AP2 : Job Monitoring

ursprüngliche Ziele

- Entwicklung eines Monitoring-Tools, welches job-basierte Informationen zum Zustand des Rechensystems gibt
- Entwicklung eines Job-Monitoring-Tools, welches die Ausführung von Jobs (Shell, Python) überwacht (im Gegensatz zum Systemmonitoring)
- automatische Klassifizierung von Job-Abbrüchen

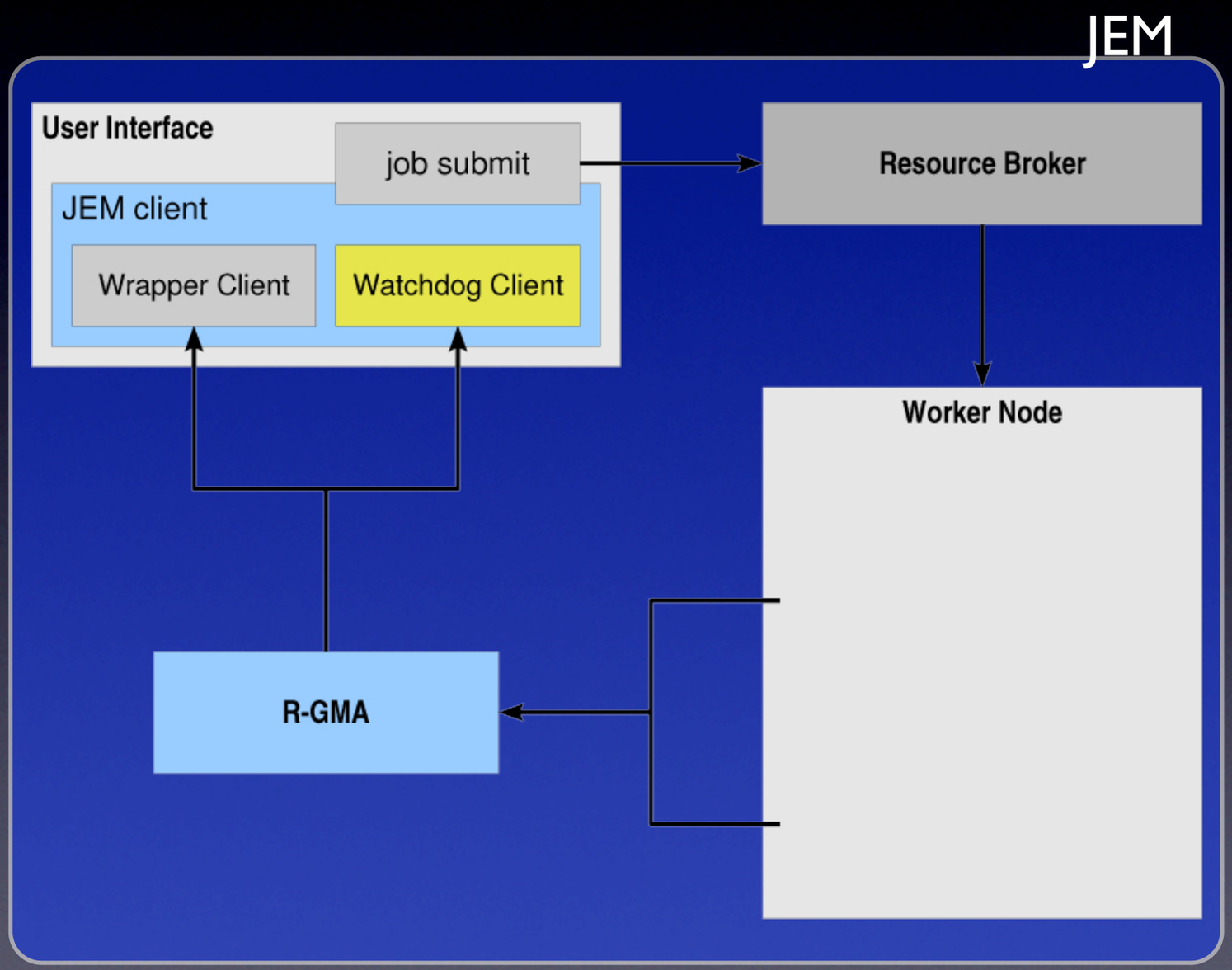
erreichte ~~ursprüngliche~~ Ziele

- Entwicklung eines Monitoring-Tools, welches job-basierte Informationen zum Zustand des Rechensystems gibt ✔
- Entwicklung eines Job-Monitoring-Tools, welches die Ausführung von Jobs (Shell, Python) überwacht (im Gegensatz zum Systemmonitoring) ✔
- automatische Klassifizierung von Job-Abbrüchen ☺

① Monitoring Tool: JEM

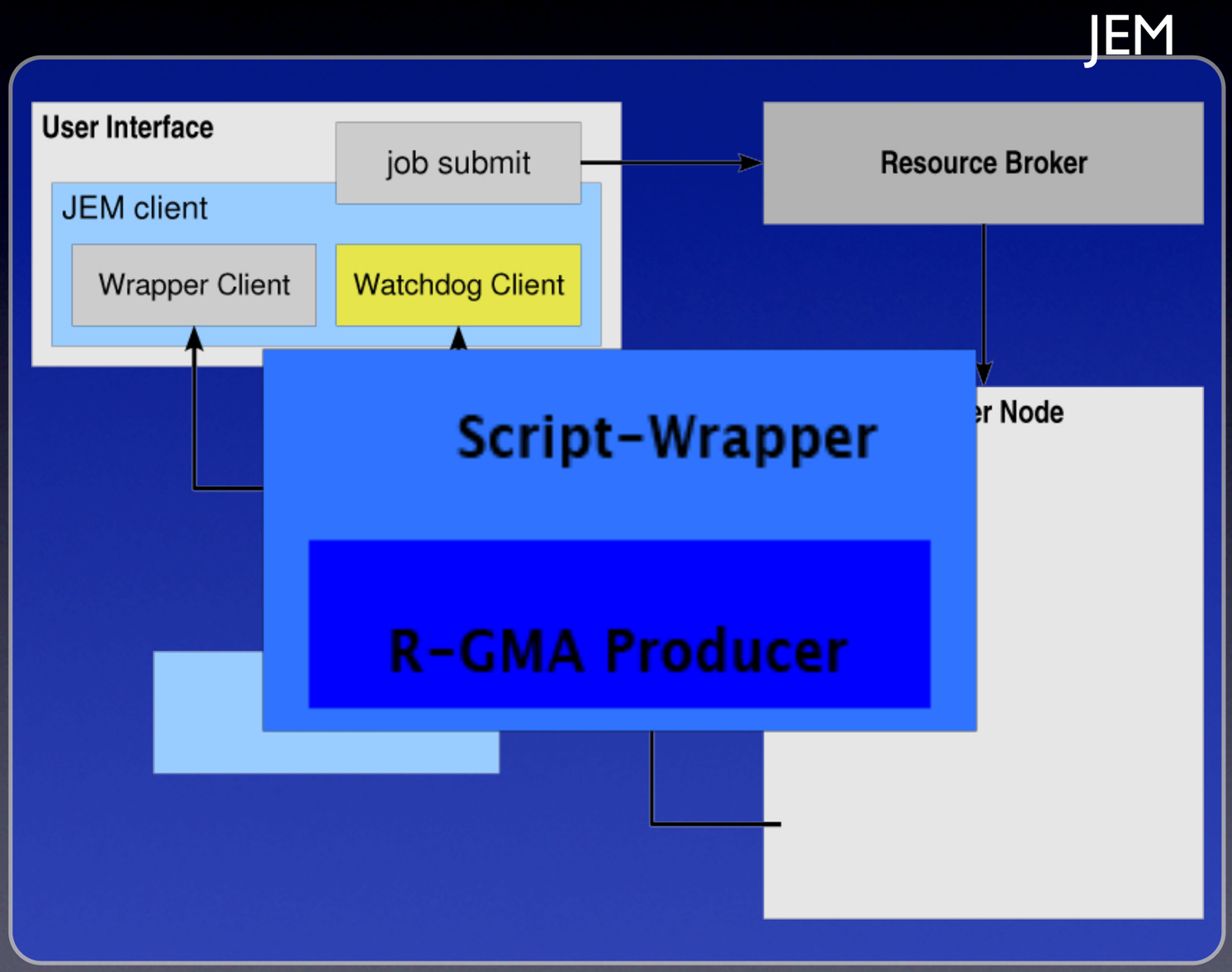
- Hintergrund:
  - Erfahrung von DØ: Probleme in verteilten Systemen für den Benutzer schwer zu identifzieren
  - Grid-Systeme (EDG, LCG) geben zu wenig Rückmeldung an den Benutzer

