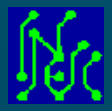
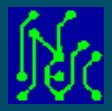
Approaches to High-Performance Computing

João Coelho Garcia
INESC / IST
Distributed Systems Group
joao.c.garcia@inesc.pt



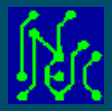
Fundamental Concerns

- What are my system's problems?
  - Time : Computation Duration / Response Time
  - Space: Storage (Memory/Disk) Occupation
- What should I do about it?
  - Optimize the model? YES, very often undervalued!
  - Buy additional (new) hardware (and port model to it)?
  - Move to a distributed system?



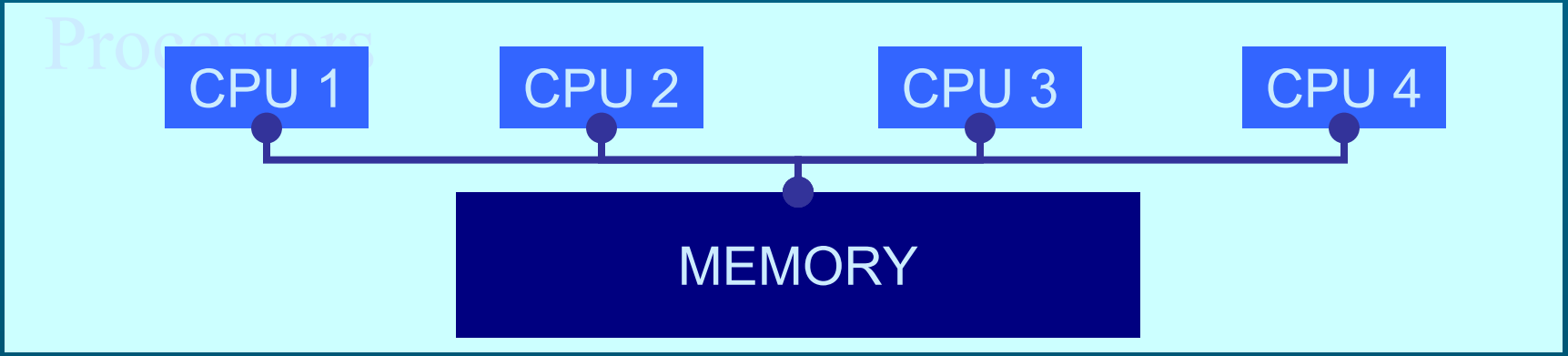
Paradigms

- Standard Approach:
  - Sequential Programming
- Parallel Computers:
  - Multi-threaded Programming
- Distributed Systems:
  - Message Passing
  - Distributed Shared Memory
  - (Modular Decomposition)



Parallel Programming

> Supercomputers are a dying breed:
Parallel Architecture = Simmetric Multi-Processors

