JÜLICH
FORSCHUNGSZENTRUM

System monitoring with LLview and the Parallel Tools Platform

November 25, 2014 | Carsten Karbach

Content

Part I: LLview

November 25, 2014 | Carsten Karbach

JÜLICH FORSCHUNGSZENTRUM

Why system monitoring?

- For administrators
  - Global overview of system utilization
  - Throughput optimization
  - Batch system configuration optimization
  - Adaptive change of scheduling parameters
- For users
  - Controlling own running and waiting jobs
  - Planning job submissions
  - Use of idling resources

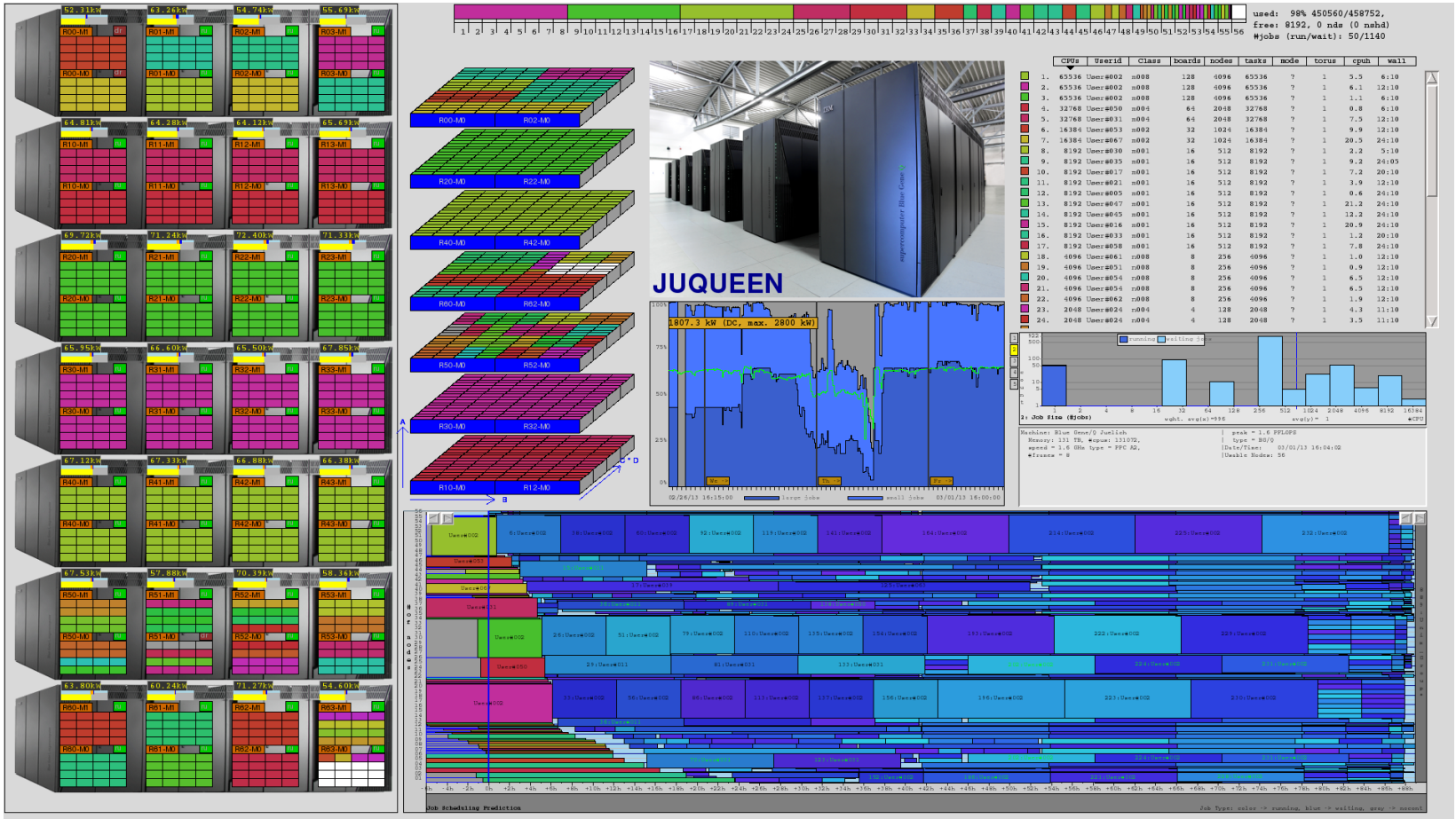
$\Rightarrow$ LLview

  - Compact display of all usage data in one window
  - Easy access to system's status data
  - Interactive display for linking information

JÜLICH FORSCHUNGSZENTRUM

LLview

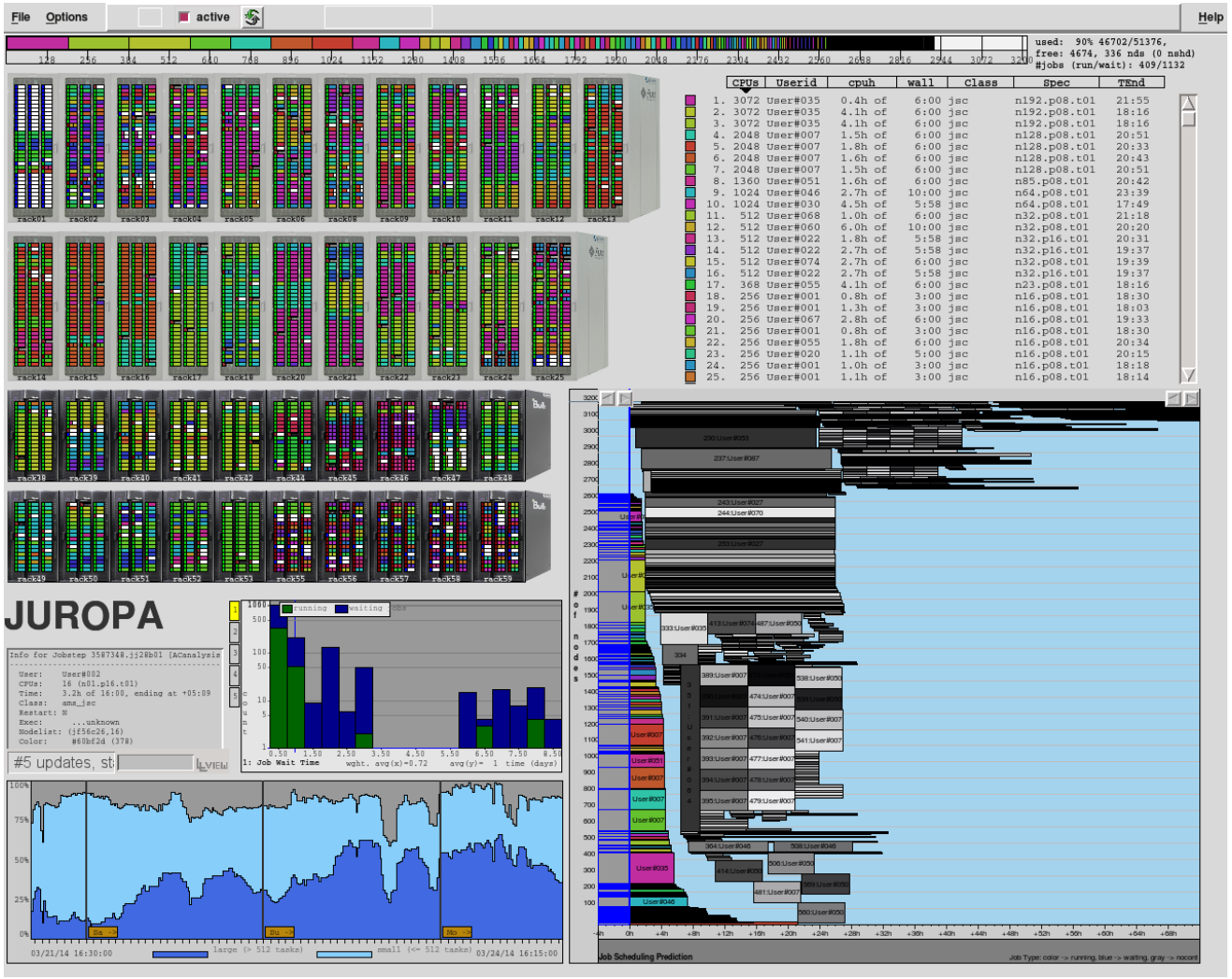
→ Visualizes supercomputer status on a **single screen**



Source: Screenshot LLview for JUQUEEN (5.9 PFlops, 458K cores, LoadLeveler)

