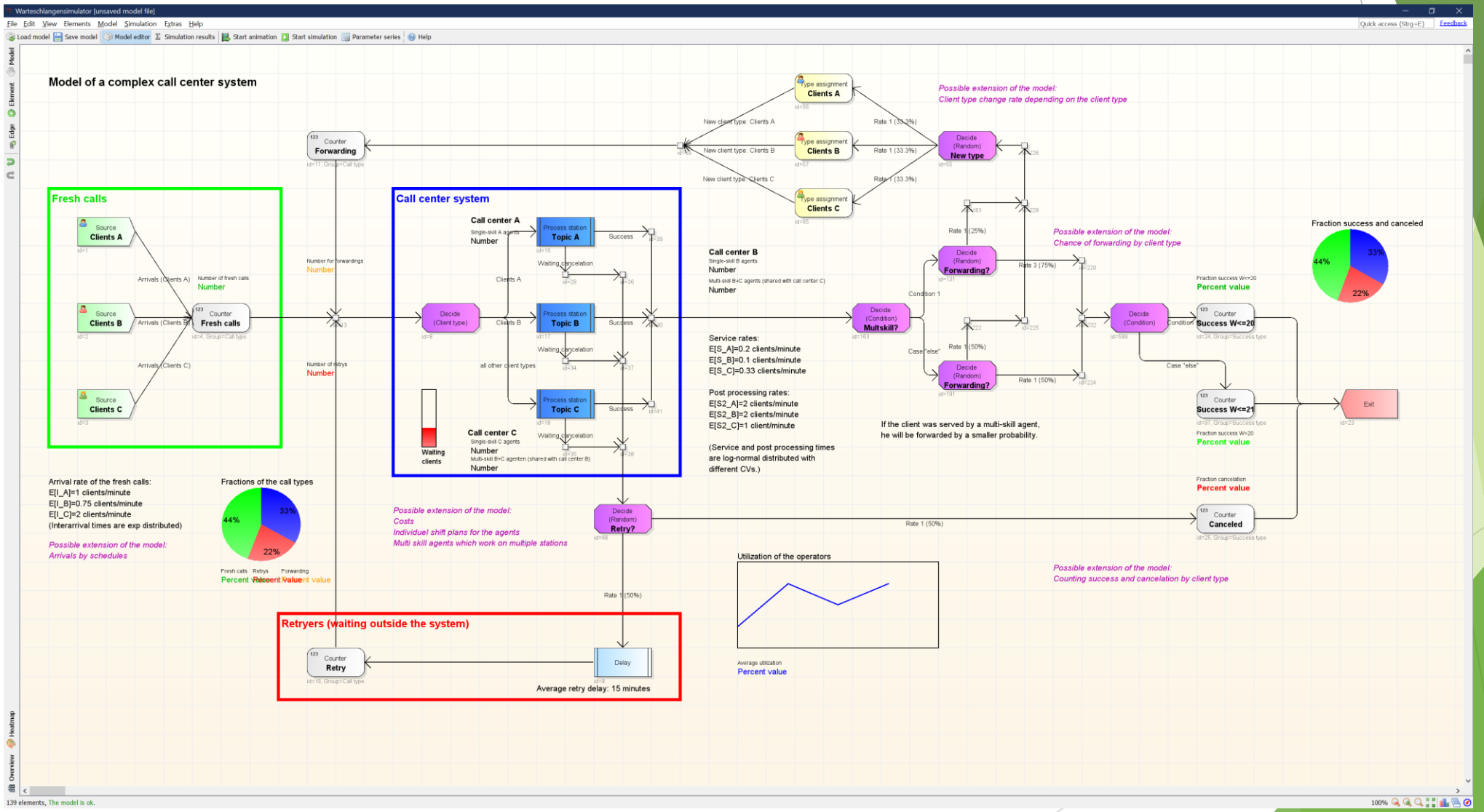
Warteschlangen-simulator

An open-source tool for modelling, simulation and animation of queueing processes

Alexander Herzog
TU Clausthal / SWZ
www.simzentrum.de

The graphical user-interface

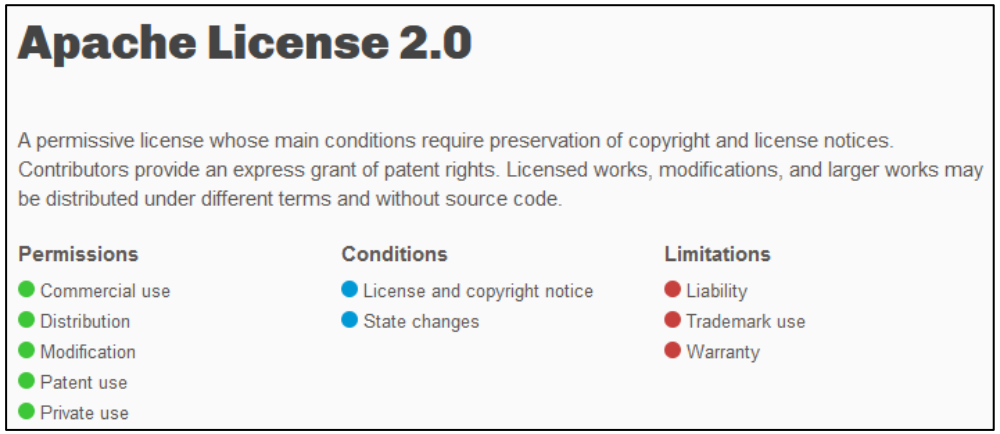


Overview

- Simulation and animation of queueing processes.
- Queueing processes are modeled in form of flow charts.
- No hard-coded model that can only be parameterized,
  but complete freedom in creating models
  (with on the other side no need to write program code).
- Full statistics recording during simulation.
  In program statistics viewer and report generator available.
- Automation (parameter studies, optimizations) available.
- Can be used as a desktop program with full graphical user-interface or in command-line mode (for example as part of a tool chain on a server).