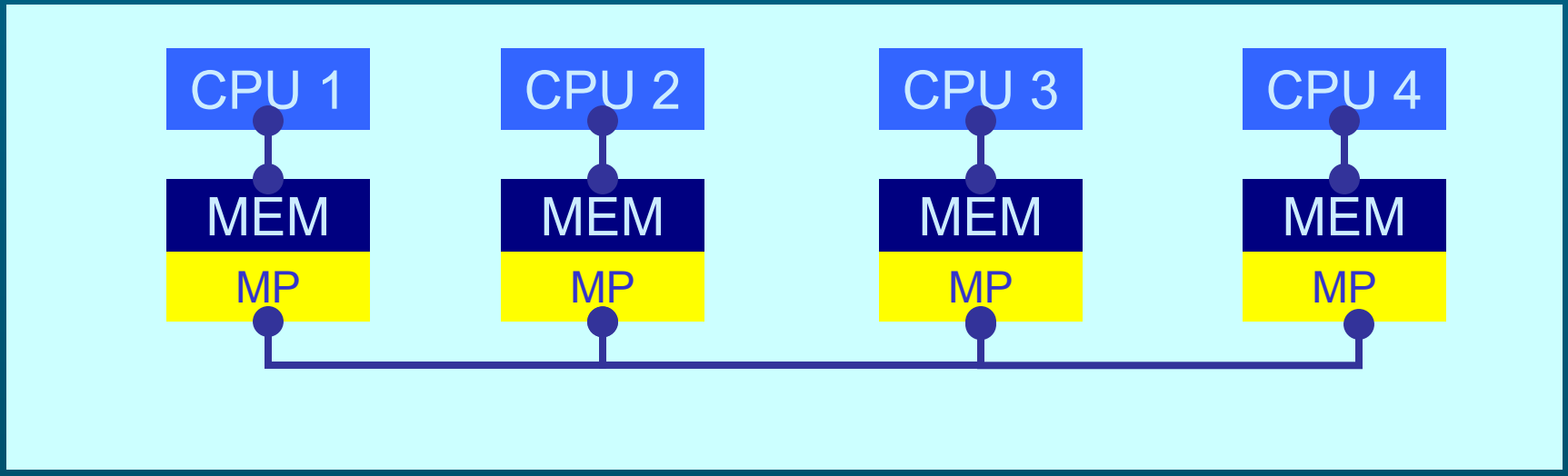


> Hardware abstraction (*almost* perfect)

> Hardware is *somewhat* cheaper than manpower, but... supercomputers DO NOT degrade gracefully!

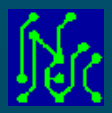
Message Passing

> Data items are exchanged explicitly



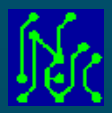
Message Passing

- Explicit Data Transfer/Sharing:
  - send(data_item, to) ⇔ receive(data_item, from)
- Best Known Implementations: MPI, PVM
- Simple, widespread
- Good adaptation to heterogeneity



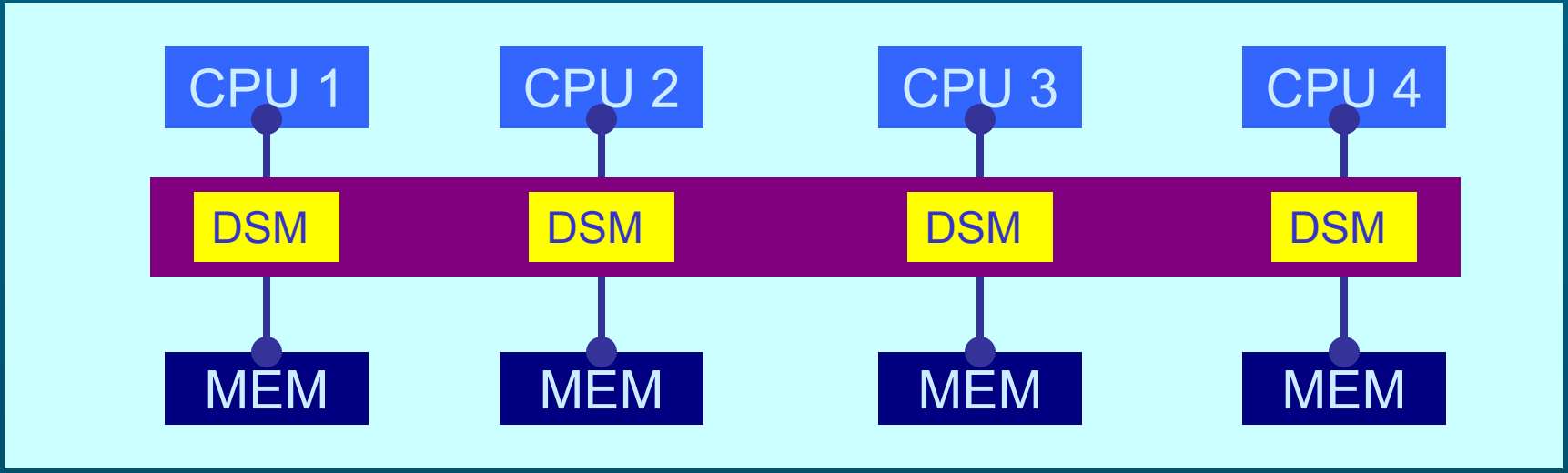
Message Passing: Pros and Cons

- PRO - Total Control of Data Transfer: Possibility of Ideal Performance
- PRO - Natural Synchronization: Data reception is an implicit synchronization point; "wait for data before proceeding…"
- PRO - Better known; more expertise available
- CON – Programming is VERY error prone; embedded description of the network



