TRƯỜNG ĐẠI HỌC
BÁCH KHOA HÀ NỘI

TRUNG TÂM
TÍNH TOÁN HIỆU NĂNG CAO

NGHIÊN CỨU

CÁC HỆ THỐNG TÍNH TOÁN HIỆU NĂNG CAO VÀ ỨNG DỤNG MÔ PHỎNG VẬT LIỆU VI MÔ

Trong Khuôn Khổ Hợp Tác Khoa Học Công Nghệ
Theo Nghị Định Thư Với Ấn Độ Giai Đoạn 2004 – 2005

Chủ Nhiệm Đề Tài: PGS. TS. Nguyễn Thanh Thuỷ

Đề tài nhánh

PHẦN MỀM MÁY CHỦ TÍNH TOÁN

(Tài liệu kỹ thuật: Kiến trúc hệ thống)

(2/4)

Hà Nội, 3-2006

Lời Nói Đầu

Tài liệu này được thực hiện sau một thời gian làm việc về Tính toán hiệu năng cao của nhóm nghiên cứu High Performance Computing tại Trung tâm Tính Toán Hiệu Năng Cao trường Đại Học Bách Khoa Hà Nội.

Trong tài liệu này, chúng tôi xin trình bày, mô tả kiến trúc của **BKluster** – một hệ thống tính toán song song ghép cụm dựa trên kiến trúc hệ Beowulf và mô hình lập trình truyền thông điệp cùng với bộ phần mềm **BKlusware** - một tập các phần mềm hỗ trợ tối đa nhiều người dùng sử dụng hệ thống BKluster ở nhiều mức độ khác nhau. Nội dung của tài liệu được chia thành 5 chương:

Chương 1: Giới thiệu về mục đích và các hướng tiếp cận của nhóm nghiên cứu khi xây dựng hệ thống.

Chương 2: trình bày về kiến trúc của hệ thống dưới nhiều góc nhìn khác nhau: góc nhìn tổng thể, góc nhìn theo hướng phân tầng, góc nhìn theo hướng phân hệ.

Các chương 3, 4, 5 trình bày về kiến trúc của 3 hệ thống con tạo thành BKluster: Hệ thống phát triển tích hợp, Hệ thống Thực thi tính toán, Hệ thống giám sát và quản trị.

Mục Lục

Lời Nói Đầu 1
Mục Lục 2
Danh mục hình vẽ 3
CHƯƠNG 1 GIỚI THIỆU HỆ THỐNG 4
CHƯƠNG 2 KIẾN TRÚC TỔNG THỂ 7
2.1 Kiến trúc phân tầng 9
2.2 Kiến trúc phân hệ 12
2.3 Kiến trúc vật lý 12
2.3.1 Thiết lập cấu hình phần mềm trên hệ thống 14
2.3.2 Thiết lập hệ thống lưu trữ 15
CHƯƠNG 3 HỆ THỐNG PHÁT TRIỂN TÍCH HỢP 17
3.1 Module phát triển chương trình 17
3.2 Module hỗ trợ gỡ rối chương trình – BKPD 20
3.2.1 Thành phần Bộ điều khiển cục bộ 21
3.2.2 Thành phần Bộ điều khiển toàn cục 22
3.2.3 Bộ giao tiếp với client 24
3.2.4 Thành phần Server Daemon 25
3.2.5 Giao thức 26
CHƯƠNG 4 HỆ THỐNG THỰC THI TÍNH TOÁN 32
CHƯƠNG 5 HỆ THỐNG GIÁM SÁT VÀ QUẢN TRỊ 35
5.1 Module quản lý các gói phần mềm 36
5.2 Module quản lý cấu hình cluster 37
5.3 Module quản lý người dùng 38
5.4 Module giám sát hoạt động hệ thống 39
5.5 Module đánh giá hiệu năng hệ thống 40
5.5.1 Thành phần đánh giá năng lực tính toán của hệ thống 40
5.5.2 Thành phần đánh giá năng lực truyền thông trong hệ thống 42
Tài liệu tham khảo 44

Danh mục hình vẽ

Hình 2-1 Kiến trúc tổng quan hệ thống BKluster 8
Hình 2-2 Kiến trúc vật lý tổng quan hệ thống BKluster 8
Hình 2-3 Kiến trúc phân tầng hệ thống BKluster 9
Hình 2-4 Sự phân bố các thành phần chính của BKluster trong các tầng 11
Hình 2-5 Kiến trúc phân hệ BKluster 12
Hình 2-6 Kiến trúc vật lý hệ thống BKluster 13
Hình 2-7 Các dịch vụ cài đặt trên máy chủ tính toán 14
Hình 2-8 Các dịch vụ cài đặt trên nút tính toán 15
Hình 2-9 Hệ thống lưu trữ trong BKluster 16
Hình 3-1 Hệ thống phát triển tích hợp 18
Hình 3-2 Kiến trúc module phát triển chương trình 19
Hình 3-3 Kiến trúc module hỗ trợ gỡ rối chương trình 20
Hình 3-4 Bộ điều khiển cục bộ trong module Gỡ rối chương trình 22
Hình 3-5 Bộ điều khiển toàn cục trong module gỡ rối chương trình 23
Hình 3-6 Thành phần Server Daemon trong module gỡ rối chương trình 25
Hình 4-1 Hệ thống thực thi tính toán 33
Hình 4-2 Công cụ đệ trình công việc 34
Hình 5-1 Hệ thống giám sát và quản trị 36
Hình 5-2 Kiến trúc module quản lý gói phần mềm 37
Hình 5-3 Kiến trúc module quản lý cluster 38
Hình 5-4 Kiến trúc module quản lý người dùng 39
Hình 5-5 Kiến trúc module giám sát hệ thống 40
Hình 5-6 Kiến trúc thành phần đánh giá hiệu năng tính toán hệ thống 41
Hình 5-7 Thành phần đánh giá năng lực truyền thông hệ thống 43

CHƯƠNG 1
GIỚI THIỆU HỆ THỐNG

Nhu cầu về các hệ thống tính toán lớn đã tạo ra những hướng nghiên cứu rất mạnh trên nhiều lĩnh vực kiến trúc máy tính, phần cứng và phần mềm. Trong thập niên cuối cùng của thế kỷ trước, hướng tiếp cận ghép các hệ thống nhỏ thành một hệ thống lớn hơn được xem là khả thi, đặc biệt phù hợp với điều kiện kinh tế, công nghệ của các nước đang phát triển.

