Hệ thống điều khiển phân tán

Chương 7: Xử lý thời gian thực và xử lý phân tán

13.09.06

Chương 7: Nội dung

7.1 Khái niệm “thời gian thực”

7.2 Hệ điều hành thời gian thực

7.3 Khái niệm “xử lý phân tán”

7.4 Các kiến trúc xử lý phân tán

7.5 Các cơ chế giao tiếp trong hệ phân tán

7.1 Khái niệm thời gian thực

Tại sao cần nghiên cứu về *xử lý thời gian thực*

- *Xử lý thời gian thực* là nguyên lý làm việc cơ bản của mỗi bộ điều khiển, nhìn từ quan điểm *tin học*
- Chất lượng điều khiển và độ tin cậy của hệ thống điều khiển không chỉ phụ thuộc vào *thuật toán điều khiển*, *công nghệ phần cứng*, mà còn phụ thuộc một cách tất yếu vào phương pháp *xử lý thời gian thực*
- Chúng ta còn biết quá ít về cơ chế thực hiện các chức năng bên trong một bộ điều khiển (số)
- Chúng ta cũng còn biết tương đối ít về cơ chế giao tiếp giữa các thành phần mềm trong một hệ thống điều khiển phân tán

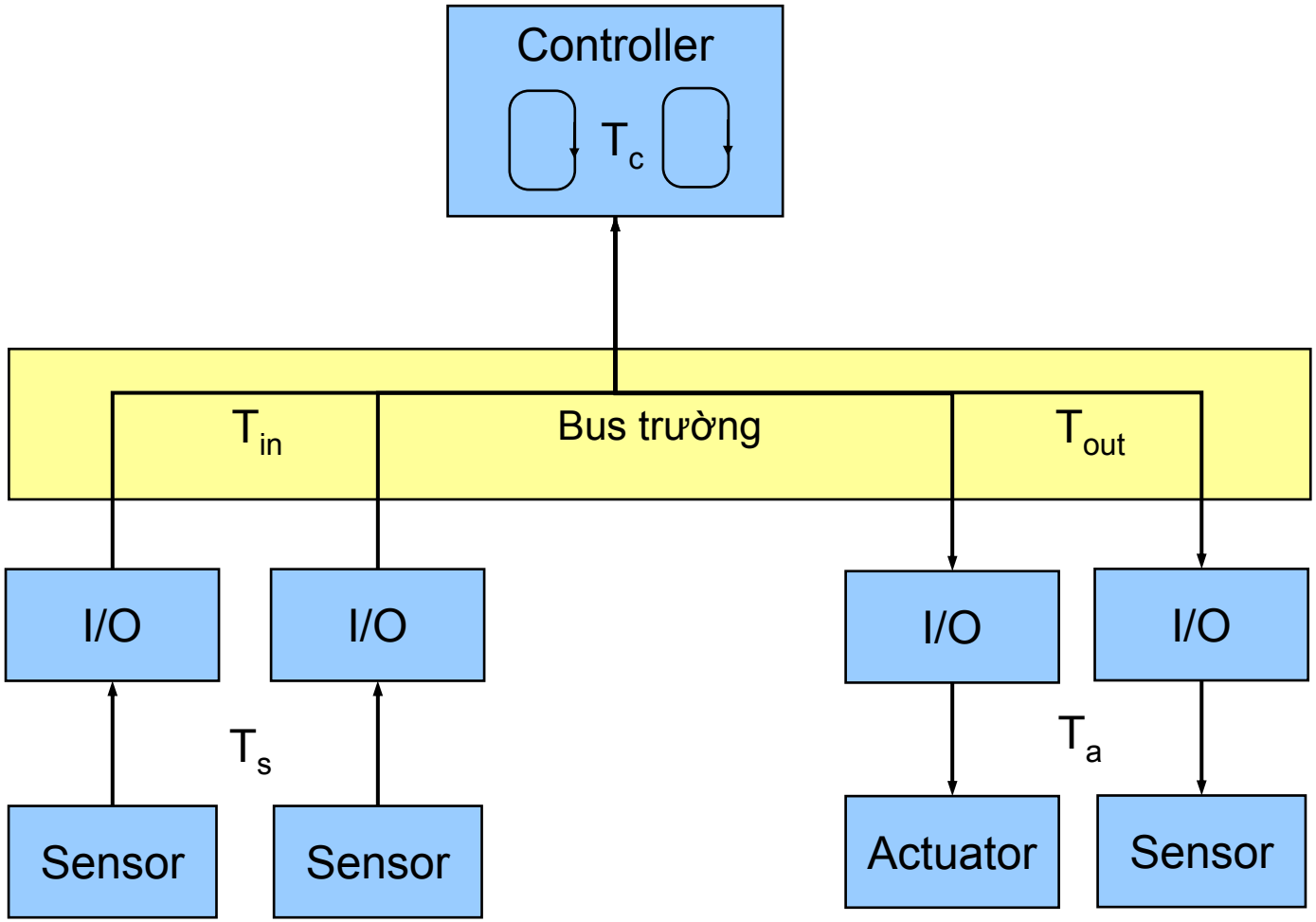
Hệ thời gian thực là gì?

A real-time system is one in which the correctness of the system depends not only on the logical results, but also on the time at which the results are produced.

