Framework of the program interface of the simulator

start
Main
Relevant methods:
`static void main(args)`

---

ui.commandline
CommandLineSystem
Relevant methods:
`File checkLoadFile(args)`
`boolean run(args)`
`protected List<> getCommands()`

- Contains a list of all command-line commands
- Checks if a file is to be loaded or if a command is to be executed.

(SimTools) systemtools.commandline
AbstractCommand
Relevant methods:
`String getName()`
`String prepare(…)`
`void run()`

- Abstract base class for all command-line commands.
- The commands only need to define a name, the requested parameter types and have to implement a run() method.

Folder containing help files

---

ui.help
Help
Relevant methods:
`static void topic(parent, topic)`
`static void topicModal(parent, topic)`

- Displays a page from the online help.

---

ui
MainFrame
- Displays the main window.
- Contains a panel of type MainPanelBase, which shows the window contant.
- Menu and toolbar will be displayed in the window and outside MainBasePanels, but have to be provided by the panel.

---

ui
MainPanel
Relevant methods:
`JToolBar createToolBar()`
`JMenuBar createMenu()`

- Content of the program window.
- Contains editor, wait panel and statistics viewer.

---

ui.compare
ComparePanel
- Allows to compare different statistic results.

---

ui.tools
WaitPanel
Relevant methods:
`void setSimulator(Simulator,Runnable)`

- Shows the simulation progress.
- Changes to the statistics viewer when simulation is done.

---

ui
ModelViewerFrame
- Displays a window for viewing the model connected to some statistics results.
- Contains a read only EditorPanel object.

---

ui
EditorPanel
Relevant methods:
`protected void buildGUI()`
`protected void loadGUIDataFromModel()`
`protected void writeGUIDataToModel()`
`void setModel(…)`
`EditModel getModel()`
`String loadModel(File)`
`String saveModel(File)`

- Complete model editor.
- Comminicates by the shown methods with MainPanel.

---

ui.statistics
StatisticsPanel
Relevant methods:
`void setStatistics(…)`
`Statistics getStatistics()`
`String loadStatistics(File)`
`String saveStatistics(File)`

- Complete statistics viewer.
- Communicates by the show methods with MainPanel.

(SimTools) systemtools.statistics
StatisticNode
- Represents a node in the statistics tree.
- Contains one concrete viewer.

---

simulator.editmodel
EditModel

---

simulator.statistics
Statistics

(SimSystem) statistics
Statistics*PerformanceIndicator

---

ui.statistics
StatisticViewer*
- Base class for displaying statistics texts, tables and diagrams.
- Gets the statistics object and prepares data on this basis.
- StatistikPanel can build a tree from this viewers and will also offer methods for saving as pdf, docs, xlsx and html.

Simulator

SimSystem and SimTools

Framework of the core of the multi-core event-driven stochastic simulator

GUI

xml Package

Methods for reading and writing xml files (including the ability to read/write to or from zip files).

Simulator

- This base class combines all simulation parts.
- The relevant sub classes are: RunData, RunModel and Statistics.

Relevant methods:
```
Simulator(editModel,null)
prepare()
start()
isRunning()
finalizeRun()
getStatistics()
```

Statistics*PerformanceIndicator

- Multiple classes for recording counters, distributions, ...
- Only a name for the corresponding xml-tag has to be given, then loading/saving can also be automated.

EditModel

Relevant methods:
```
clone()
compareToEditModel(model)
```
+ Methods for loading and saving

- Data as free texts
- Version identification to be able to warn when loading new models into old program versions
- Functions for cloning and comparing models (helpful in the editor)

EditModel

Statistics

Statistics

- Can record data during simulation thread local
- Allow combining multiple statistic objects

Relevant fields and methods:
```
editModel
addData(statistics)
calc()
```
+ Methods for loading and saving

Per Thread:

SimulationData

- Contains the event manager etc.
- The simulation is controlled by the methods provided by this class.

Relevant fields and methods:
```
runModel
runData
statistics
eventManager
getEvent(eventClass)
loggingActive
logEventExecution(eventName,info)
```

Combining for global statistics

RunModelCreator

Statistics

xml Package

Methods for reading and writing xml files (including the ability to read/write to or from zip files).

RunModel

Relevant methods:
```
static getRunModel(editModel)
```

- Runtime model
- All stations etc. are connected by references
- Model is read only and will not be changed during simulation

RunModel

RunData

Relevant methods:
```
initRun(nr,simData)
doneRun(now,simData)
```

- Contains all dynamic simulation data
- Above Methods will be ran at the start and the end of each repitation. By this methods needed objects can be created.

Derived classes for events

Event

- Abstract Base class for all events

Relevant methods:
```
init(time) (instead of a constructor)
time
run(simData)
addNextEvent
```

mathtools Package

- **Probability distributions**: Classes for distributions, functions to load / save, compare, generate random numbers, graphical editors
- **Numbers and times**: Parsing and formatting functions
- **Tables**: Functions for managing, loading and saving tables

EventManager

Relevant methods:
```
addInitialEvents(list)
addEvent(event)
deleteEvent(event,simData)
```

Simulation procedure
```
EditModel editModel=new EditModel();
editModel.<…>
Simulator simulator=new Simulator(editModel,null);
String error=simulator.prepare();
if (error!=null) {System.out.println(error); return;}
simulator.start();
Statistics statistics=simulator.getStatistics();
statistics.<…>
```