Với mong muốn đáp ứng các nhu cầu tính toán của các phòng thí nghiệm các viện nghiên cứu trong và ngoài trường Bách Khoa, chúng tôi xây dựng một ***hệ thống tính toán song song ghép cụm dựa trên kiến trúc hệ Beowulf và mô hình lập trình truyền thông điệp - hệ thống BKluster.***

Với mong muốn giúp người dùng tiếp cận, khai thác một cách hiệu quả BKluster, chúng tôi xây dựng một tập các phần mềm hỗ trợ tối đa nhiều người dùng ở nhiều mức độ khác nhau. Bộ phần mềm này là BKlusware. Nói cách khác

BKluster = một cluster theo mô hình Beowulf + BKlusware

Xét từ phương diện một hệ thống tính toán, trước hết BKluster cho phép người dùng thực hiện một tác vụ tính toán. Hơn thế, BKluster còn phải là một hệ đa nhiệm, đa phiên làm việc, nghĩa là cho phép nhiều người cùng khai thác hệ thống cùng một lúc một cách độc lập. Ngoài ra, chúng tôi cũng đề xuất ra các công cụ hỗ trợ việc phát triển một chương trình tính toán được thực thi trên BKluster. Các công cụ phát triển hỗ trợ người dùng ở nhiều mức từ người dùng thuần túy qua một ngôn ngữ mô tả đơn giản PCS cho đến những người lập trình chuyên nghiệp có hiểu biết về lập trình nói chung và lập trình theo mô hình truyền thông điệp nói chung BKluster cũng có các công cụ hỗ trợ người quản trị. Thông qua bộ công cụ quản trị và giám sát hệ thống, người quản trị có khả năng nắm bắt toàn bộ thông

tin của hệ thống. Hơn nữa, các thông tin này được hiện thị một cách trực quan giúp người quản lý có những chiến lược cụ thể nhằm khai thác hệ thống một cách tối đa.

Với phương châm tận dụng tối đa các nguồn tài nguyên sẵn có, chúng tôi xây dựng BKlusware theo mô hình phát triển các bộ phần mềm mã nguồn mở. Trong các phần việc, chúng tôi cố gắng tìm kiếm một gói phần mềm mã nguồn mở khả dĩ, phù hợp với yêu cầu bài toán. Dựa trên phần mềm đó, chúng tôi tỉnh chỉnh và bổ sung các module cần thiết để có thể tích hợp phần mềm sẵn có vào hệ thống một cách trơn tru. Đi theo hướng này, có thể kể đến gói phần mềm PBS trong việc xây dựng một bộ quản lý tác vụ của BKluster, GDB trong bộ công cụ hỗ trợ tìm lỗi chương trình BKDB hay Ganglia trong phần việc giám sát hệ thống. Ngoài việc tinh chỉnh và bổ sung cho những phần mềm sẵn có để được một phần mềm hoàn thiện phù hợp với yêu cầu cụ thể, chúng tôi cũng xây dựng khá nhiều gói phần mềm khác như gói phần mềm quản lý các gói phần mềm trên một cluster, bộ dịch cho ngôn ngữ PCS – một ngôn ngữ mô tả đơn giản cho phép mô tả các thao tác cơ bản của đại số tuyến tính.

Trong các phần tiếp sau, chúng tôi sẽ mô tả về hệ thống theo trình tự sau:

- Chương 2 trình bày kiến trúc tổng thể của hệ thống. Chúng tôi đề xuất kiến trúc này với mục đích BKluster là một hệ thống tính toán hoàn thiện với đầy đủ 3 nhóm chức năng chính: tính toán, xây dựng chương trình tính toán và quản trị, giám sát hệ thống. Ba chức năng này sẽ tương ứng với 3 hệ thống con, bao gồm: hệ thống thực thi chương trình, hệ thống trợ giúp phát triển chương trình và hệ thống quản trị giám sát hệ thống. Ngoài tiêu chí đầy đủ chức năng, kiến trúc được đề xuất phải đảm bảo một sự linh hoạt trong việc mở rộng cũng như phát triển tiếp sau. Sự linh hoạt này được đảm bảo bởi kiến trúc phần tầng minh bạch của hệ thống bao gồm: tầng phục vụ (tầng client), tầng dịch vụ và tầng nút. Trong đó, hầu hết các hệ thống con được thiết kế theo mô hình client-server giúp mở rộng phạm vi hoạt động của hệ thống cũng như tăng cường khả năng phục vụ đa nhiệm, đa phiên làm việc.

- Chương 3 trình bày chi tiết hệ thống thực thi tương tứng với chức năng tối thiểu của một hệ thống tính toán – hệ thống thực thi chương trình. Thành phần cốt lõi của hệ thống này là bộ quản lý công việc được xây dựng dựa trên PBS.

- Chương 4 đề cập tới kiến trúc của hệ thống phát triển chương trình. Hệ thống gồm 2 thành phần chính nằm trên 2 tầng: tầng dịch vụ và tầng phục vụ. Các dịch vụ như quản lý mã nguồn, biên dịch chương trình, tìm lỗi ... thuộc tầng dịch vụ được đưa tới người dùng thông qua giao diện của một môi trường phát triển tích hợp (IDE) thuộc tầng phục vụ. Tầng nút của hệ thống này gồm chủ yếu các gói phần mềm sẵn có như GDB (GNU DeBugger) hay LAM/MPI.

- Chương 5 bàn về hệ thống quản trị, giám sát hệ thống. Mặc dù đây không phải là chức năng bắt buộc trong một hệ thống tính toán nhưng nó lại khá hưu ích, đặc biệt đối với người quản trị . Một đặc điểm nổi bật của hệ thống này là sự phức tạp vì nó gồm rất nhiều thành phần nhỏ. Sự phức tạp này la do hệ thống gồm rất nhiều chức năng và các chức năng này còn liên quan tới rất nhiều các thành phần trong các hệ thống con khác.

Lưu ý rằng, mặc dù hệ thống được chia làm 3 hệ thống con tương ứng với 3 nhóm chức năng riêng biệt, độc lập nhưng vẫn có những thành phần được sử dụng chung cho nhiều hệ thống. Thành phần này có thể được xuất hiện nhiêu lần trong các mô tả kiến trúc của các hệ thống con.