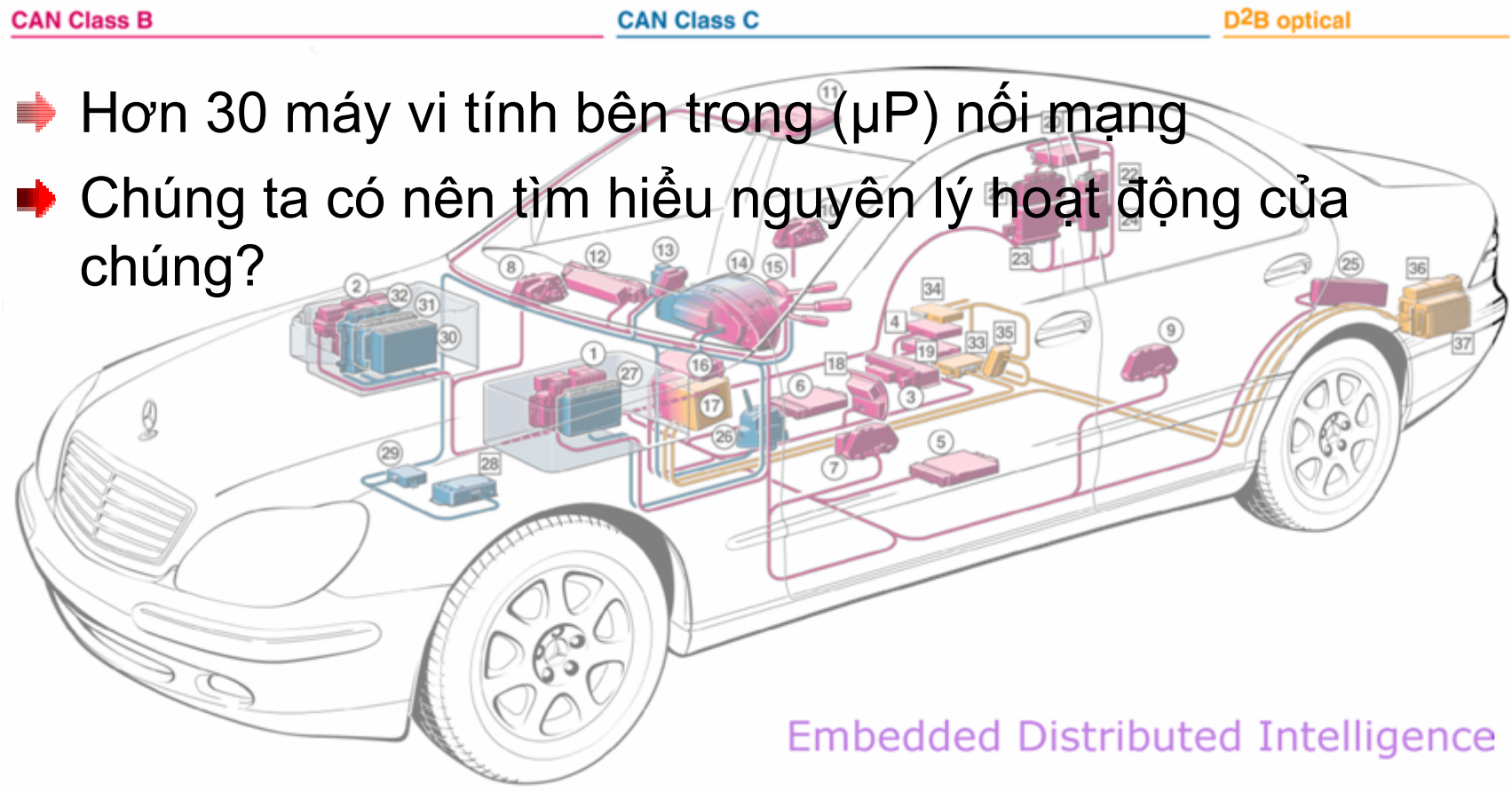
JOHN A. STANKOVIC ET AL.: Strategic Directions in Real-Time and Embedded Systems. ACM Computing Surveys, Vol. 28, No. 4, December 1996

- Mỗi hệ thống điều khiển là một hệ thời gian thực
- Phần lớn các hệ thời gian thực là các hệ thống điều khiển

