

OMIS v2.0

Eine universelle Schnittstellenspezifikation für
Online-Monitorssysteme

Thomas Ludwig

Übersicht

- Software-Werkzeuge bei parallelen und verteilten Systemen
- Werkzeugumgebungen ohne und mit OMIS
- Ziele des OMIS Projekts
- Anforderungen an eine Schnittstellenspezifikation
- Das Systemmodell
- Das Überwachungssystem
- Die Monitorschnittstelle
- Erweiterungen in OMIS v2.0
- Beispiele: Fehlersuche und Leistungsanalyse
- OCM: Die Implementierung von OMIS
- Projektstatus und Weiterführung

Software-Werkzeuge

Einsatzbereiche

Entwicklung, Optimierung und Laufzeitunterstützung paralleler und verteilter Programme

Benutzungsmethodik

Interaktiv — Automatisch

Benutzungszeitpunkt

Zur Laufzeit (online) — Nach Programmende (offline)

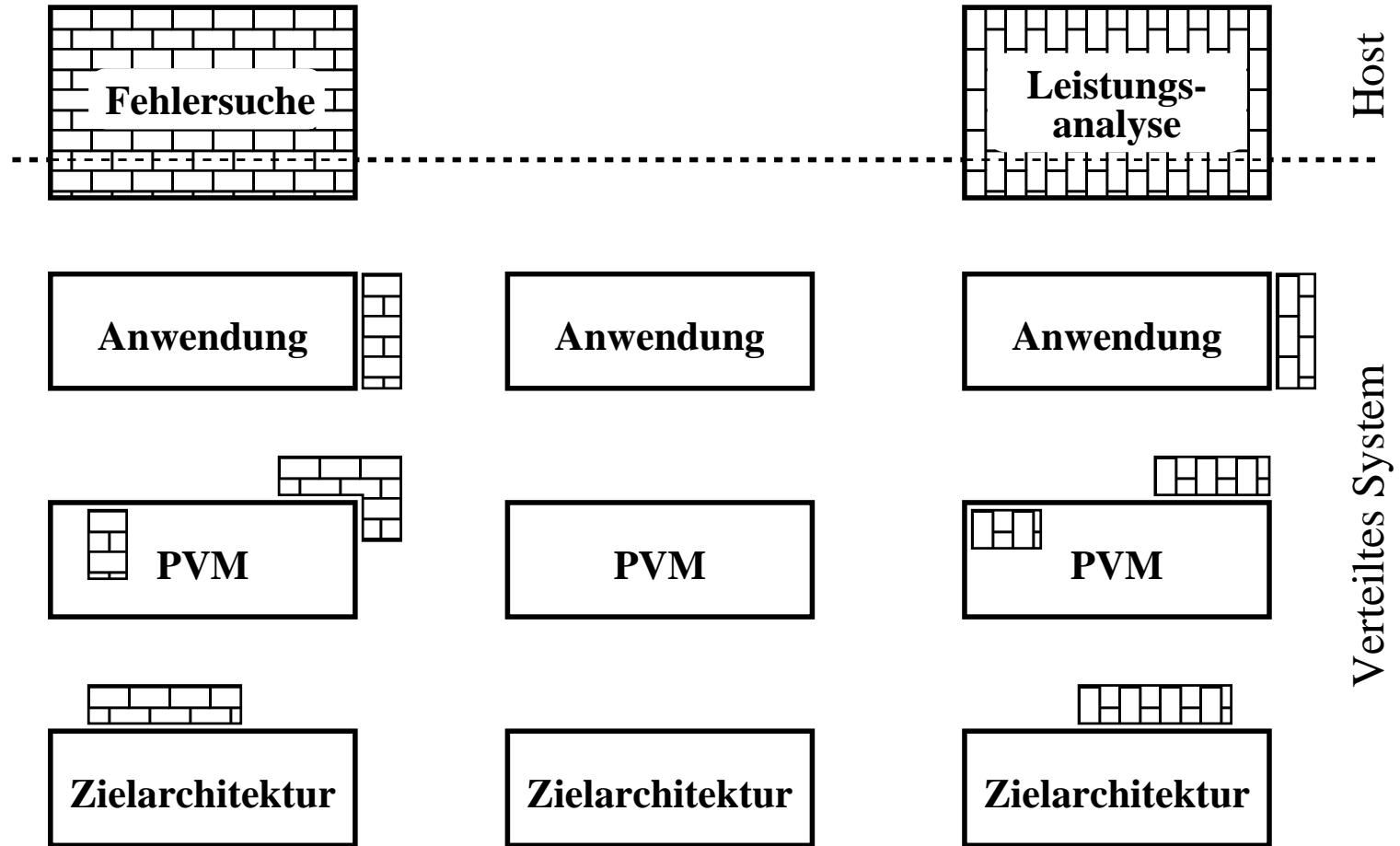
Klassen der Programminteraktion

Beobachtung — Manipulation

Mittel der Programminteraktion

Überwachungssystem (monitoring system)