License & download

- Warteschlangensimulator is **Apache 2.0 licensed open-source** which allows private as well as commercial usage.
- Hosted on GitHub.
- Binaries and source code available for download: **https://github.com/A-Herzog/Warteschlangensimulator**



Source: https://choosealicense.com/licenses/apache-2.0/

Technical data

- Written in Java, i.e. **platform independent** (Windows, Linux).
  Only a Java runtime in version 8 or higher is needed.
- Comes with full **graphical user-interface** but can also be controlled via **command-line** (for example model creation on a slow Windows desktop computer and than simulation of the model on a fast, remote Linux compute server).
- Windows version comes with full **installer**, uninstaller, auto updater etc.
- **Portable version** is also available.

Main features (1)

Model editor

- Working similar to vector drawing programs.
- Allows a fast overview of the model.
- But models can also be partially or completely generated by external programs for automation due to the open and documented xml file format for models.

Simulation

- High speed multi-core simulation.
- Automatic recording of all statistics indicators (no need to a priori define which indicators are of interest).

Main features (2)

Animation

- For easy understanding of model correlations – and because it's fun.
- Animation can be recorded as videos.
- Option to display live indicators in form of values, diagrams etc. directly on the model surface during animation.

Statistics

- Automatic recording of all statistics indicators during simulation.
- Presentation of the results as texts, tables and graphics.
- Option to directly copy the results to external programs like spreadsheet processors.
- Report generation and free assembly of results also available.

Main features (3)