Aufbau von JEM



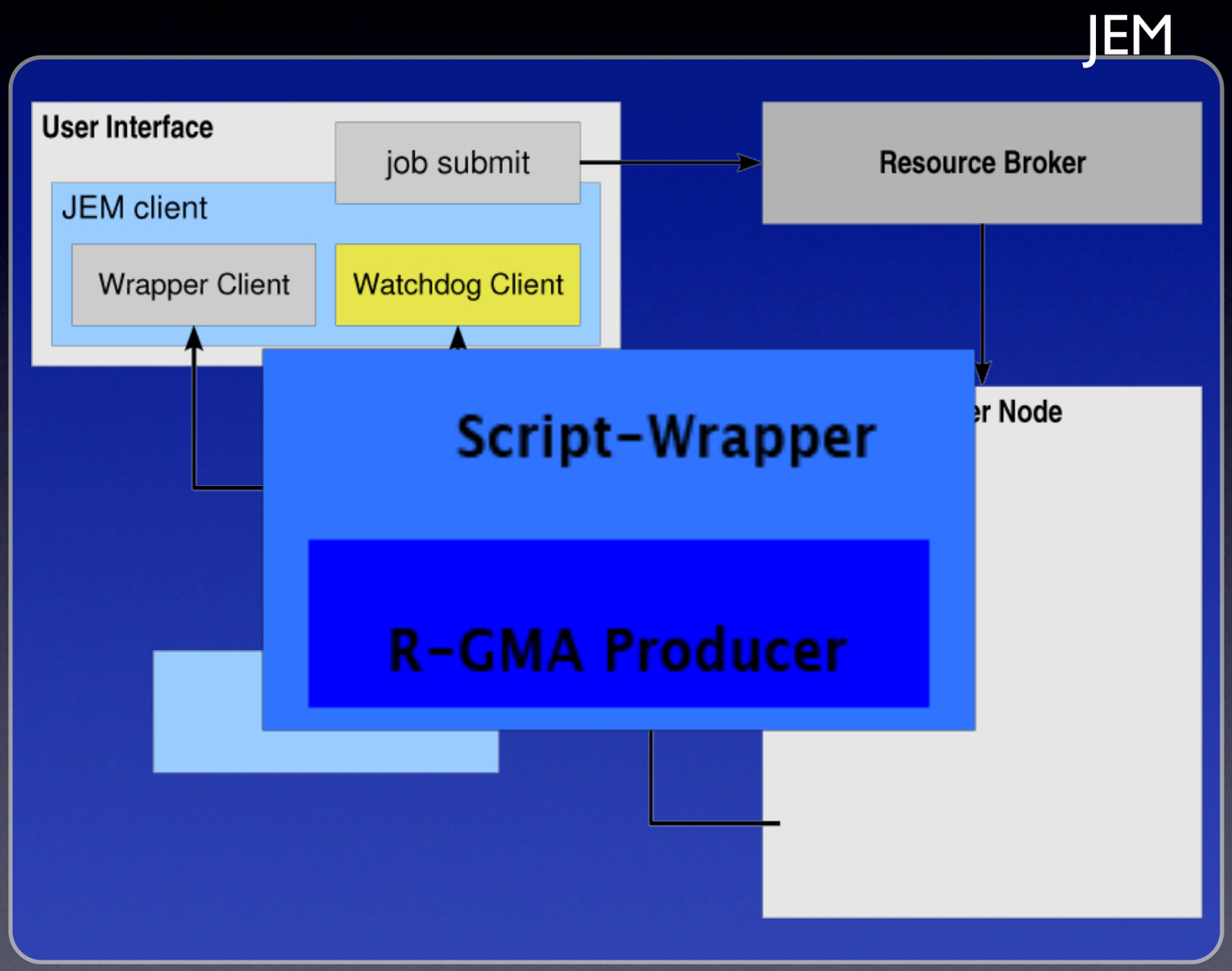
Aufbau von JEM

- script-wrapper



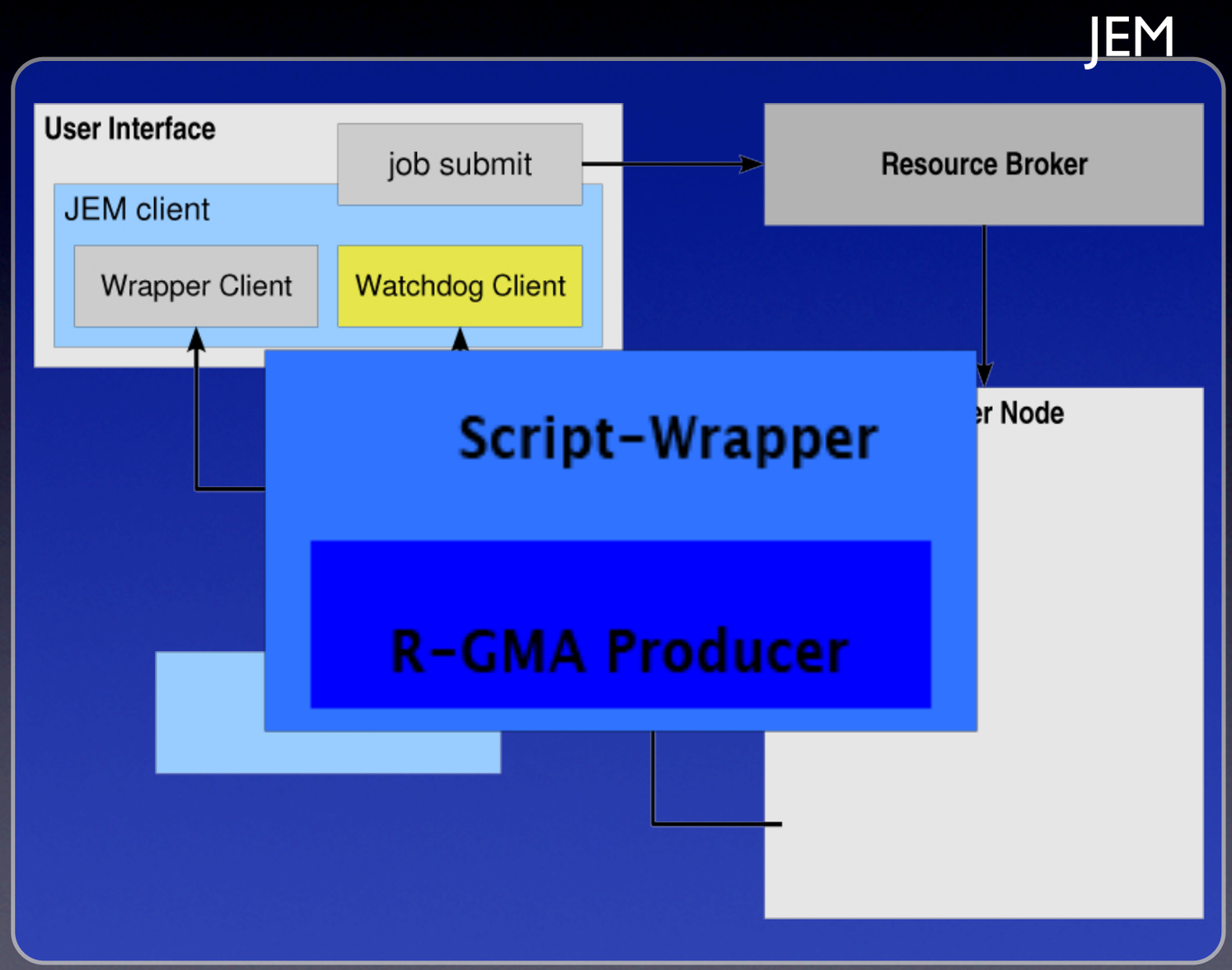
Aufbau von JEM

- script-wrapper
  - step-by-step execution.