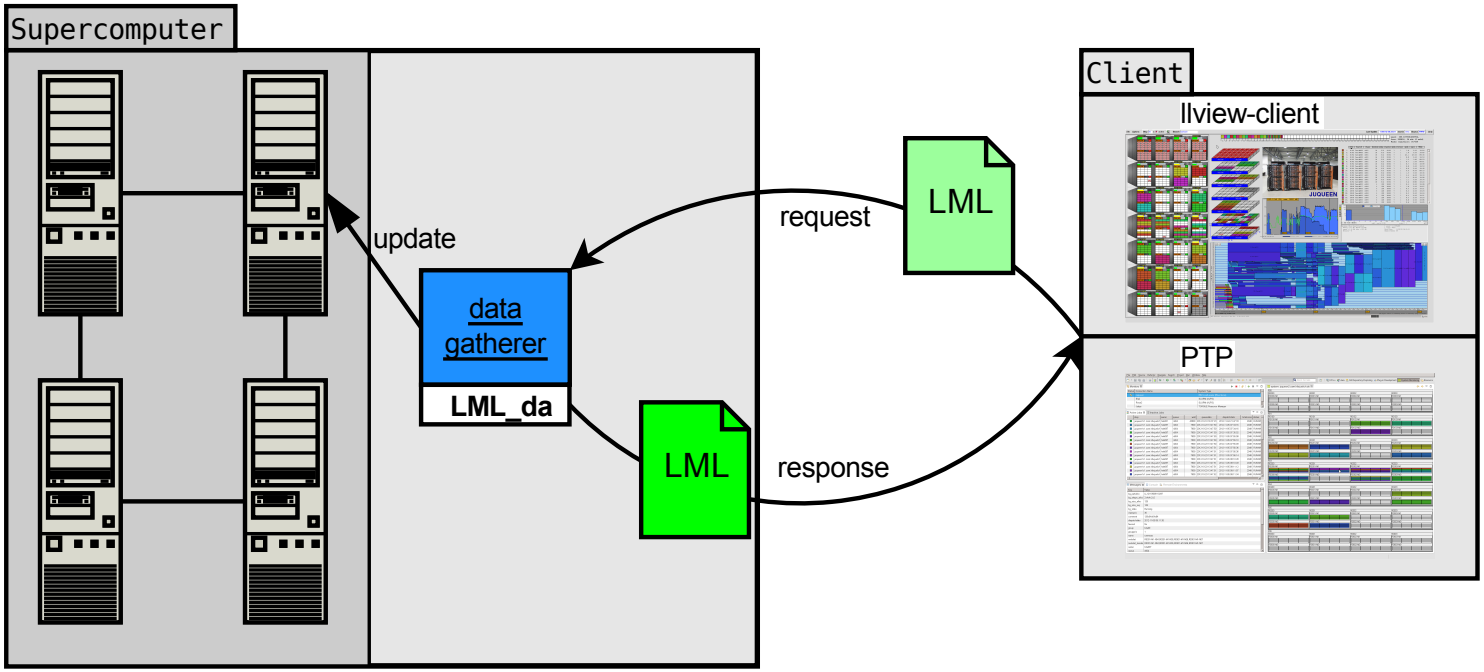
JÜLICH FORSCHUNGSZENTRUM

LLview Example: JUROPA



Source: Screenshot LLview for JUROPA (207 TFlops, 26K cores, Moab/Torque)

Monitoring Architecture I

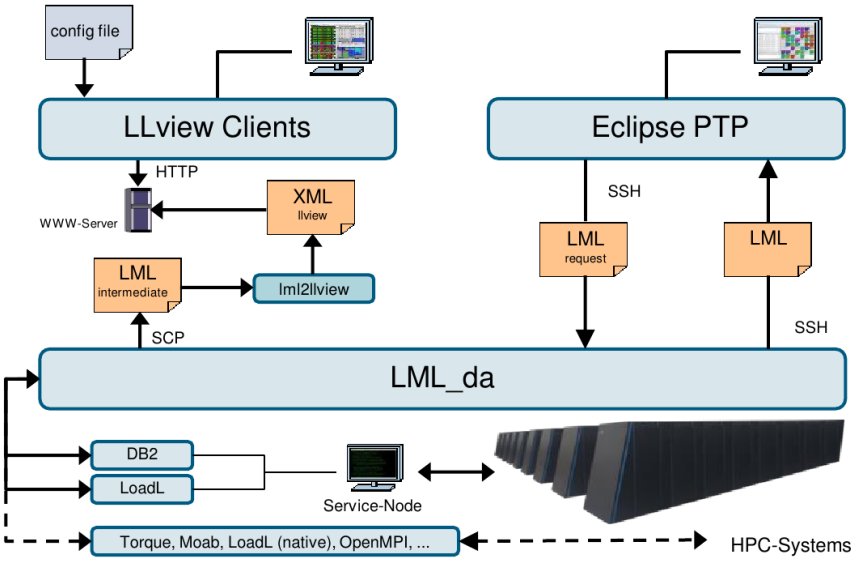


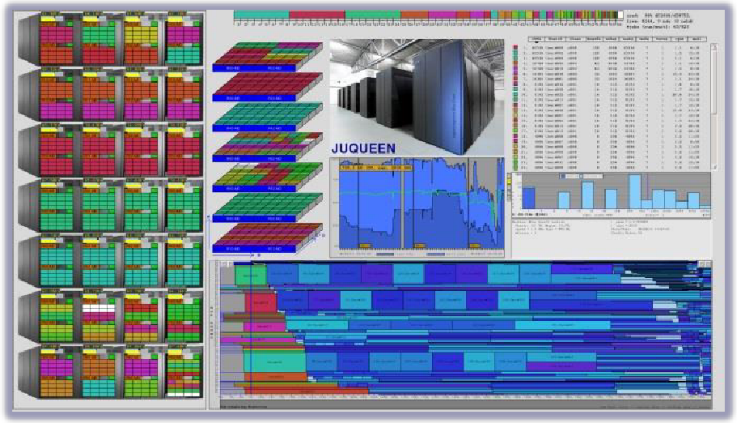
JÜLICH FORSCHUNGSZENTRUM

Monitoring Architecture II

- **LML_da** gathers status information,
  calls target system's remote commands, written in Perl
- Automatic deploy of LML_da, no installation required
- **LML** is a data format for status information
  of supercomputers
- LML request: requested data and layout
- LML response: contains the request and status information
- LML as abstraction layer
  $\rightarrow$ thin clients, **re-use** of LML_da functions

LLview Architecture Details





- Client-Server architecture, LML_da as backend
- Wide range of supported batch systems, minimal effort for extension
- Minor performance impact on monitored system, only **central batch** system is queried

Part II: PTP

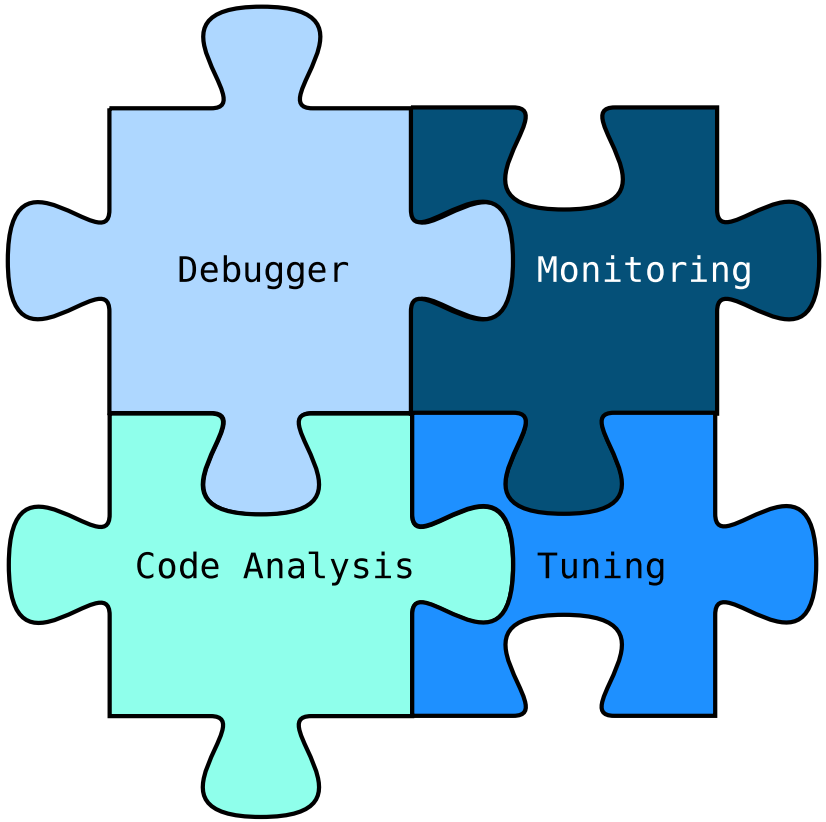
November 25, 2014 | Carsten Karbach

JÜLICH FORSCHUNGSZENTRUM

PTP – Parallel Tools Platform

What is PTP?

