Development of an interactive simulation environment in Python

Complex physical models are becoming increasingly used in industry for simulation and optimization. Modeling languages such as Modelica allows model libraries of physical components, which in turn can be used to compose system models of, e.g., vehicle systems, power plants and electronic systems. In this project, the task is to develop a user-friendly application in Python that allows the user to simulate large-scale dynamic systems. Important aspects such as interactivity of the simulation, visualization and performance need to be considered. As a starting point for the project, existing software for simulating dynamic models in Python (specifically the Assimulo package) will be used. Also, the Functional Mock-up Interface standard will be used as an exchange format for compiled dynamic models: in this way models exported by several Modelica tools, e.g., Dymola and SimulationX, can be simulated in the application. The project is done within the scope of the JModelica.org open source project.

Student profile: One or two skilled and highly motivated students with interest in numerical algorithms and programming. Prior knowledge of the Python programming language and design of user interfaces is considered a merit, but is not mandatory.

Contact: Johan Åkesson, Modelon AB

High-performance storage of simulation data

Modern tools for simulation of large-scale dynamic systems are becoming a common technology in a wide range of industrial domains, including avionics, automotive, power plants, electronics and robotics. State of the art tools are capable of simulating models with more than 100,000 equations, which in turn generates large amounts of simulation data that needs to be stored in an efficient manner. To meet this need, a new data storage format is under development, combining two industrial standards: the Functional Mock-up Interface (FMI) for exchange of compiled dynamic models, and HDF5, which is a general purpose storage format based on XML and binary data storage. The task in this project is to implement support for the new storage format in the open source Modelica platform JModelica.org. The project includes a number of challenges, e.g., development of a user-friendly software design, learning two industrial standards, and obtaining high performance. The implementation will be validated on industrial grade simulation models.

The project is done within the scope of the JModelica.org open source project.

Student profile: One skilled and highly motivated student with interest in numerical algorithms and programming. Prior knowledge of the C programming language, Modelica and XML is considered a merit, but is not mandatory.

Contact: Johan Åkesson, Modelon AB
Interfacing external models in Modelica

Tool interoperability and model exchange is getting increasingly important in industrial design processes in a wide range of domains including automotive, avionics, electrical, robotics and power plants. There are several tools for simulation of dynamic models on the market, all with different specialities, and it is commonly recognized that one single tool cannot meet all needs. Therefore, increased attention has been given to development of tool-independent model exchange formats, e.g., the Functional Mock-up Interface (FMI). The task in this project is to develop a module in the JModelica.org open source platform for importing FMI compliant models into Modelica models. This includes automatic generation of a Modelica model based on FMI models and implementing a simulation run-time system in Modelica. The task includes several challenges in the areas of simulation of large-scale dynamic models, programming and tool integration. The resulting implementation will be tested on realistic industrial models to validate functionality and performance.

The project is done within the scope of the JModelica.org open source project.

Student profile: One skilled and highly motivated student with interest in numerical algorithms and programming. Prior knowledge of the C programming language, Modelica and XML is considered a merit, but is not mandatory.

Contact: Johan Åkesson, Modelon AB

Optimization interfaces for large-scale dynamic models

Optimization of dynamic models is an increasingly used technology in engineering design processes. Both off-line applications, such as parameter identification and design optimization, and on-line applications such as model predictive control are of interest. In order to make efficient use of state of the art numerical optimization algorithms, development of standardized model and optimization interfaces are important. In this project, the industrial standard for exchange of compiled dynamic models, the Functional Mock-up Interface (FMI) will be extended to support also optimization. This includes design and implementation of a C API for cost functions, constraints and optimization tuners. The starting point for the work is the existing FMI standard and previous work on optimization done in the JModelica.org platform. The proposed design will be validated on industrial grade benchmarks.

The project is done within the scope of the JModelica.org open source project.

Student profile: One skilled and highly motivated student with interest in numerical algorithms and programming. Prior knowledge of the C and Python programming languages is considered a merit, but is not mandatory.

Contact: Johan Åkesson, Modelon AB

Real-time simulation of physical models

Dynamic models of technical systems, e.g., cars, power plants and robots, are commonly used in the development of control systems. The use of detailed physical models decrease time to market, control systems designs can be evaluated in computer simulation prior to deployment. Another industrially widespread technology for assessing correctness of the implementation of a control system is Hardware In the Loop Simulation (HILS), where a model of the physical system to be controlled is simulated, in real-time, in a computer and the control system executes on the target hardware. This scenario is challenging for the simulation software, since the dynamic system, often of large scale, needs to be integrated in real-time. The task of this project is to implement symbolic and numerical techniques for
real-time simulation. These include graph algorithms for inline integration and tailored numerical integration schemes. The design will be evaluated on industrial grade benchmark problems.

The project is done within the scope of the JModelica.org open source project.

**Student profile:** One or two skilled and highly motivated student with interest in numerical algorithms and programming. Prior knowledge of the Java and C programming languages and Modelica is considered a merit, but is not mandatory.

**Contact:** Johan Åkesson, Modelon AB

---

**Eclipse-based editor for embedded HTML documentation in Modelica code and for graphical representations for Modelica classes**

The proposed master’s thesis project is in two parts, both concerning user-friendly editing of auxiliary information stored in Modelica code in Eclipse.

One part concerns documentation for Modelica models. It is embedded in the Modelica code in HTML form. The object of this part of the project is to render and edit this documentation in a user-friendly manner. This entails creating a specialized Eclipse view that uses some existing framework for rendering HTML, possibly after adding some auto-generated information to the documentation. It should also be possible to edit it in a simple manner, preferably without requiring the user to know HTML (i.e. a WYSIWYG-editor).

The second part concerns the graphical representations of Modelica classes. Since Modelica is primarily intended to be edited graphically, the graphical representations are very important. They are stored in text form as a collection of graphical primitives. The object of this part is a graphical editor for editing these graphical representations, using a framework for graphical editors for Eclipse. Some of the parts for this are already present, such as the graphical primitives from the code.

**Student profile:** One or two motivated and skilled students with interest in IDE development, compilers, and Java programming.

**Contact:** Jesper Mattsson, Modelon AB

---

**Robust Modelica parser using Bridge Parsing**

The proposed master's thesis project targets the development of a more robust parser for Modelica than the one currently used in JModelica.org. By using bridge parsing or other parser error recovery techniques, the parser's ability to recover from syntax errors would be improved. This also includes integration of the new parser into the existing compiler and editor. Improving the parser serves two goals:

- To improve the usability of the Modelica editor by ensuring that it always has the information necessary to provide support function like code completion and cross referencing.
- To improve the amount of feedback to the user for syntax errors, probably to about the same level as the standard Java compiler.

**Student profile:** One highly motivated and skilled student, who has taken the course EDA180 “Compiler Construction” (preferably with good grades), and with interest in compilers and Java programming.

**Contact:** Jesper Mattsson, Modelon AB