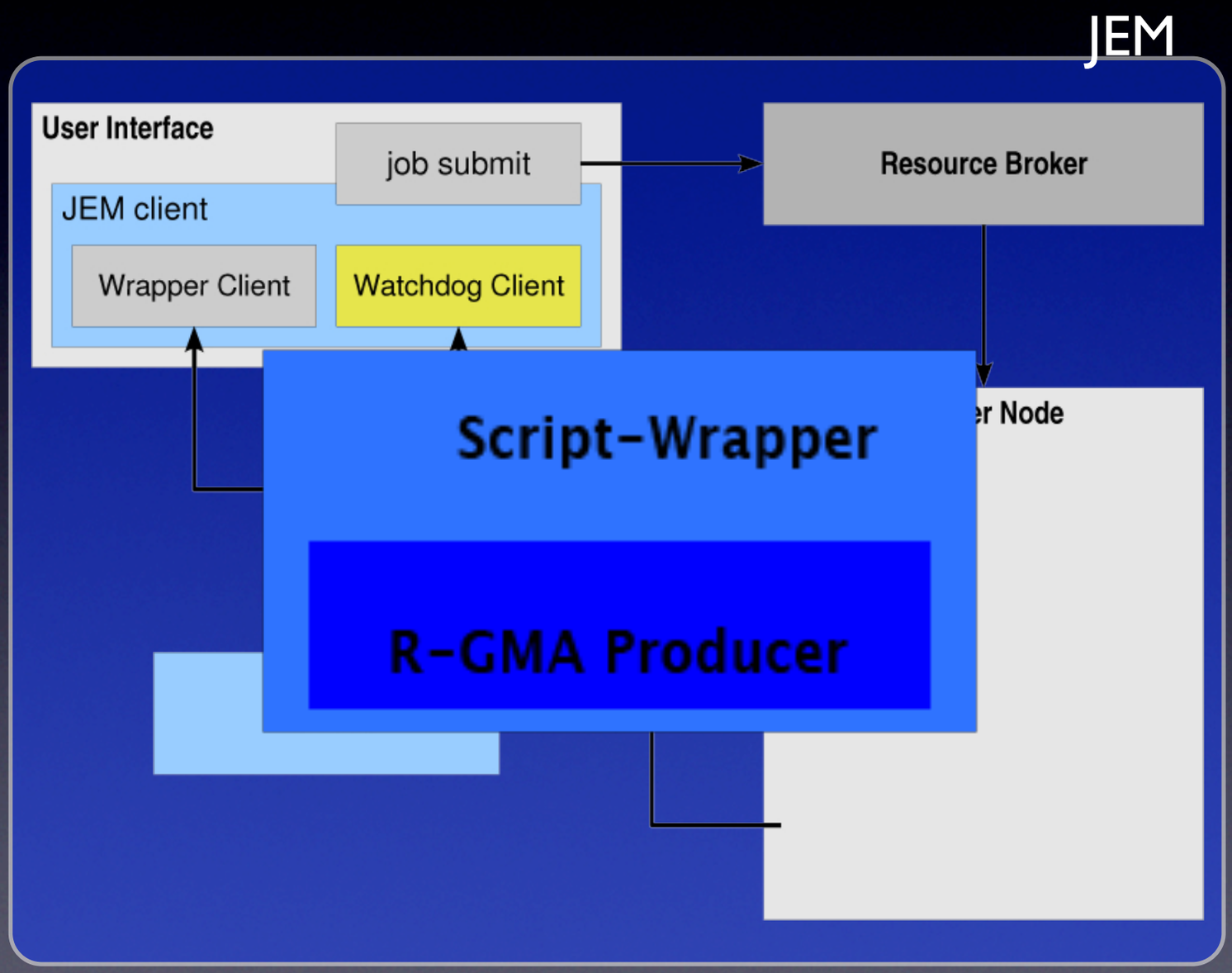
Aufbau von JEM

- script-wrapper
  - step-by-step execution.
  - supervisor of jobscript.



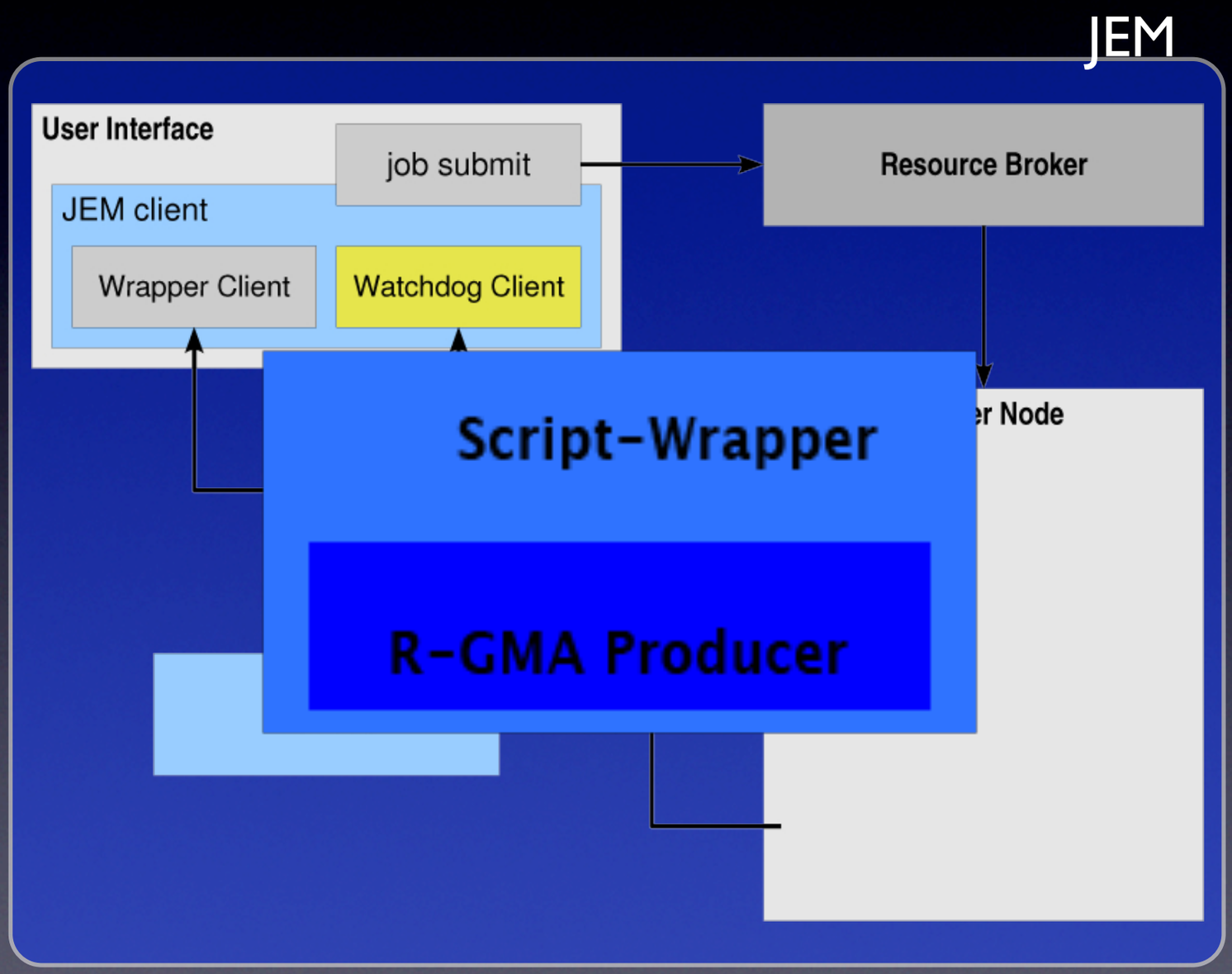
Aufbau von JEM

- script-wrapper
  - step-by-step execution.
  - supervisor of jobscript.
  - I/O-logging.