CHƯƠNG 2
KIẾN TRÚC TỔNG THỂ

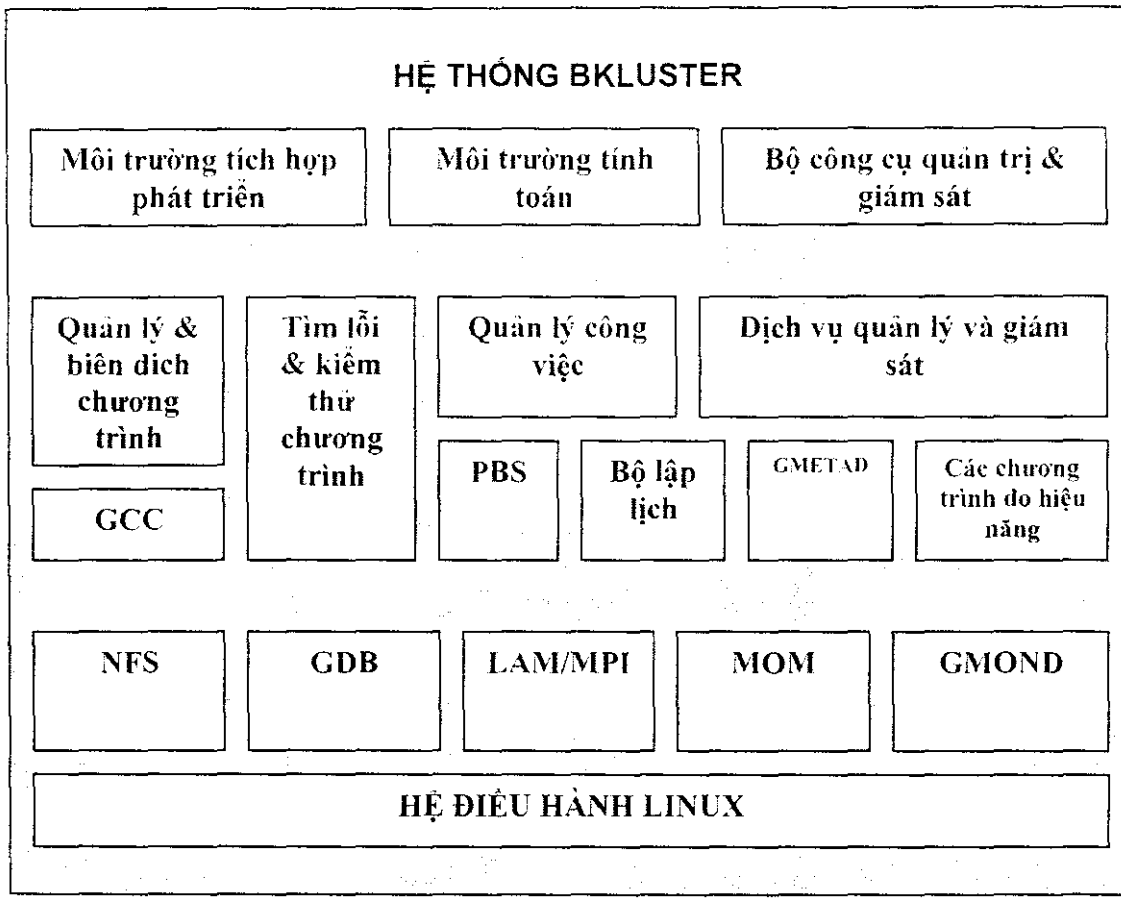
Kiến trúc hệ thống được đề xuất với các tiêu chí:

- Hệ thống BKluster có đầy đủ các chức năng của một hệ thống tính toán bao gồm tính toán, phát triển và quản trị giám sát.
- Hệ thống BKluster hỗ trợ nhiều người dùng một lúc kể cả người dùng từ xa
- Hệ thống BKluster có khả năng mở rộng linh hoạt dựa trên sự phân tầng chức năng minh bạch.
- Sự phức tạp trong tổ chức phân tán trở nên trong suốt đối với người dùng.

Trong chương này mô tả hệ thống một cách khái quát. Kiến trúc tổng thể được xem xét trên một số phương diện:

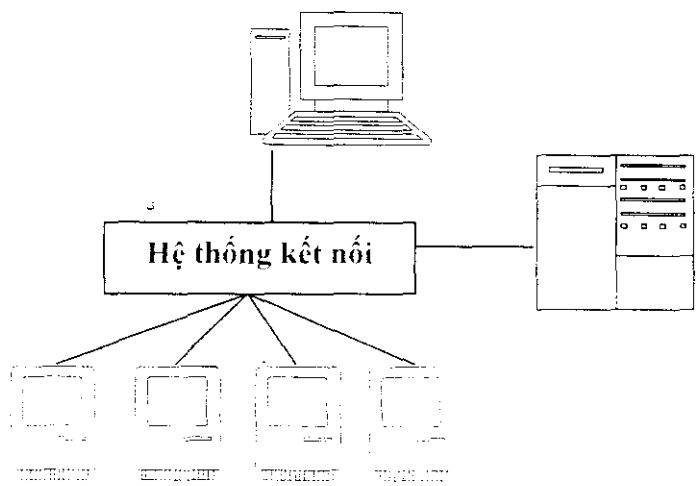
- Kiến trúc phân tầng
- Kiến trúc các phân hệ
- Kiến trúc vật lý và tổ chức trên từng thành phần vật lý

Trong đó, hai phương diện đầu, hệ thống BKluster được xem như một "máy tính" ảo, yếu tố phân tán các thành phần tính toán trở nên trong suốt. Hai cách nhìn này quan tâm chủ yếu tới yếu tố tổ chức phần mềm hệ thống. Hình vẽ sau mô tả các thành phần chính Lưu ý rằng, trong hình vẽ, chúng tôi đã lược đi một số thành phần nhỏ hoặc ghép các thành phần nhỏ vào một khối lớn, ví dụ dịch vụ quản lý và giám sát gồm rất nhiều dịch vụ nhỏ khác hay PBS bao gồm PBS server và bộ lập lich. Chi tiết của từng thành phần cụ thể sẽ được đề cập trong các phần sau. Ở đây chúng tôi mong muốn thông qua hình vẽ, người đọc có những ý niệm ban đầu về hệ thống.



Hình 2-1 Kiến trúc tổng quan hệ thống BKluster

Trong cách nhìn sau cùng, hệ thống được phân chia theo các thành tố phần vật lý cấu thành lên nó, bao gồm tập các nút tính toán, hệ thống kết nối và hệ thống máy chủ quản lý.



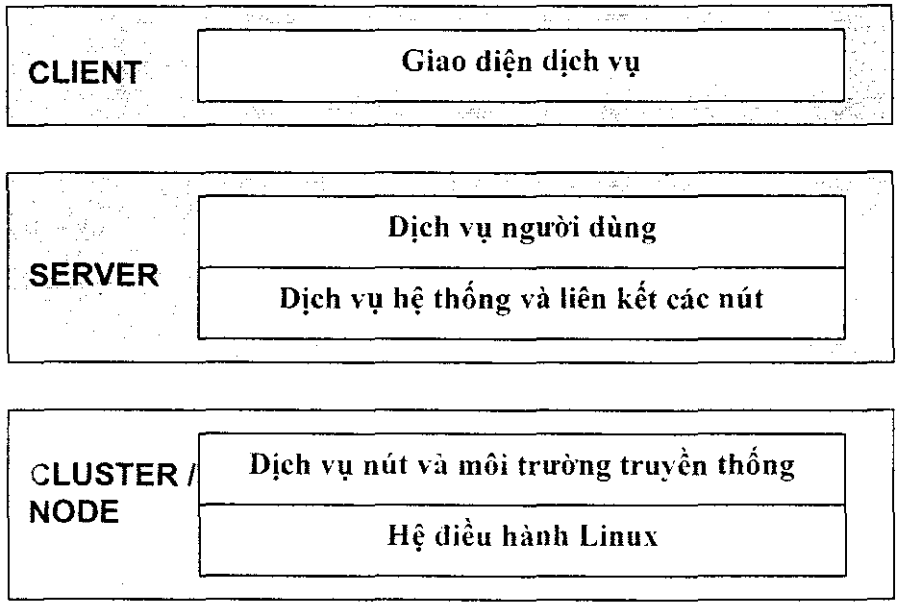
Hình 2-2 Kiến trúc vật lý tổng quan hệ thống BKluster