Vấn đề thời gian trong hệ ĐK qua mạng

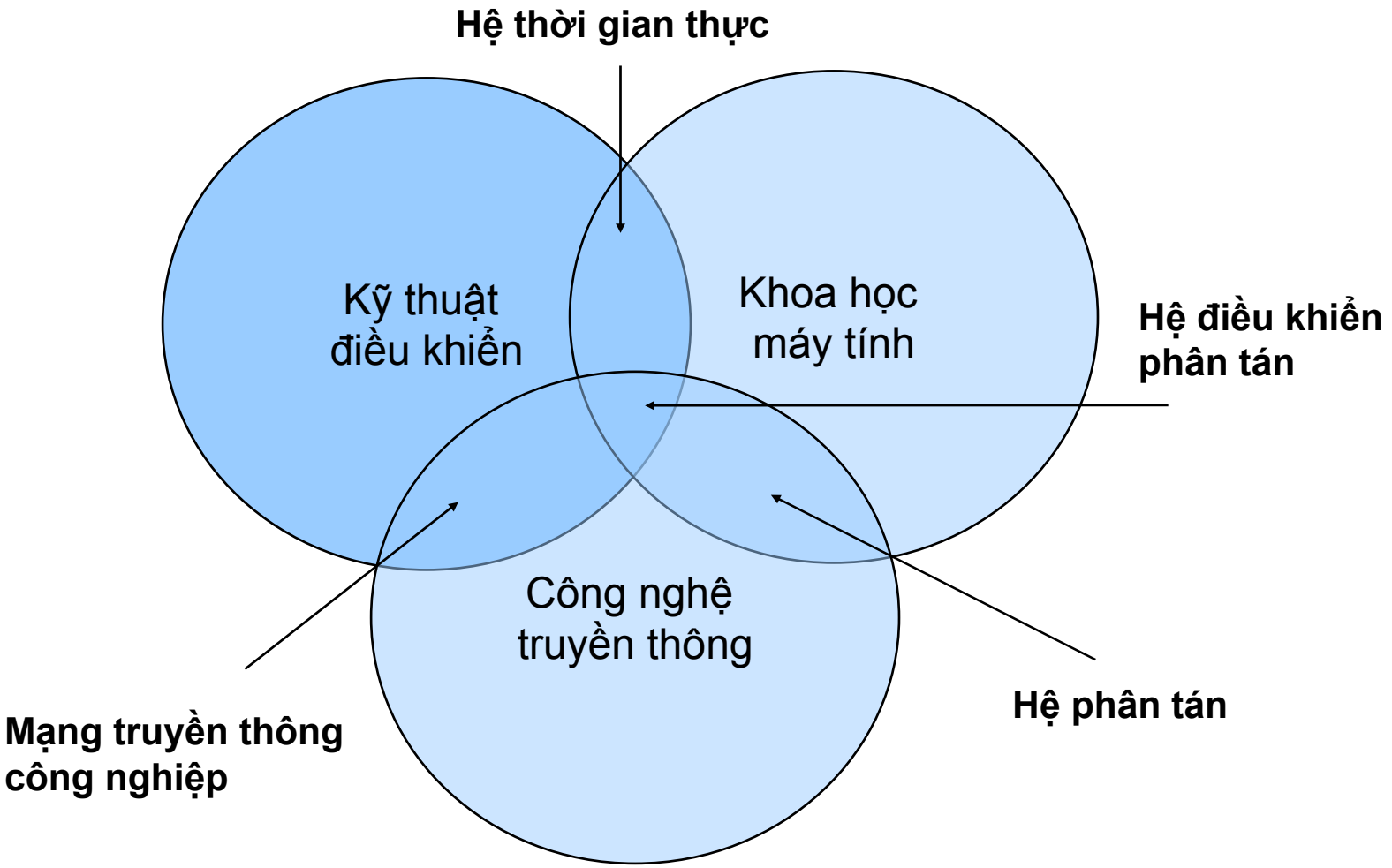


Chiếc xe hơi có là một hệ thời gian thực?



- Hơn 30 máy vi tính bên trong (μP) nối mạng
- Chúng ta có nên tìm hiểu nguyên lý hoạt động của chúng?

Nội dung liên ngành



Một hệ thời gian thực có các đặc điểm:

- *Tính phản ứng*: Hệ thống phải phản ứng với các sự kiện xuất hiện vào các thời điểm không biết trước.
- *Tính nhanh nhạy*: Hệ thống phải xử lý thông tin một cách nhanh chóng để có thể đưa ra kết quả phản ứng một cách kịp thời.
- *Tính đồng thời*: Hệ thống phải có khả năng phản ứng và xử lý đồng thời nhiều sự kiện diễn ra.
- *Tính tiền định*: Dự đoán trước được thời gian phản ứng tiêu biểu, thời gian phản ứng chậm nhất cũng như trình tự đưa ra các phản ứng.

Xử lý thời gian thực là gì?

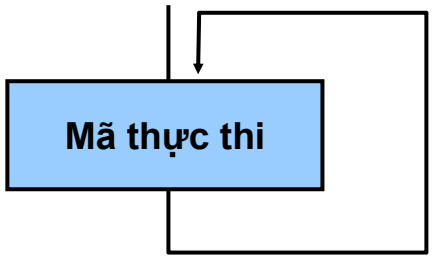
Xử lý thời gian thực là hình thức xử lý thông tin trong một hệ thống để đảm bảo tính năng thời gian thực của nó.

- Luôn liên quan với các *sự kiện bên ngoài* (*tính phản ứng*)
- Yêu cầu cao về *hiệu suất phần mềm* (*tính nhanh nhạy*)
- Đòi hỏi *xử lý đồng thời* nhiều tác vụ (*tính đồng thời*)
- Đòi hỏi cơ sở lý thuyết chặt chẽ phục vụ phân tích và đánh giá (*tính tiền định*)

Khái niệm "tác vụ" (task)

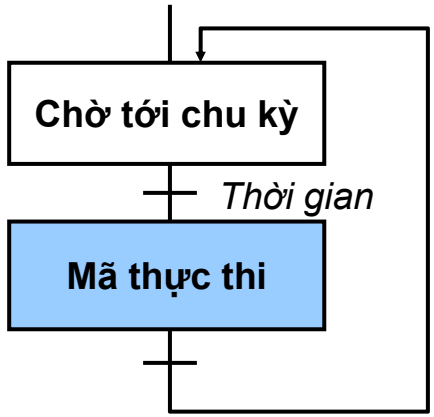
- Một quá trình tính toán cho một nhiệm vụ cụ thể, có thể được thực hiện đồng thời, ví dụ:
  - Các tác vụ xử lý giá trị vào/ra
  - Các tác vụ điều chỉnh
  - Các tác vụ điều khiển logic
  - Các tác vụ xử lý biến cố
  - ...
- Một tác vụ là sự thi hành một chương trình hoặc một phần chương trình
  - Một chương trình chạy nhiều lần => nhiều tác vụ
  - Một đoạn mã chương trình (ví dụ một hàm) được gọi tuần hoàn với các chu kỳ khác nhau => nhiều tác vụ khác nhau
- Multitasking (đa nhiệm): khả năng thi hành đồng thời nhiều tác vụ

Phân loại tác vụ (IEC 61131-3)