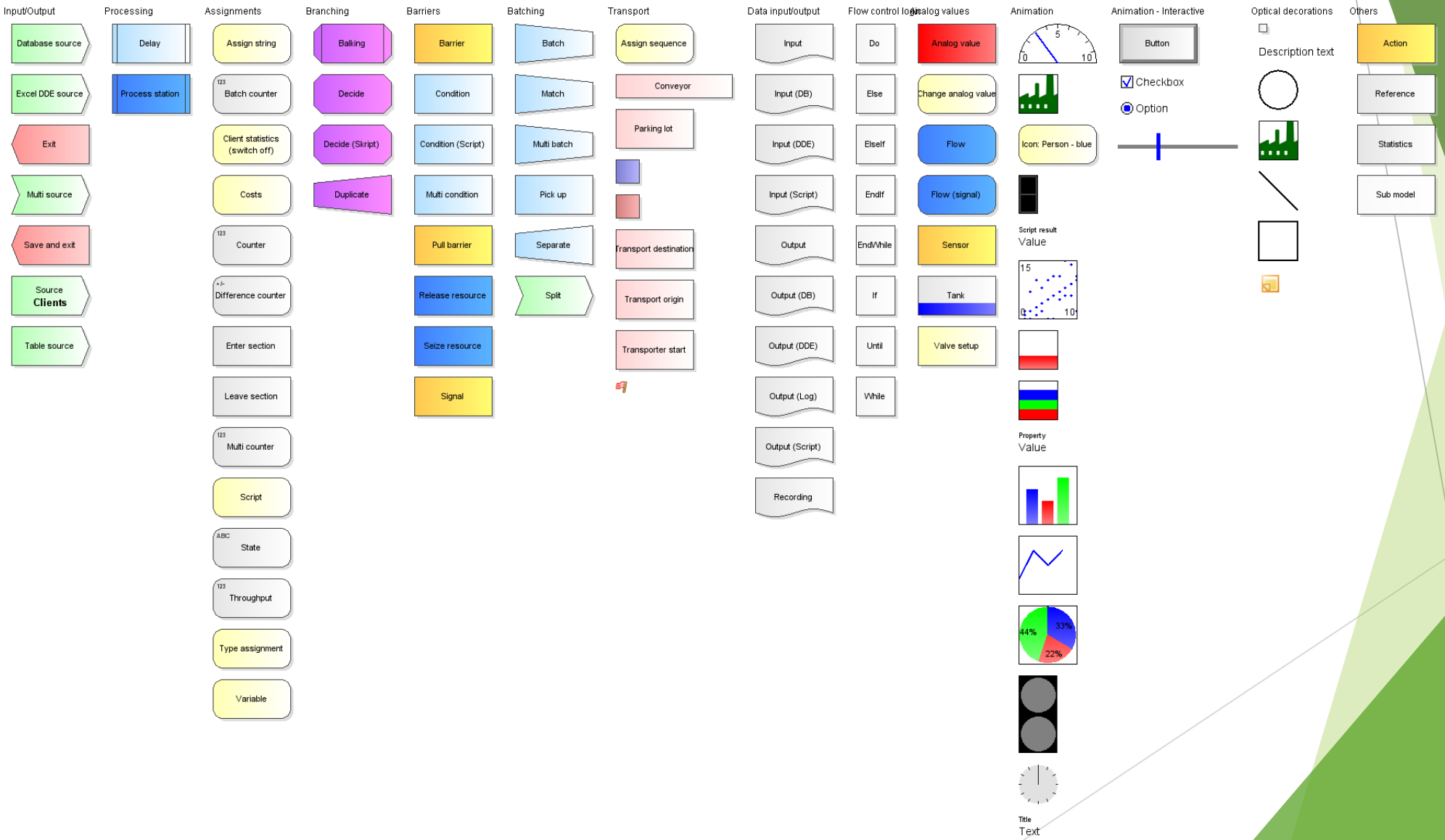
Parameter studies

- Easy creating of what-if studies.
- Results of multiple simulation runs can be combined directly in the user-interface – or of course exported.
- Parameter study setups can be stored and then executed on a faster remote computer.

Optimization

- Optimizer for finding best input parameter setups to maximize or minimize some performance indicators.
- Different optimization strategies including genetic algorithms available.

