OMIS v2.0 — A Universal Interface for On-line Monitoring Systems

Roland Wismüller, Thomas Ludwig

LRR-TUM
Institut für Informatik
Technische Universtität München, Germany
e-mail: wismuell@informatik.tu-muenchen.de

http://wwwbode.informatik.tu-muenchen.de/˜omis

Motivation

Powerful tools for parallel and distributed programming need on-line monitoring facilities for **observation** and **manipulation** of programs during runtime.

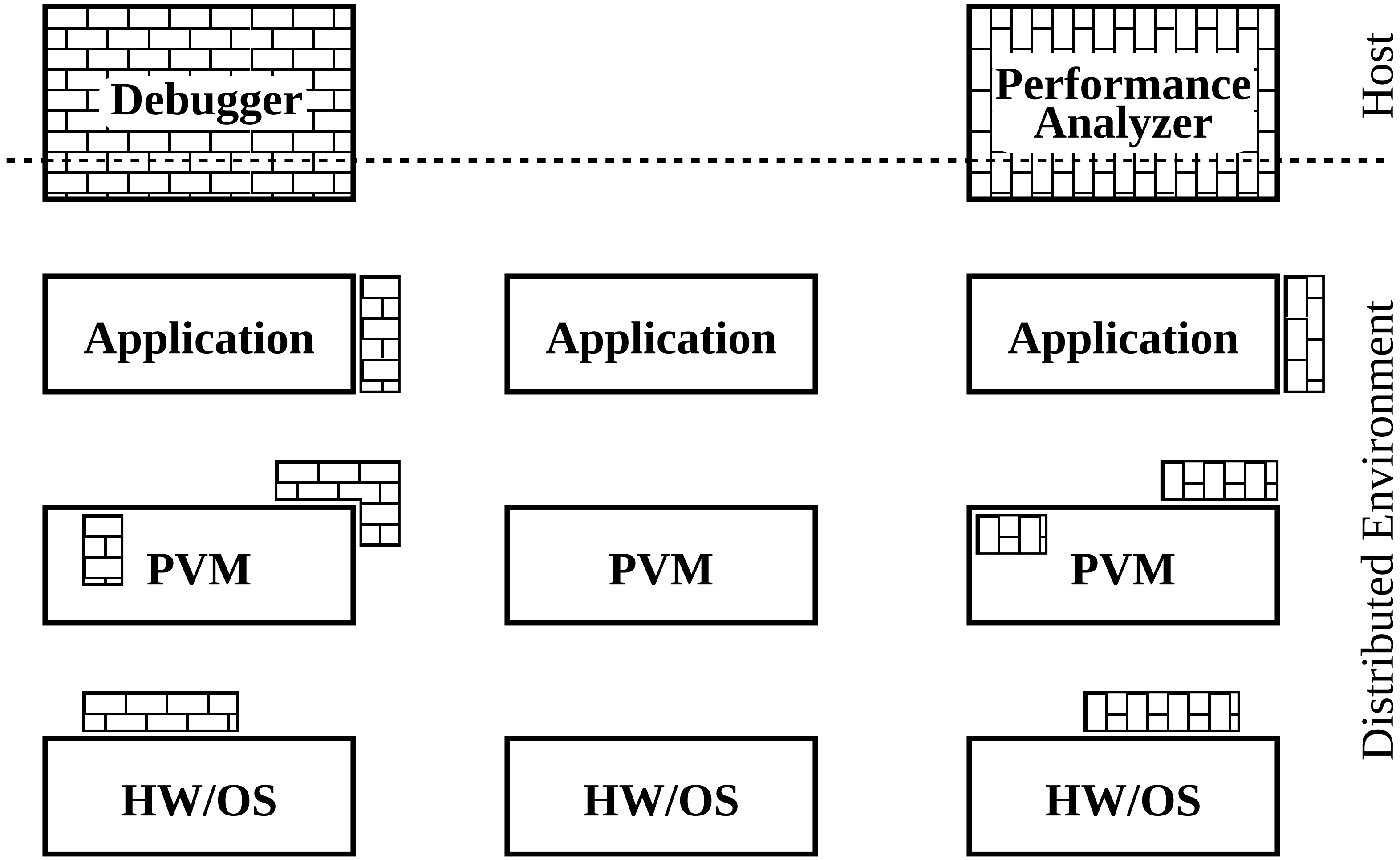
E.g. debugging, performance analysis, load balancing, computational steering ...

