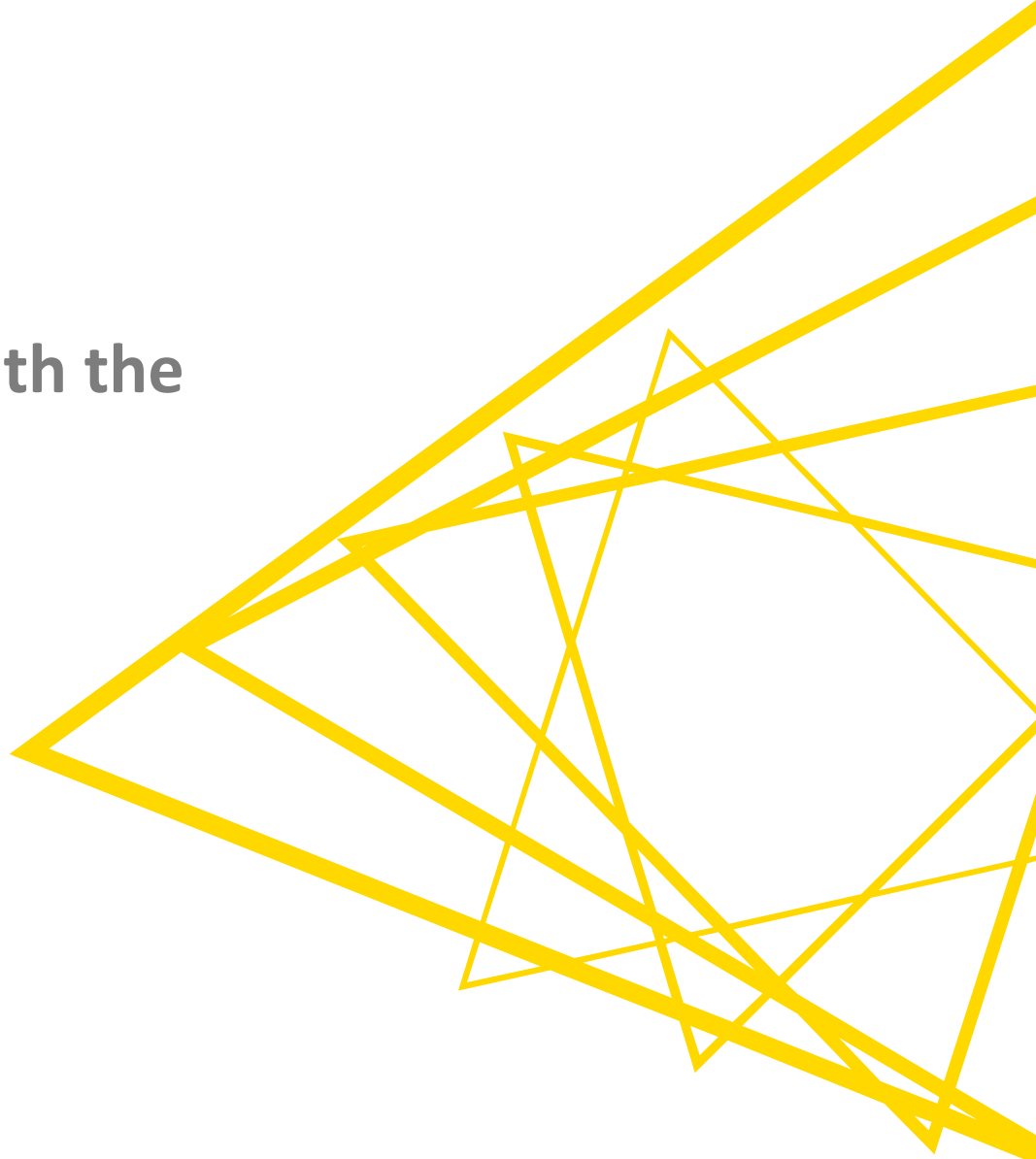
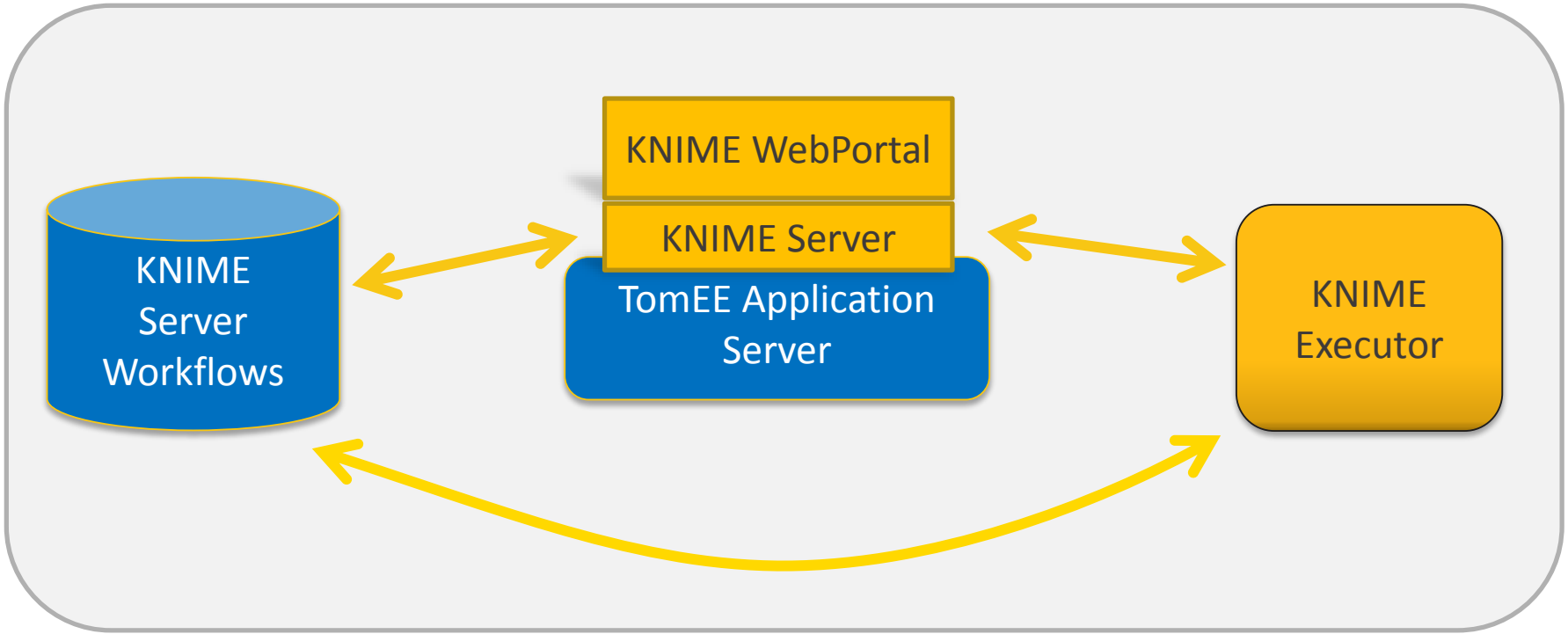
Open for Innovation ®
KNIME