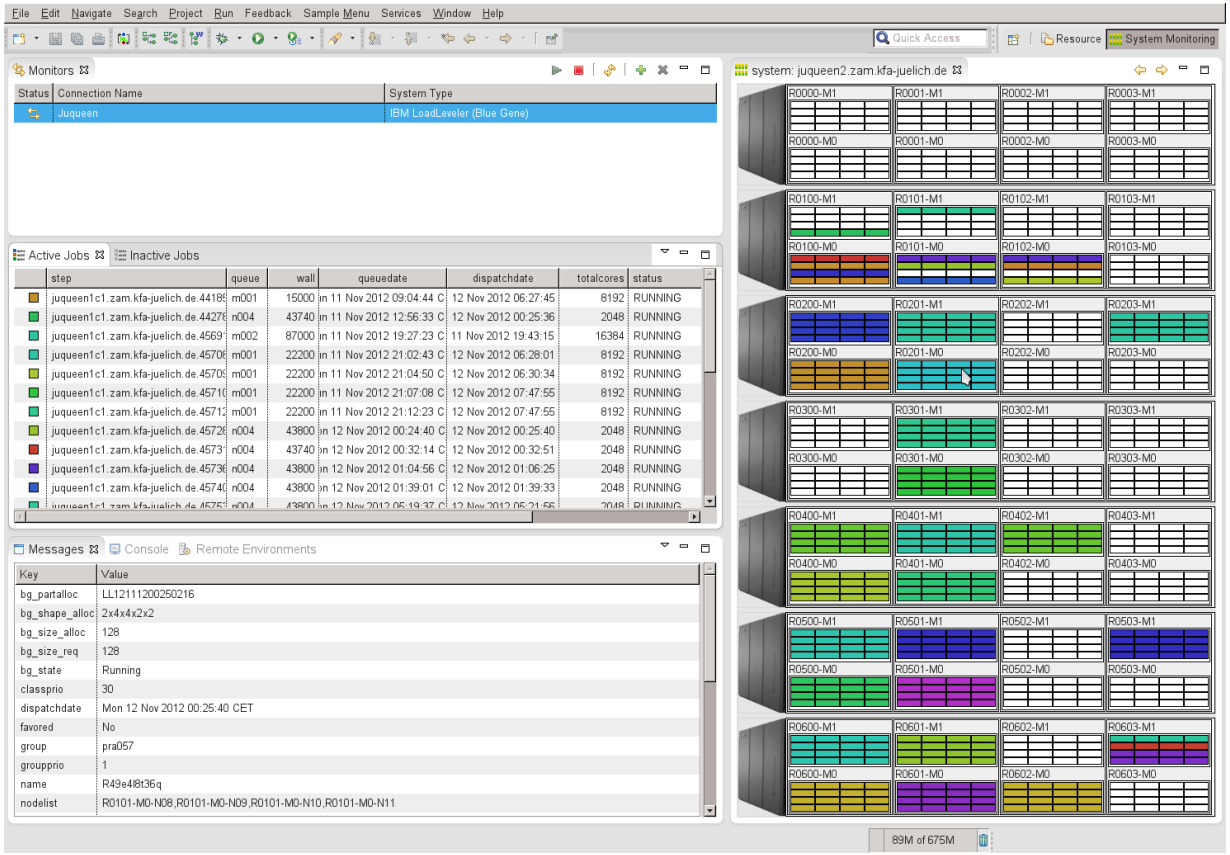
- **IDE** for parallel application development
- Based on **Eclipse**
- **Open-source** project
- Developers: IBM, U.Oregon, UTK, Heidelberg University, NCSA, UIUC, JSC, ...



JÜLICH FORSCHUNGSZENTRUM

PTP with LLview

→ Main components of LLview in **PTP's monitoring perspective**



Monitoring example: JUQUEEN

JÜLICH FORSCHUNGSZENTRUM

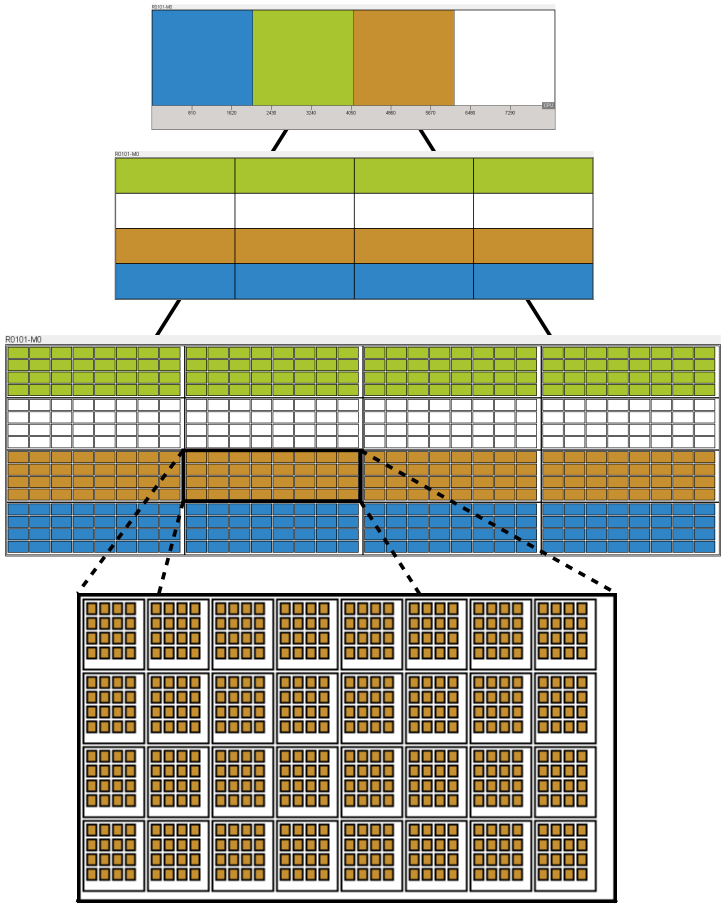
PTP monitoring – features

- Support for many target systems:
  Loadleveler, Torque, PBS, Grid Engine, SLURM, LSF
- Authentication and communication via **SSH**
- **Client-server** architecture
- **LML** (large-scale system markup language) as communication language
- Integrated into PTP workflow
  - Job monitoring
  - Job cancellation
  - View job output data
- Automatic deployment of LML_da

JÜLICH FORSCHUNGSZENTRUM

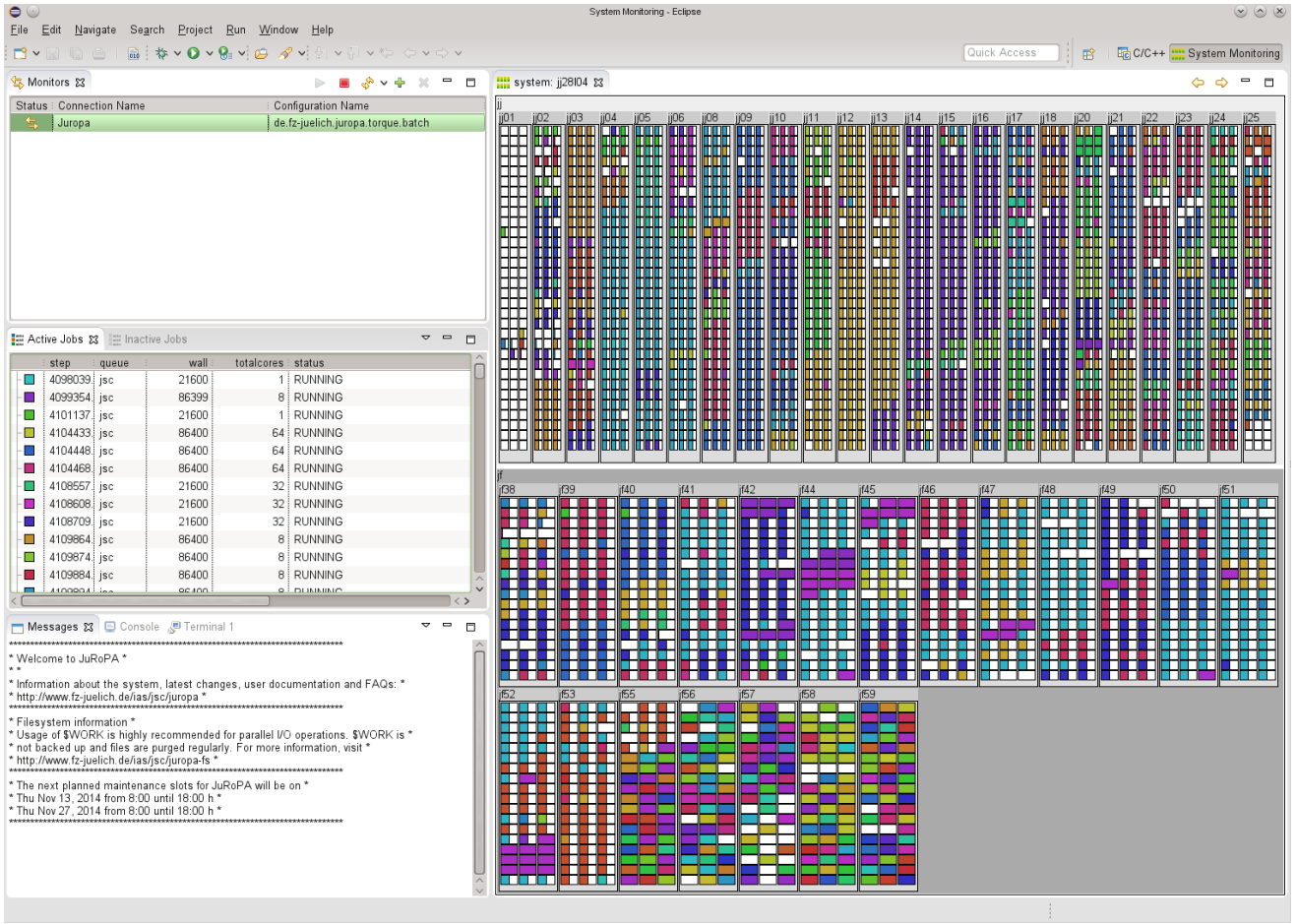
Scalability

1. Scalable **server scripts** LML_da
   - Select only required data
2. Scalable **data format** LML
   - Data compression
   - Redundancy avoidance
   - Uses system hierarchy
3. Scalable **visualization** PTP
   - Filter status data
   - Levels of detail
   - Zoom function