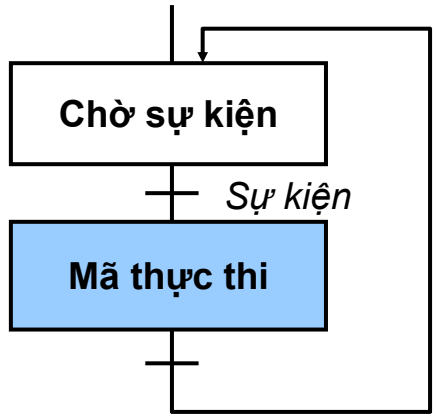
Tác vụ mặc định

Ví dụ:
- Điều khiển logic
- Kiểm tra lỗi



Tác vụ tuần hoàn

Ví dụ:
- Điều chỉnh vòng kín
- Xử lý truyền thông



Tác vụ sự kiện

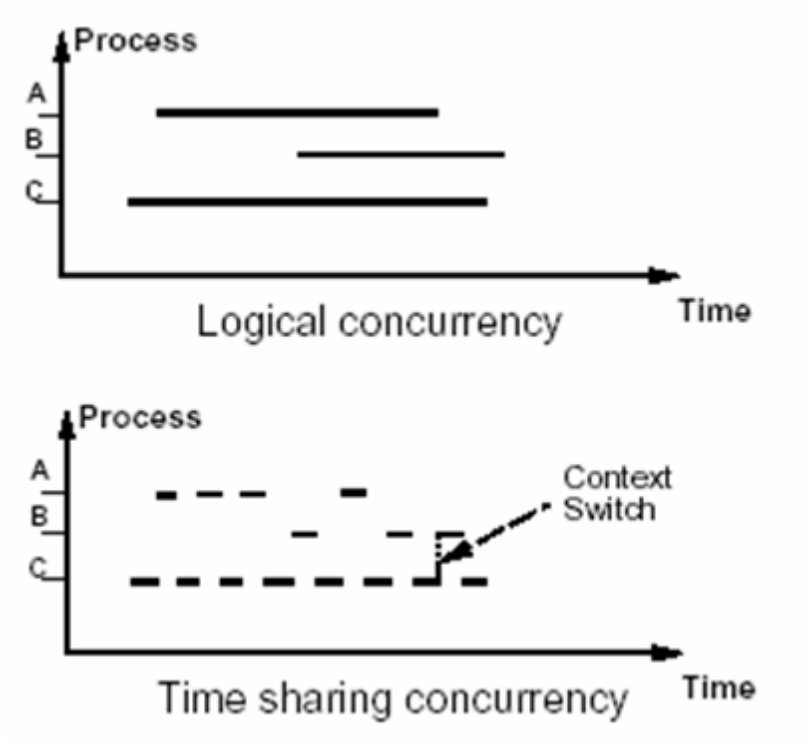
Ví dụ:
- Điều khiển trình tự
- Xử lý sự cố

Các hình thức xử lý đồng thời

- *Xử lý song song*: Các tác vụ (*task*) được phân chia thực hiện song song trên nhiều bộ xử lý
- *Xử lý cạnh tranh*: Nhiều tác vụ chia sẻ thời gian của một bộ xử lý.
- *Xử lý phân tán*: Mỗi (nhóm) tác vụ được thực hiện riêng trên một máy tính (trường hợp đặc biệt của xử lý song song).

- *Xử lý cạnh tranh là hình thức quan trọng nhất trong các hệ thống điều khiển (có thể kết hợp với xử lý phân tán)*

Xử lý cạnh tranh



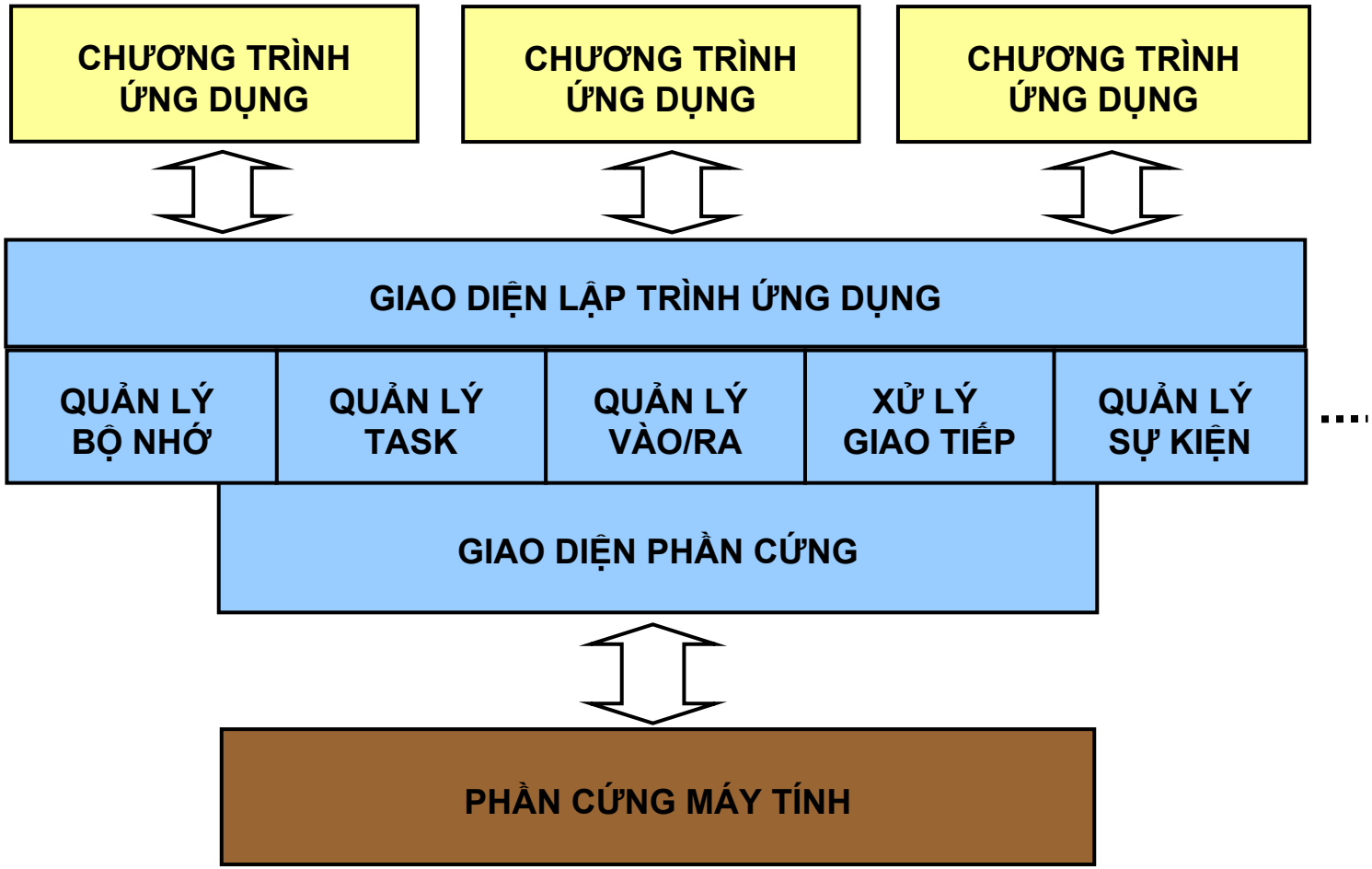
- Các vấn đề:
  - Tổ chức, lập lịch phân chia tài nguyên cho các tác vụ
  - Giao tiếp giữa các tác vụ
  - Đồng bộ hóa giữa các tác vụ

