Workshop for advanced users

Payment and settlement simulator
seminar - 2006

Matti Hellqvist, Bank of Finland

Topics

- Forthcoming, new and advanced features
- Performance optimization
- Some database tools and tricks
- Process flow and logic of the simulator
- User modules

New features in v. 2.2.0

- 64-bit version available
  - plus MySQL 5.0 compatibility & Java update to JDK 1.5.0
- New algorithms
  - Group Codes i.e. DVP-linking of arbitrary number of transactions (for GC_MNS i.e. ”optimization”, ask from BoF)
  - Exact liquidity injections
- GUI updates
  - SystemID definition

Features added in v. 2.0.0

- Intrady liquidity management features
  - Known limits for multilateral or bilateral flow of liquidity or granted credit can be replicated
  - Bilateral limits in v.2.0.0
- Bilateral statistics
  - Statistics on intraday flow on bilateral level: (Set large enough values for bilaterla limits)
  - Can be utilized e.g. to reveal bilateral limits used by the participants by observing history of actual bilateral balances
- DB defragmentation
  - Reported to prevent slowing down of simulations in projects with massive data sets and repeated simulation runs.
- Time / Date transpositions

Some existing possibilities

- Assessing scenarios in network of interlinked systems.
  - Simulations with several interlinked systems with different logic are possible. See e.g. examples 2&3
- Borrowing liquidity from specified account when necessary
  - Mimic behavioral responce to liquidity shock (?)
  - implemented with ”Liquidity injections”
  - Cost or friction terms are lacking however.

More existing possibilities

- Simulation batches
  - Combine multiple (timely non overlapping) simulations into one.
  - Define batch from a set of simulation ID:s.
    => Changes in underlying data or system specifications create efficiently new scenarios.
- Arbitrary queue order
  - LIFO, Smallests first, alphabethical…
  - All these in strict queue order or bypass-mode
  - Use QUUSEDEF algorithms and import the sorting criteria in user defined fields.

Coming soon…

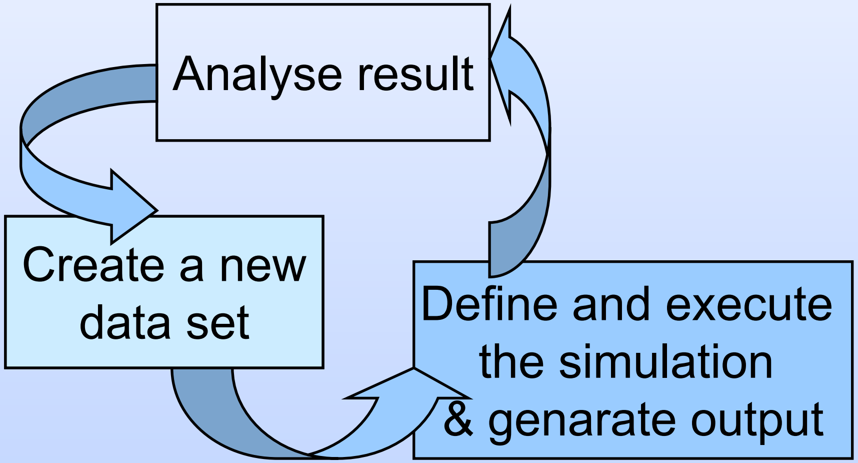
- Script language or ”command interface” for BoF-PSS2
  - Enhanced batch run possibilities
  - Integration of BoF-PSS2 to other analysis software (e.g. Matlab) via command interface
  - Fully automatic calculation of e.g. periodical paysys statistics is possible. ”STP simulations”
- Network topology analysis tool of input / output data
  - Stochastic data creation based on network structure (?)

Script language continued

- Server-client structure, similar as in MySQL
- Commands for most common and repetitive tasks
    - data import
    - simulation configuration
    - simulation execution
    - Output statistics export
- One-off actions not included (yet)
    - Creation of import/export templates
    - System definitions
    - Creating new project

Script language example

1. Start the BoF-PSS in server mode
2. Create the txt-file with your batch of commands
3. Drop the command txt-file to Bo-PSS2 client