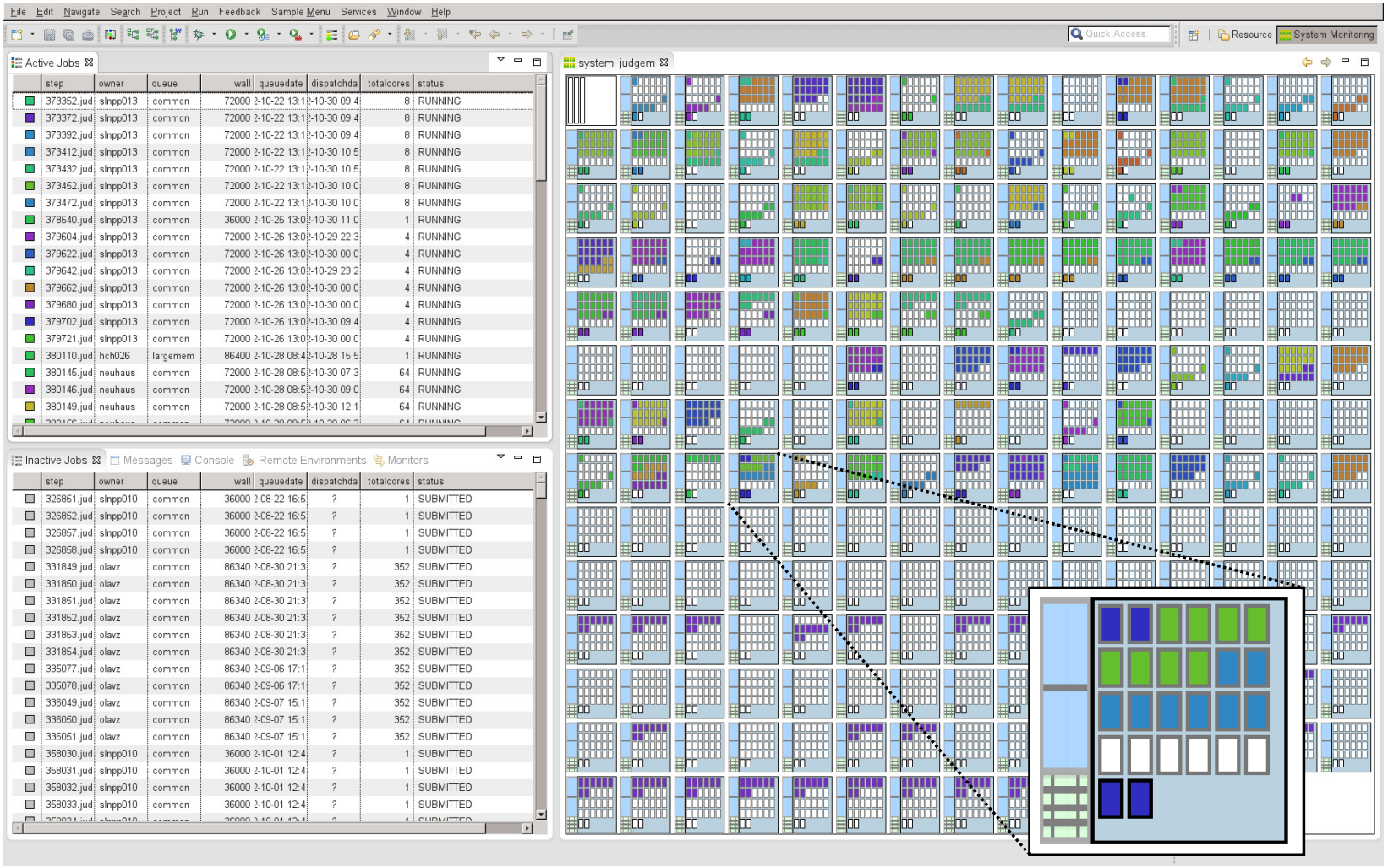
JÜLICH FORSCHUNGSZENTRUM

Monitoring example: JUROPA



JÜLICH FORSCHUNGSZENTRUM

Monitoring example: JUDGE



JÜLICH FORSCHUNGSZENTRUM

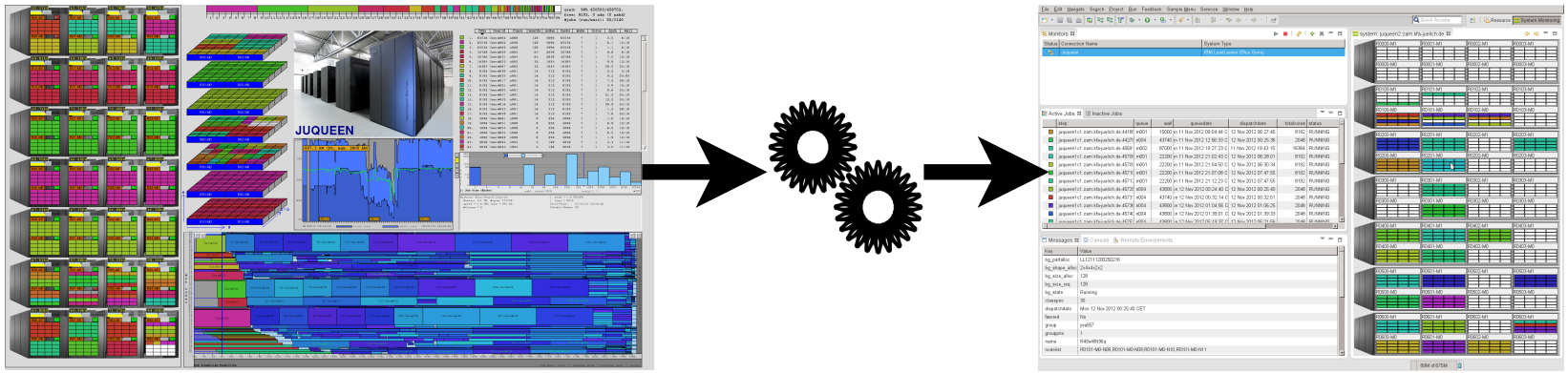
Development history

2009 Start of 3-years project on
A Scalable Development Environment for Peta-Scale Computing

2010 Design of LML and integration into LLview

2011 First PTP release including system monitoring

2013 Collaboration with NCSA and IBM within Seed Fund Project
Advanced System Monitoring for PTP



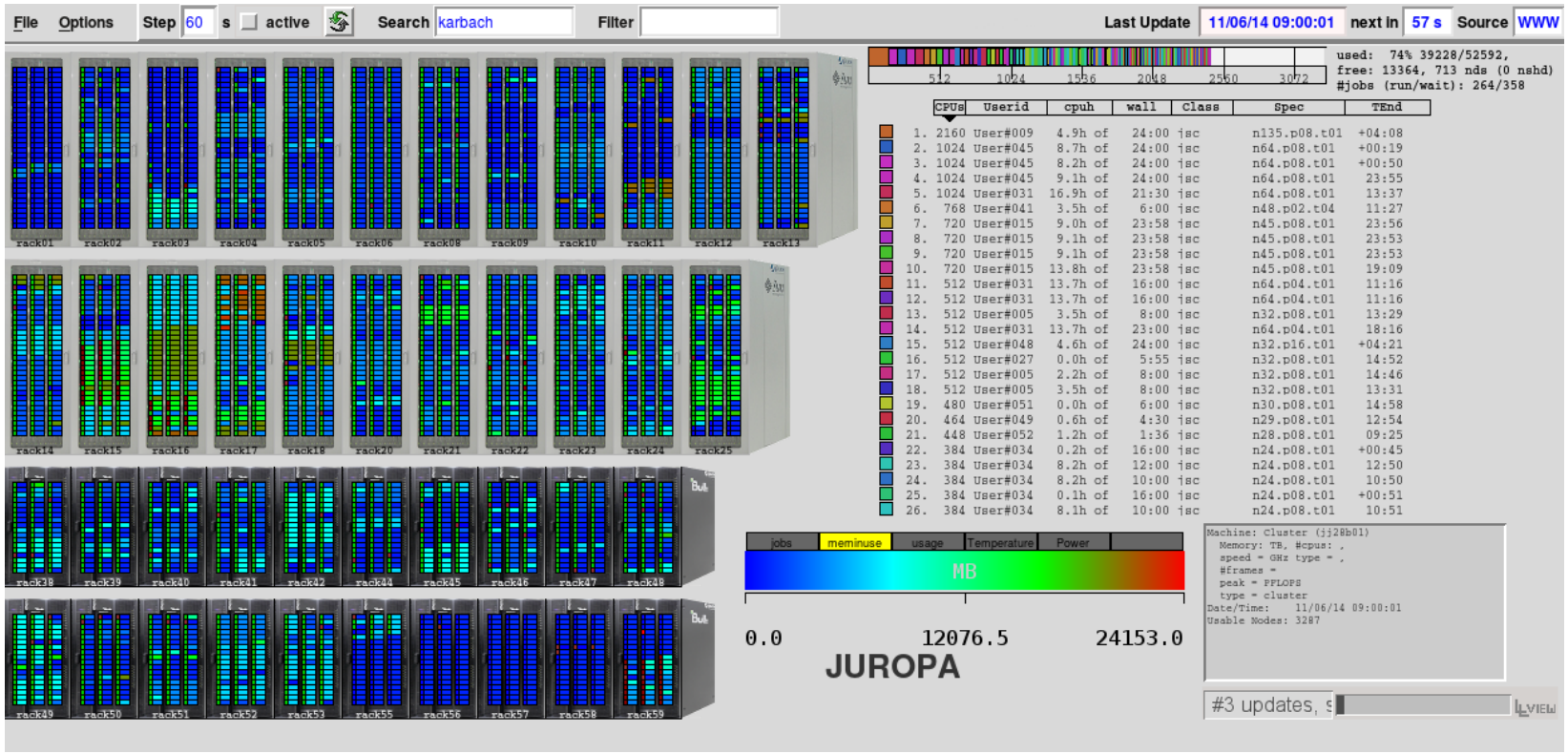
Part III: Latest features

November 25, 2014 | Carsten Karbach

JÜLICH FORSCHUNGSZENTRUM

Node level metrics

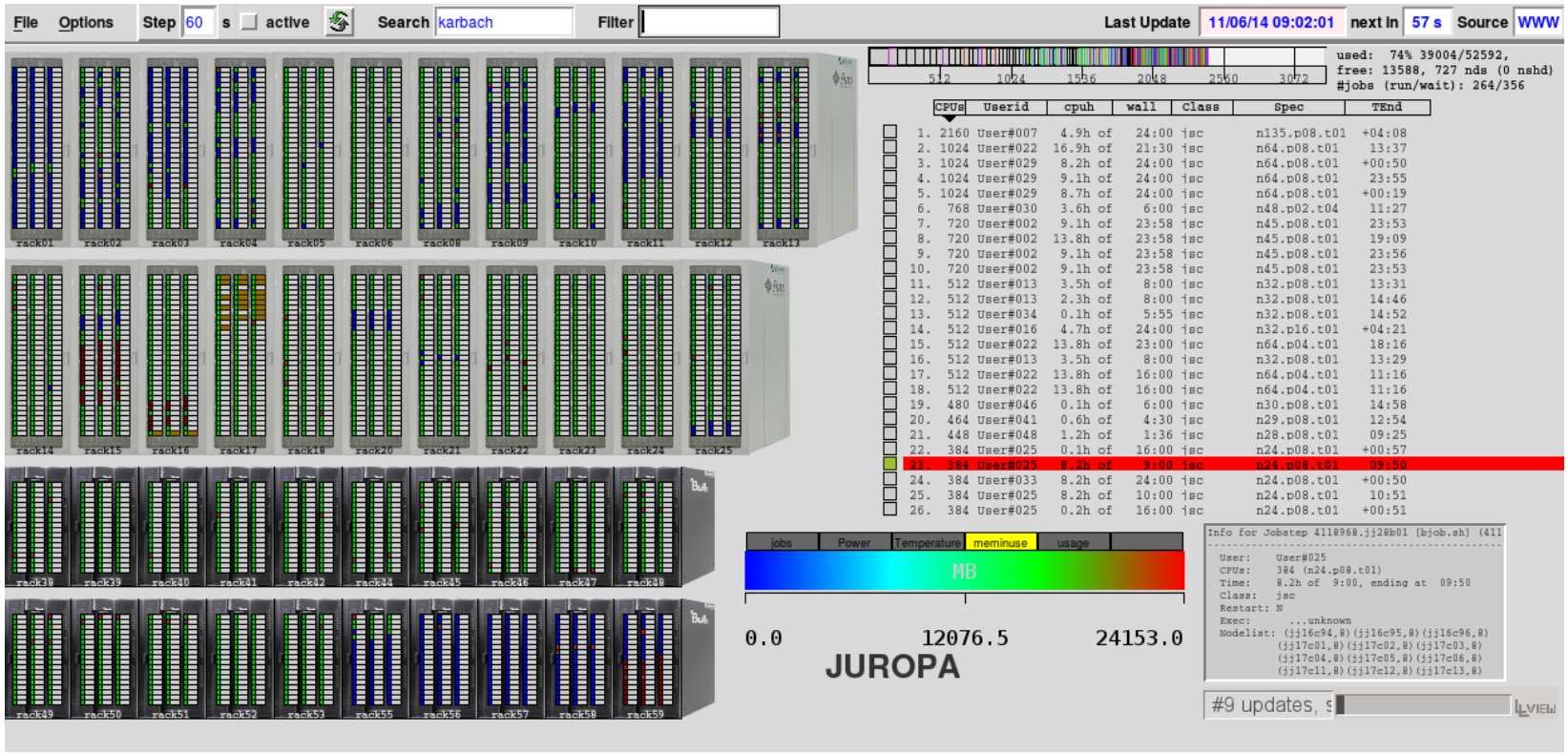
- Add compute node attribute visualization
- Enrich LML data with metrics like temperature, memory/power usage and load
- Use nodes display and color map for visualization