Sophisticated tools already exist, **but**:
all use proprietary monitoring systems which are incompatible to each other!

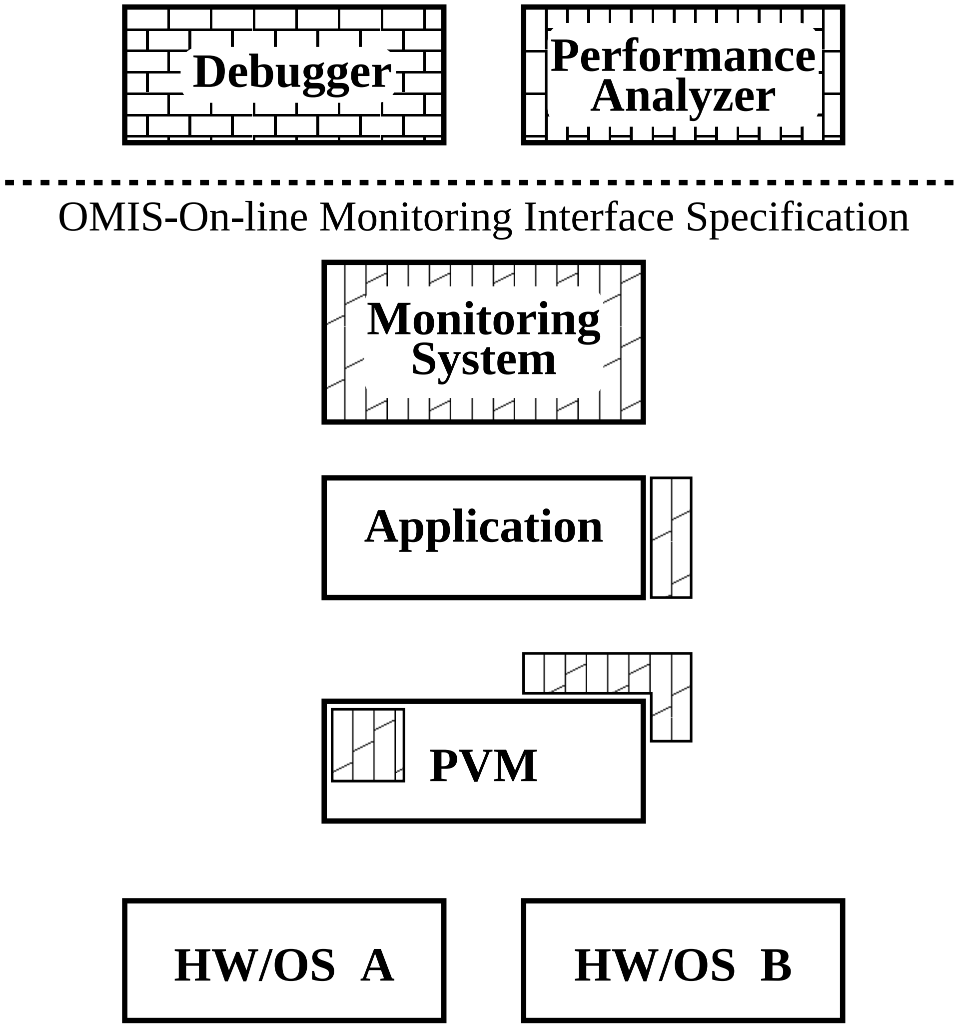
No standard infrastructure exists to connect tools to running systems

- ⇒ every new tool requires to implement a new monitoring system
- ⇒ no interoperable tools
- ⇒ no uniform tool environments

Tool Environments Without OMIS



Tool Environments With OMIS



OMIS v2.0 — On-line Monitoring Interface Specification

OMIS has been started in Summer 1996 as a joint project between Emory University and LRR-TUM.