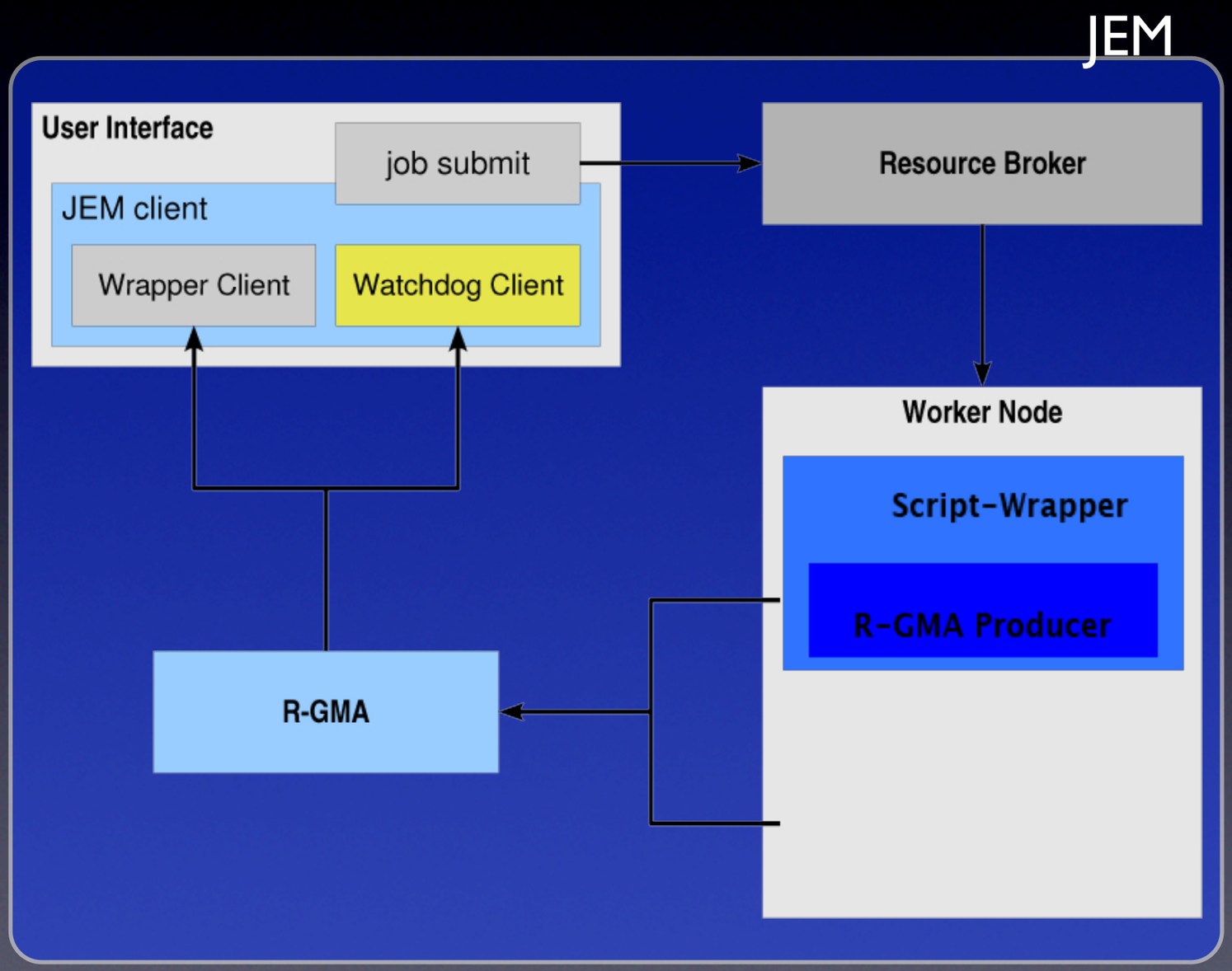
Aufbau von JEM

- script-wrapper
  - step-by-step execution.
  - supervisor of jobscript.
  - I/O-logging.
  - frequent reports to user.



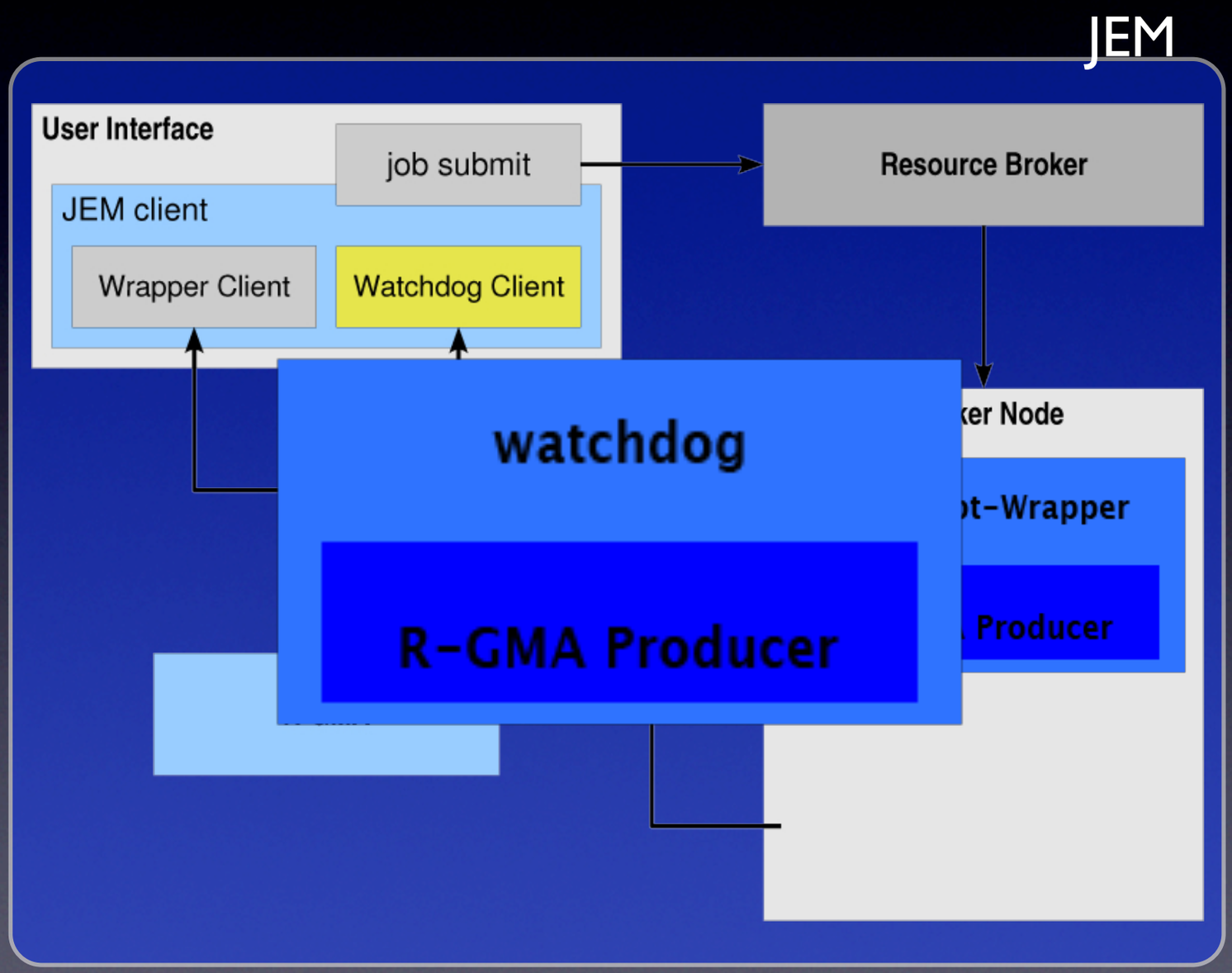
Aufbau von JEM

- script-wrapper
  - step-by-step execution.
  - supervisor of jobscript.
  - I/O-logging.
  - frequent reports to user.