JÜLICH FORSCHUNGSZENTRUM

Node level metrics

- Add compute node attribute visualization
- Enrich LML data with metrics like temperature, memory/power usage and load
- Use nodes display and color map for visualization

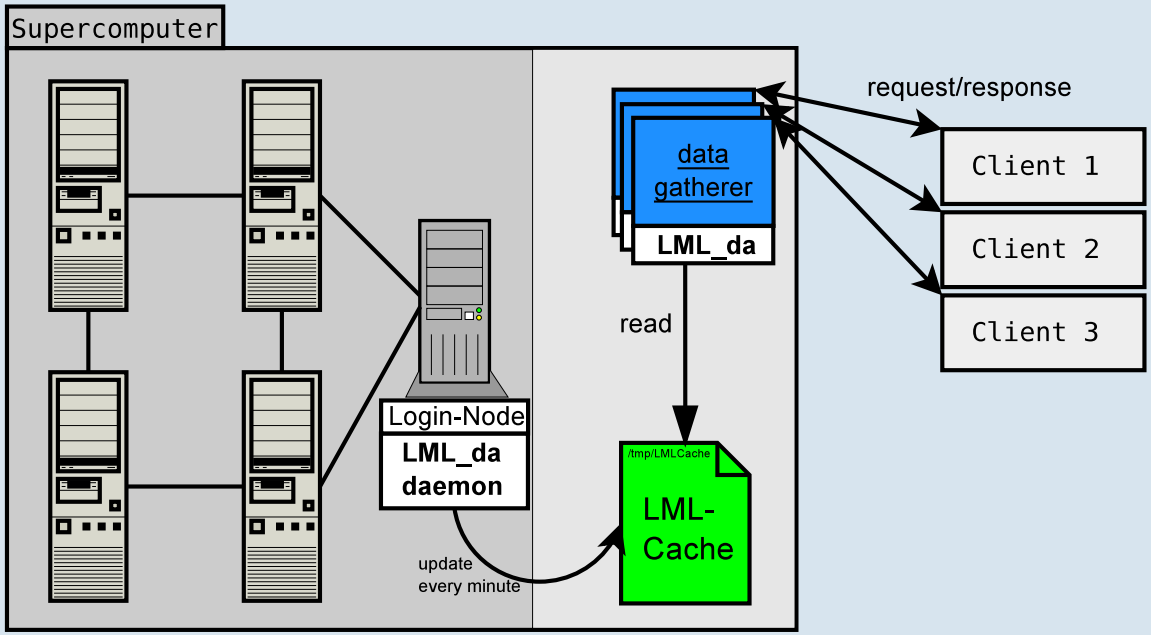


JÜLICH FORSCHUNGSZENTRUM

Server Caching

- **multiple users** on the same target system
- **cache** LML file in public directory (e.g. /tmp), use LML cache as data source
- default: each client triggers independent status data update
- caching: daemon retrieves status data, clients use cached data

Cache workflow



JÜLICH FORSCHUNGSZENTRUM

Advantages of caching

- Faster response time

System	Cache [s]	No Cache [s]
JUDGE	1.1	2.8
JUROPA	5.8	11.1
JUQUEEN	1.7	34.4

- Recording of **history** data is possible
- Enhancement with data not accessible to normal user
- **Decreased load** for the system