Project Goals:

- define a universally usable on-line monitoring interface
- separate tool development from monitoring system development
  - ⇒ R&D groups involved in tool design and implementation
  - ⇒ R&D groups involved in monitor design and implementation
- provide basis for interoperable tools and uniform tool environments

OMIS v1.0 finished in Feb. 1996

Design of an OMIS implementation and feedback from other research groups lead to revised version 2.0

Requirements on a Universal Monitoring Interface

The interface should ideally be

- usable for any kind of run-time tools (interactive/automatic, centralized/distributed)
- independent of the target platform (hardware, operating system, programming library)
- implemented efficiently with minimal intrusion

$\rightsquigarrow$ To meet all of these requirements at the same time is hard!

Approach:

- provide a very flexible interface
- provide means to extend the interface in an easy and well-defined way

Flexibility and Extendibility

Flexibility:

- interface for centralized and distributed tools
- event-action paradigm:
  - request consists of event definition and action list
  - events needed by different tools are fairly the same
  - actions may differ $\Rightarrow$ extensions for specific tools

Extendibility:

- distributed tool extensions
  define derived events and new actions, platform independent
- monitor extensions
  define new objects and new events, platform dependent

The Monitoring Interface

Interface procedure (usable by tools and extensions):

```
Omis_reply
omis_request( char * request,
              void (* callback)(Omis_reply reply, void *param),
              void *param,
              Omis_flags flags )
```

General syntax of requests:

request	::=	[*event_definition*] : *action_list*
event_definition	::=	*service_name* (*parameters*)
action_list	::=	*action* | *action* [;] *action_list*
action	::=	*service_name* (*parameters*)

An action list may execute concurrently; a semicolon acts as a global synchronization (barrier).

Enhancements in OMIS v2.0

Feedback on OMIS:

- OMIS v1.0 is very dependent on PVM and NOWs

  $\Rightarrow$ new object based model of the observed system (threads, SMPs)

  $\Rightarrow$ each tool defines the observed system explicitly

  $\Rightarrow$ all PVM specific details are now in an extension

- Explicit specification of nodes is often cumbersome

  $\Rightarrow$ location transparency

- Reply string is a source of inefficiency

  $\Rightarrow$ reply is more structured now

- Services are not completely specified

  $\Rightarrow$ OMIS v2.0 contains full specifications (optional and required parts)

Support for different target platforms

Object based approach

- Abstract objects: nodes, processes, threads, messages, and message queues
- Hardware and process model suitable for DMPs, NOWs, SMPs, and clusters of SMPs
- Identification by object handles, automatic conversions (localization, expansion)
- OMIS core only defines platform independent services (others: platform specific extensions)

Location transparency

- Events and actions work on any sets of objects
- Tools *need* not care about the location of these objects

Example: Debugging

```
thread_has_started_lib_call([],"pvm_send") :
    proc_get_info([$proc],12)  thread_get_backtrace($thread,1)
```

Whenever a process calls pvm_send, return its tid, its argument vector, and the return address of the call

```
thread_reached_addr([p_12,p_34,p_44],0xfe08) : thread_stop([])
```

When one of the specified processes reaches the breakpoint, stop the whole application

```
thread_creates_proc([p_34]) : proc_attach($new_proc)  thread_stop($new_proc)
```