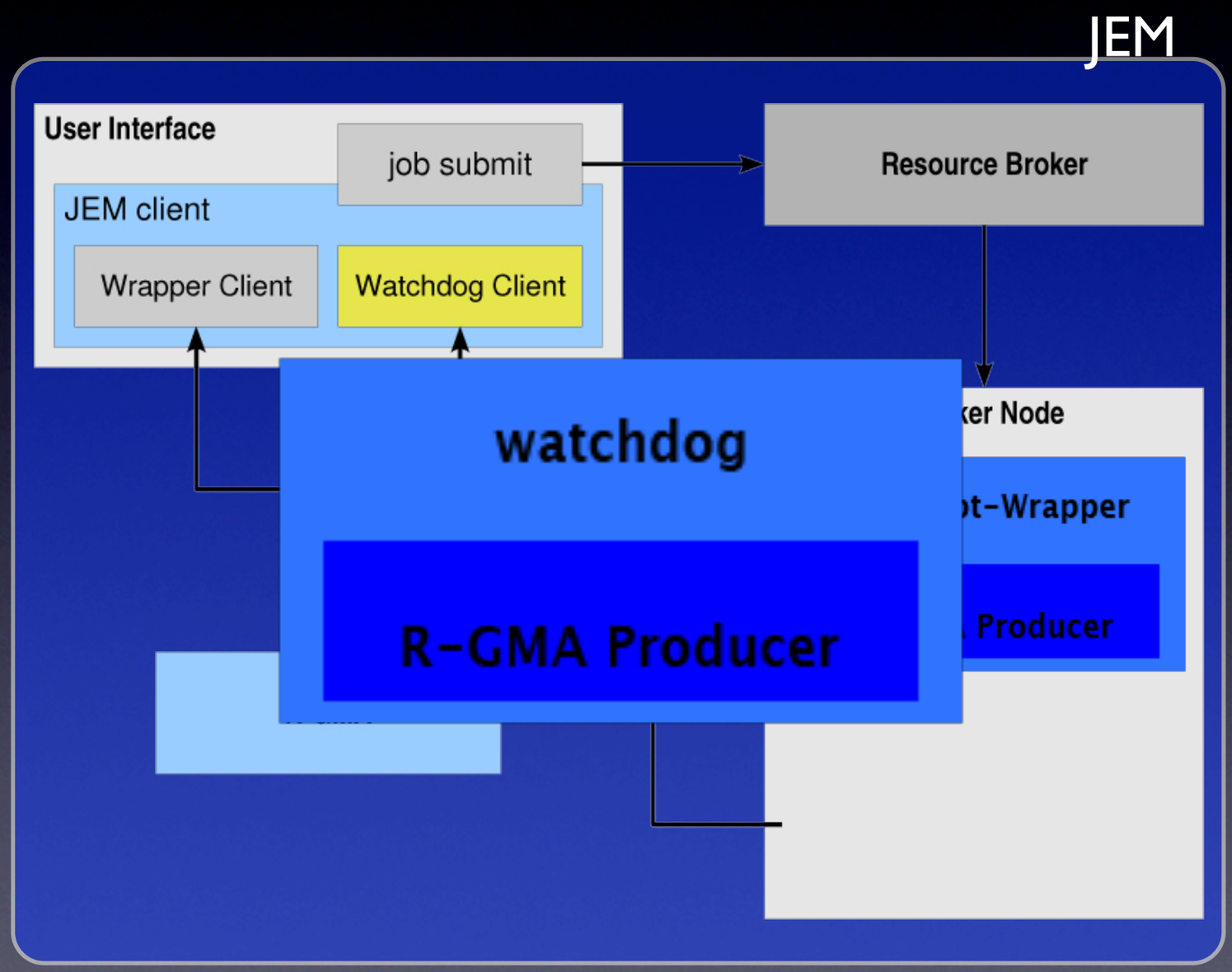
Aufbau von JEM

- script-wrapper
  - step-by-step execution.
  - supervisor of jobscript.
  - I/O-logging.
  - frequent reports to user.
- watchdog



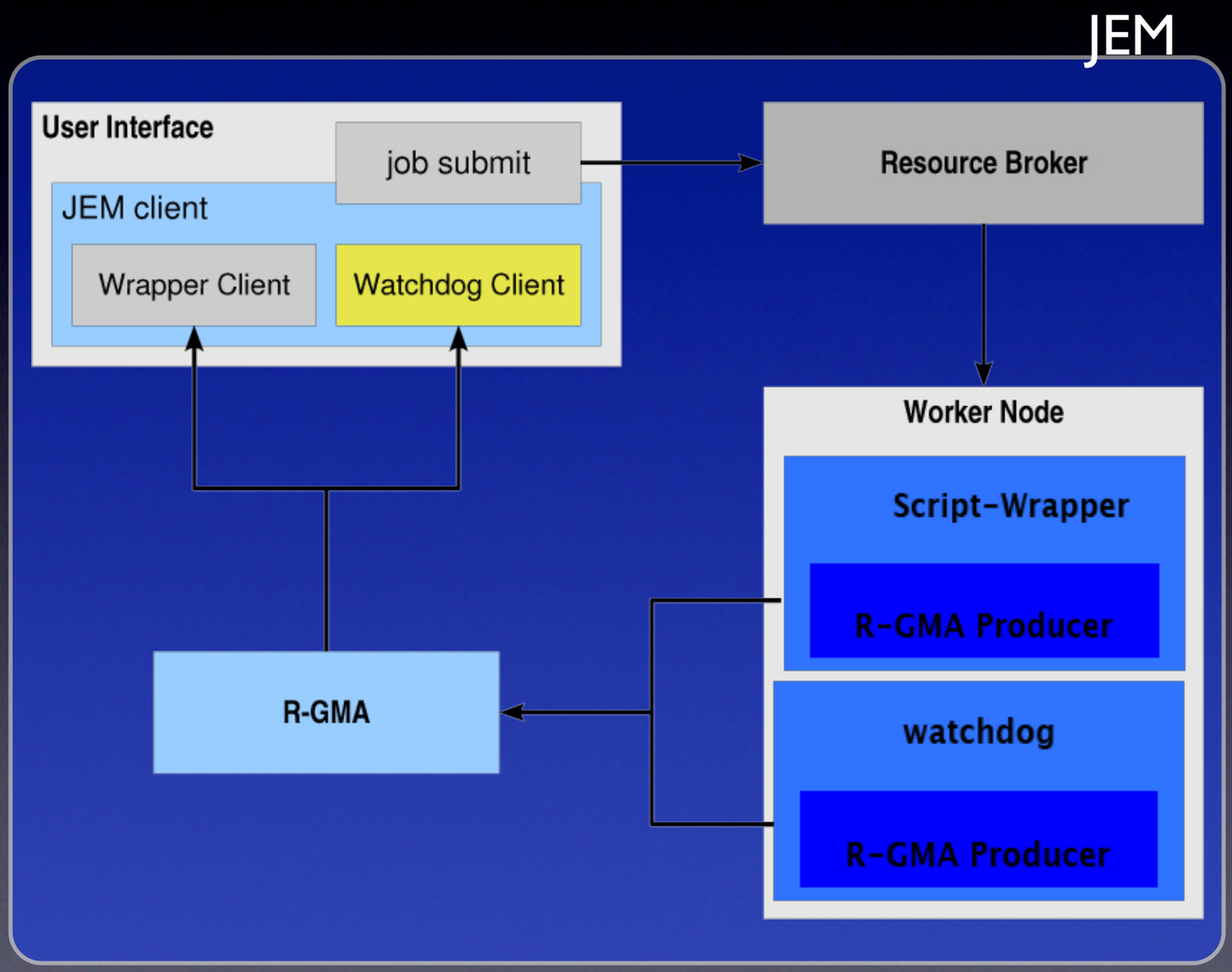
Aufbau von JEM

- script-wrapper
  - step-by-step execution.
  - supervisor of jobscript.
  - I/O-logging.
  - frequent reports to user.
- watchdog
  - gather System-Informations.



Aufbau von JEM

- script-wrapper
  - step-by-step execution.
  - supervisor of jobscript.
  - I/O-logging.
  - frequent reports to user.
- watchdog
  - gather System-Informations.



JEM-Overview

JEM-Overview

- 'python'-written program.