7.2 Hệ điều hành thời gian thực

- Hệ điều hành thời gian thực là một hệ điều hành hỗ trợ các chương trình ứng dụng xử lý thời gian thực
- Hầu hết các bộ điều khiển công nghiệp (PLC, DCS,...) đều hoạt động trên nền một hệ điều hành thời gian thực (RTOS, *Real-time Operating System*)
- Bản thân hệ điều hành thời gian thực cũng là một hệ thời gian thực
- Một hệ điều hành thời gian thực bao giờ cũng là một hệ đa nhiệm (*multitasking*), hỗ trợ xử lý cạnh tranh hoặc/và xử lý song song.

Các nhiệm vụ chính của hệ điều hành thời gian thực trong một bộ điều khiển

- Nạp chương trình, hỗ trợ thử nghiệm, gỡ rối chương trình
- Quản lý dữ liệu vào/ra và quản lý truyền thông
  - Giúp các chương trình ứng dụng dễ dàng truy cập dữ liệu mà không cần quan tâm tới cơ chế phần cứng cụ thể
- Quản lý tác vụ:
  - *Lập lịch*: Phân chia thời gian CPU cho thi hành các tác vụ khác nhau (trong xử lý cạnh tranh)
  - *Hỗ trợ đồng bộ hóa quá trình*: Giúp các tác vụ chia sẻ tài nguyên sử dụng chung (bộ nhớ, cổng vào/ra,..)
  - *Hỗ trợ giao tiếp liên quá trình*: Giúp các tác vụ thực hiện giao tiếp, trao đổi dữ liệu hoặc phối hợp hoạt động
- Các chức năng kiểm tra, chẩn đoán lỗi


CHƯƠNG TRÌNH ỨNG DỤNG
CHƯƠNG TRÌNH ỨNG DỤNG
CHƯƠNG TRÌNH ỨNG DỤNG
GIAO DIỆN LẬP TRÌNH ỨNG DỤNG
QUẢN LÝ BỘ NHỚ
QUẢN LÝ TASK
QUẢN LÝ VÀO/RA
XỬ LÝ GIAO TIẾP
QUẢN LÝ SỰ KIỆN
....
GIAO DIỆN PHẦN CỨNG
PHẦN CỨNG MÁY TÍNH

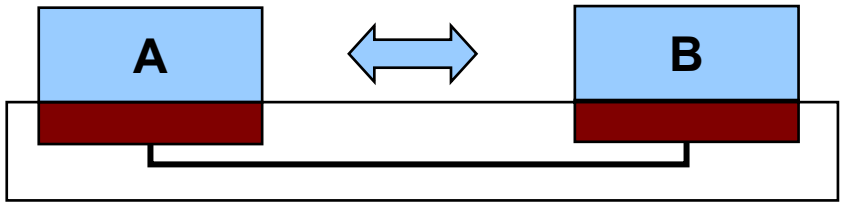
Phương pháp lập lịch

- Cơ chế lập lịch
  - Lập lệnh tĩnh: thứ tự thực hiện các tác vụ được xác định trước khi hệ thống đi vào hoạt động.
  - Lập lệnh động: thứ tự thực hiện các tác vụ được xác định trong khi hệ thống đang hoạt động.
- Sách lược lập lịch
  - FIFO: đến trước sẽ được thực hiện trước.
  - Mức ưu tiên cố định/động: các tác vụ được đặt các mức ưu tiên cố định hoặc có thể thay đổi nếu cần.
  - Preemptive: chen hàng, chọn một tác vụ để thực hiện trước các tác vụ khác.
  - Non-preemptive: không chen hàng, các tác vụ được thực hiện bình thường dựa trên mức ưu tiên của chúng.
- Thuật toán lập lịch
  - Rate monotonic: càng thường xuyên càng được ưu tiên.
  - Deadline monotonic: càng gấp càng được ưu tiên.
  - Least laxity: tỷ lệ thời gian tính toán/thời hạn cuối cùng (deadline) càng lớn càng được ưu tiên.

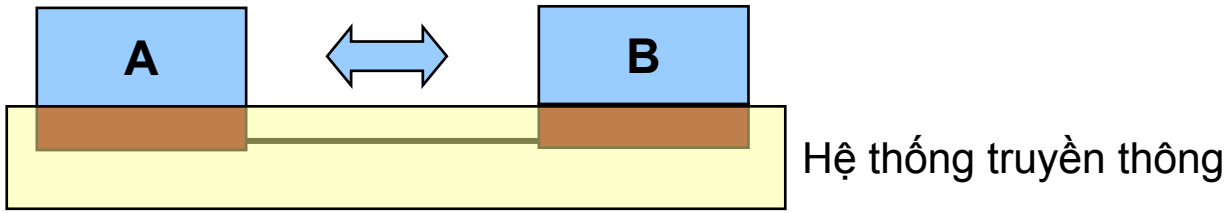
7.3 Khái niệm xử lý phân tán

- *Xử lý phân tán* là hình thức xử lý thông tin tất yếu của các hệ thống phân tán nói chung và các hệ thống điều khiển phân tán nói riêng
- Xử lý phân tán giúp nâng cao năng lực xử lý thông tin của một hệ thống, góp phần cải thiện tính năng thời gian thực, nâng cao độ tin cậy và tính linh hoạt của hệ thống.
- Phân biệt các khái niệm:
  - Xử lý cục bộ => ứng dụng đơn độc
  - Xử lý cạnh tranh => ứng dụng đa nhiệm
  - Xử lý tập trung => ứng dụng tập trung
  - Xử lý nối mạng => ứng dụng mạng (giao tiếp hiện)
  - Xử lý phân tán => ứng dụng phân tán (giao tiếp ngầm)