2.1 Kiến trúc phân tầng

Hệ thống được phân định thành 3 tầng rõ ràng với các chức năng xác định, tách biệt:

- Tầng giao phục vụ (client layer) cung cấp các giao diện người dùng giúp đơn giản hóa các tác vụ cũng như hiển thị thông tin một cách trực quan
- Tầng dịch vụ (server layer) bao gồm các chức năng chính của hệ thống. Tầng này được chia thành 2 tầng con dịch vụ người dùng và dịch vụ hệ thống
- Tầng cụm (cluster layer) đảm nhận chức năng liên kết và các chức năng được phân tán trên các nút tính toán. Tầng này cũng được chia thành 2 tầng con: Dịch vụ nút và môi trường truyền thông và Hệ điều hành Linux.

Kiến trúc phân tầng của BKluster được mô tả trong hình sau. Kiến trúc phân tầng này không chỉ áp dụng cho cả hệ thống mà còn được áp dụng trong thiết kế của các thành phần con.



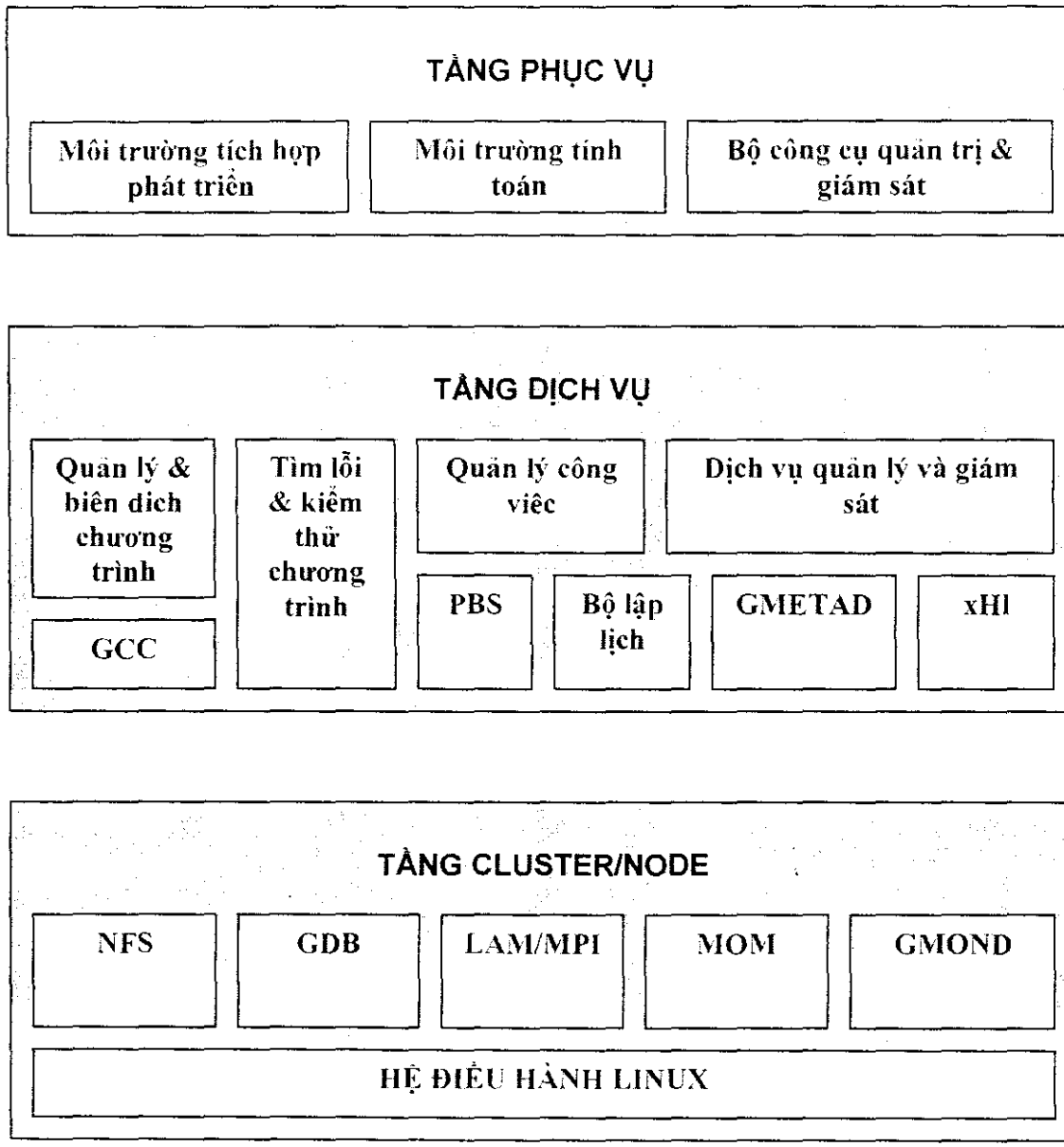
Hình 2-3 Kiến trúc phân tầng hệ thống BKluster

Tầng phục vụ gồm 3 giao diện tương ứng với 3 chức năng lớn: môi trường phát triển tích hợp, môi trường tính toán và bộ công cụ quản lý. Các môi trường làm việc hầu hết chỉ đóng vai trò là các giao diện, ở tầng này không có chức năng xử lý thông tin. Một điểm cần nhấn mạnh nữa là mô hình trao đổi thông tin giữa tầng dịch vụ và tầng phục vụ là mô hình client- server, trong đó, các giao diện thuộc tầng phục vụ đóng vai trò là client.

Các chức năng cơ bản của hệ thống được đặt trong tầng dịch vụ. Tầng này được phân thành 2 tầng con là tầng dịch vụ người dùng và dịch vụ hệ thống. Các giao diện trên tầng phục vụ chỉ liên hệ với các dịch vụ thuộc tầng dịch vụ người dùng. Tầng con Dịch vụ hệ thống và liên kết các nút con làm nhiệm vụ tạo ra cách dịch vụ dựa trên tập hợp các dịch vụ nút. Nói cách khác, tầng này đóng vai trò đầu mối cho các dịch vụ phân tán trên các nút. Do vậy, tầng con này đóng vai trò che lấp sự phức tạp do tính phân tán của các dịch vụ trên nút. Ngoài ra, việc phân tầng dịch vụ thành 2 tầng con còn nhằm mục đích tích hợp tối đa các dịch vụ sẵn có vào hệ thống. Khi đó, tầng con dịch vụ người dùng đóng vai trò một bộ ghép nối các dịch vụ sẵn có này vao hệ thống.