Part IV: Future development

November 25, 2014 | Carsten Karbach

JÜLICH FORSCHUNGSZENTRUM

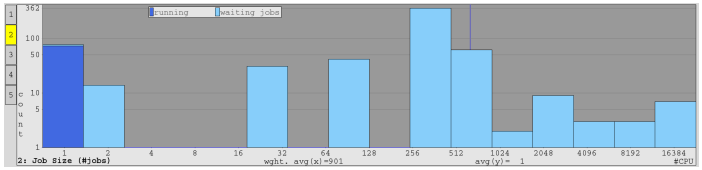
Customized LML layouts

- understand system architecture and hierarchy
- **map topology** into LML layout
- advantages: level of detail, automatic job filtering, display node names, improved performance
- workplan: tutorial on LML layouts, contact system administrators of partner XSEDE/PRACE sites, support writing of LML layouts, ask for feedback

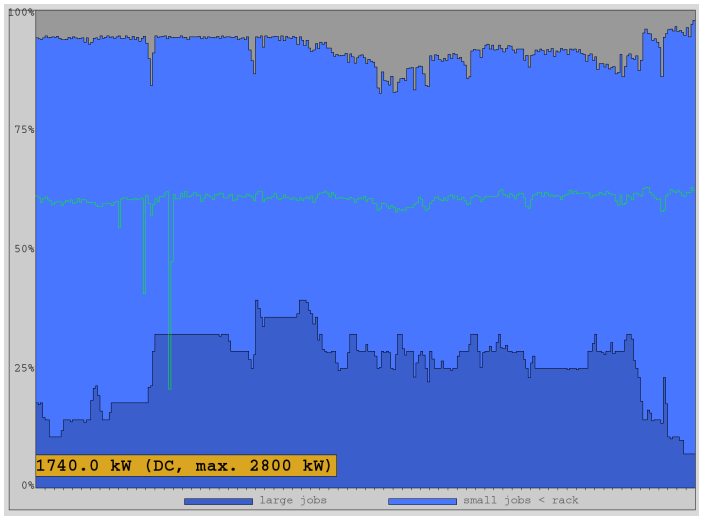
JÜLICH FORSCHUNGSZENTRUM

LLview diagrams

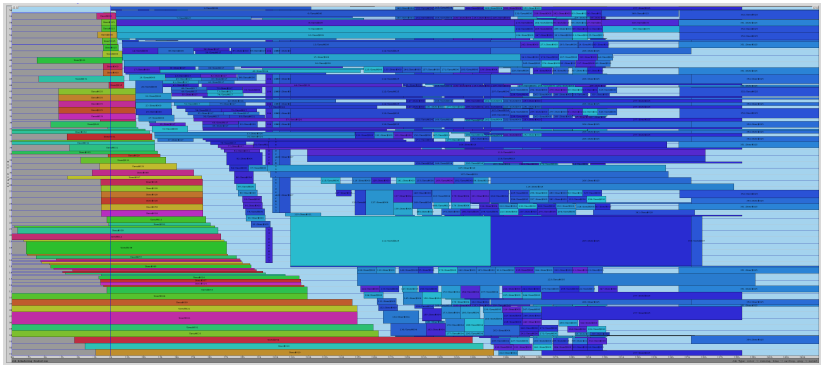
- histograms



- load history



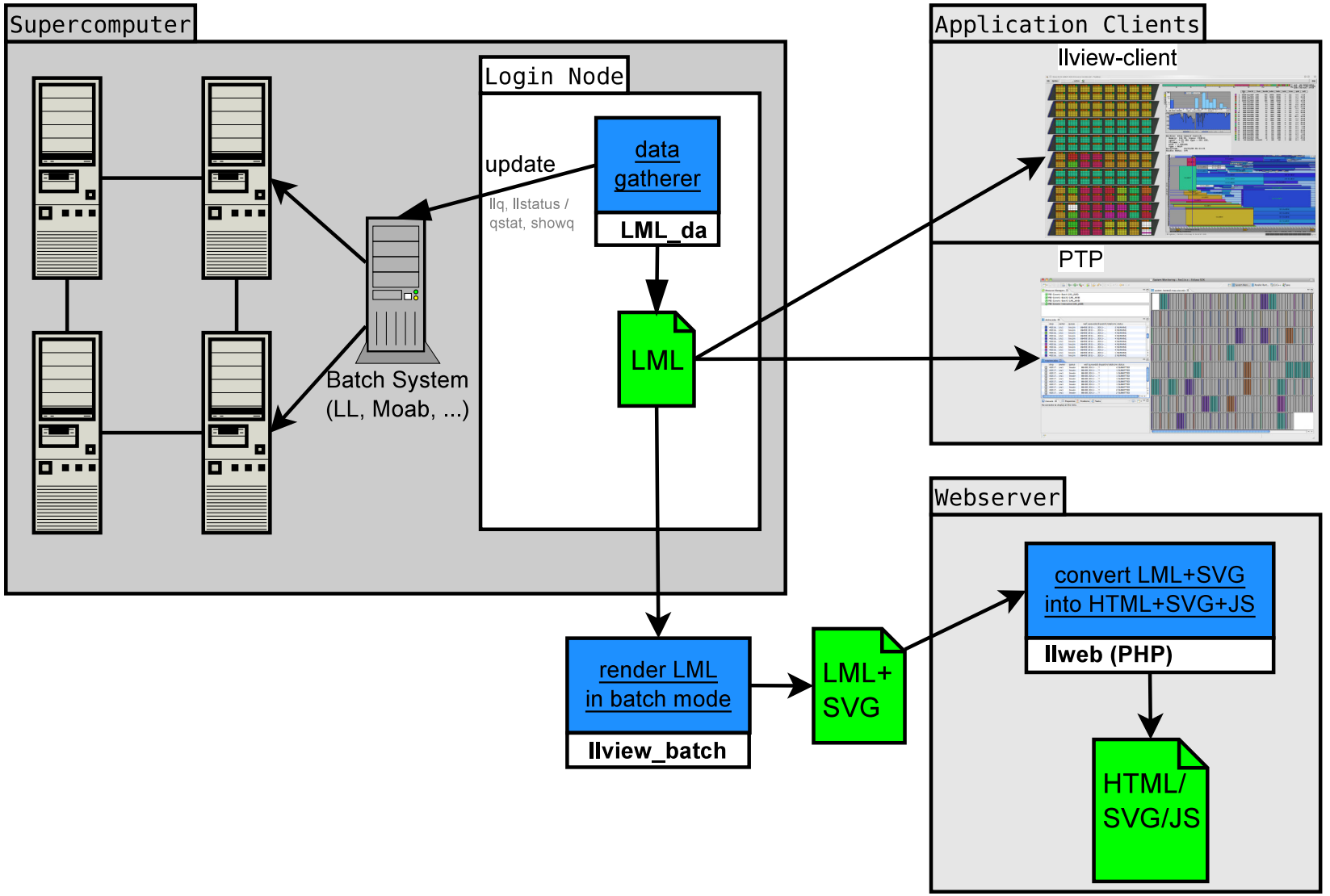
- prediction



Challenges

- Re-implementation in Java?
- Double update for LLview and PTP

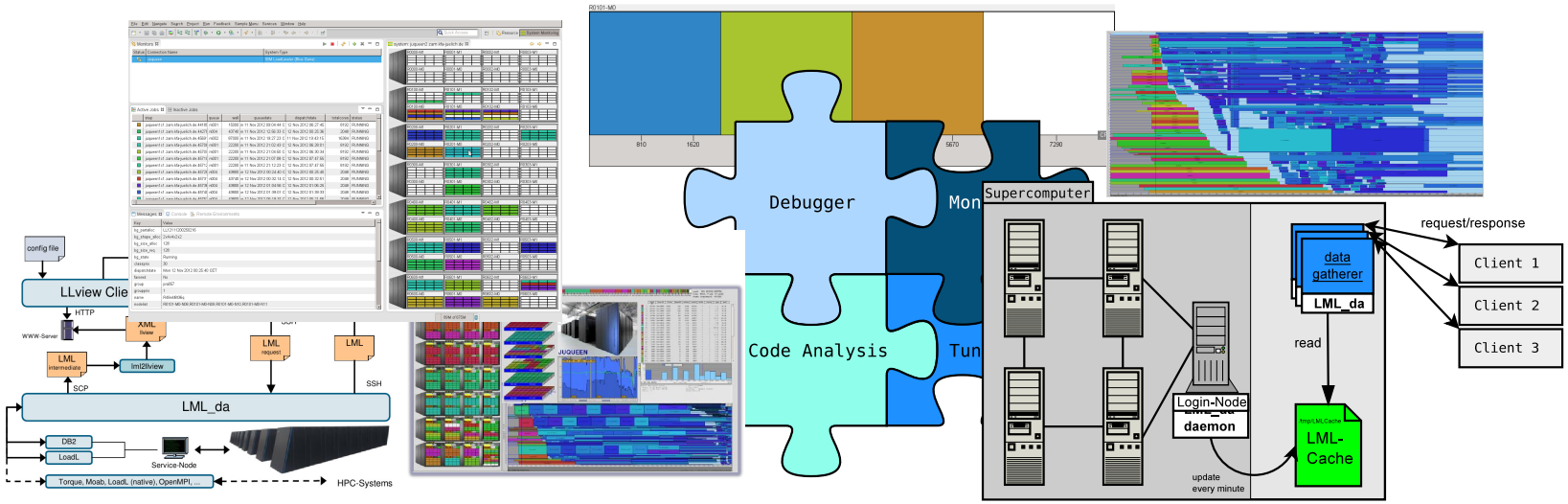
First implementation step



JÜLICH FORSCHUNGSZENTRUM

Conclusion

- Monitoring is **vital for production** of HPC systems
- **LLview** visualizes system status in a single screen
- **PTP** integrates monitoring into development workflow
- Future development will optimize PTP's monitoring workflow for large scale systems