Giao tiếp ngầm ⇔ Giao tiếp hiện



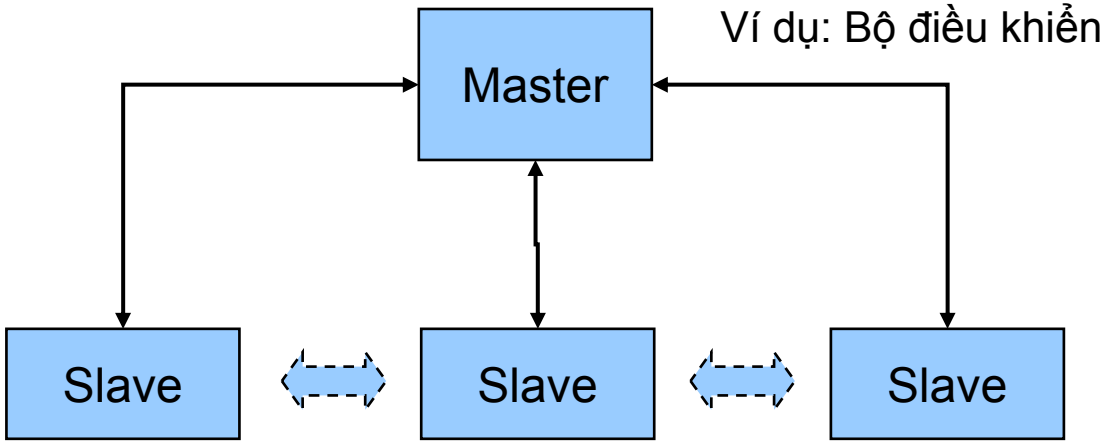
- Giao tiếp hiện (*explicit communication*):
  - Hoạt động giao tiếp được coi là chức năng riêng
  - Sử dụng dịch vụ giao tiếp (ví dụ lập trình) cần biết rõ về hệ thống truyền thông (kiến trúc giao thức)



- Giao tiếp ngầm (im*plicit communication*):
  - Hoạt động giao tiếp được thực hiện ngầm khi cần thiết
  - Sử dụng dịch vụ giao tiếp (ví dụ lập trình) cần biết rõ về hệ thống truyền thông (kiến trúc giao thức)

7.4 Các kiến trúc xử lý phân tán

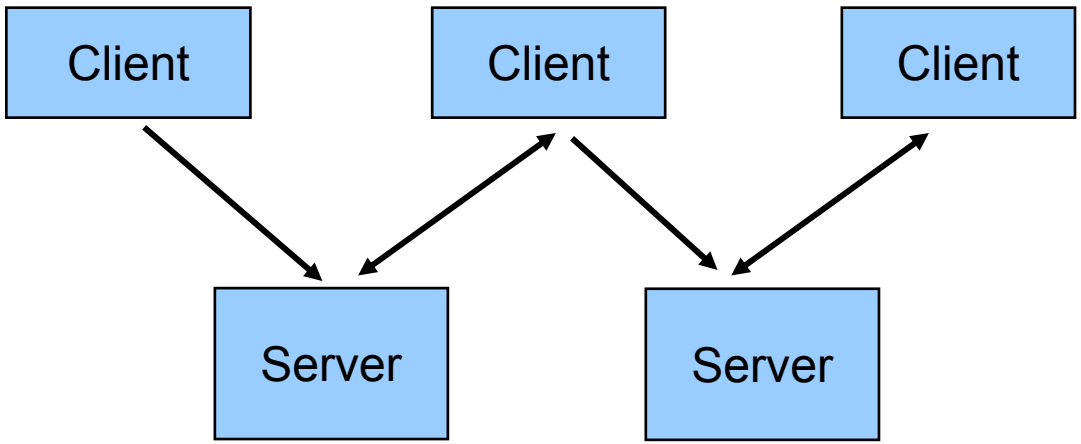
- Kiến trúc Master/Slave
  - Chức năng xử lý thông tin được phân chia trên nhiều trạm tớ
  - Một trạm chủ phối hợp hoạt động của nhiều trạm tớ
  - Các trạm tớ có vai trò, nhiệm vụ tương tự như nhau
  - Các trạm tớ có thể giao tiếp trực tiếp, hoặc không



Ví dụ: Các vào/ra phân tán, các thiết bị trường

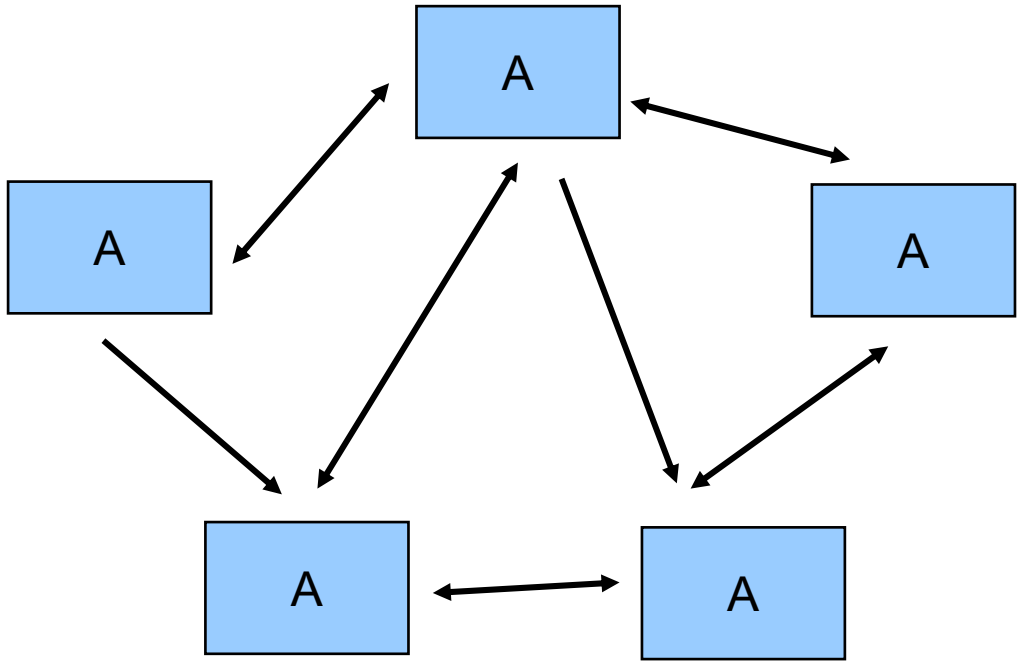
- Kiến trúc Client/Server
  - Chức năng xử lý thông tin được phân chia thành hai phần khác nhau, phần sử dụng chung cho nhiều bài toán được thực hiện trên các server, phần riêng thực hiện trên từng client.
  - Giữa các client không cần thiết có giao tiếp trực tiếp
  - Vai trò chủ động trong giao tiếp thuộc về client