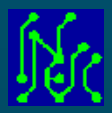
Distributed Shared Memory

> Items in Distributed Shared Memory are shared automatically



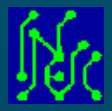
Distributed Shared Memory

- Many machines, but applications see ONE memory: software abstraction
- There is a cable in between. Where´s the magic?
  - "Automatic" Access Detection
- Distribution ⇒ Synchronization
- Great Paradigm ⇒ EASE of Programming
- Implementations: TreadMarks, DiSOM



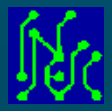
DSM: Pros and Cons

- PRO - Data is always shared
- PRO - Programming is easy
- CON - False sharing
- CON - Communication costs typically high
- Access detection:
  - Hardware: quick but coarse
  - Software: detailed but slow

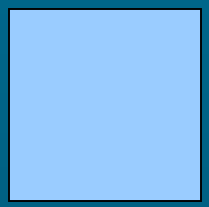


Options & Trade-offs

- SMP vs. MP/DSM:
  - Hardware Costs vs. Porting Costs
  - PC farms are CHEAP!
  - Interconnects are improving fast
- MP vs. DSM - Experience proves:
  - Programming synchronization is easier than communication
  - DSM penalty not necessarily big

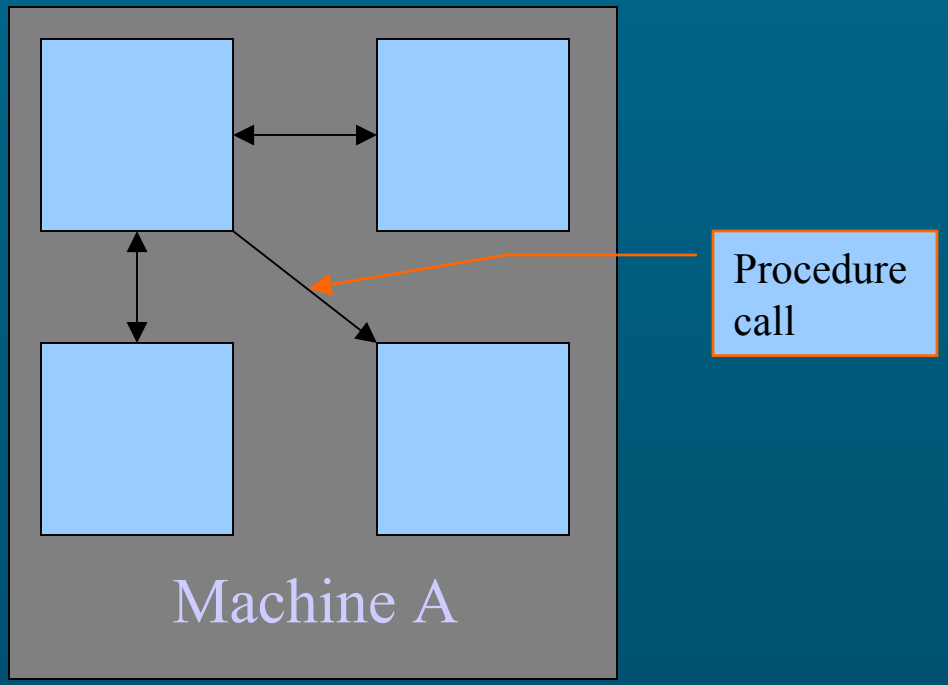


Modular Decomposition

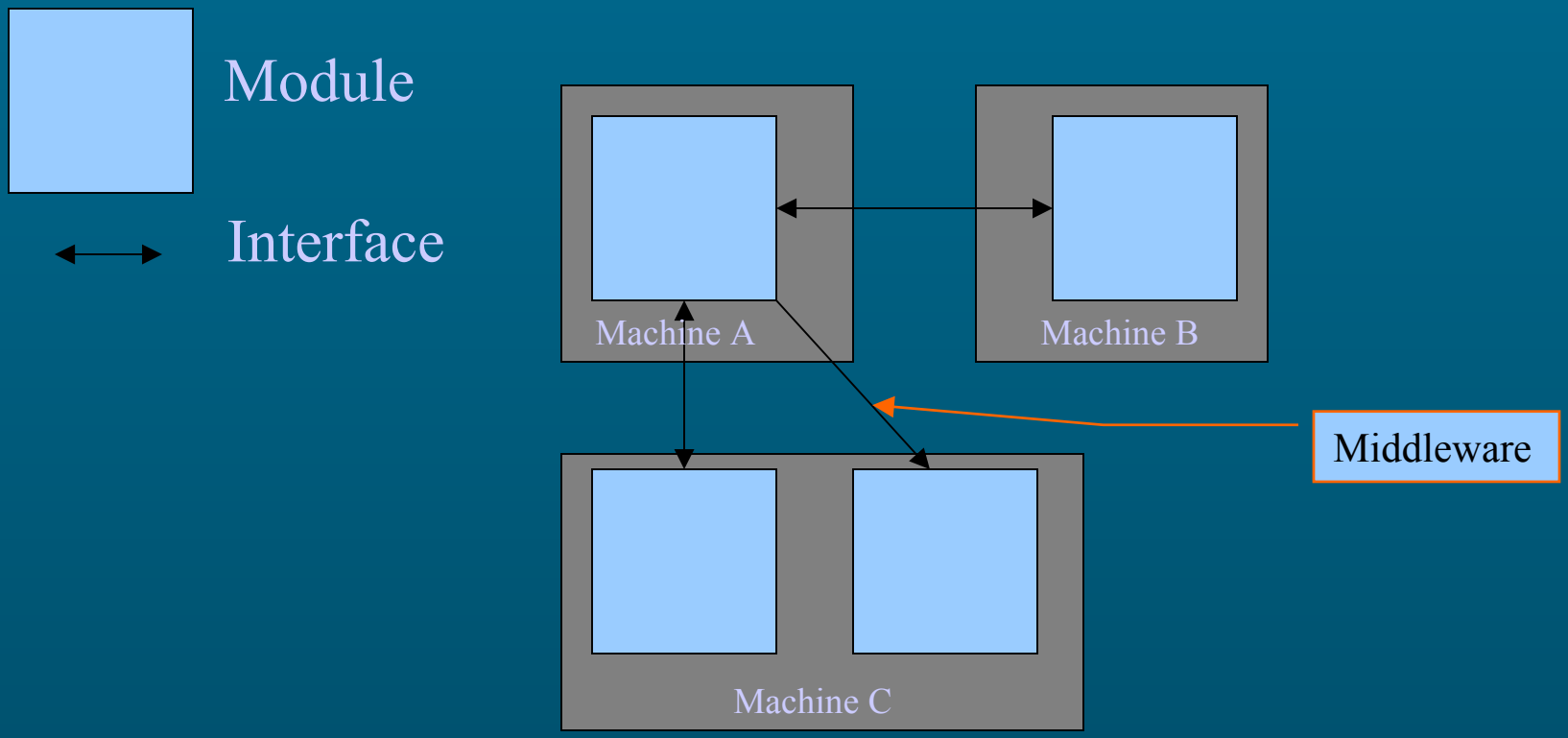


Module

Interface

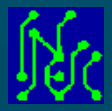


Modular Decomposition



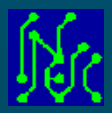
Modular Decomposition

- Useful for:
  - multi-program models: legacy code
  - decomposable models
  - optimisation of resource utilisation (heterogeneous PC/workstation farm)
- Technology:
  - Middleware: DCOM, Java RMI, CORBA



Past & Current Projects at GSD

- Past Projects
  - ESPRIT: PerDiS, **FLASH**
  - **DiSOM**, COMANDOS I & II
- Consulting and Partnership Interests
  - Experience in Scientific (e.g. shipyards) and Commercial Fields (e.g. banks)



Future Research at GSD

- Middleware for wireless netwoks (OBIWAN)
- (Memory Management and Security)
- Expert Support to Modelling/Simulation Projects

http://www.gsd.inesc.pt

or

gsd@mail.gsd.inesc.pt