Proven scalability

- PTP/LLview were successfully tested on:

System	Batch system	Cores	Peak	#Jobs
JUQUEEN	LoadLeveler	458K	5.9 PF	100-1000
Jaguar	Torque/Moab	224K	1.8 PF	–
Mogon	LSF	34K	204 TF	-15000
JUROPA	Torque/Moab	26K	207 TF	200-2000
Lonestar	SGE	22K	311 TF	–

JÜLICH
FORSCHUNGSZENTRUM

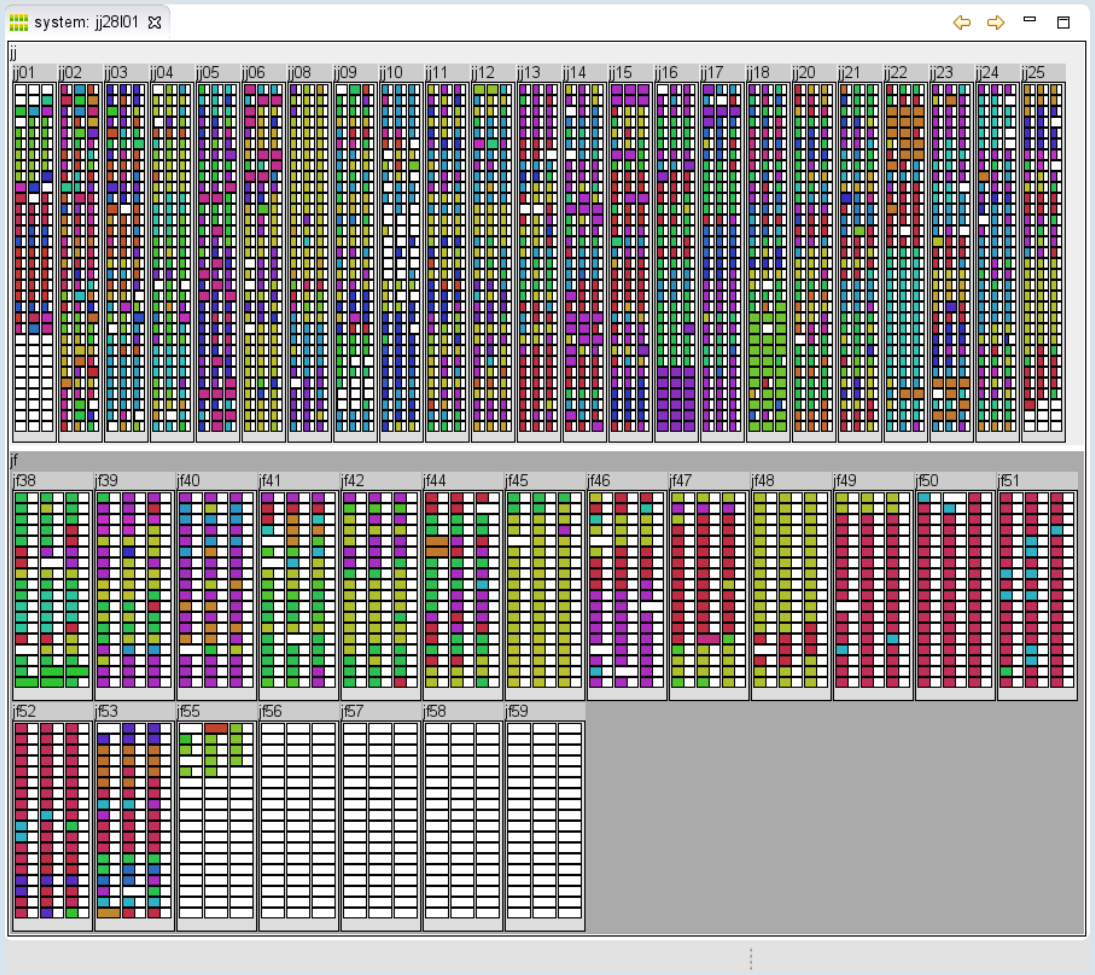
Part I: Examples for customized layouts

November 25, 2014 | Carsten Karbach

JÜLICH FORSCHUNGSZENTRUM

Example JUROPA

Customized layout

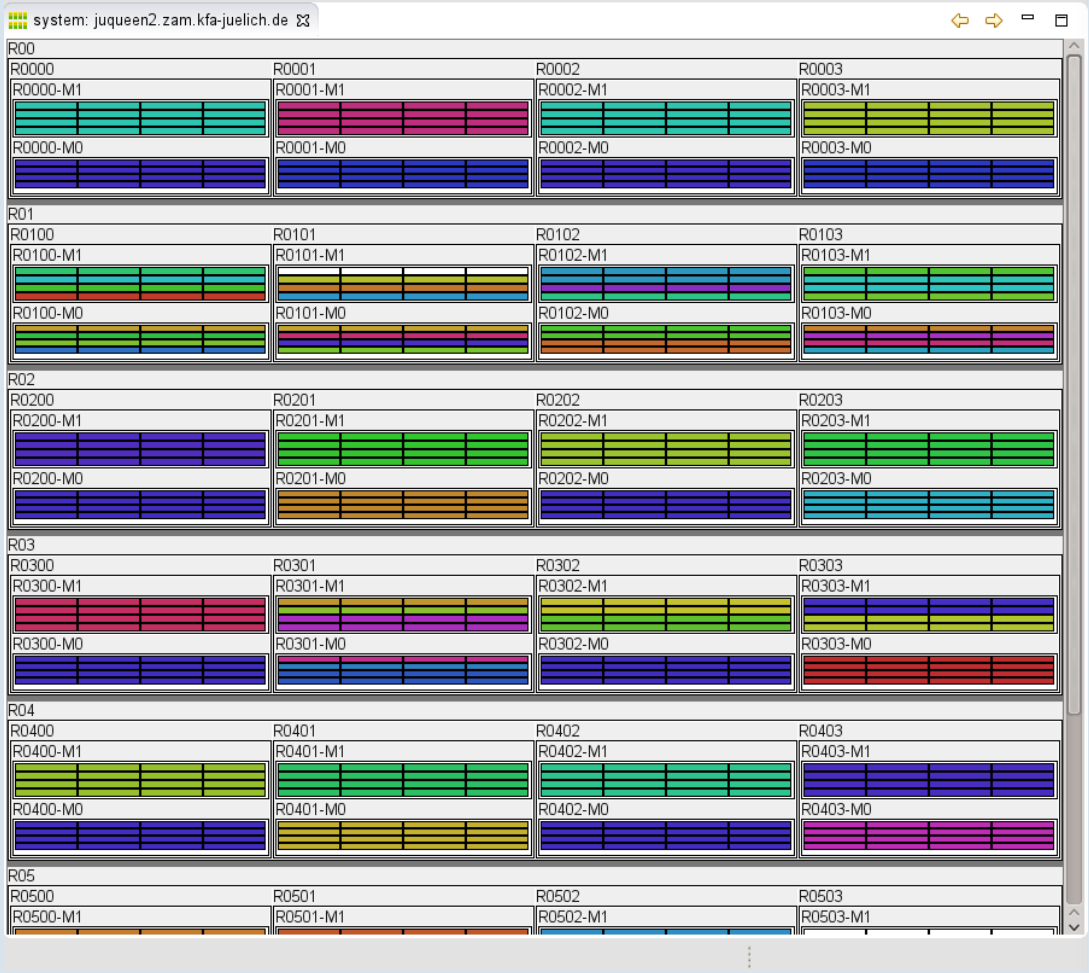


- four level hierarchy: partition, rack, node, core
- level of detail (only with layout)
- better overview