Ví dụ: Các trạm vận hành



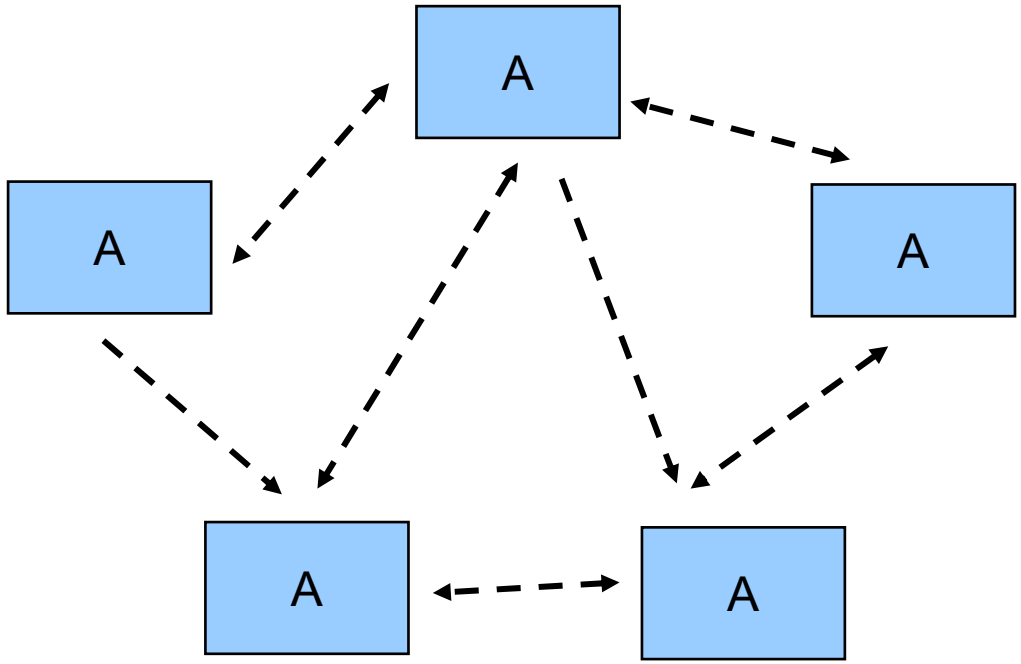
Ví dụ: Các bộ điều khiển hoặc các trạm quản lý dữ liệu

- Kiến trúc bình đẳng
  - Các trạm có vai trò bình đẳng, phải phối hợp hoạt động, hình thức giao tiếp trực tiếp với nhau không qua trung gian



Ví dụ: Các trạm điều khiển phân tán (kiến trúc PLC/DCS) hoặc các thiết bị trường thông minh (kiến trúc FCS)

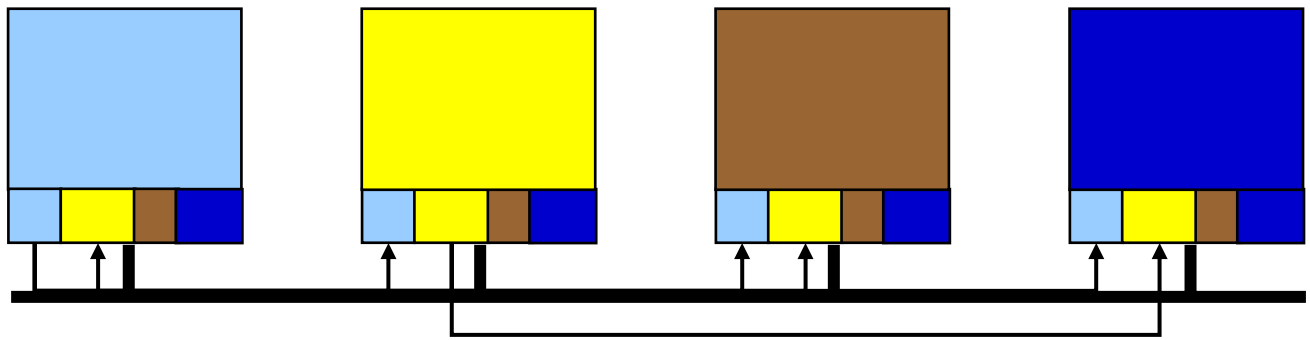
- Kiến trúc tự trị
  - Các trạm có vai trò bình đẳng, có thể hoạt động hoàn toàn độc lập nhưng sự phối hợp hoạt động tạo hiệu quả cao nhất