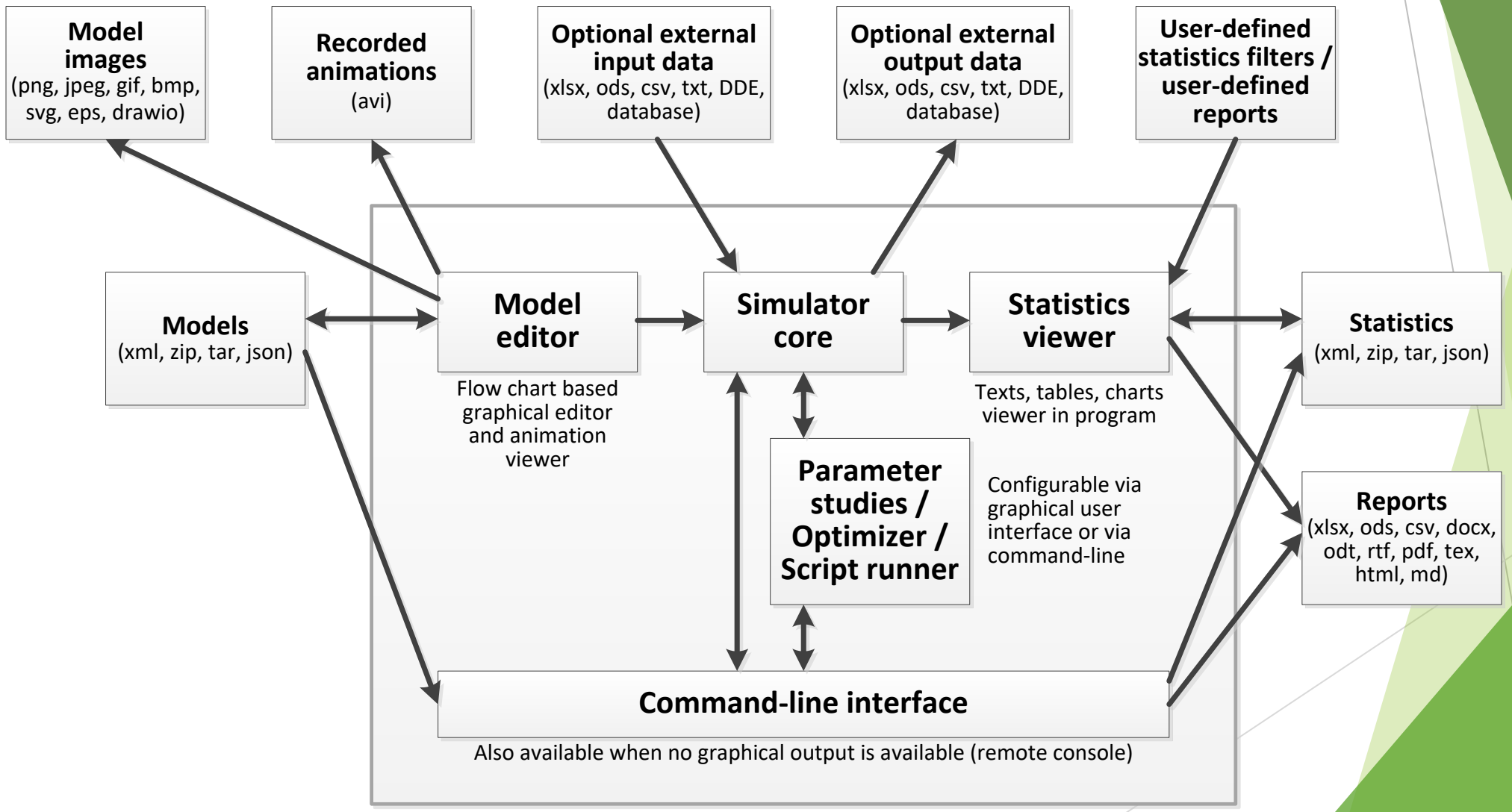
More than 100 station types available



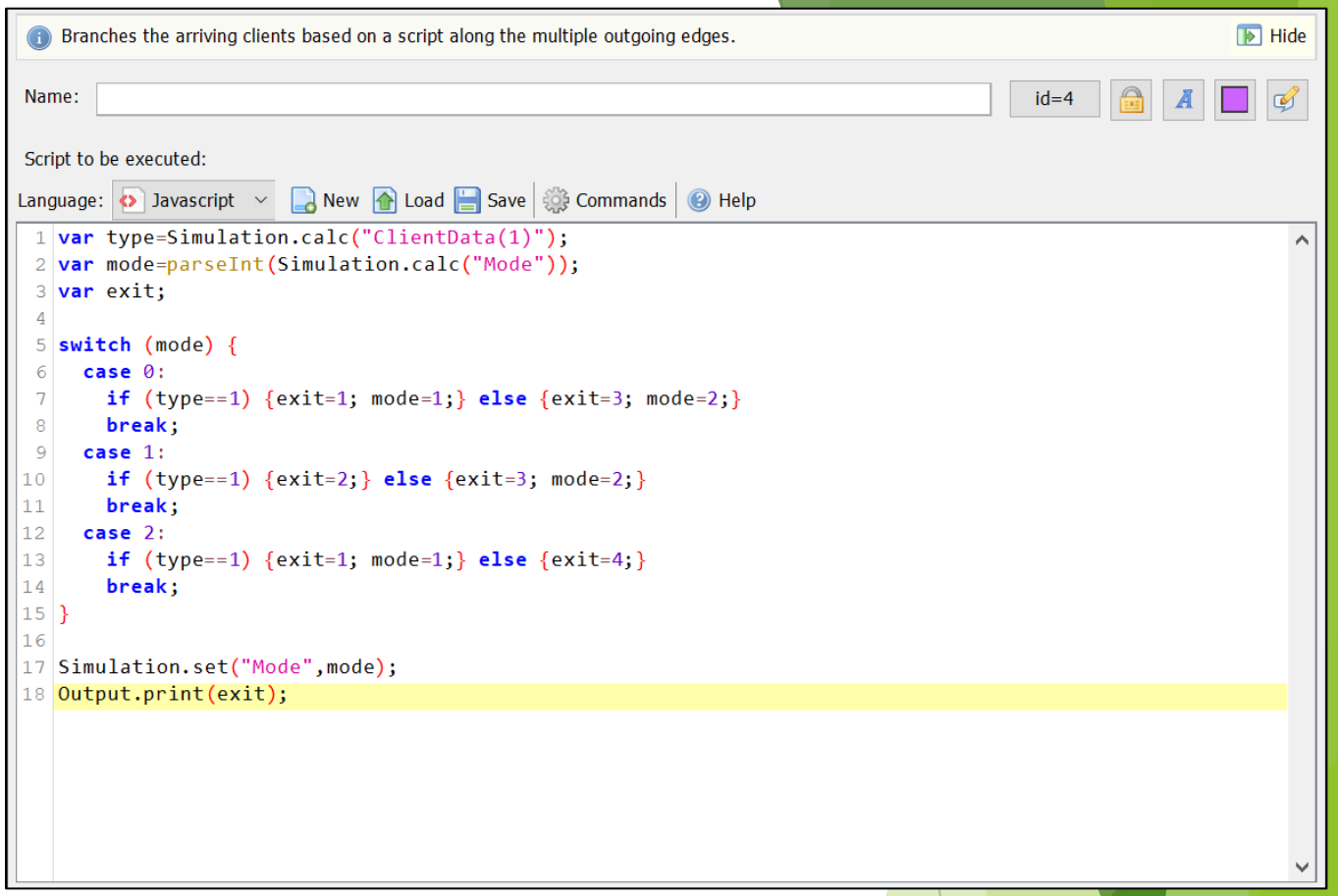
File formats (1)

- Native file format for models and statistics data: xml (dtd and xsd documentation available).
- Can read/write models and statistics also as zip, tar, tar.gz and json.
- Input and output of table data: xlsx, ods, csv, txt, via DDE from/to Excel, via database connectors.
- Output of statistics data: xlsx, ods, csv, txt, docx, odt, pdf, tex, html, md.
- Exporting models: different bitmap formats, svg, eps, pptx, html, drawio.

File formats (2)



Scripting



- Complex branching or holding clients rules can be defined by scripts.
- This gives the simulator strong modelling capabilities.
- Scripts can be written using Javascript or Java.
- Java code can use external class files.
- Syntax highlighting and code competition is available in the built-in editor.
- Java code is compiled at simulation start and therefore can be executed with native system speed.
- Scripts can also be used for filtering statistics data.

Server mode

- Of source slow and fast computers can be combined by transferring model files created on the slow desktop computer to the fast remote server and there executed via command-line interface.
- But Warteschlangensimulator also has three built-in servers for simplifying this:
  - When starting the built-in **web server** Warteschlangensimulator will accept model files via a web interface and offer the statistics results of the finished simulations for download. The web server also offers a **REST interface**.
  - The built-in **simulation server** allows to directly connect two instances of Warteschlangensimulator. The desktop instance (where the user-interface runs on) will transparently use the calculation power of the remote server when starting a simulation.
  - The Warteschlangensimulator simulation server can also register as a **MQTT client** at a MQTT server to get simulation tasks via MQTT.
- Server modes can be started directly (via GUI or command-line) but scripts for making Warteschlangensimulator **Docker** images are also available.