Tầng phân cụm là tầng dưới cùng gồm các dịch vụ được triển khai trên các nút và môi trường truyền thông giữa các nút. Có thể nói đây là tầng cốt lõi đóng vai trò là nhân của hệ thống tính toán phân cụm. Môi trường truyền thông được thiết lập dựa trên các phần mềm trao đổi thông điệp giữa các tiến trình. Trong trường hợp sử dụng các phần mềm phổ biến như LAM/MPI hay MPICH để thiết lập môi trường truyền thông, các phần mềm này vẫn sử dụng các hàm của hệ điều hành LINUX. Xu thế hiện tại, người ta xây dựng các phần mềm trao đổi thông điệp đặc thù cho một hệ thống, khi đó, các phần mềm này tương tác trực tiếp với phần cứng. Vì thế tầng con môi trường truyền thông sẽ nằm ngang hàng cùng tầng hệ điều hành.

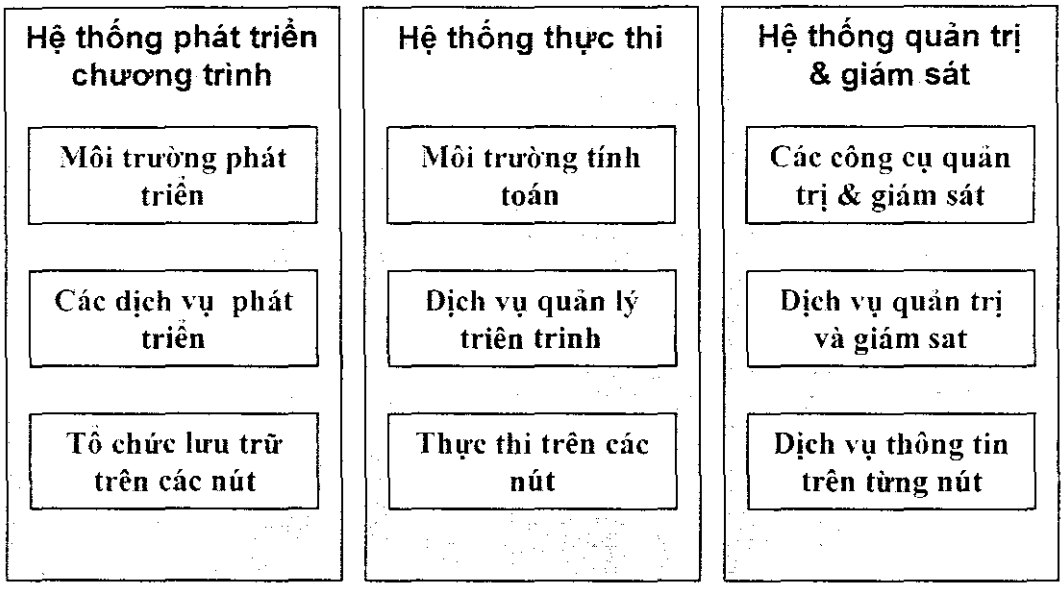
Hình vẽ sau mô tả sự phân bố các thành phần chính của BKluster trong các tầng:



Hình 2-4 Sự phân bố các thành phần chính của BKluster trong các tầng

2.2 Kiến trúc phân hệ

Nếu kiến trúc phân tầng phân chia hệ thống BKluster theo chiều ngang thì kiến trúc phân hệ phân chia BKluster theo chiều dọc.