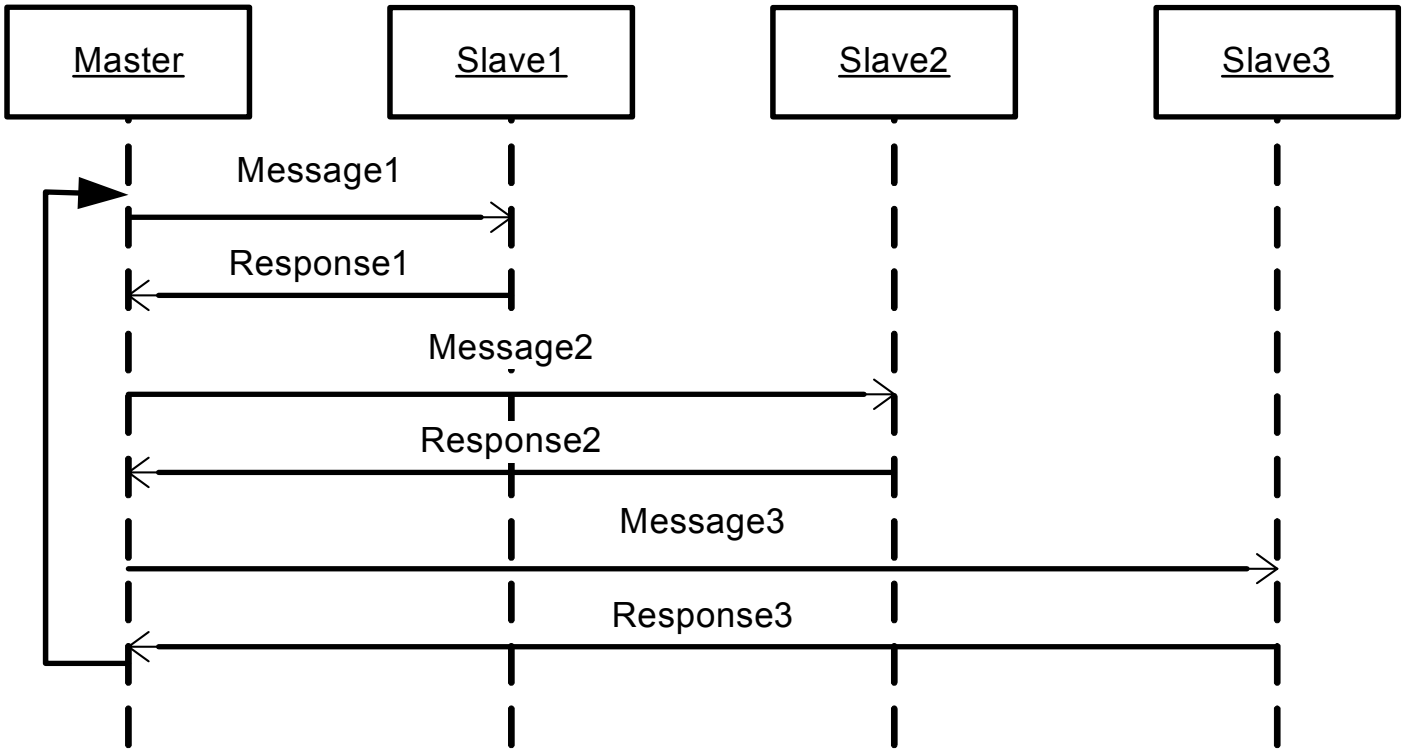
Ví dụ: Các hệ thống xây dựng theo công nghệ Agent, Multi-Agent

7.5 Các cơ chế giao tiếp trong hệ ĐKPT

- Dữ liệu toàn cục (Global Data)
  - Giống như một vùng nhớ chung
  - Mỗi trạm đều chứa một ảnh của bảng dữ liệu toàn cục, trong đó có toàn bộ dữ liệu cần trao đổi của tất cả các trạm khác
  - Mỗi trạm gửi phần dữ liệu của nó tới tất cả các trạm, mỗi trạm tự cập nhật ảnh của bảng dữ liệu toàn cục
  - Đơn giản, tiền định nhưng kém hiệu quả
  - Áp dụng cho lượng dữ liệu nhỏ, tuần hoàn, thích hợp trong kiến trúc bình đẳng (ví dụ giữa các trạm điều khiển).



- Hỏi tuần tự (Polling, Scanning)
  - Một trạm đóng vai trò Master
  - Cơ chế hỏi/đáp tuần tự theo trình tự đặt trước
  - Đơn giản, tiền định, hiệu quả cao
  - Áp dụng cho trao đổi dữ liệu tuần hoàn, thích hợp trong kiến trúc Master/Slave



- Tay đôi (Peer-To-Peer)
  - Hình thức có liên kết hoặc không liên kết, cấu hình trước hoặc không cấu hình trước, có xác nhận hoặc không xác nhận, có yêu cầu hoặc không có yêu cầu
  - Linh hoạt nhưng thủ tục có thể phức tạp
  - Áp dụng cho trao đổi dữ liệu tuần hoàn hoặc không tuần hoàn, thích hợp cho tất cả các kiến trúc khác nhau.
- Chào/đặt hàng (Subscriber/Publisher)
  - Nội dung thông báo được một trạm chủ chào và các trạm client đặt theo cơ chế tuần hoàn hoặc theo sự kiện
  - Thông báo chỉ được gửi tới các trạm đặt (có thể gửi riêng hoặc gửi đồng loạt)
  - Linh hoạt, tiền định, hiệu suất cao
  - Áp dụng cho trao đổi dữ liệu tuần hoàn hoặc không tuần hoàn, thích hợp cho kiến trúc Client/Server hoặc kiến trúc bình đẳng.

- Hộp thư (Mailbox)
  - Các trạm sử dụng một môi trường trung gian như files, một cơ sở dữ liệu hoặc một chương trình server khác để ghi và đọc dữ liệu
  - Mỗi bức thư mang dữ liệu và mã căn cước (nội dung thư hoặc/và người nhận)
  - Gửi và nhận thư có thể diễn ra tại bất cứ thời điểm nào
  - Linh hoạt nhưng kém hiệu quả, không đảm bảo tính năng thời gian thực
  - Áp dụng cho trao đổi dữ liệu có tính chất ít quan trọng, thích hợp cho kiến trúc Client/Server hoặc kiến trúc tự trị.