Werkzeugumgebungen ohne OMIS



Leitgedanken des OMIS-Projekts

OMIS — On-line Monitoring Interface Specification

Trennung der Entwicklung von

- Software-Werkzeugen und
- Überwachungssystemen

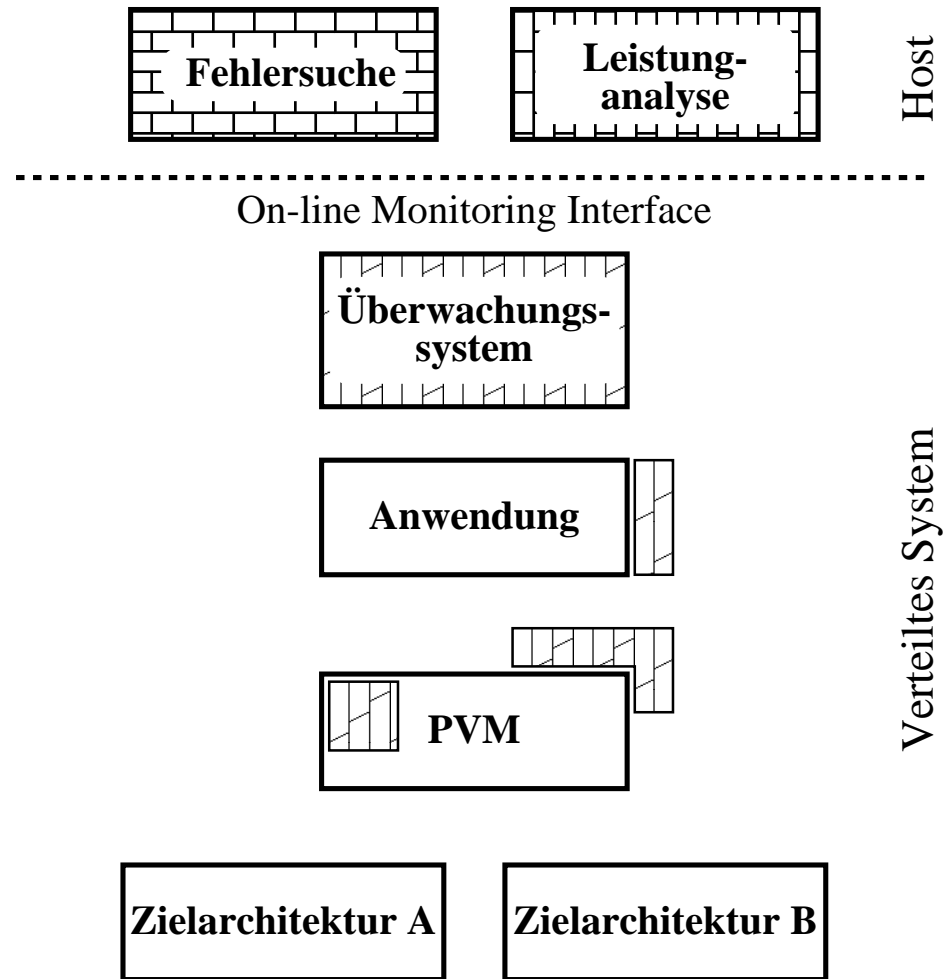
Bereitstellung interoperabler Software-Werkzeuge

- Basierend auf einer gemeinsamen Schnittstelle zum Überwachungssystem
- Z.B. Umschaltung von Visualisierung zur Fehlersuche

Bereitstellung uniformer Werkzeugumgebungen

- Identische Werkzeuge auf allen Zielarchitekturen mit OMIS Überwachungssystem

Werkzeugumgebungen mit OMIS



Ziele des OMIS-Projekts

1. Spezifikation einer erweiterbaren Schnittstelle zwischen Überwachungssystemen und Werkzeugen (OMIS — on-line monitoring interface specification)
2. Entwurf und Implementierung eines OMIS Überwachungssystems für eine ausgewählte Zielarchitektur
3. Entwicklung einer Werkzeugumgebung für dieses Überwachungssystem
4. Verwaltung und Ausbau der Schnittstellenspezifikation