JÜLICH FORSCHUNGSZENTRUM

Example JUQUEEN

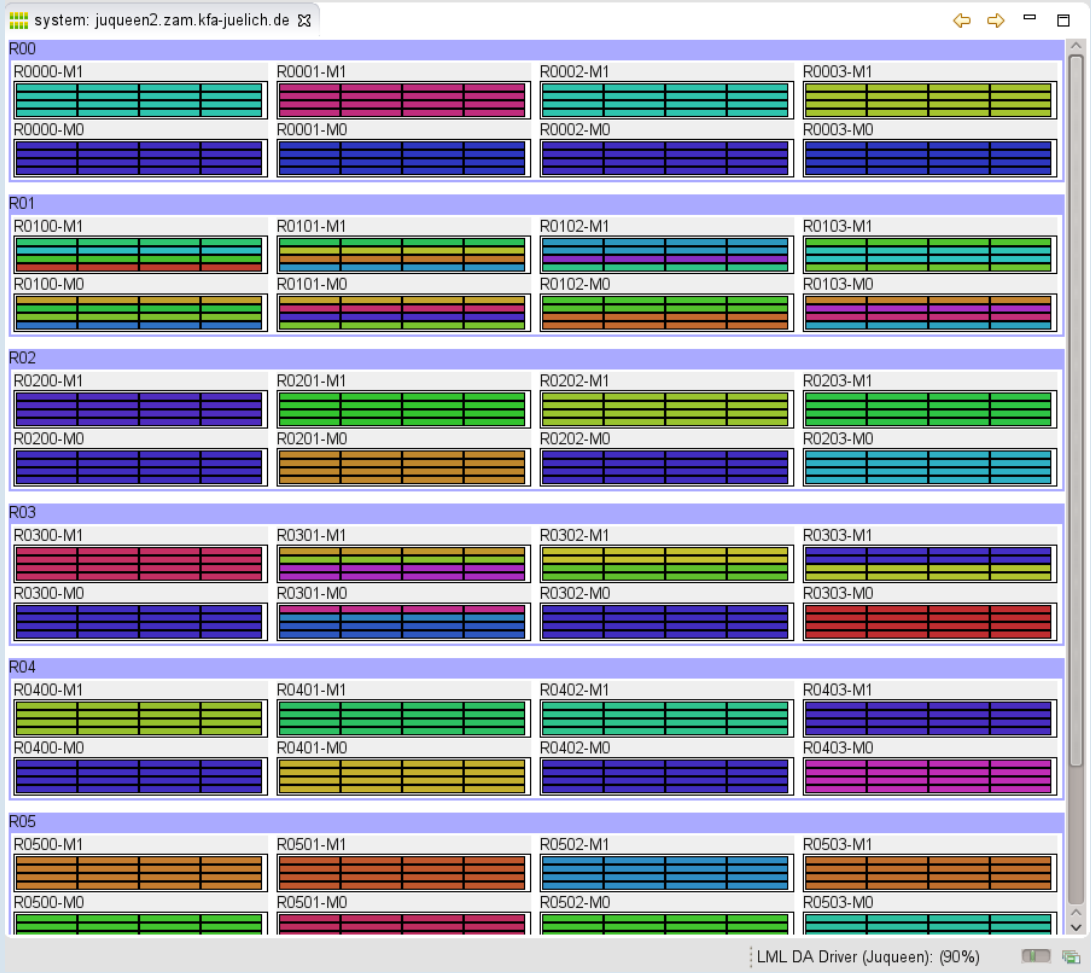
Default layout



JÜLICH FORSCHUNGSZENTRUM

Example JUQUEEN

Customized layout



- colored rows
- do not display rack names

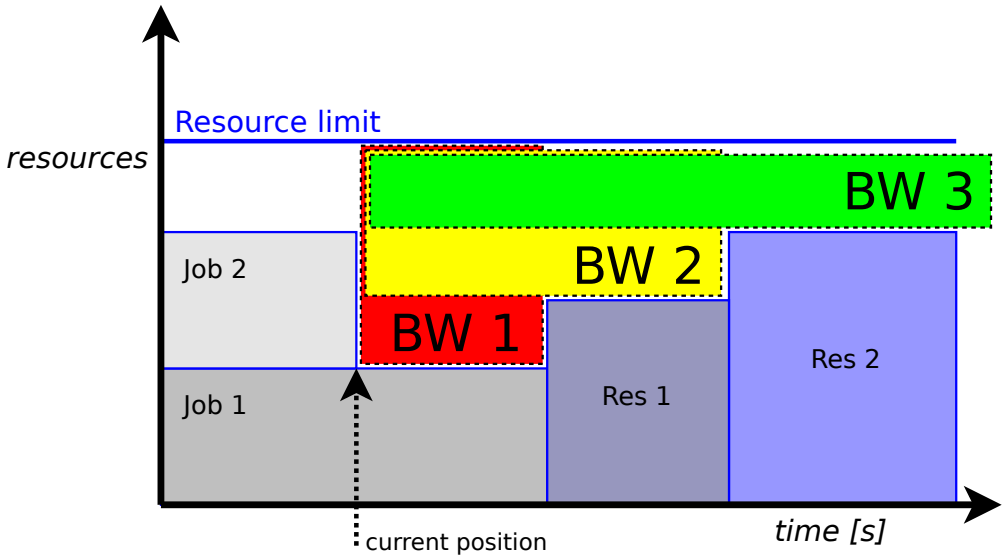
Part II: JuFo

November 25, 2014 | Carsten Karbach

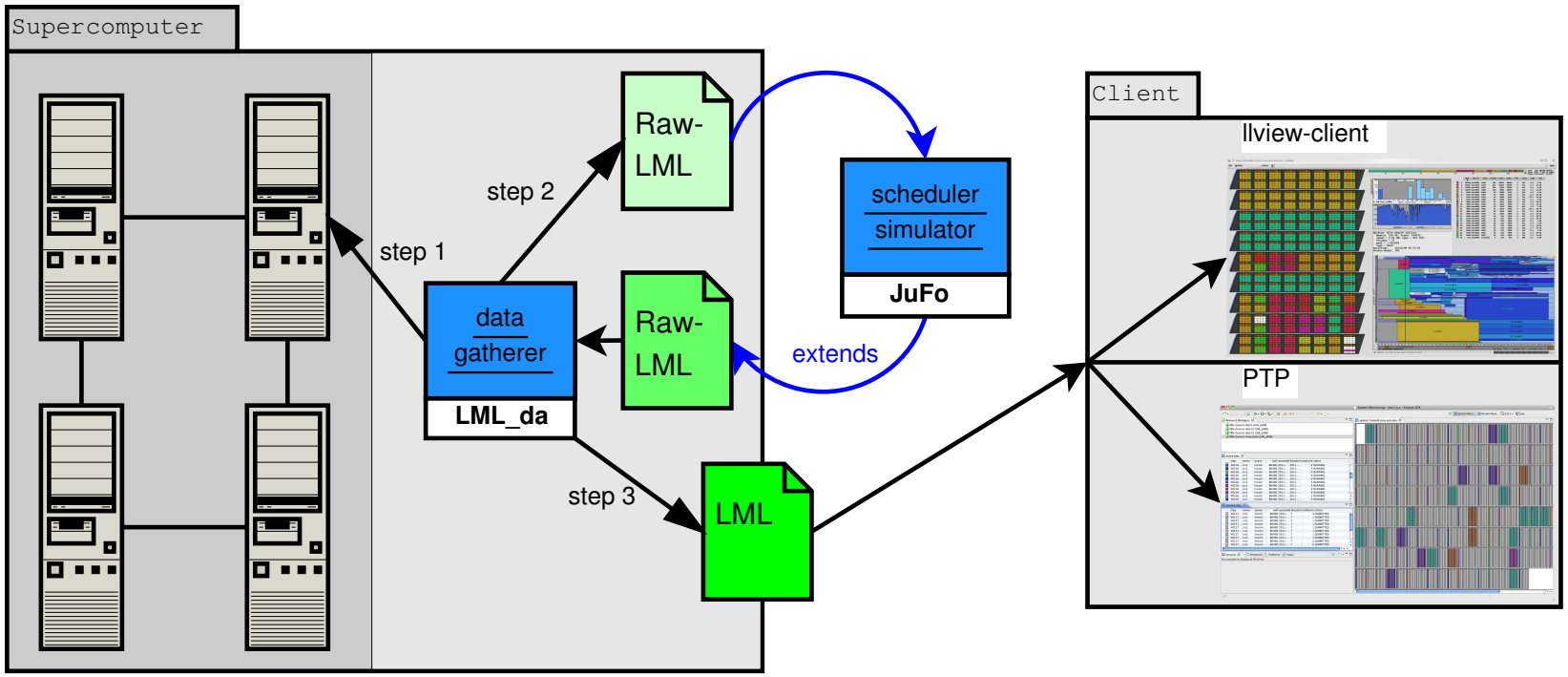
JÜLICH FORSCHUNGSZENTRUM

Overview

- **Configurable** simulator for global job schedulers for **on-line prediction** of job dispatch dates
- Based on analysis of JSC batch systems **Moab** and **Loadleveler**
- **Integrated** with monitoring system **LLview**
- **LML** as configuration and communication data format
- **Use-cases**:
  - **User** predicts start dates of submitted jobs
  - **Administrator** simulates job scheduler performance with various input parameters



Architecture



JÜLICH FORSCHUNGSZENTRUM

Features

- Supported **scheduling algorithms**
  - First-Come-First-Served
  - List-Scheduling
  - Backfilling
- Available **simulation parameters**
  - Generic job **prioritization**
  - Advanced **reservations**
  - Jobs can request CPUs, GPUs, memory
  - **Nodesharing**
  - **Queue** constraints
- Test framework for evaluating JuFo's accuracy