Database connection

- Client arrivals and variable values can be read from database tables.
- Variable values and also any calculation results can be written to database tables during simulation.
- Directly supported database formats: SQLite (direct file access), HSQLDB (direct file access and via network connection), PostgreSQL (via network connection), MariaDB (via network connection), Firebird (via network connection), Access (direct file access).
- More databases can be accessed via JDBC: Just a driver jar file and a record in the configuration file is needed.

Performance

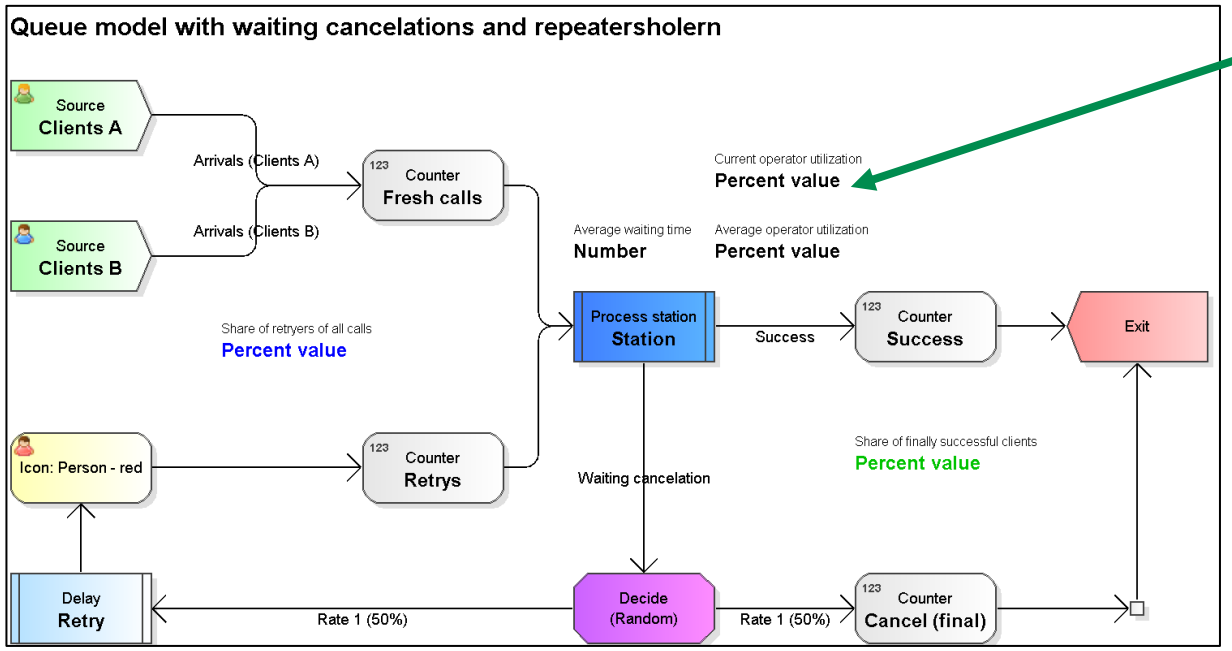
- Typical runtimes for models with 1.000.000 – 10.000.000 simulated arrivals: 5 – 30 seconds.
- This allows a very interactive analysis of the models (no need to wait for hours for the results of a simulation).
- Multiple CPU cores will automatically be utilized when ever possible.
- Warteschlangensimulator is 10x – 100x faster than many commercial event-driven simulation tools.

Support

- Online help (English and German) available in the program.
- Optionally an instant help can be displayed next to the program window on selecting stations.
- Interactive tutorial available.
- Many example models loadable direct from the file menu.
- Pdf documentation (short introduction, reference of all station types, reference of all calculation and scripting commands, reference of all command-line parameters and hotkey reference) available.
- Wiki on GitHub homepage.
- Introduction videos available on GitHub homepage.
- Source code is 100% JavaDoc documented.

Usage example: Impatient customers (1)