OMIS wurde im Sommer 1996 in Kooperation mit der Emory University Atlanta begonnen

OMIS v1.0 veröffentlicht im Februar 1996; OMIS v2.0 erscheint im Juli 1997

Anforderungen an eine Schnittstellenspezifikation

Allgemeingültigkeit und Flexibilität

- Verschiedene Werkzeuge zu unterschiedlichen Zwecken
- Verschiedene Werkzeugarchitekturen
- Verschiedene programmiersprachliche Objekte

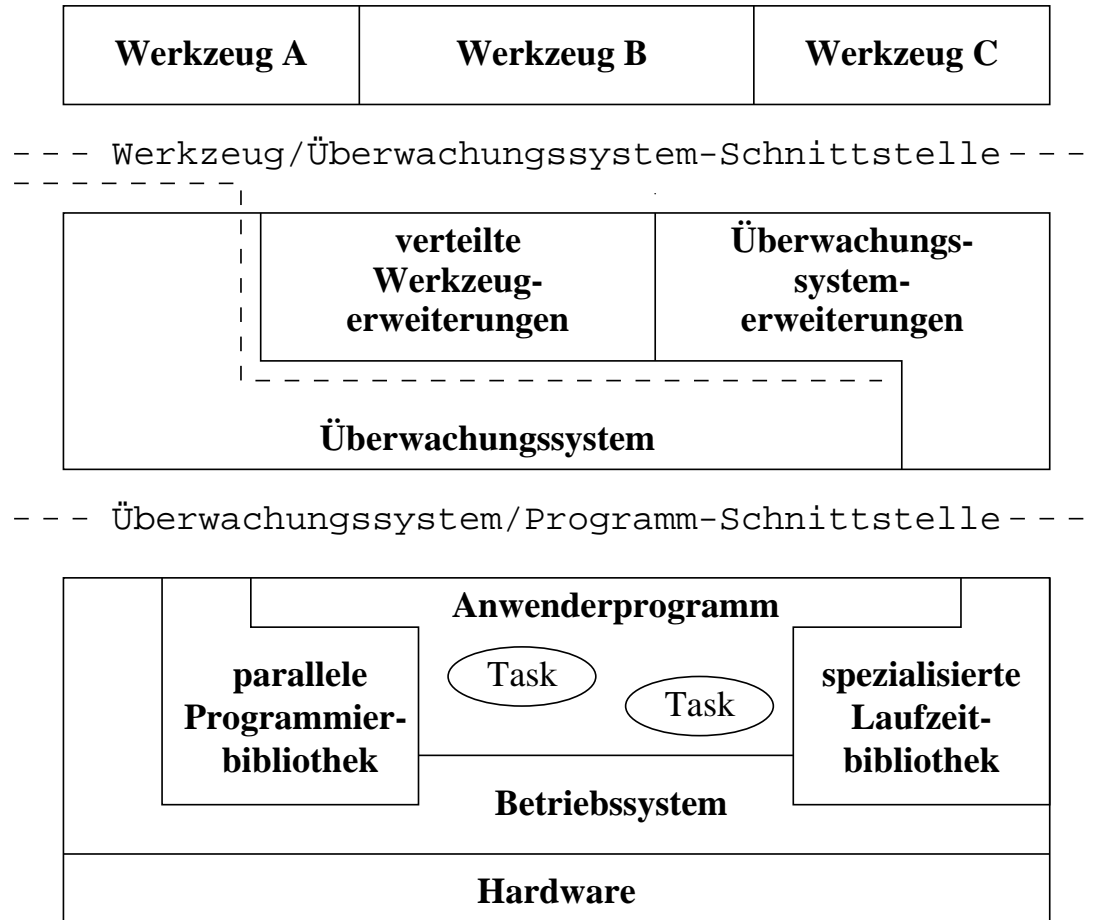
Erweiterbarkeit

- Neue Werkzeugkonzepte
- Neue zu überwachende Objekte

Effizienz

- Geringes Kommunikationsaufkommen zwischen Werkzeugen und Überwachungssystem

Das Systemmodell



Das Überwachungssystem

Arbeitsprinzip: Ereignis/Aktions-Modell

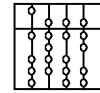
- Ereignis = interessierender Zustandsübergang im überwachten Programm
- Aktion = erwünschte Beobachtung oder Manipulation des Programms
- Verhalten ist durch Ereignis/Aktions-Relationen bestimmt
- Programmierung der Relationen über die definierte Schnittstelle
- Aktionen jeweils ausgelöst, wenn das Ereignis erkannt wird