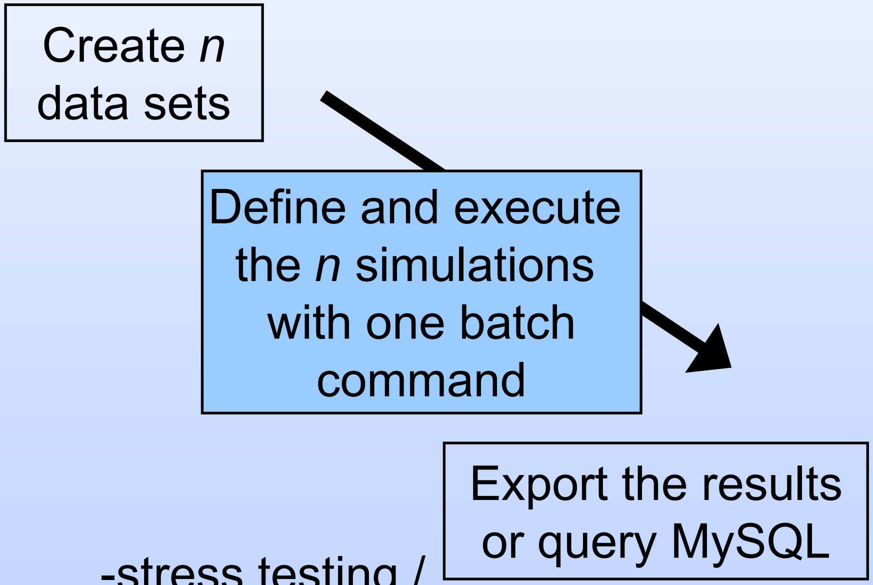


Analyse result

Create a new data set

Define and execute the simulation & genarate output

-Behavioral reactions in data creation
-Optimization



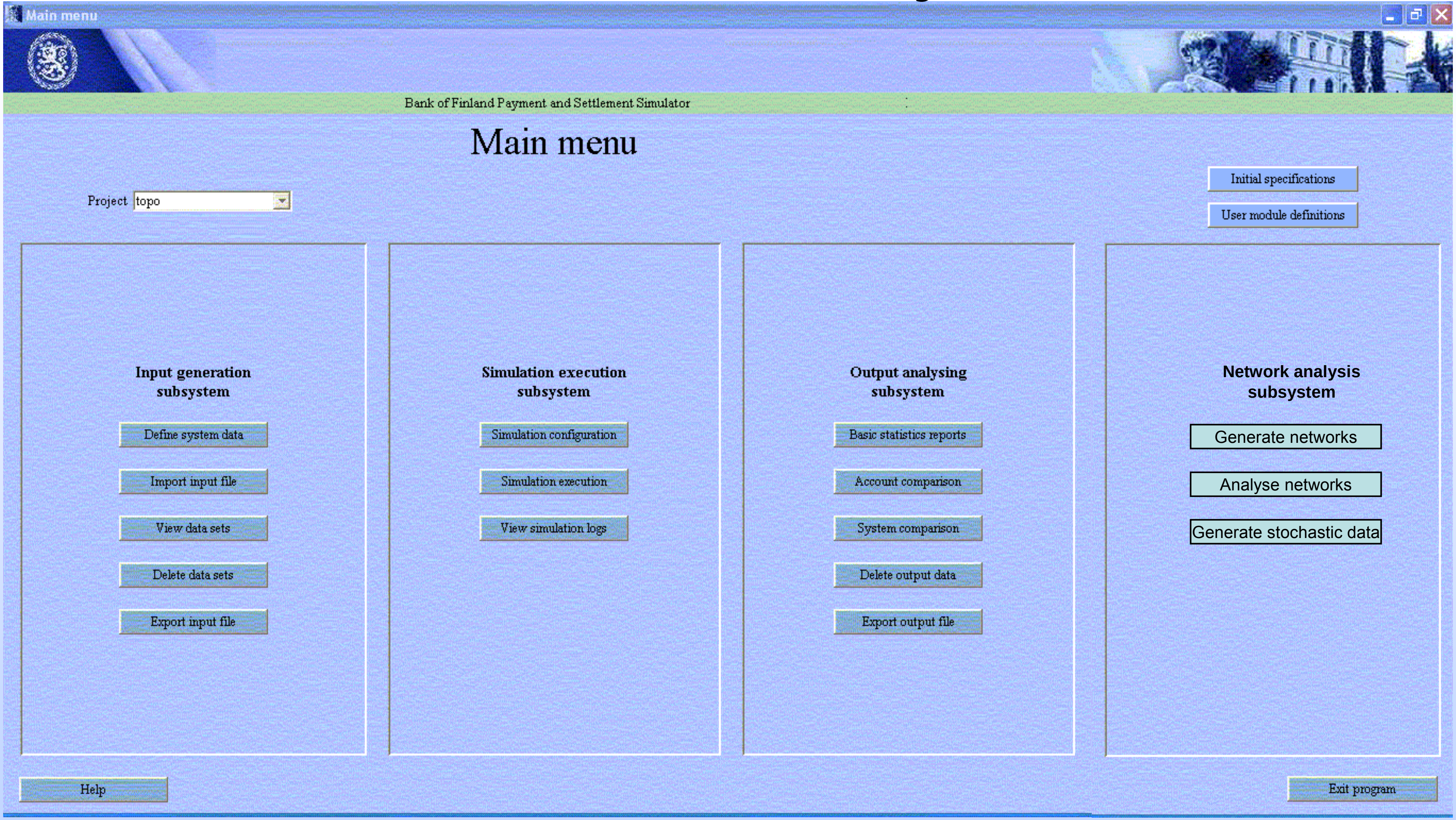
Create *n* data sets

Define and execute the *n* simulations with one batch command

Export the results or query MySQL

-stress testing / distribution of impacts

Network analyser



Main menu

Bank of Finland Payment and Settlement Simulator

Main menu

Project topo

Initial specifications

User module definitions

Input generation subsystem

Define system data

Import input file