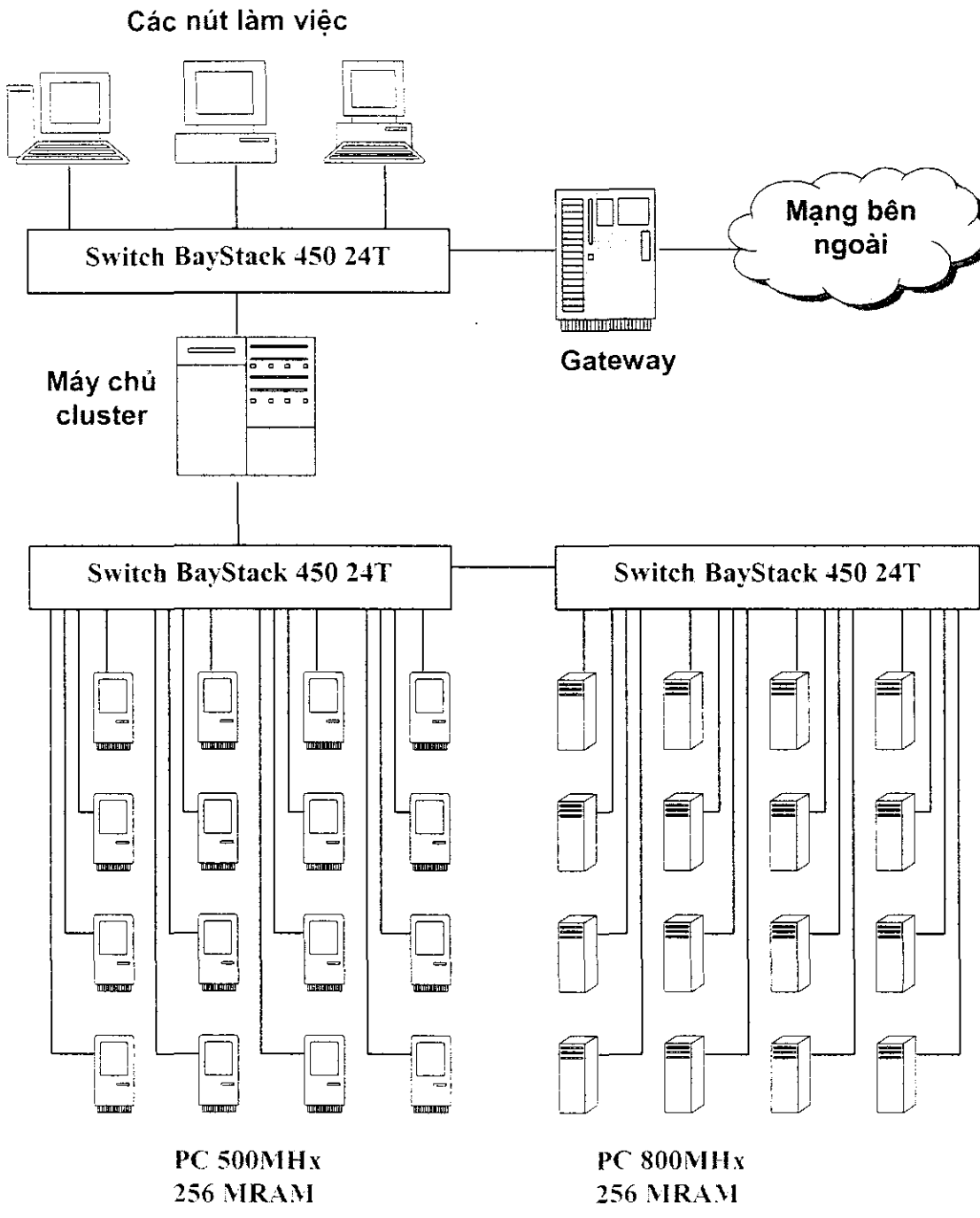
Hình 2-5 Kiến trúc phân hệ BKluster

Trên thực tế, việc phân chia hệ thống BKluster thành 3 hệ thống con chỉ mang tính chất tương đối và chỉ thực sự rõ ràng và tách bạch ở tầng trên cùng – tầng phục vụ. Ở các tầng dưới, các dịch vụ phát triển, dịch vụ quản lý tiến trình và dịch vụ quản trị giám sát có những thành phần chung. Nguyên nhân không thể tạo ra sự tách bạch là việc sử dụng các gói phần mềm nguồn sẵn có, ví dụ như PBS được dùng cho cả dịch vụ quản lý tiến trình lẫn dịch vụ quản trị. Do vậy, trong hình vẽ trên, chúng tôi chỉ minh họa ở mức rất chung mà tránh đi vào chi tiết nhằm có thể tách bạch 3 hệ thống con. Thiết kế chi tiết về từng hệ thống con sẽ được đề cập tới trong các phần tiếp theo.

2.3 Kiến trúc vật lý

Kiến trúc vật lý mô tả cụ thể việc triển khai hệ thống BKluster. Hệ thống bao gồm 32 nút tính toán gồm 16 PC với tốc độ 500 MHz và 16 PC 800 MHz. Một máy chủ HP Netserver 6000 vừa đóng vai trò là front-end của hệ thống, vừa đóng

vai trò là hệ thống lưu trữ:



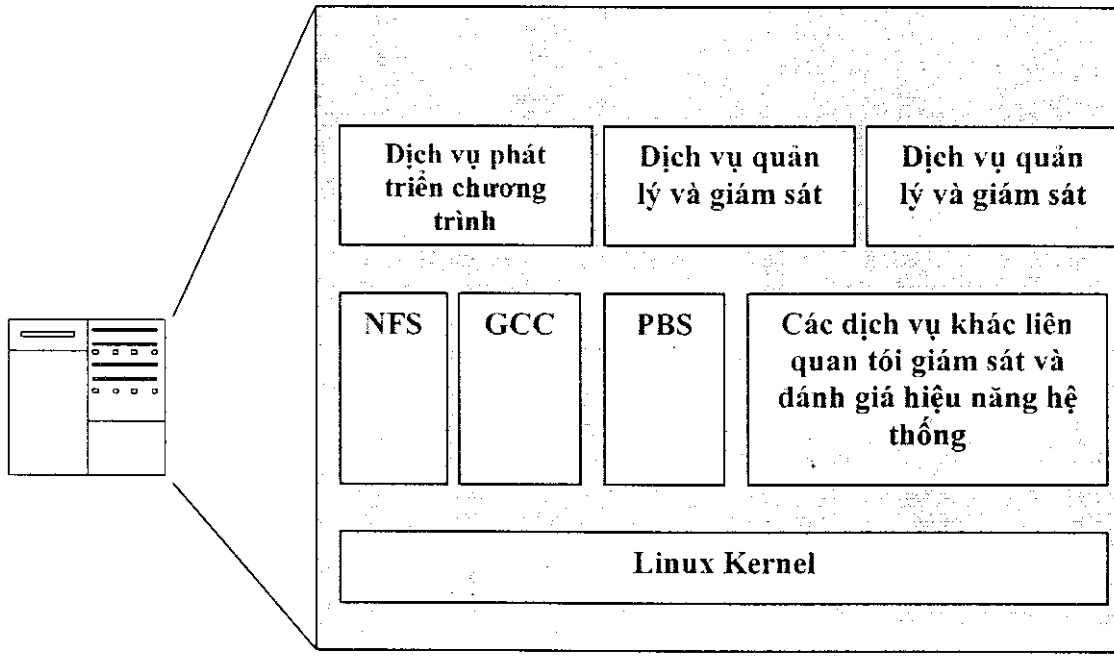
Hình 2-6 Kiến trúc vật lý hệ thống BKluster

2.3.1 Thiết lập cấu hình phần mềm trên hệ thống

Các dịch vụ được cài đặt trên hệ thống chia làm 2 lớp :

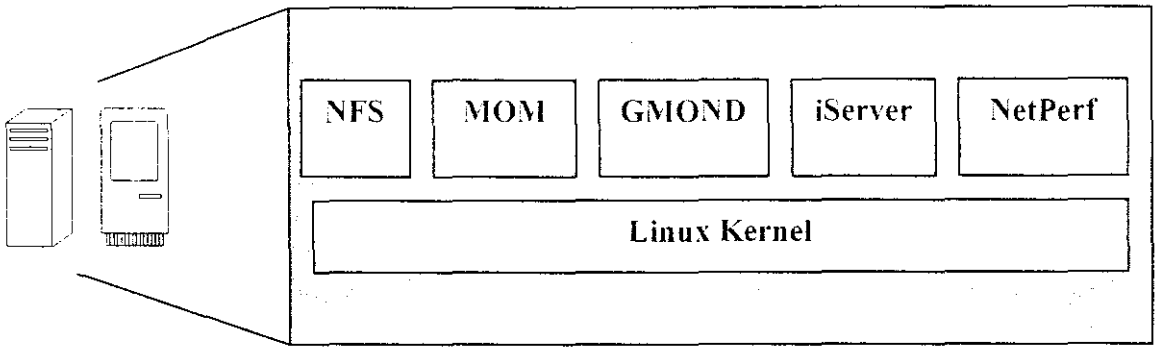
- Các dịch vụ cài đặt trên máy chủ tính toán
- Các dịch vụ cài trên các nút tính toán

Các dịch vụ cài đặt trên máy chủ gồm các dịch vụ trên tầng phục vụ. Các dịch vụ ở tầng này chia ra làm hai lớp tương ứng với hai nhiệm vụ lớn: các dịch vụ hệ thống đảm trách vai trò của một bộ tập trung các dịch vụ nút, các dịch vụ người dùng đóng vai trò là người cung cấp dịch vụ tới người dùng.



Hình 2-7 Các dịch vụ cài đặt trên máy chủ tính toán

Các dịch vụ trên nút tính toán đóng vai trò là các thành phần xử lý trong hệ thống. Ngoài các dịch vụ, trên các nút cần thiết lập hệ thống truyền thông chó phép trao đổi thông điệp giữa các nút trong quá trình tính toán.