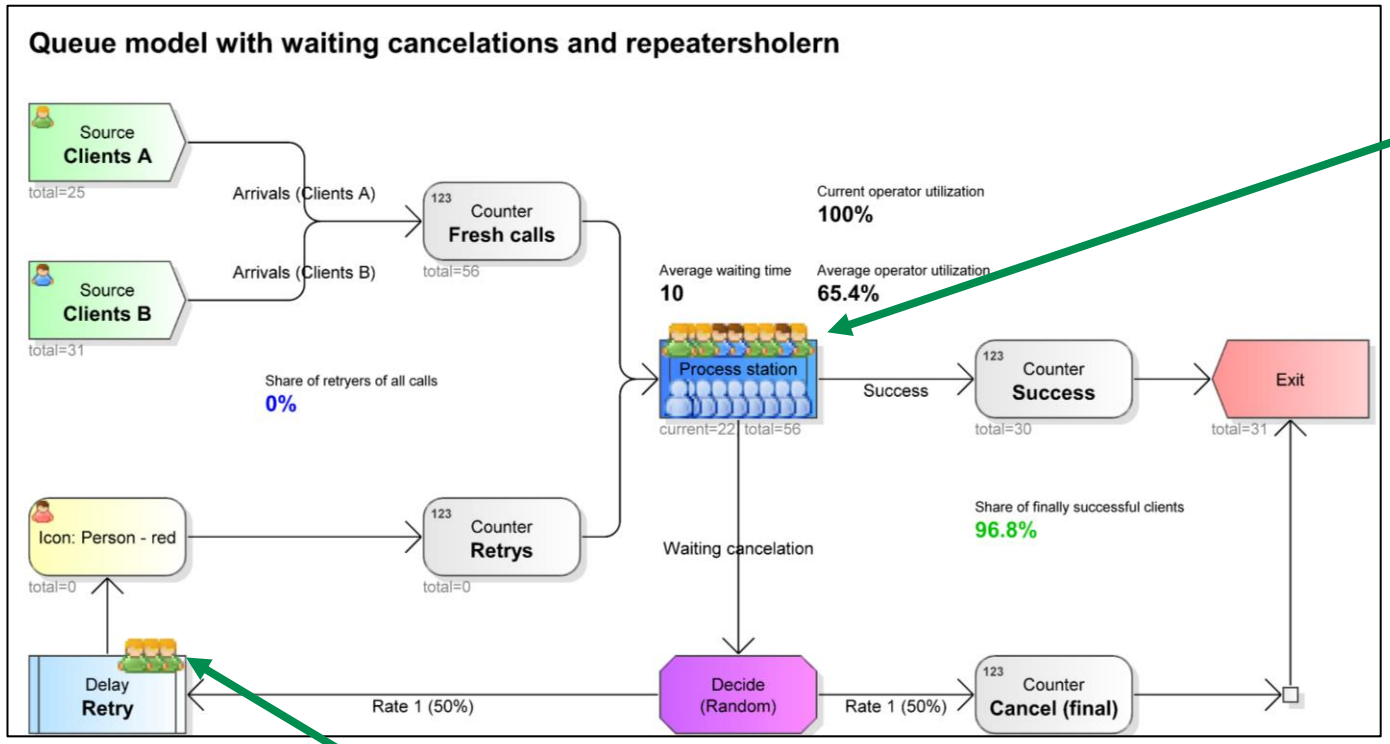
- On high operator utilization (= long queues), some customers will cancel waiting.
- A part of these cancelers will start another attempt later (and thus further increase the total load of the system).



Screenshot from model editor, therefore no values here yet.

Usage example: Impatient customers (2)

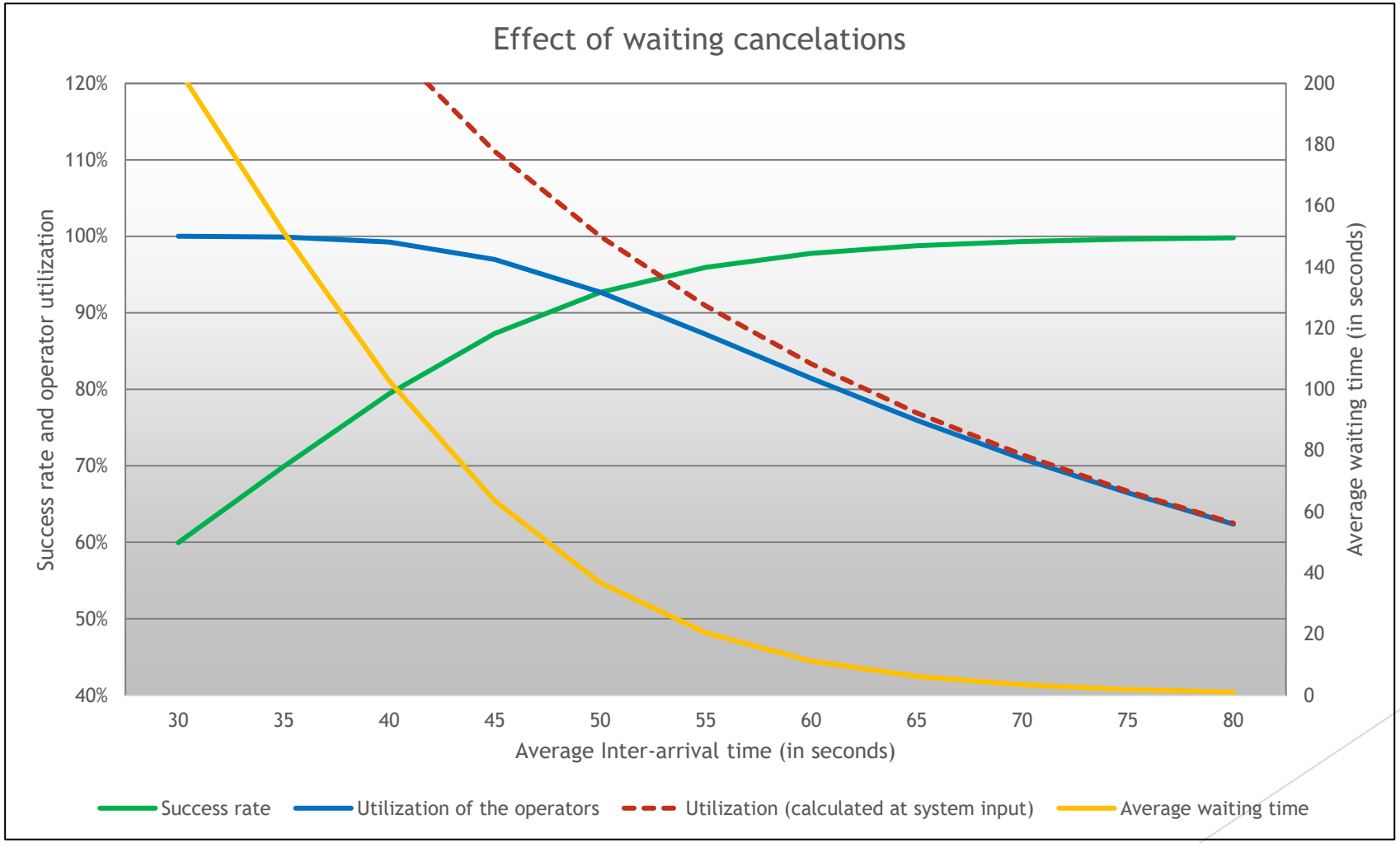
Screenshot of a running animation:



Clients waiting and in service process

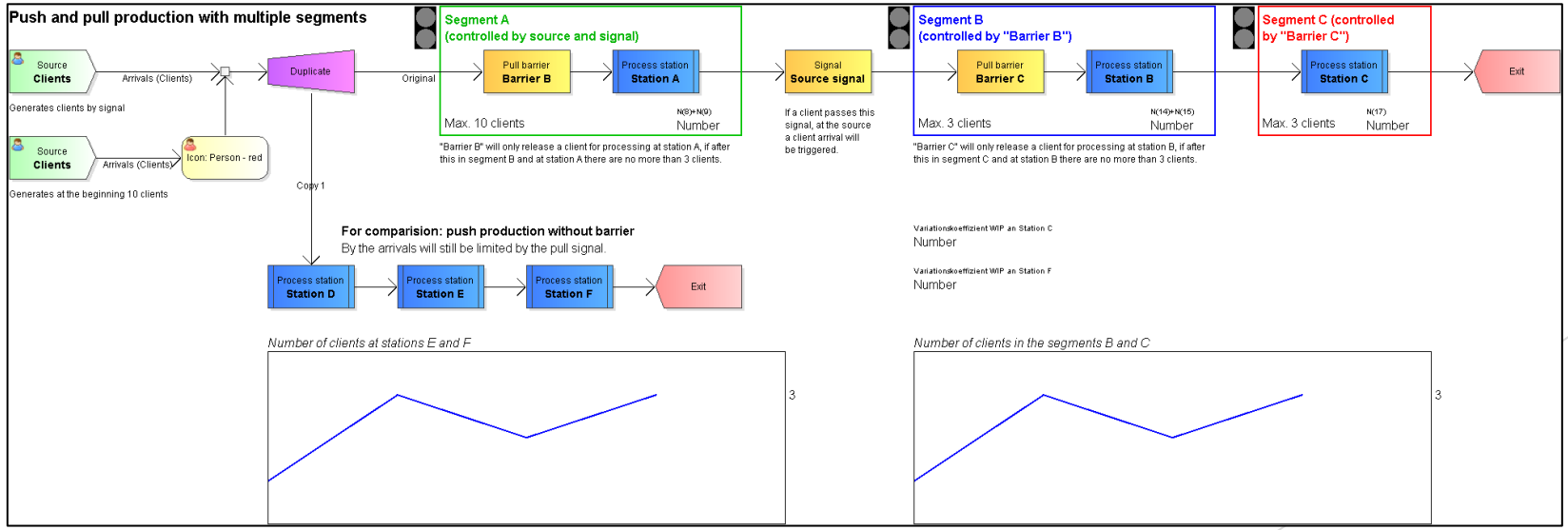
Clients starting a new attempt soon

Usage example: Impatient customers (3)