View data sets

Delete data sets

Export input file

Simulation execution subsystem

Simulation configuration

Simulation execution

View simulation logs

Output analysing subsystem

Basic statistics reports

Account comparison

System comparison

Delete output data

Export output file

Network analysis subsystem

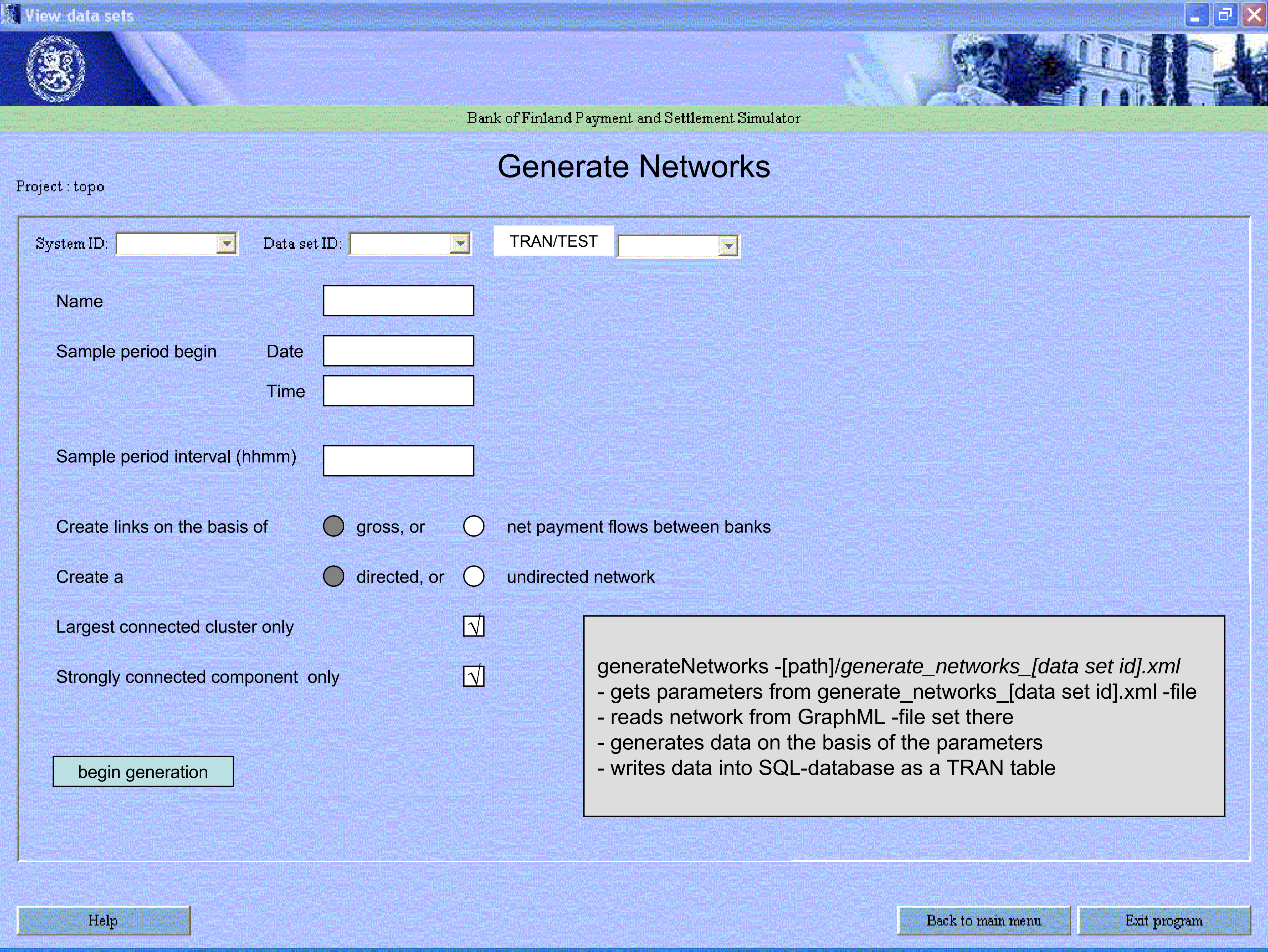
Generate networks

Analyse networks

Generate stochastic data

Help

Exit program



Bank of Finland Payment and Settlement Simulator

Generate Networks

Project : topo

System ID: Data set ID: TRAN/TEST

Name

Sample period begin Date

Time

Sample period interval (hhmm)

