On-line Distributed Debugging on Scalable Multicomputer Architectures*

Roland Wismüller
Technische Universität München
Institut für Informatik

20.04.1994

* Partly funded by the German Science Foundation, Contract SFB342, TP A1

Debugging with TOPSYS

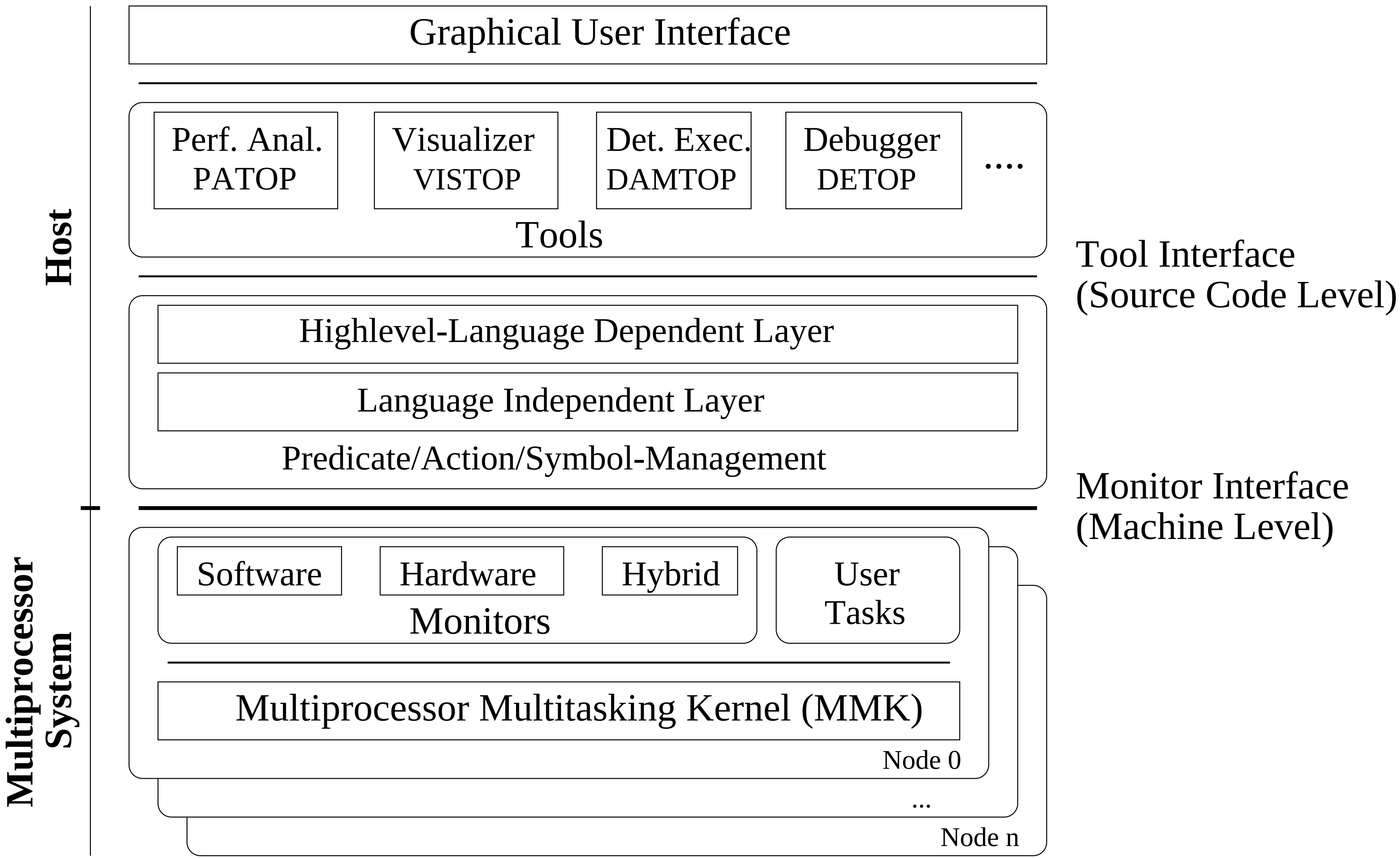
Originally developed for MMK programming model:

- object based: tasks, mailboxes and semaphores
- dynamic: all objects can be created and deleted during runtime
- location transparent: access to objects is independent of their location

Design of TOPSYS:

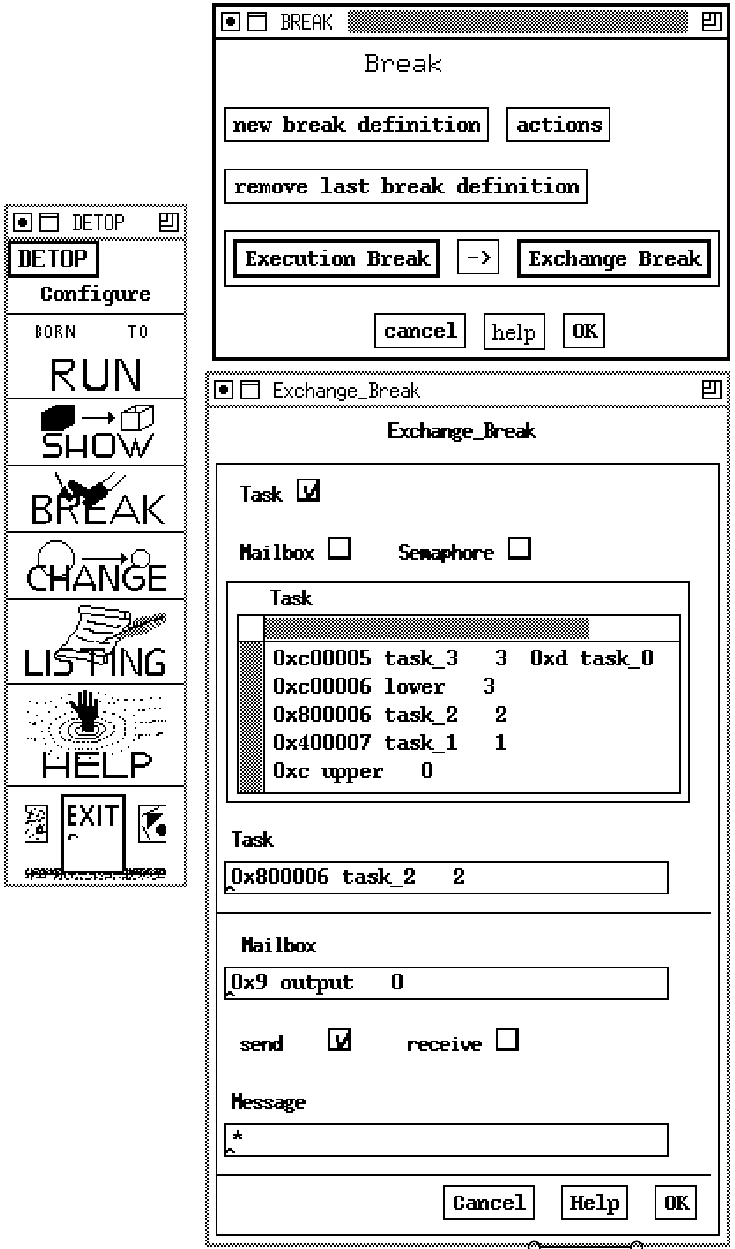
- integrated tool environment
- on-line tools for monitoring and control execution of parallel programs
- provides global view of distributed program
- hierarchical design based on distributed monitoring system

Hierarchical Structure of TOPSYS



DETOP Commands

- execution control
  - ⤳ start and stop tasks, single step
- inspect program state
  - ⤳ existing objects, mapping to nodes
  - ⤳ state and stack backtrace of tasks
  - ⤳ messages and queues of mailboxes
  - ⤳ type and value of variables
- modify program state
- breakpoints and traces
  - ⤳ global breakpoints
  - ⤳ trace of selected simple events
- miscellaneous
  - ⤳ configuration parameters
  - ⤳ source code listings