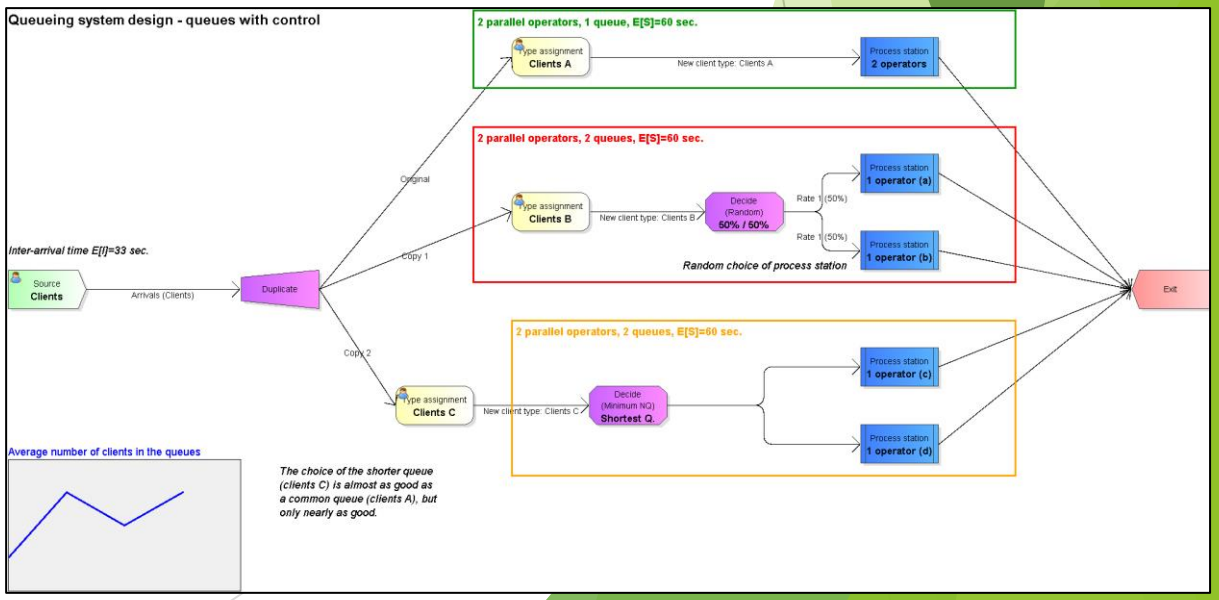
Usage example: Push/pull production

- By reducing the number of workpieces at the individual stations, it is possible to reduce the inventory in circulation and thus the residence times.
- At the same time, the variation in the residence times is reduced (= better delivery reliability).
- However, a too strong reduction of the maximum stock level leads to the fact that the available service capacity can no longer be fully utilized.



Usage example: Assignment strategies

- In many cases the arriving customers are distributed to several process stations (several cash registers, several servers behind a load balancer, several partial call centers in a complex system, several parallel machines on a production line, ...).
- If the distribution of the customers has to take place as soon as they arrive, the effect can occur that customers will wait in one queue while another operator is in idle mode.
- Simulation can be used to investigate how strong these effects are and whether it is worth changing the overall system (for example to a common queue) to remedy this.



More typical research questions (1)

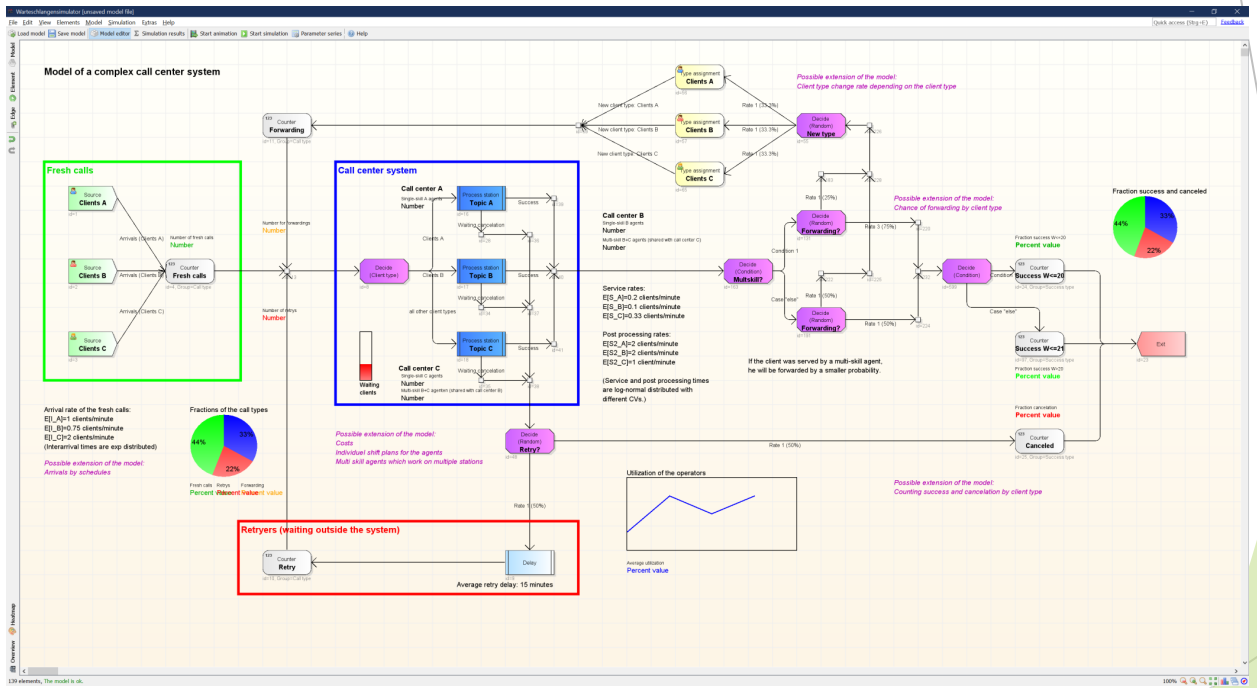
- **Queuing networks**
  Changes in one place can have effects in very different other places.
- **Different customer types**
  If different types of customers or products are treated in the same system, they influence each other. Different prioritization allows virtually different service levels to be generated for different customers and still make optimal use of the available service.
- **Batch arrivals and batch service**
  If the customers do not arrive individually or if the service is provided in groups, additional waiting times will occur.

More typical research questions (2)

- **Transports**
  Components are often transported between process stations via conveyors or vehicles. These do not only have a limited capacity, but must also be in the right place at the right time.
- **Combination of partial components**
  If components (e.g. car body and engine) have to be brought together at one station, the slower line blocks the faster one.
- **Shift plans / failures**
  Often operators are not available continuously. Conversely, the operating performance can possibly also be actively adapted to the respective arrival rate.

Warteschlangensimulator

- Easy to use
- Fast
- Platform independent
- Open-source



https://github.com/A-Herzog/Warteschlangensimulator