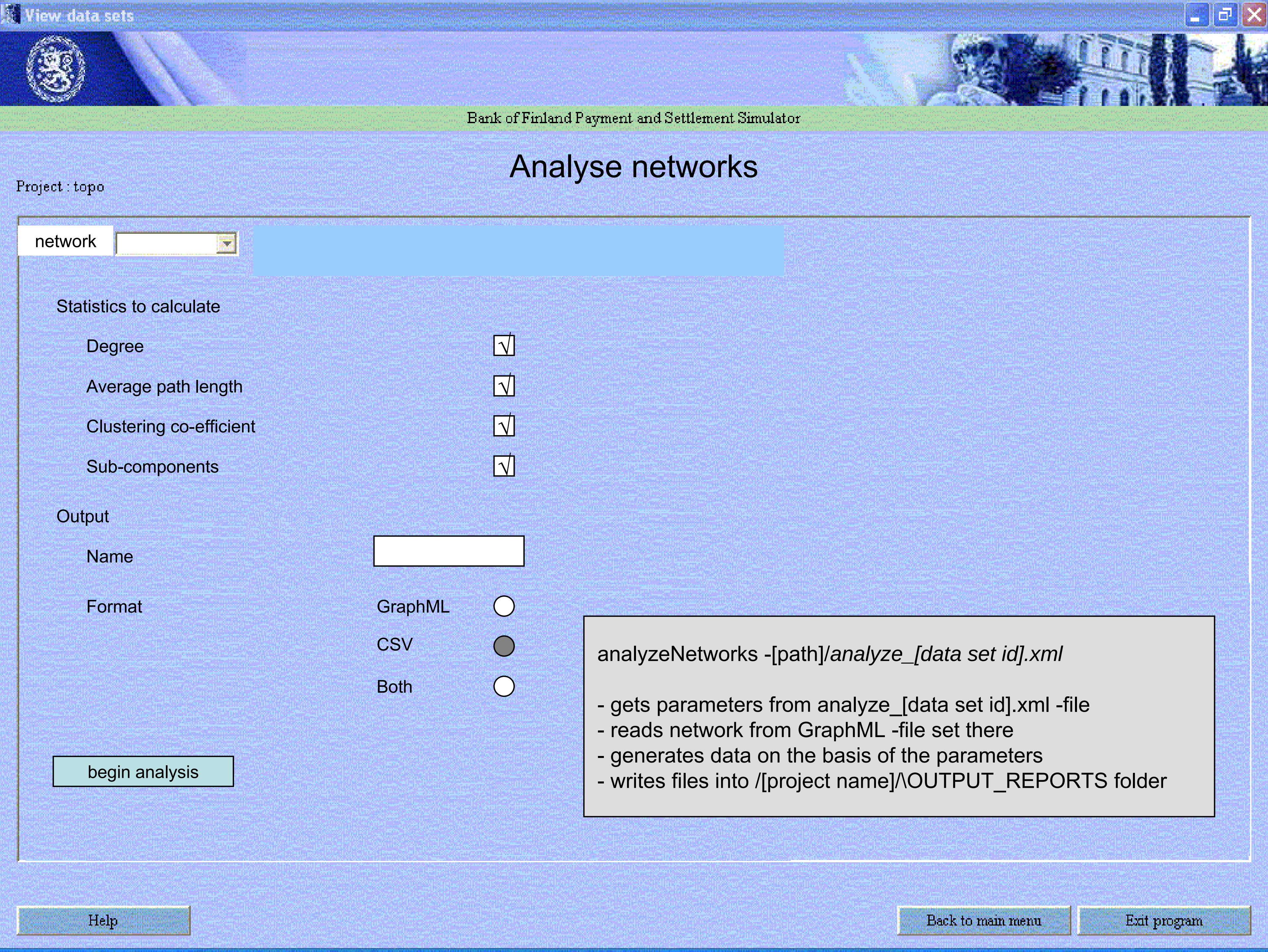
Create links on the basis of ● gross, or ○ net payment flows between banks

Create a ● directed, or ○ undirected network

Largest connected cluster only ☑

Strongly connected component only ☑

generateNetworks -[path]/*generate_networks_[data set id].xml*
- gets parameters from generate_networks_[data set id].xml -file
- reads network from GraphML -file set there
- generates data on the basis of the parameters
- writes data into SQL-database as a TRAN table