JEM-Overview

- 'python'-written program.
- automatically invoked by a special job-submission.

JEM-Overview

- 'python'-written program.
- automatically invoked by a special job-submission.
- direct data-flow to
  - R-GMA
  - MonALISA
  - socket

JEM-Overview

- 'python'-written program.
- automatically invoked by a special job-submission.
- direct data-flow to
  - R-GMA
  - MonALISA
  - socket
- indirect data-flow via Output-Sandbox.

JEM-Features

JEM-Features

- 'add-on' to existing monitoring architectures possible.

JEM-Features

- 'add-on' to existing monitoring architectures possible.
  - accessable command properties like: scriptname, line-number, function-name, time-stamp, stdin/out/err and exit-code.

JEM-Features

- 'add-on' to existing monitoring architectures possible.
  - accessable command properties like: scriptname, line-number, function-name, time-stamp, stdin/out/err and exit-code.
  - no need to modify job itself.

JEM-Features

- 'add-on' to existing monitoring architectures possible.
  - accessable command properties like: scriptname, line-number, function-name, time-stamp, stdin/out/err and exit-code.
  - no need to modify job itself.
  - runs stable with Athena.

JEM-Features

JEM-Features

- emergency-mode

JEM-Features

- emergency-mode
  - activated on error within JEM itself.

JEM-Features

- emergency-mode
  - activated on error within JEM itself.
  - runs the job without any Job Monitoring.

JEM-Features

- emergency-mode
  - activated on error within JEM itself.
  - runs the job without any Job Monitoring.
  - prevents additional job errors

JEM-Features

JEM-Features

- little overhead

JEM-Features

- little overhead
  - largest overhead was messured within python.

JEM-Features

- little overhead
  - largest overhead was messured within python.
  - a small overhead with bash scripts.

JEM-Features

- little overhead
  - largest overhead was messured within python.
  - a small overhead with bash scripts.
  - no overhead on binary execution.

Pluses on JEM

Pluses on JEM

- different verbosity levels in command logging.

Pluses on JEM

- different verbosity levels in command logging.
- values of python variables during errors.

Pluses on JEM

- different verbosity levels in command logging.
- values of python variables during errors.
- known-critical commands my be modified / hardened.

Pluses on JEM

- different verbosity levels in command logging.
- values of python variables during errors.
- known-critical commands my be modified / hardened.
- a history of System-Resources available.

Pluses on JEM

- different verbosity levels in command logging.
- values of python variables during errors.
- known-critical commands my be modified / hardened.
- a history of System-Resources available.
- realtime informations of:

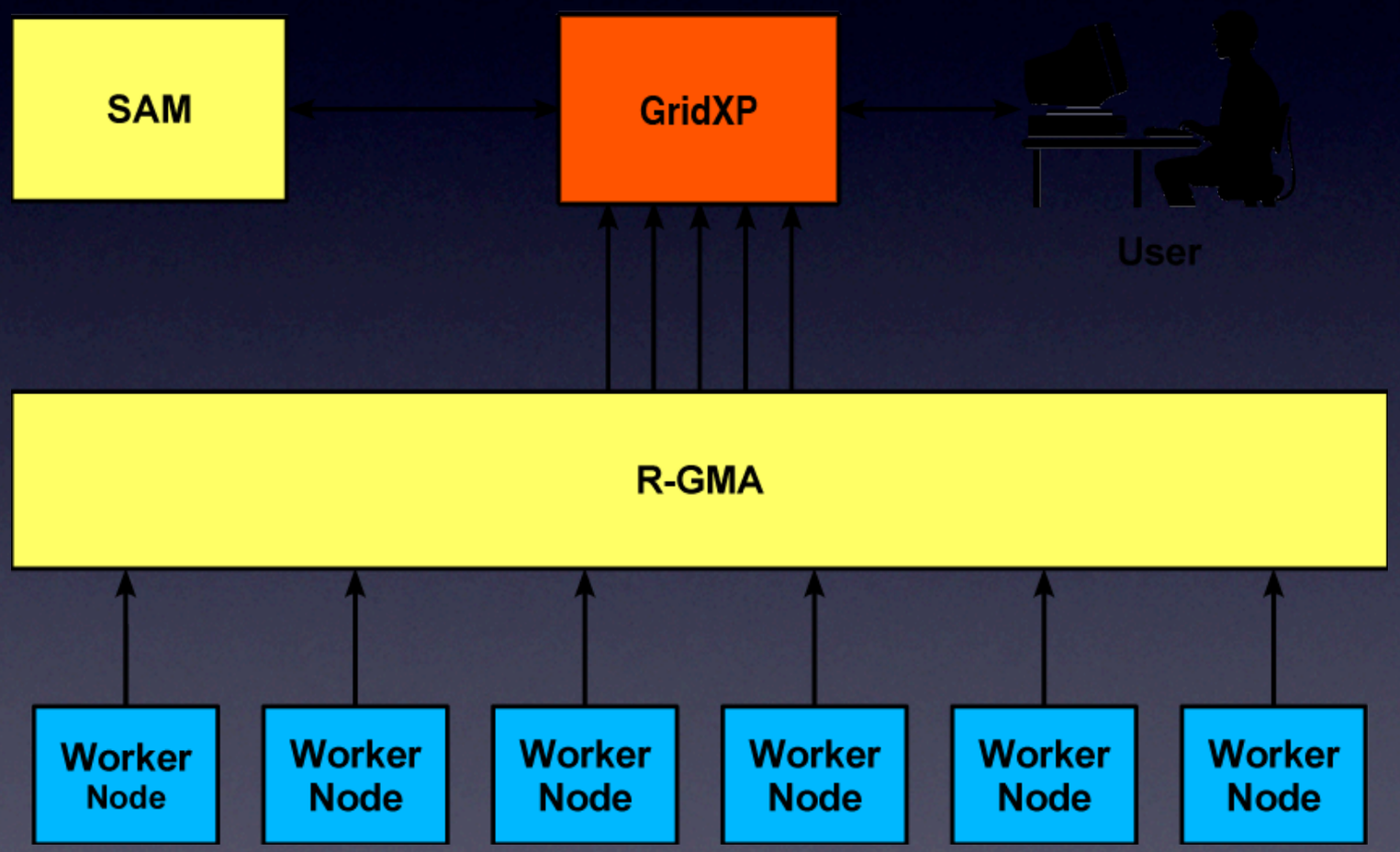
Pluses on JEM

- different verbosity levels in command logging.
- values of python variables during errors.
- known-critical commands my be modified / hardened.
- a history of System-Resources available.
- realtime informations of:
  - currently executed command.