Distributed Executors with the KNIME Server

Thorsten Meinl
KNIME



Status quo



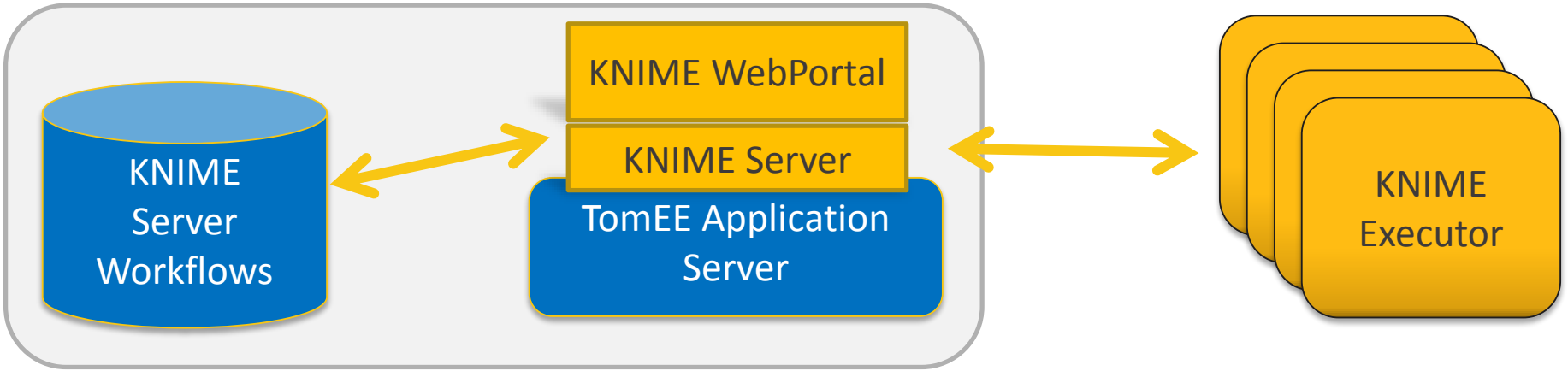
Communication via RMI, HTTP, and local file system