Analyse networks

Project : topo

network

Statistics to calculate

Degree ☑

Average path length ☑

Clustering co-efficient ☑

Sub-components ☑

Output

Name

Format

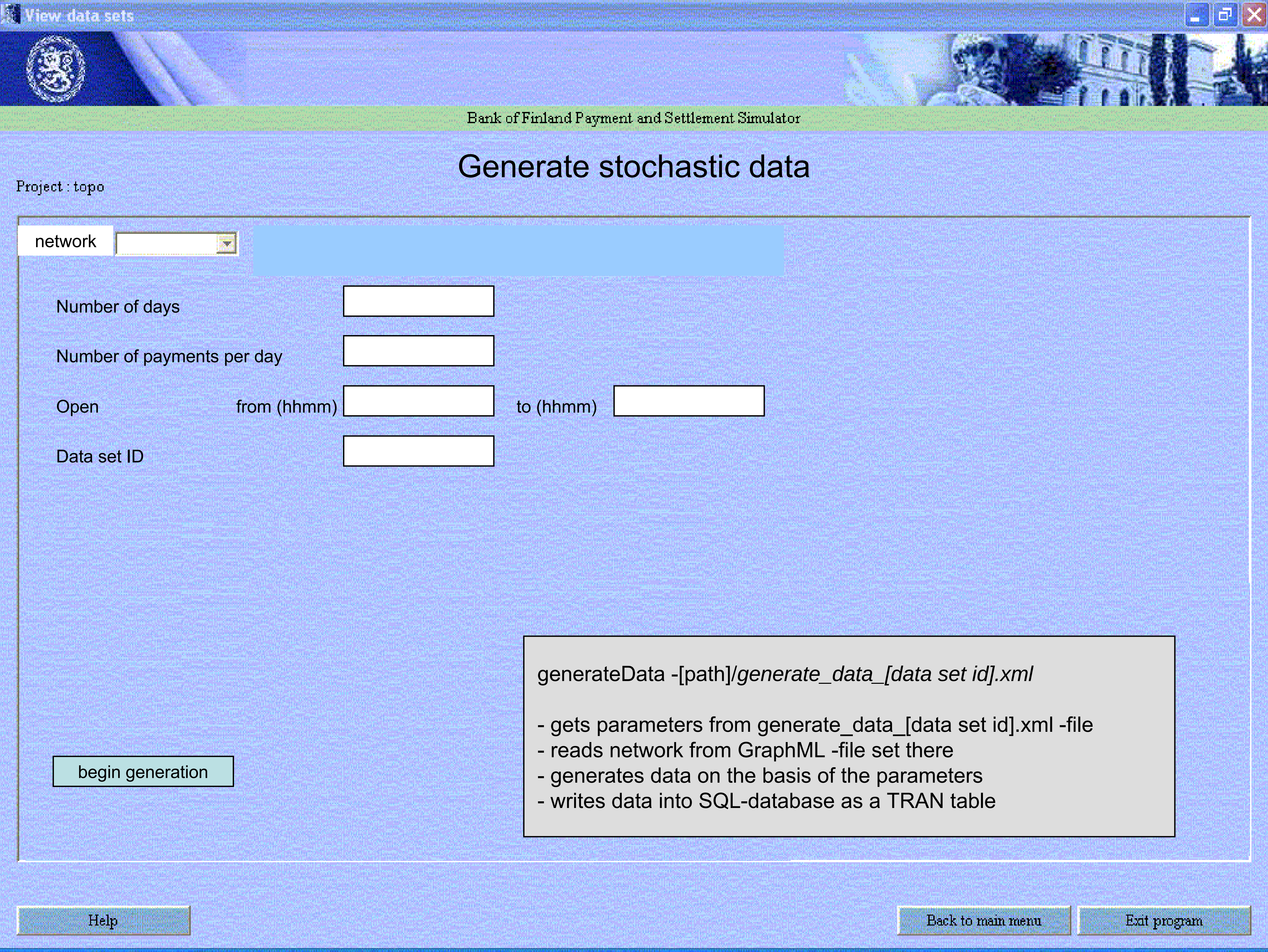
GraphML ○

CSV ●

Both ○

analyzeNetworks -[path]/*analyze_[data set id].xml*

- gets parameters from analyze_[data set id].xml -file
- reads network from GraphML -file set there
- generates data on the basis of the parameters
- writes files into /[project name]/\OUTPUT_REPORTS folder



Generate stochastic data

Project : topo

generateData -[path]/*generate_data_[data set id].xml*

- gets parameters from generate_data_[data set id].xml -file
- reads network from GraphML -file set there
- generates data on the basis of the parameters
- writes data into SQL-database as a TRAN table