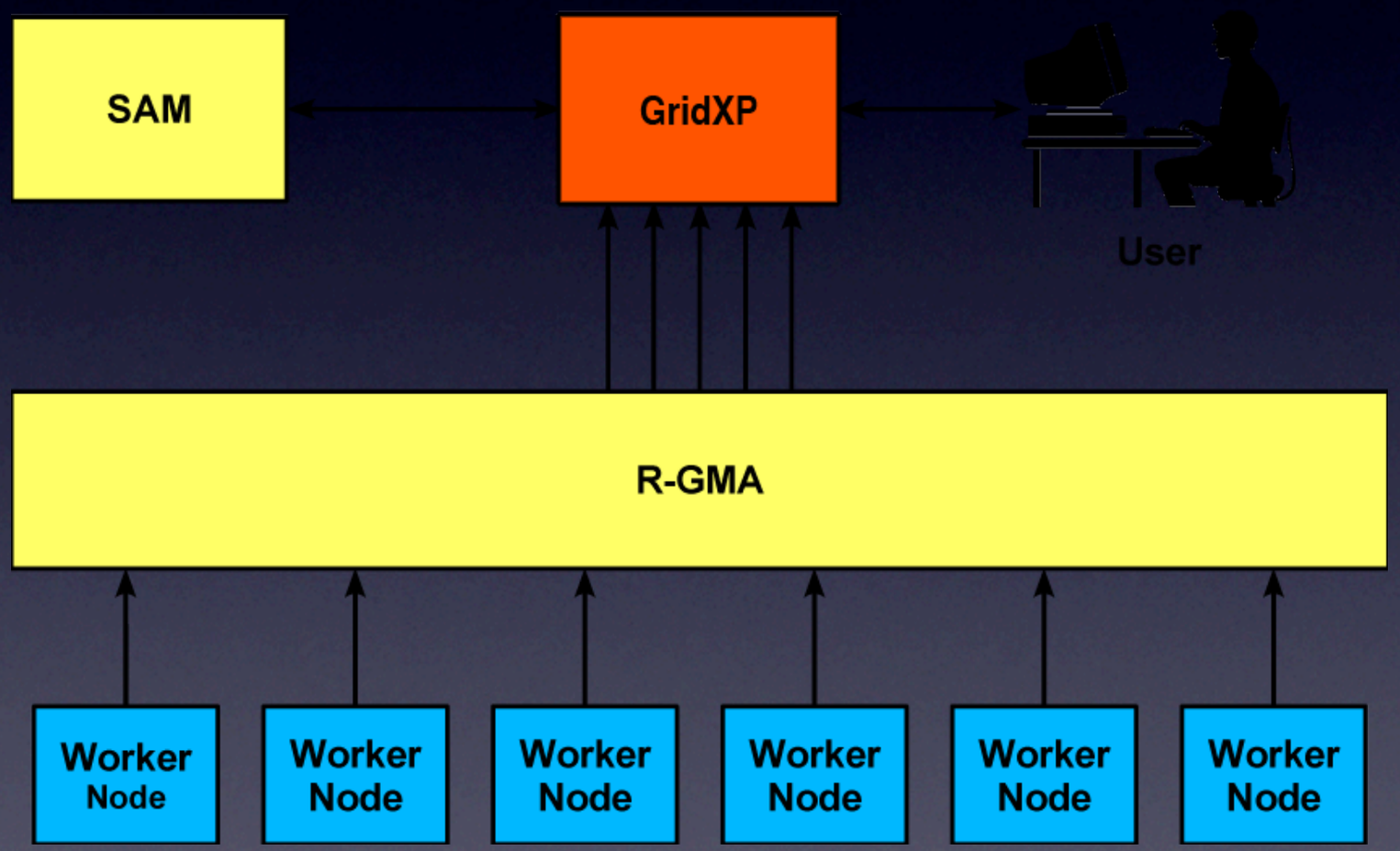
Pluses on JEM

- different verbosity levels in command logging.
- values of python variables during errors.
- known-critical commands my be modified / hardened.
- a history of System-Resources available.
- realtime informations of:
  - currently executed command.
  - access to stdout / stderr and exit codes.

② GridXP

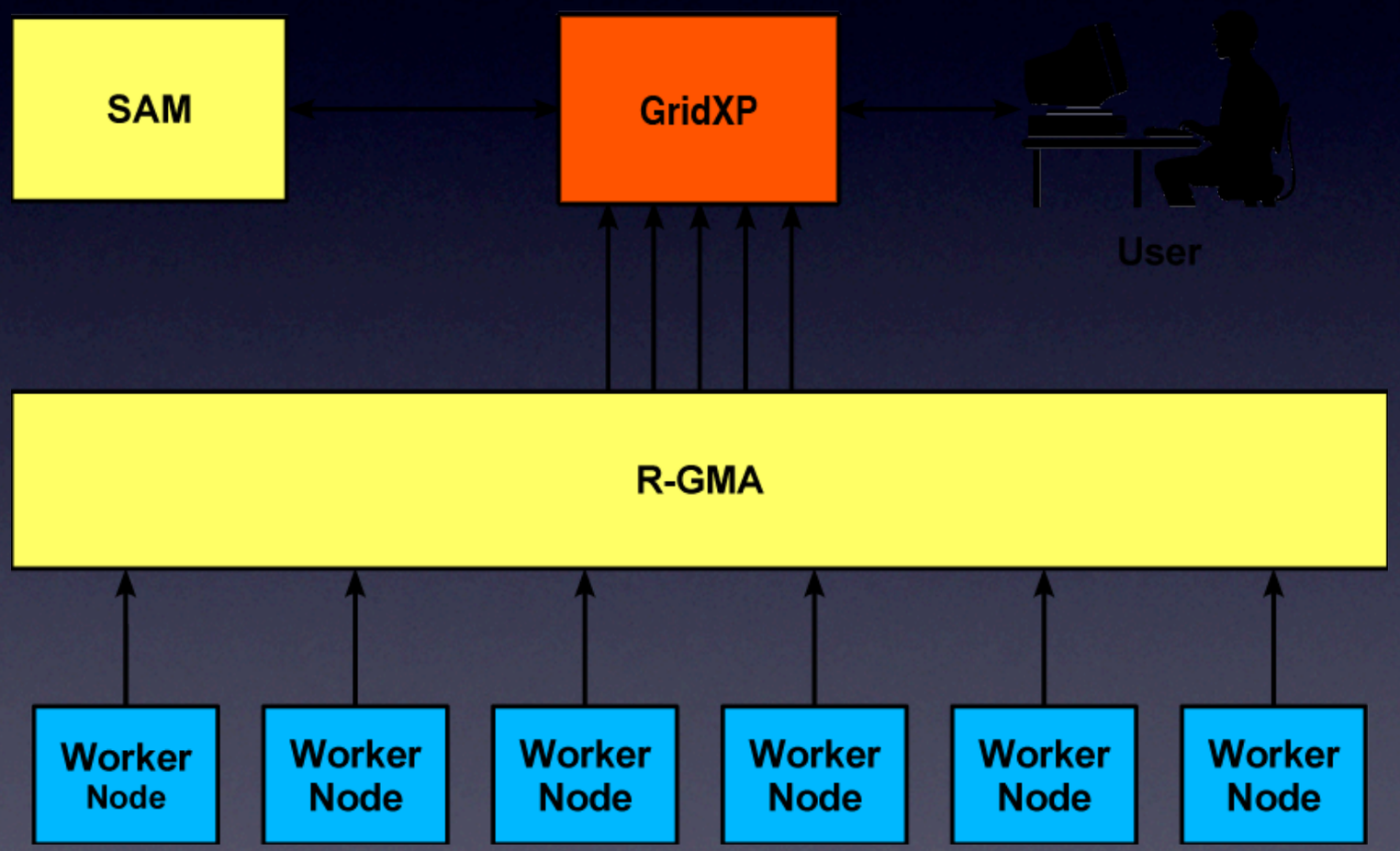

SAM
GridXP
User
R-GMA
Worker Node
Worker Node
Worker Node
Worker Node
Worker Node
Worker Node