Diensteklassen

Informationsdienste — Manipulationsdienste — Ereignisdienste

Dienste für hierarchische Objektklassen

- Knotenobjekte
Knotenanzahl, Statusmeldungen
- Prozeß- und Threadobjekte
Anhalten, Starten, Einzelschritt, Informationsabfrage
- Warteschlangen- und Nachrichtenobjekte
Warteschlangenmanipulation, Nachrichtenobjekt-Information
- Objekte des Überwachungssystems
Zählermanipulation, Uhrmanipulation



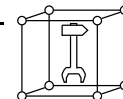
Schnittstellenfunktion (für Werkzeuge und Erweiterungen):

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

Syntax der Ereignis-Aktionsliste:

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

Eine Aktionsliste kann nebenläufig ausgeführt werden; ein Strichpunkt wirkt als globale Synchronisation



Erweiterungen in OMIS v2.0

- OMIS v1.0 zu sehr PVM- und Cluster-spezifisch
 - + neues objektbasiertes Systemmodell (threads, SMPs)
 - + jedes Werkzeug definiert seine Sicht explizit
 - + PVM-spezifische Funktionen in einer Erweiterung
- Explizite Knotenangabe bei Objekten ist oft umständlich
 - + Ortstransparenz eingeführt
- Antwortstring ineffizient
 - + Antworten als Datenstruktur definiert
- Diensteliste noch unvollständig
 - + OMIS v2.0 enthält vollständige Spezifikation (geforderte und optionale Dienste)

Beispiel: Fehlersuche

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

Wenn irgendein Prozeß `pvm_send` aufruft: liefere seine `tid`, seinen Argumentenvektor und die Rücksprungadresse des Aufrufs

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

Wenn einer der spezifizierten Prozesse den Haltepunkt erreicht: halte die gesamte Anwendung an

Beispiel: Leistungsanalyse

```
thread_has_started_lib_call([p_21], "MPI_Send") :  
  pt_integrator_start(pt_i_1) pt_counter_add(pt_c_1, $par5)
```

Wenn Prozeß p_21 einen Sendeaufufruf startet: aktiviere einen integrierenden Zähler und addiere die Nachrichtenlänge (\$par5) auf einen Zähler

```
thread_has_ended_lib_call([p_21], "MPI_Send") : pt_integrator_stop(pt_i_1)
```

Wenn der Prozeß den Aufruf beendet: halte den integrierenden Zähler an

Resultat: Verweilzeit im Sendeaufufruf und gesendete Nachrichtenmenge

OCM: Die Implementierung von OMIS

OCM = OMIS compliant monitoring system

Implementierung von OMIS für PVM-basierte Anwendungen auf Rechnernetzen

Projektziele:

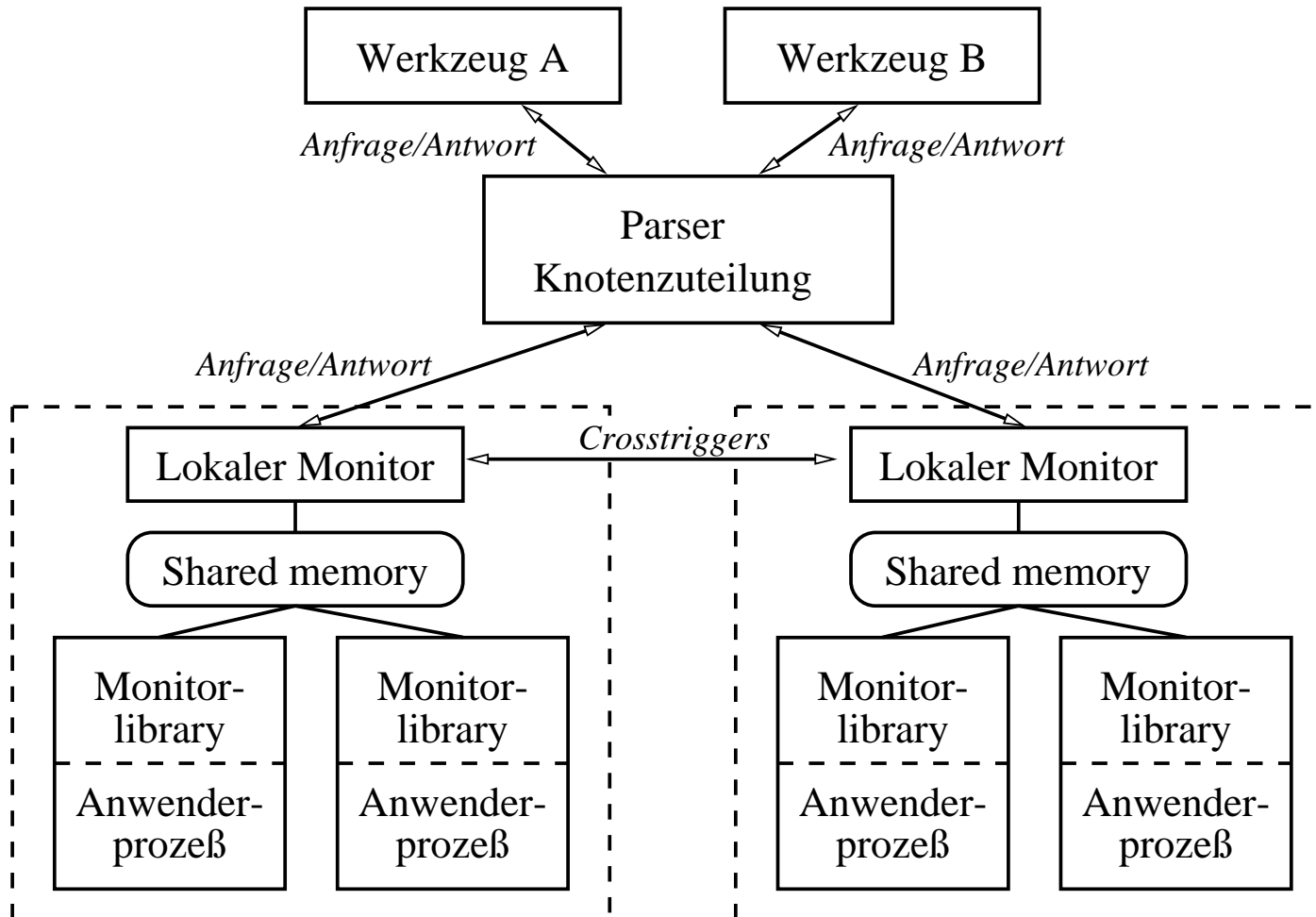
- Verifikation der Anwendbarkeit von OMIS
- Implementierungsplattform für THE TOOL-SET
- Implementierungsplattform für Werkzeuge Dritter
- Referenz für andere OMIS Implementierungen

OCM wird unter GNU Lizenzbedingungen erhältlich sein

Entwurfskonzepte

- Das Überwachungssystem wird aus kommunizierenden Monitoren gebildet, jeweils einem pro Rechnerknoten der virtuellen Maschine
- Globale Aktionsspezifikationen werden zur Definitionszeit an die Knoten verteilt.
- Kommunikation zwischen Monitoren mittels PVM-Mechanismen
- Im ersten Schritt: Zentrales Parsen der Diensteanforderungen
- Ereignisgetriebenes Ausführungsmodell der (Einzelknoten-)Monitore
- Ereigniserkennung und Aktionsausführung sowohl im Monitor als auch in der Anwendung
- Kommunikation zwischen knotenlokalen Monitorkomponenten über gemeinsamen Speicher und Signale

Grobstruktur von OCM



Status des Projekts

OMIS

- Version 1.0 veröffentlicht Februar 1996
- Überarbeitung: Version 2.0 (Juli 1997)

OCM

- Implementierung begonnen Januar 1997
- Erster Prototyp: Juli 1997
- Erste Werkzeuge des TOOL-SET: Oktober 1997

Kooperationen

- ENS Lyon, France (Dosmos)
- KFKI-MSZKI Research Institute, Budapest, Hungary (GRADE)
- GUP, Universität Linz, Österreich (MAD)

Weiterführung

- Abschluß der OCM-Implementierung und Werkzeuganpassung
- Optimierung der OCM-Implementierung
- Integration neuer Werkzeuge: Sicherungspunkterstellung, Lastausgleich, deterministische Programmausführung
- OMIS für Architekturen mit verteiltem gemeinsamem Speicher
- Kooperationen mit anderen Forschungsgruppen

<http://www.bode.informatik.tu-muenchen.de/~omis>