Topics

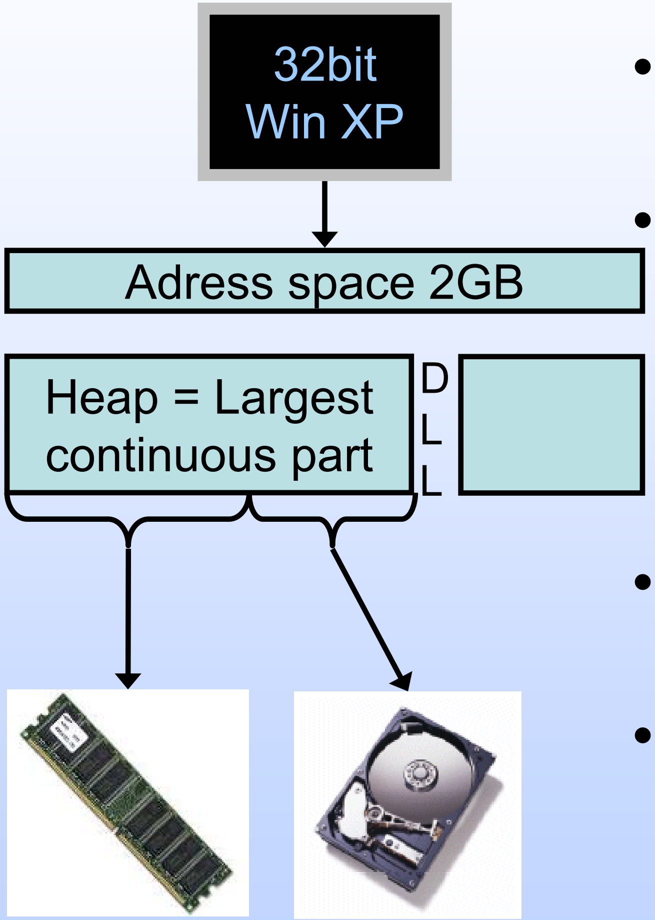
- Forthcoming, new and advanced features
- Performance optimization
- Some database tools and tricks
- Process flow and logic of the simulator
- User modules

When ”tuning” is needed?

- If you make hardware modifications after simulator installation
- If the simulations run out of memory due to
  - Increased number of transactions / day
  - Decreased liquidity and accumulating queues
  - Increased nr of liquidity constraints
    - Bilateral limits

Complex processing rules will allways take their time (DVP, continuous gridlock resolution etc.)

OS constraints



Available resources

- 32bit Windows => max 2G memory per application (”address space”)
- Java requires continuous block of address space for ”heap” = available amount of memory for e.g. the simulator.
  - Max ~1.5G
- Available ”virtual memory” used: RAM (+ Hard disk = ”paging”)
- 64Bit OS and hardware remove the 2G memory limit

One detailled explanation:
http://forum.java.sun.com/thread.jspa?threadID=584329&messageID=3009798

Start up parameters

- Simulator:

  c:\BoF-PSS\startup.bat

  ```
  "jre-1.3\bin\java -Xms128m -Xmx512m ..."
  ```

  - Xms = initial heap size
  - Xmx = Maximum heap size
- MySQL: c:\my.cnf
  - – For alternative configurations see: c:\bof-pss\program\ or c:\mysql\

Simulations with paging will be **severy** slower regardless which program is out of memory (MySQL or BoF-PSS)

Topics

- Forthcoming, new and advanced features
- Performance optimization
- Some database tools and tricks
- Process flow and logic of the simulator
- User modules

Managing projects

- Creating a project creates the structure of folders and databases.
- After this, contents can be changed simply by copying files

⇒Easy backups, cloning, transfering etc. of entire contents of a project.

(Handy also for reporting bugs)

Database tools and tricks

- Database of the simulator can be accessed directly for
  - Modifying the installation (templates, projects, user modules…)
  - viewing (or altering) the data directly
  - More powerful or tailored exports / imports of data

Database tools and tricks

- In practice
  1. Start the database server: c:\bof-pss2\program\database.bat
  2. Open viewing or editing tool. Command line and graphical versions available
  - Simulator must not be running (DB locking)
- **Carefully** with the direct modifications…
  - e.g. removing the project defined to be default will paralyse the simulator

MySQL tools

- Several easy to use monitor applications available
  - – MySQL Query browser (freeware by MySQL)

    http://www.mysql.com/products/tools/query-browser/

    (connect to ”localhost” as ”root”)
  - – MySQL Front, MyCon, … (Shareware)
- ODBC drivers for MySQL
  - – Allows connection with e.g. MS Access, SAS, Stata, …

    http://www.mysql.com/products/connector/odbc/

Topics

- Forthcoming, new and advanced features
- Performance optimization
- Some database tools and tricks
- Process flow and logic of the simulator
- User modules

The actual building blocks

Main algorithms

Submission: What happens next?

Common for all systems simulated concurrently

Entry: Initial processing for transactions: settle immediately if possible, call sub algorithms if defined or send to queue.

Settlement: Execute sub algorithms to settle trans from queue.

End: Perform final procedures of day or settlement period.

Sub algorithms

Queue:Settle individual transactions from queue in defined order.

