Distributed Systems

Winter Term 2024/25

Roland Wismüller
Universität Siegen
roland.wismueller@uni-siegen.de
Tel.: 0271/740-4050, Büro: H-B 8404

Stand: December 19, 2024

Distributed Systems

Winter Term 2024/25

10 Distributed Shared Memory

10 Distributed Shared Memory ...

Contents

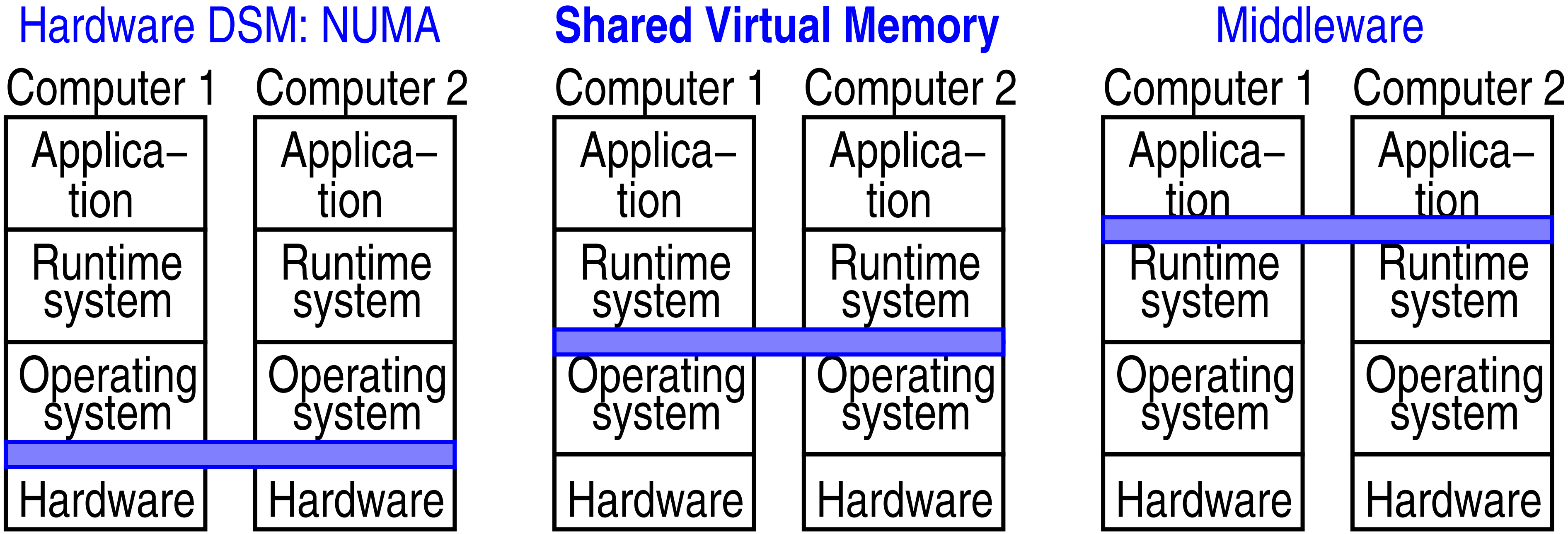
- Introduction
- Design alternatives

Literature

- Colouris, Dollimore, Kindberg: Kap. 16.1-16.3

10 Distributed Shared Memory ...

- Goal: shared memory in distributed systems
- Basic technique considered here:
  - page-based memory management on the nodes
  - on demand: loading pages over the network
  - if necessary replication of pages to increase performance
- Differentiation:



Hardware DSM: NUMA

Computer 1 | Computer 2

Application | Application

Runtime system | Runtime system

Operating system | Operating system

Hardware | Hardware

Shared Virtual Memory

Computer 1 | Computer 2

Application | Application

Runtime system | Runtime system

Operating system | Operating system

Hardware | Hardware

Middleware

Computer 1 | Computer 2

Application | Application

Runtime system | Runtime system

Operating system | Operating system

Hardware | Hardware

Design alternatives

- Structure of the shared memory:
  - byte-oriented (distributed shared memory pages)
  - object-oriented (distributed shared objects)
    - e.g., Orca
  - immutable data (distributed shared container)
    - operations: read, add, remove
    - e.g., Linda Tuple Space, JavaSpaces
- Granularity (for page-based methods):
  - when changing a byte: transmission of entire page
  - with large pages: more efficient communication, less administrative effort, more false sharing

Design alternatives ...

- Consistency model: mostly sequential or release consistency
- Consistency protocol: usually local write protocol
  - i.e., writable memory page migrated to accessing process
  - with or without replication for read accesses
    - client initiated replication, i.e., reader requests copy
  - usually only one writer per page
  - mostly invalidation protocols (with push model)
  - update protocols only if write accesses can be buffered (e.g. with release consistency)

Design alternatives ...

- Management of copies
  - mostly: at any time either multiple readers or one writer
  - each page has an owner
    - writer or one of the readers (last writer)
    - manages a list of processes with copies of the page
  - before write access: process requests current copy
- Finding the owner of a page:
  - central manager
    - manages owners, forwards requests
  - fixed distribution
    - fixed mapping: page → manager

Design alternatives ...

- Finding the owner of a page ...:
  - multicast instead of manager
    - problem: concurrent requests
    - solution: totally ordered multicast, vector time stamps
  - dynamically distributed manager
    - every process knows a likely owner
    - this node forwards the request if necessary
    - the likely owner is updated,
      - when a process transfers the ownership property
      - upon receipt of an invalidation message
      - upon receipt of a requested read-only page
      - when a request is forwarded (to the requestor)

Design alternatives ...

- Problems: e.g., thrashing, especially due to false sharing
  - simple remedy:
    - a page can be migrated again only after a certain period of time
  - TreadMarks: multiple writer protocol
    - release consistency; when released, only the changed parts of the page are transferred
    - changes are then “merged”
    - in case of conflicts: result is non-deterministic