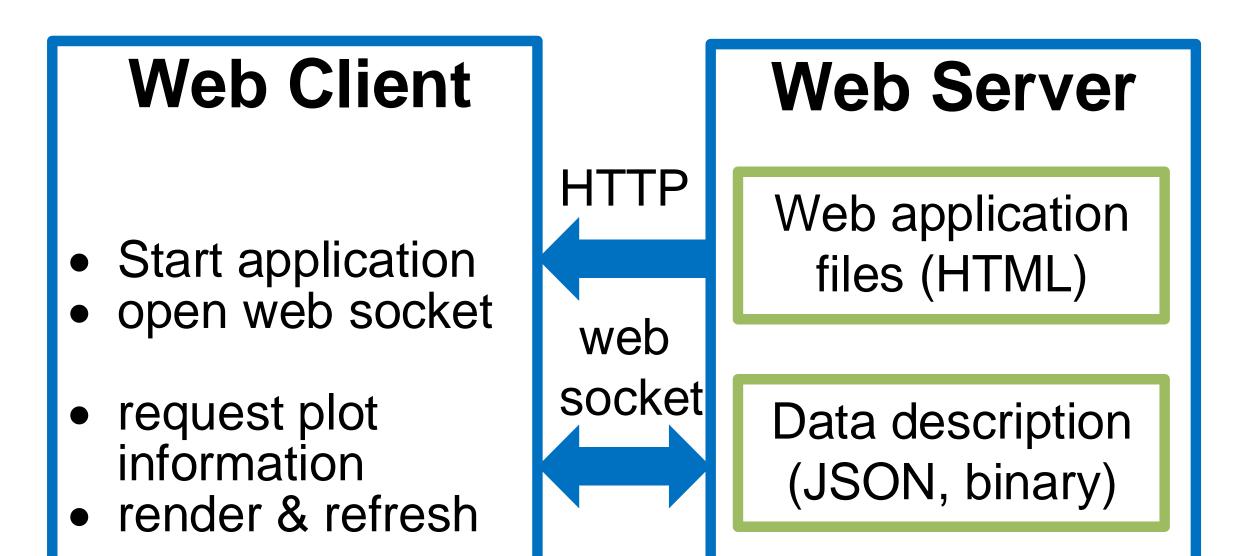


Figure 3. Web application with visualization

# Java Application:

File Edit Connection Comsol Models 🖲 🖬 Config.json Comsol Models inductive\_heating.mph (root) Inductive\_heating coil\_above\_plate.mph (root) Temperature, 3D (ih) eddy\_currents.mph (root) 🗎 Group 0 🛚 Surface 1 1D Plot Group 2 Number of Vertices: 10730 Model name: inductive\_heating.mph (root) Number of Plotgroups: 2 Number of Elements: 20512 This Render data contains Color Attributes Config.json Model eddy\_currents loaded.

Figure 4. Preprocessor

- Extract data from COMSOL Server
  Use Java API
- Upload graphics data to web server
- Multi model support

Figure 2. Web application initialization

Model name	Plot type	Size [kB]
Inductive Heating	Surface	290
<b>Power Transistor</b>	Surface	898
<b>Power Transistor</b>	Arrow Volume	120

 Table 1. Transmission size of plot data

### **Conclusions**:

The developed system for web visualization works well. By using WebGL and web sockets the demands on performance and bandwidth are minimal. The handling is intuitive due to modern and common input techniques.

**Excerpt from the Proceedings of the 2014 COMSOL Conference in Cambridge**