Debug (i.e. attach and stop) new processes dynamically created by process p_34

OCM: An Implementation of OMIS

OCM = OMIS compliant monitoring system

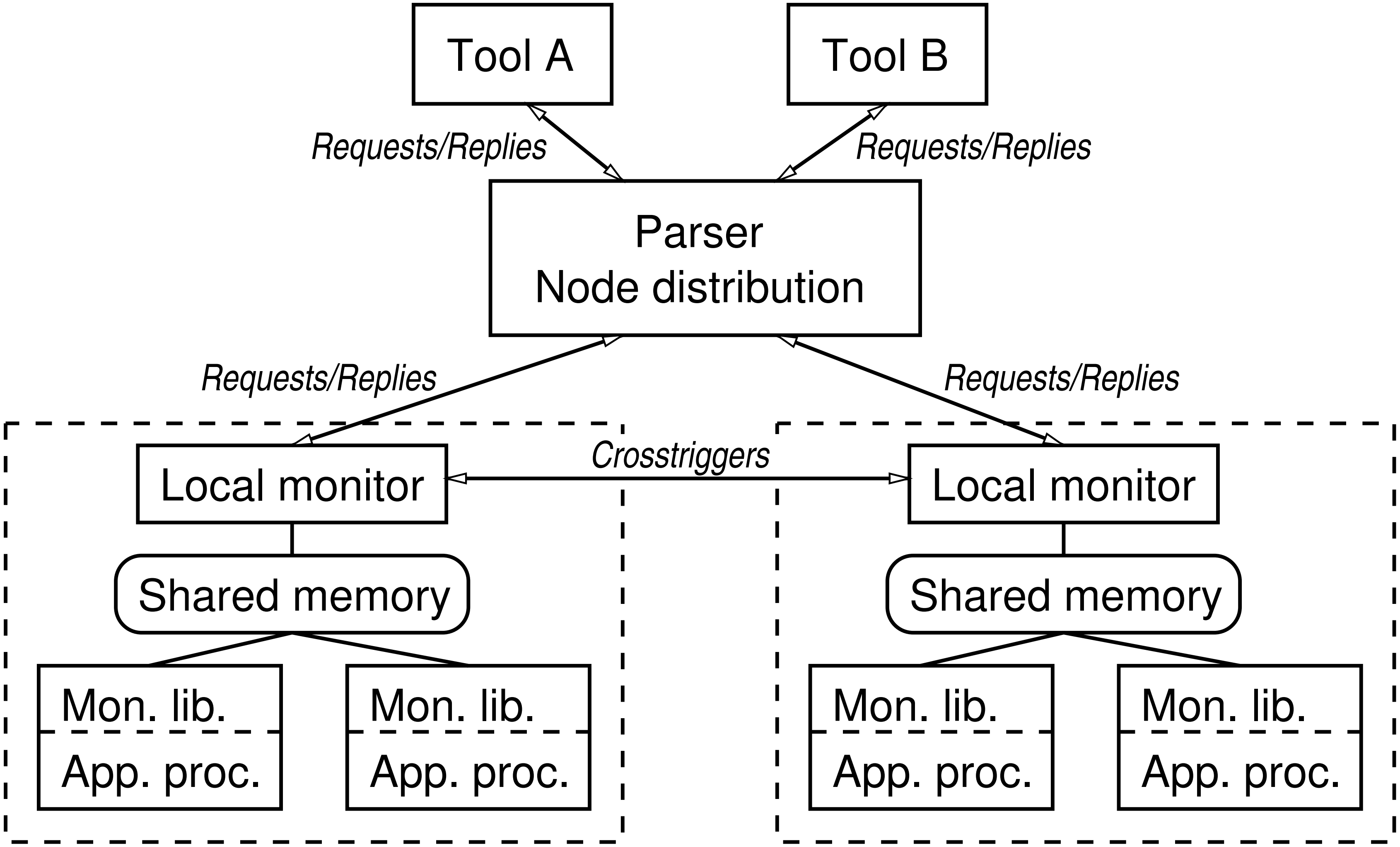
Implementation of OMIS for PVM on clusters of workstations

Project goals:

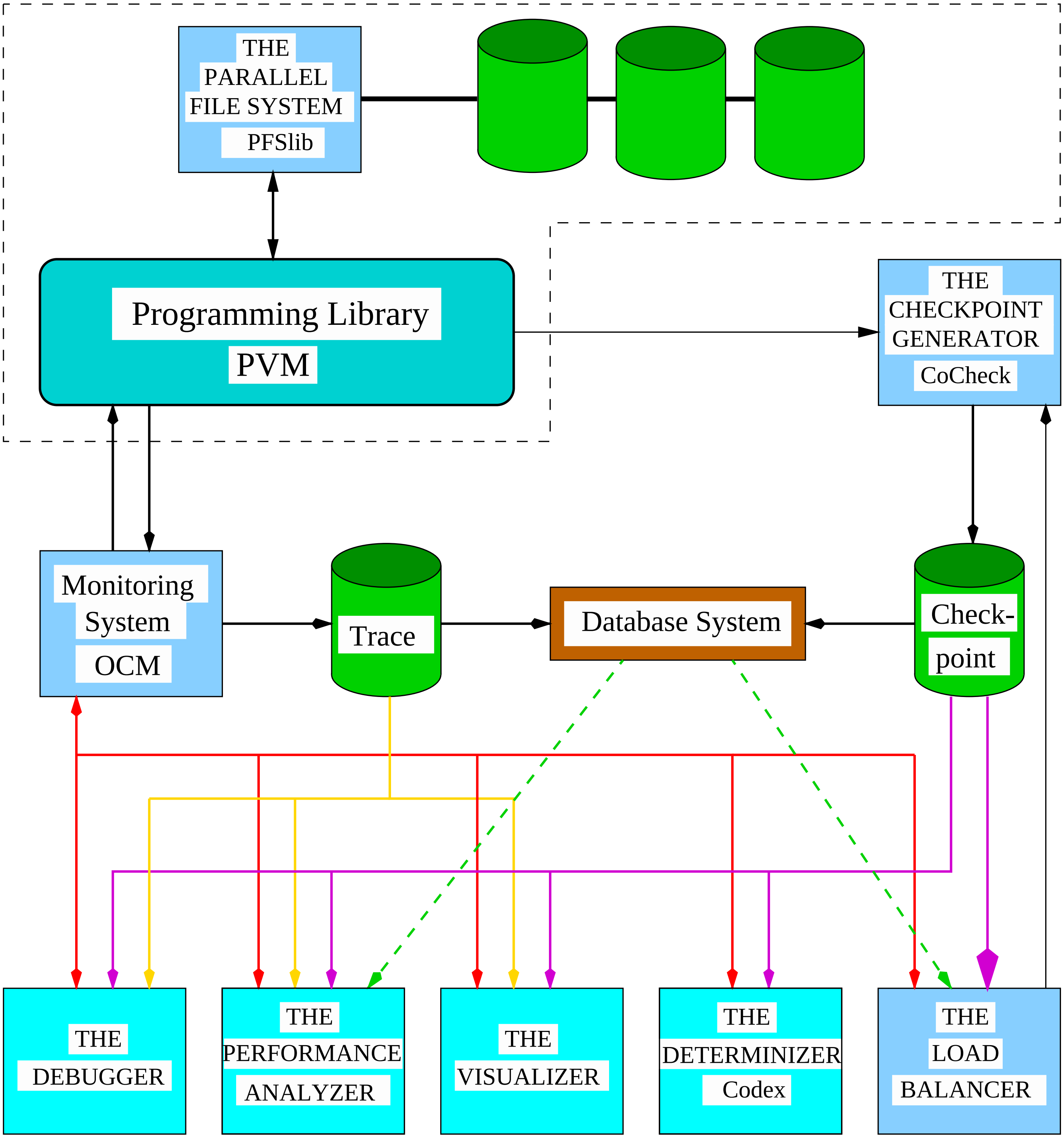
- Verify the practicability of OMIS $\Rightarrow$ feedback
- Provide an implementation platform for THE TOOL-SET
- Provide an implementation platform for other tool environments
- Prepare a reference for other OMIS implementations

OCM will be released under the GNU license conditions!

Coarse Grain Structure of OCM



Application of OCM: THE TOOL-SET for PVM



Project Status

OMIS

- Version 1.0 released Feb. 1996
- Major update: version 2.0
  draft is already finished, release in Mar./Apr. 1997

OCM

- Implementation started Jan. 1997 (3 researchers and 4 students)
- First Implementation: Jul. 1997
- First tools of THE TOOL-SET: Oct. 1997

Cooperations

- ENS Lyon, France (Dosmos)
- KFKI-MSZKI Research Institute, Budapest, Hungary (GRADE)
- GUP, Univ. of Linz, Austria (MAD)

Future Work

- Finish implementation of OCM and adaption of tools
- Optimize OCM
- Integrate new tools: checkpointing, load balancing, deterministic execution
- Integrate feedback into OMIS document
- OMIS for distributed shared memory systems
- Further discussions with other research groups

Any feedback is welcome, please look at:

http://wwwbode.informatik.tu-muenchen.de/~omis