Partial Net Settlement (PNS): Settle a subset of queued trans

Multilateral Net Settl. (MNS):Settle queues with ”All or nothing”

Bilateral offsetting: Match entered payments with queued payments having opposite direction (sender & receiver)

Splitting: Split larger trans into sub-transactions

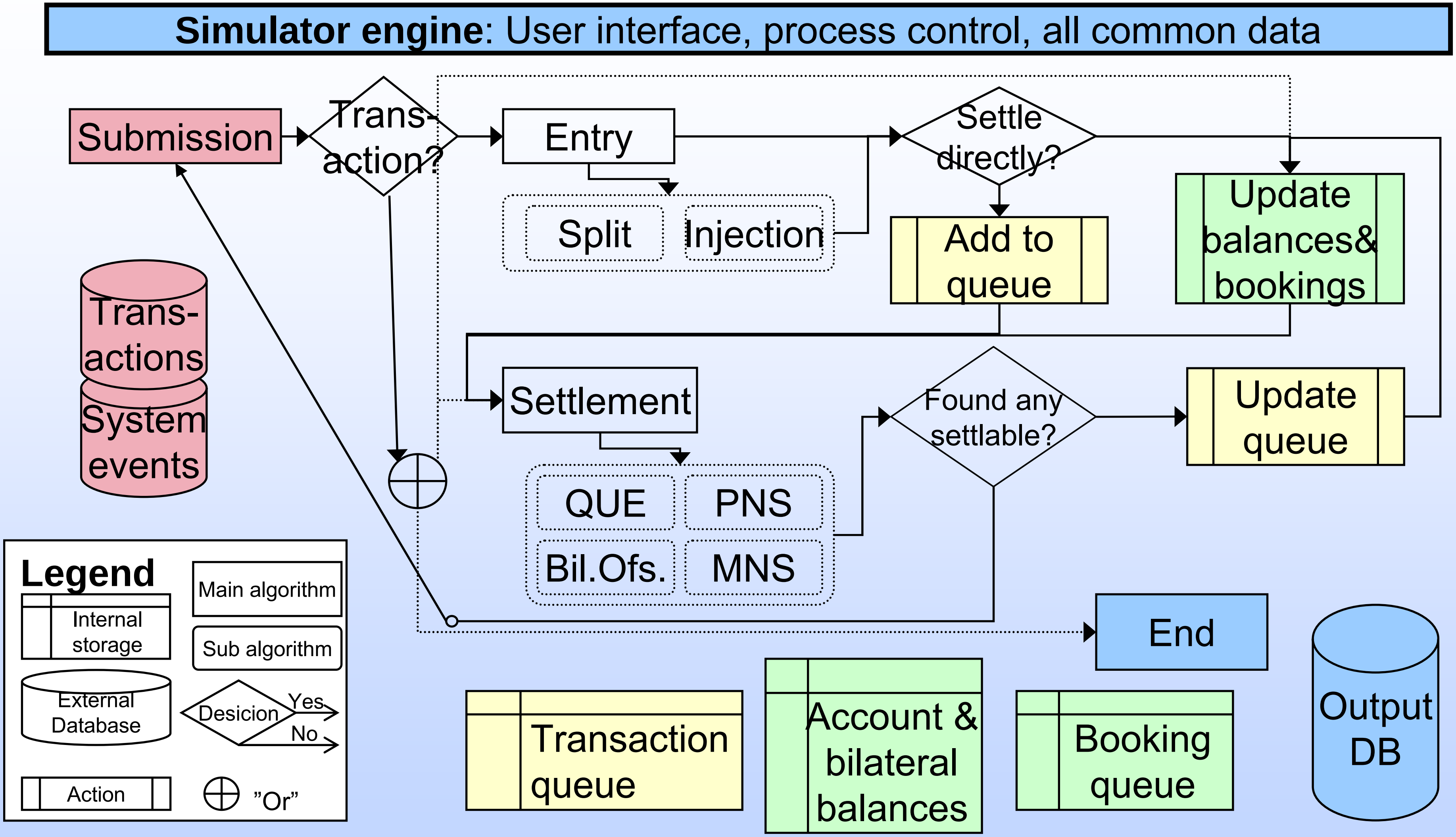
Injections: Perform liquidity transfers between defined accounts

Logics of one individual simulated system.

Some definitions

- ”Settling”: Booking or execution of a transaction. Account balances are updated.
- ”Netting” : Simultaneous settlement of independent transactions. Results in allowed balances for all involved accounts after all the transactions in the ”netting” are settled.