Status quo

- Executor on same machine as server
  - Very fast communication between server and executor
  - Limits scalability
- Communication via RMI
  - Built-in Java technology, easy to use
  - Blocking communication
  - Binary protocol, sensitive to changes
- Executor started by server via new process
  - Executor has same permissions as the server itself

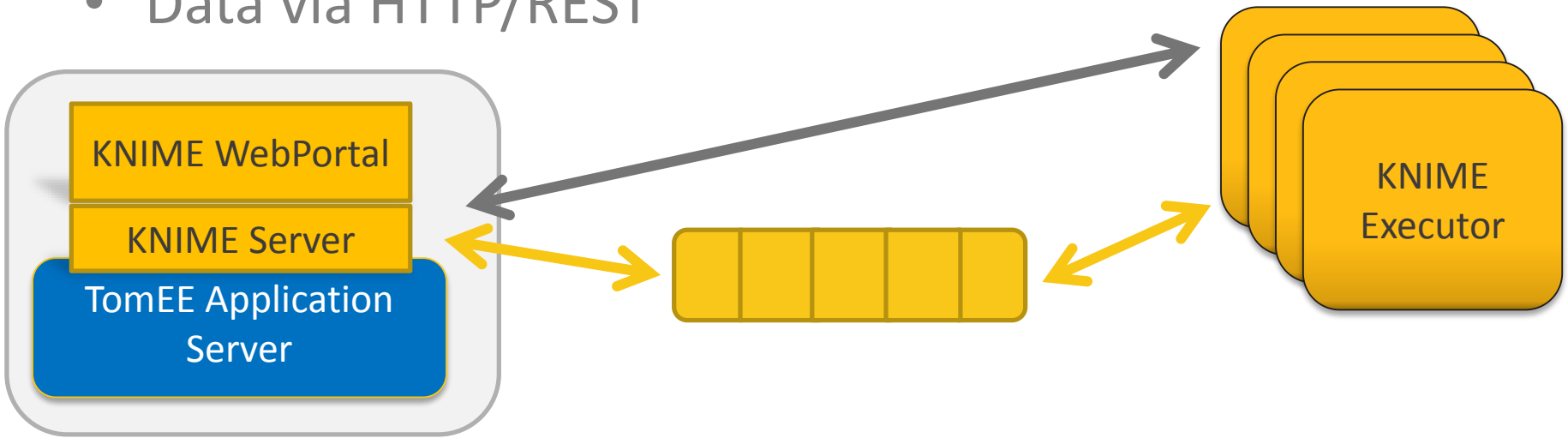
Distributed executors

- Primary goal is scalability
- One server with many executors
  - Server itself doesn‘t need many resources



Distributed communication

- Control messages via a message queue
  - E.g. RabbitMQ
- JSON as message format
- Data via HTTP/REST



Load balancing

- Message queue allows for basic load balancing
  - Mesages are distributed round-robin
  - Overloaded executors may reject work
- Currently new executors must be started externally

Required infrastructure

- RabbitMQ as message queueing system
- One host for KNIME Server
- One or more hosts for executors
- Manual setup or automated deployment with BOSH

Outlook

- Non-interactive execution available as preview with summer release
- WebPortal functionality planned for end of the year
- Future plans
  - Installation support for more HPC infrastructures
    - AWS & Azure
  - Automated executor scaling