② GridXP



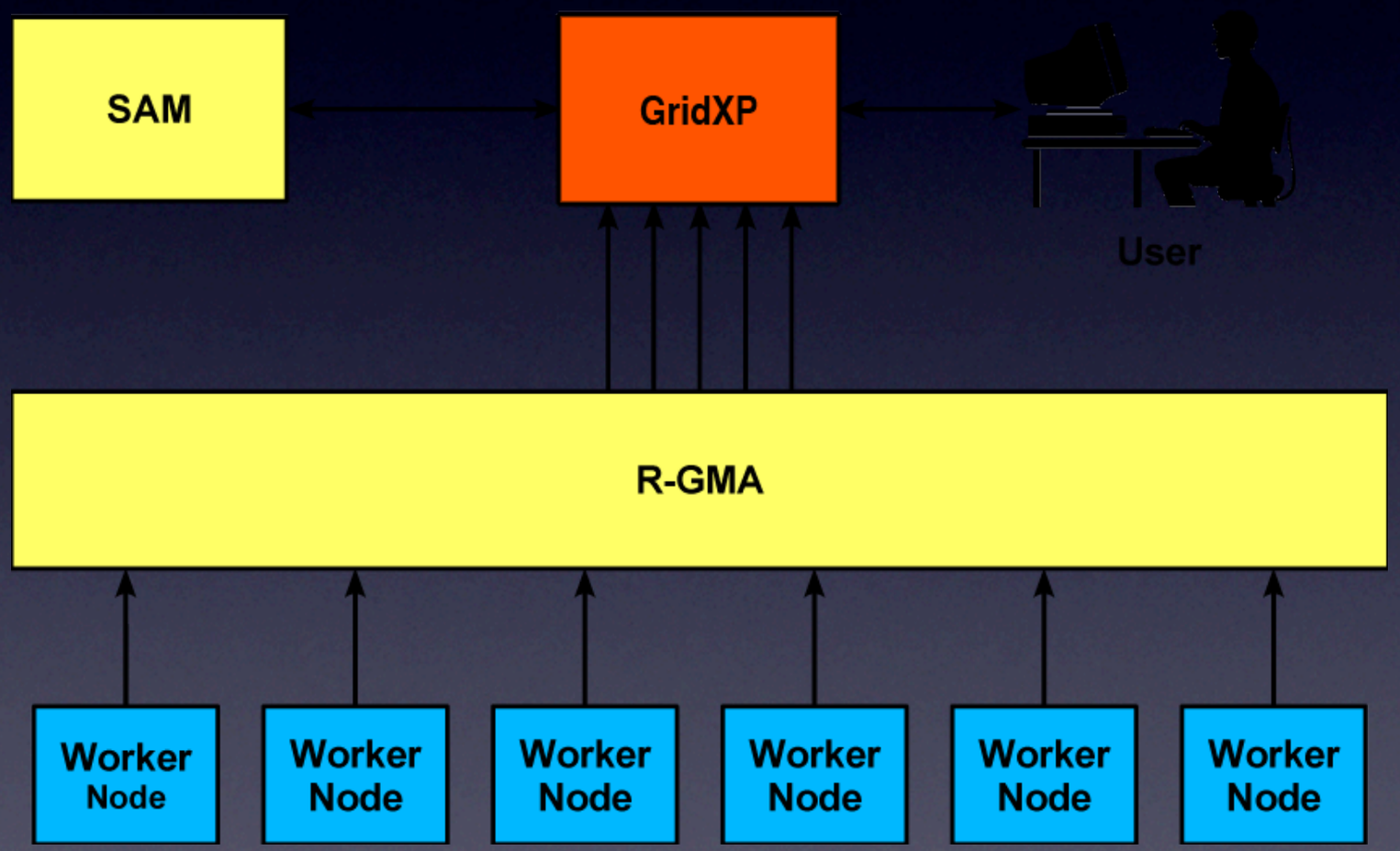
- Interface for user.

② GridXP

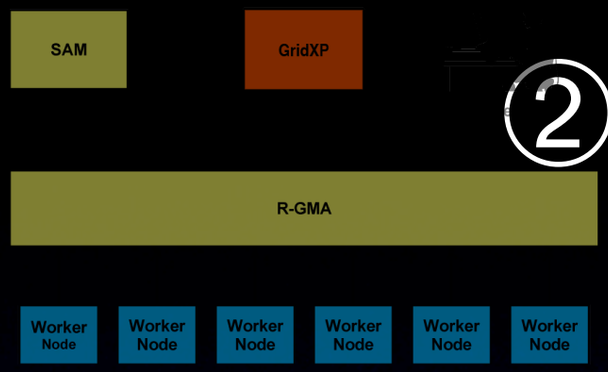


- Interface for user.
- Explains error messages.

② GridXP

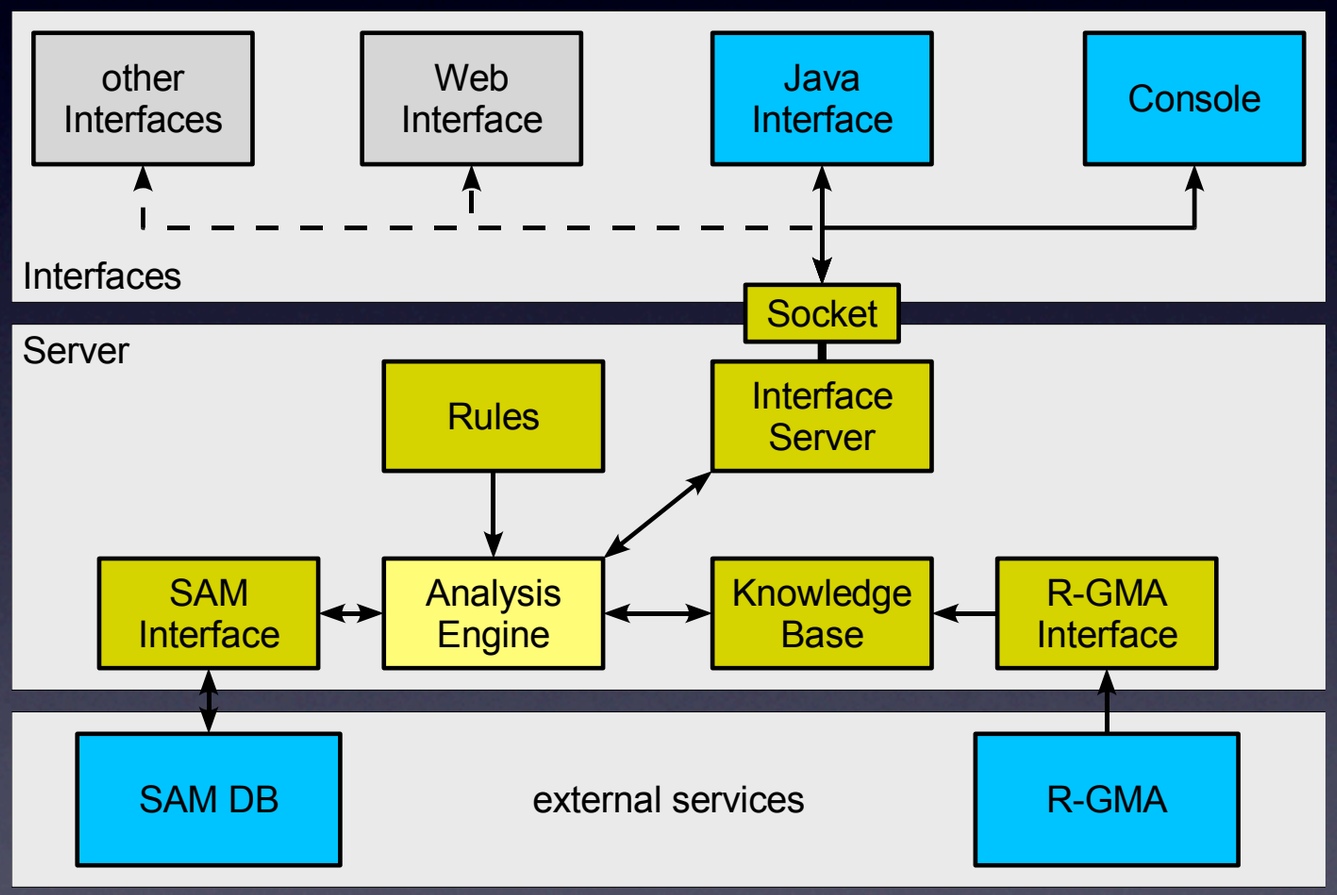


- Interface for user.
- Explains error messages.
- Makes suggestions for solutions.



② GridXP

- Interface for user.
- Explains error messages.
- Makes suggestions for solutions.



- Server collects informations from R-GMA.
- dynamically queries SAM-DB.
- matches informations against rules.
- provides result to user.

Status

- JEM:
  - Monitoring Programm fertig, „in production“, stabil und dokumentiert
  - Integration in Ganga quasi abgeschlossen
- GridXP:
  - Grundgerüst Expertensystem fertig, erste Regeln implementiert.
  - Z.Zt. laufen noch 2 Master-Arbeiten
    - Codeverbesserungen
    - Fehlerklassifizierung

Status

- http://www.grid.uni-wuppertal.de/grid/jms/

- http://www.grid.uni-wuppertal.de/grid/expertsystem/

weitere Pläne

- JMS:
  - Pflege des Codes, evtl. Bugfixing, falls durch Ganga-Produktion noch Fehler auftauchen
- GridXP:
  - Kooperation mit FH Münster/Köln/ Niederrhein:
    - automatische Klassifikation von Fehlern
    - Benutzerschnittstelle (Wissenserwerb)
    - Integration in GGUS?