Breakpoint Events and Actions

- three different event types:
  - execution events: task executes a source line or enters a procedure
  - data events: task accesses a variable
  - communication events:
    - task sends to / receives from a mailbox
    - task requests / releases a semaphore
- possible actions:
  - stop any set of tasks (global breakpoint)
  - start/stop event trace
  - any debugger command (future)

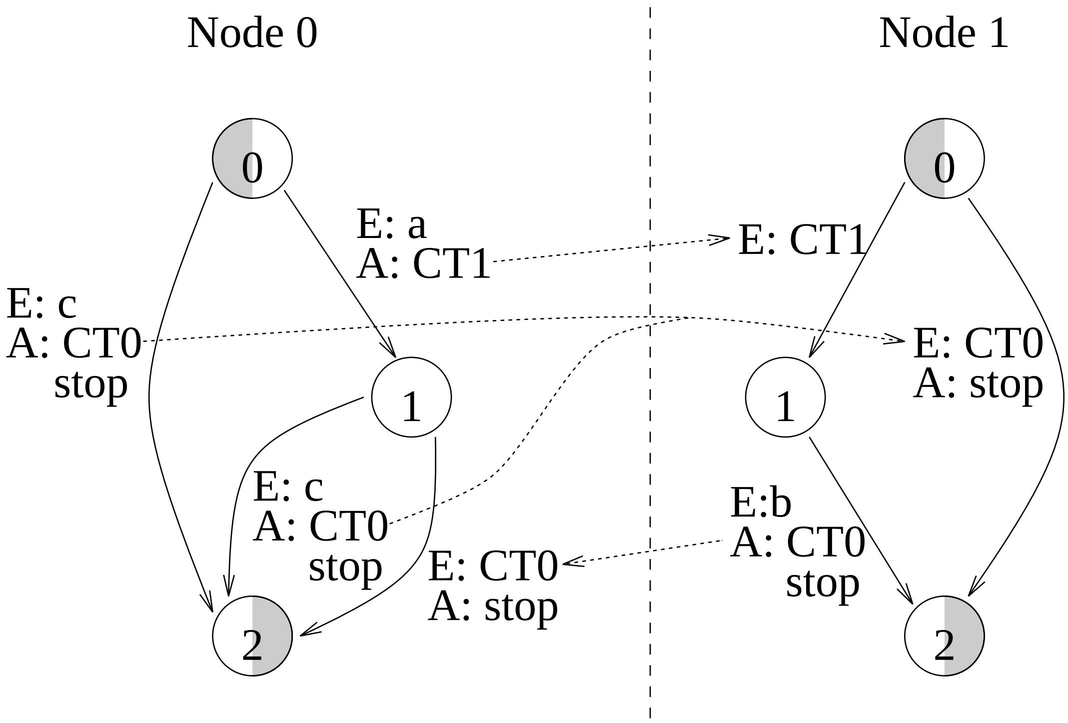
Global Event Detection

- often needed for distributed debugging
  $\rightsquigarrow$ "stop program, if task A receives a message sent by task B at line 123 in foo.c"
- in DETOP global events are combined from simple events by means of different operators (sequence, or, and)
- global event description is translated to communicating DFAs for each monitor
  $\rightsquigarrow$ efficent event detection both in soft- and hardware

Event Translation

Example: $(a \rightarrow b) \vee c$, stop all tasks

1. assign node to each simple event
   $\leadsto$ cache based localizer module
   $\leadsto$ $(a_o \rightarrow b_1) \vee c_o$
2. use crosstriggers to implement nonlocal event sequences and actions
3. build DFAs



Conclusion

The parallel debugger DETOP

- allows for global view of distributed program
- is accomodated to special needs of distributed debugging
- provides full functionality

Future Work

Other target machines and programming models

- PowerXplorer and GC/PowerPlus with PARIX (C and Fortran)
- workstation clusters (PVM)

New graphical (Motif) interface

- based on source code listings
- supports both data and code parallelism