RTGS process



Subalgorithms are executed in the same order which they have in system definition

Topics

- Forthcoming, new and advanced features
- Performance optimization
- Some database tools and tricks
- Process flow and logic of the simulator
- User modules

User modules

- When existing logics/algorithms are too limited you can build new ones of your own
  - Setting up development environment
  - Case study: Group Codes
  - JavaDoc
  - Behavioural algorithms

Development environment

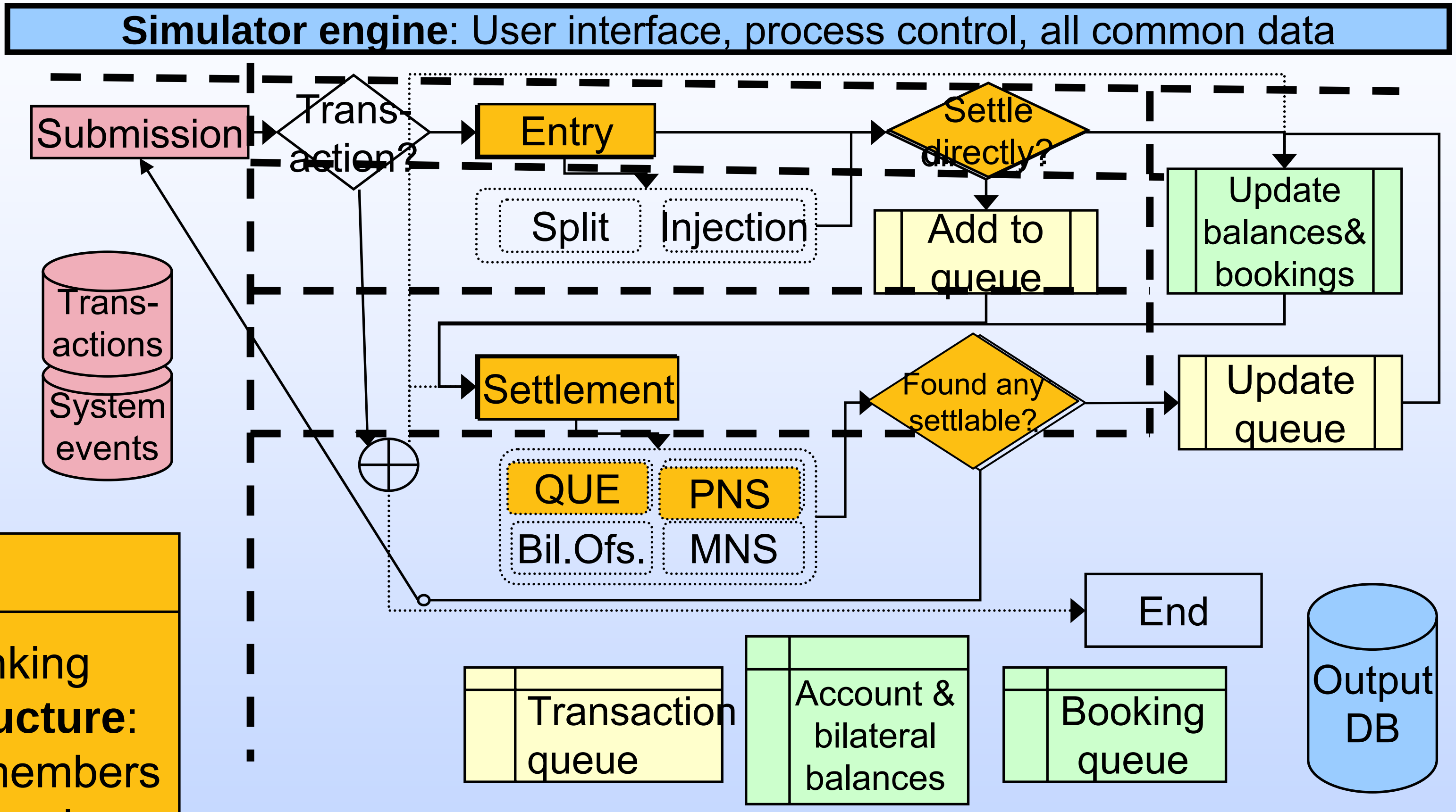
- Java coding is needed
  – Some tools to recommend: Jbuilder, eclipse (www.eclipse.org), NetBeans…

Case Group codes

- Purpose: To allow efficient linking of arbitrary many transactions together
  - DVP-link only works for pair of transactions
- Additional data required:
  - Group key, count of group members.

  ⇒Usercode 1 & 2 used to import these
- Efficient implementation requires a prepared data structure for linking transactions of one group together.

GroupCodes: implementation



Group linking **data structure**:
-List of members
-Nr. of members
-All submitted?

Java doc

- Documentation is available of the technical side of the simulator
  - Listings of all methods in all classes
    - Brief descriptions of the most important classes.
- (Extensive) HTML-document created automatically from code

Behavioral algorithms

- Natural place is in submission:
  - Observe the simulator situation & decide when to submit or delete transactions.
- Another way: Command interface
  - Behavioral reactions based on outcome of one simulation affecting the input of another
- Should be possible to write a interface algorithm and combine simulator with some other software.
    - e.g. Matlab has Java-application compatibility.