Hình 2-8 Các dịch vụ cài đặt trên nút tính toán

2.3.2 Thiết lập hệ thống lưu trữ

Hệ thống lưu trữ của BKluster được xây dựng dựa trên hệ thống file mạng NFS và được lưu trữ vật lý trên máy chủ. Thông qua NFS, hệ thống file của BKluster trở nên đồng nhất và nhất quán. Ngoài hệ thống lưu trữ thông nhất này danh cho viêc lưu trữ các chương trình của người dùng, hệ thống lưu trữ phân tán trên từng nút vẫn được duy trì dành cho việc lưu trữ các thư viện tính toán, ứng dụng song song cài đặt sẵn và các dịch vụ trên tầng nút.

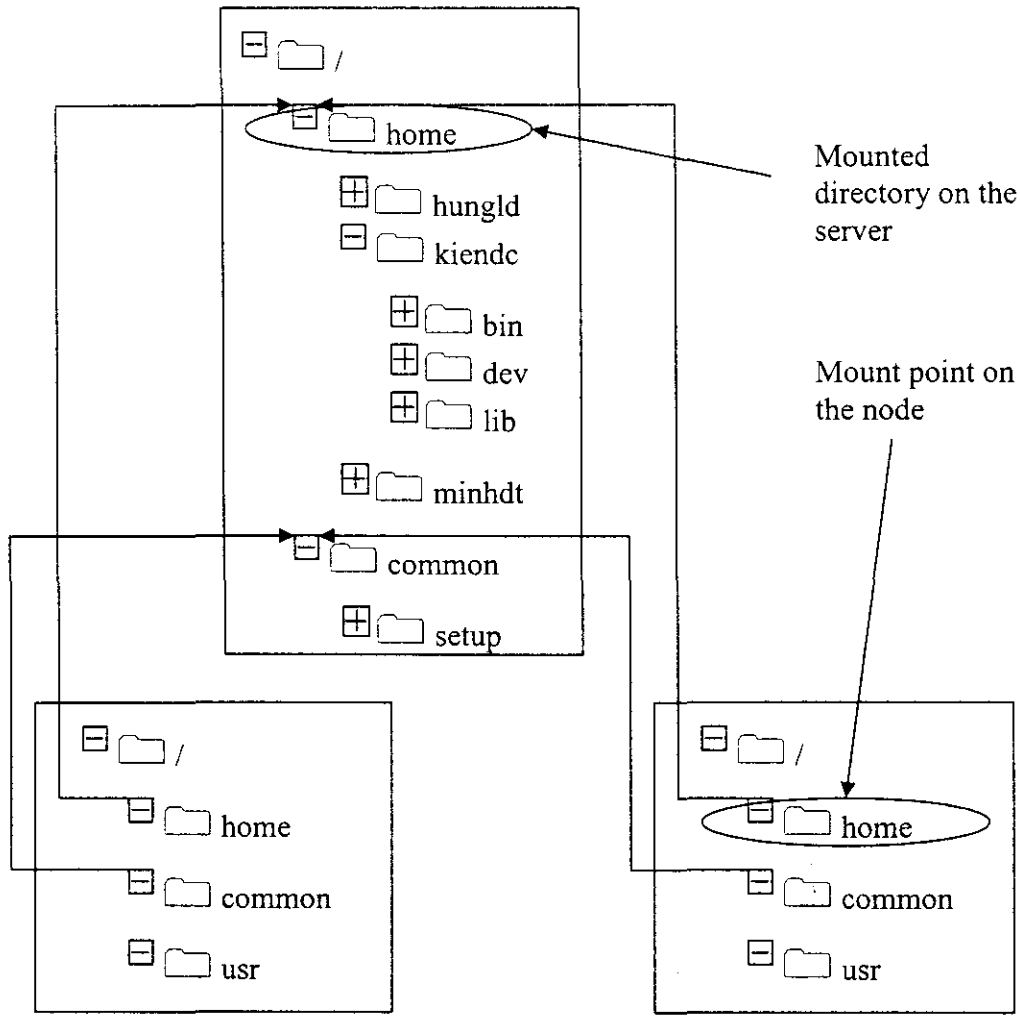
Hệ thống lưu trữ tập trung thông qua NFS bao gồm:

- Thư mục làm việc của mỗi người dùng.
- Thư mục phục vụ các mục đích chung

Mỗi người dùng có môt cây thư mục riêng. Trong thư mục này có 3 thư mục được tạo sẵn: ***bin*** chứa chương trình chạy bao gồm các chương trình muốn kiểm thử và các chương trình được đề trình, ***lib*** các thư viện phục vụ việc phát triển chương trình hoặc chương trình tính toán được đệ trình và thư mục ***dev*** chứa các dự án phát triển chương trình tính toán.

Thư mục phục vụ cho các mục đích chung chỉ cho phép người quản trị truy nhập. Việc sử dụng thư mục chung nhằm hạn chế việc truyền file xuống các nút trong các thao tác trên các nút tính toán, ví dụ, việc cài đặt các thư viên hay các dịch vụ

trên các nút.



Hình 2-9 Hệ thống lưu trữ trong BKluster

CHƯƠNG 3
HỆ THỐNG PHÁT TRIỂN TÍCH HỢP

Hệ thống phát triển tích hợp bao gồm một tập các dịch vụ hỗ trợ người dùng phát triển các chương trình cho BKluster ở nhiều mức khác nhau. Với 3 nhóm công cụ chính, hệ thống đảm trách được các công việc cơ bản trong quá trình xây dựng một chương trình bao gồm:

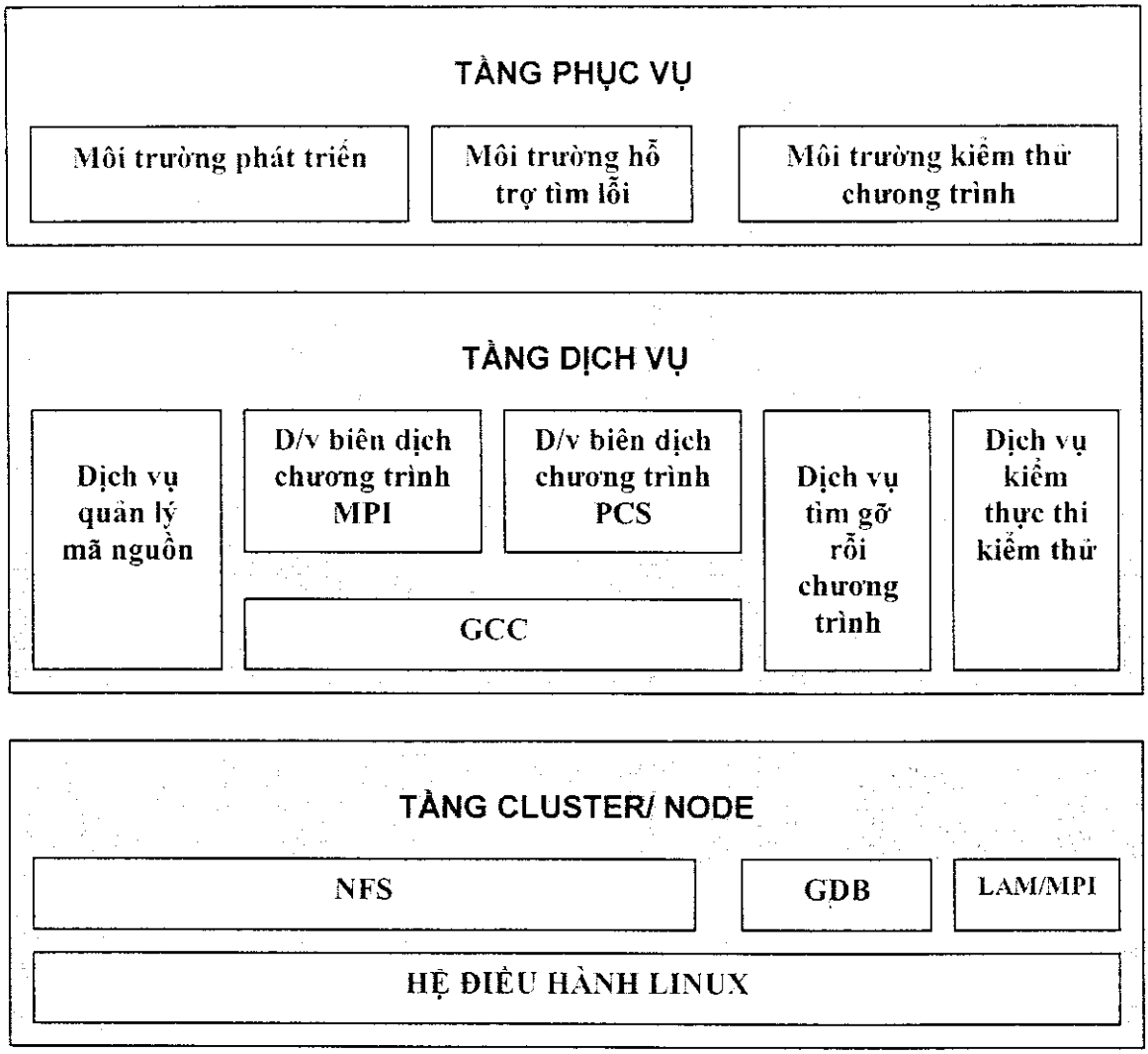
- Soạn thảo và biên dich
- Kiểm thử
- Tìm lỗi chương trình

Dựa trên 3 lớp chức năng chính mà hệ thống con này được chia nhỏ thành 3 module : module phát triển, module gỡ rối chương trình và module kiểm thử. Trong 3 module này, module phát triển là phức tạp nhất mặc dù các chức năng của nó nằm chủ yếu ở 2 tầng trên. Nó chỉ sử dụng dịch vụ NFS ở tầng dưới cùng cho phép đồng nhất mã nguồn và chương trình trên tất cả các nút. Module kiểm thử là module đơn giản nhất. Thực chất nó giống như hệ thống thực thi nhưng đơn giản hơn rất nhiều.

3.1 Module phát triển chương trình

Module phát triển chương trình có gồm các công cụ giúp người sử dụng dễ dàng phát triển môt chương trình tính toán song song. Đứng từ phương diện phát triển chương trình thuần túy, hệ thống cho phép:

- Quản lý mã nguồn của một dự án
- Hỗ trợ tối đa việc soạn thảo mã nguồn
- Biên dịch chương trình

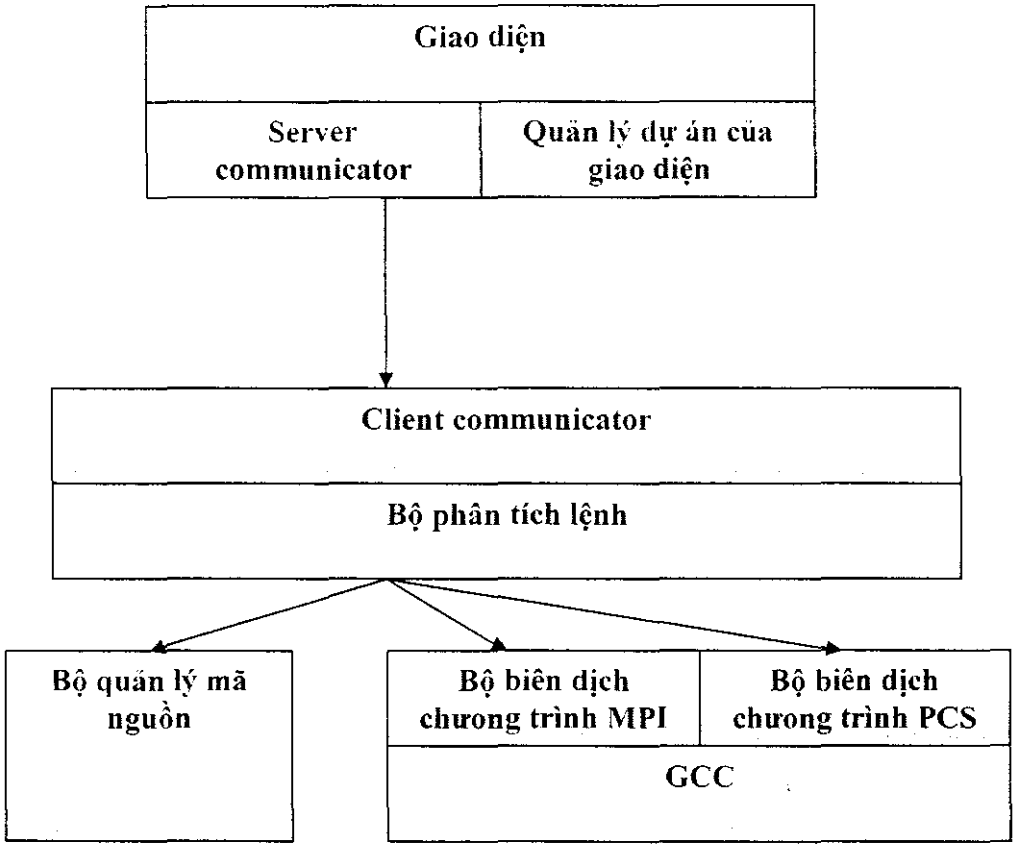


Hình 3-1 Hệ thống phát triển tích hợp

Module cũng có các công cụ hỗ trợ việc phát triển các chương trình song song ở nhiều cấp độ khác nhau :

- Các công cụ hỗ trợ người lập trình chuyên nghiệp sử dụng các ngôn ngữ C, hay FORTRAN xây dựng các chương trình song song theo chuẩn MPI

- Hỗ trợ người lập trình không hiểu rõ về lập trình song song. Lớp người dùng này có thể sử dụng ngôn ngữ đặc tả PCS để mô tả yêu cầu tính toán như một chương trình tuần tự

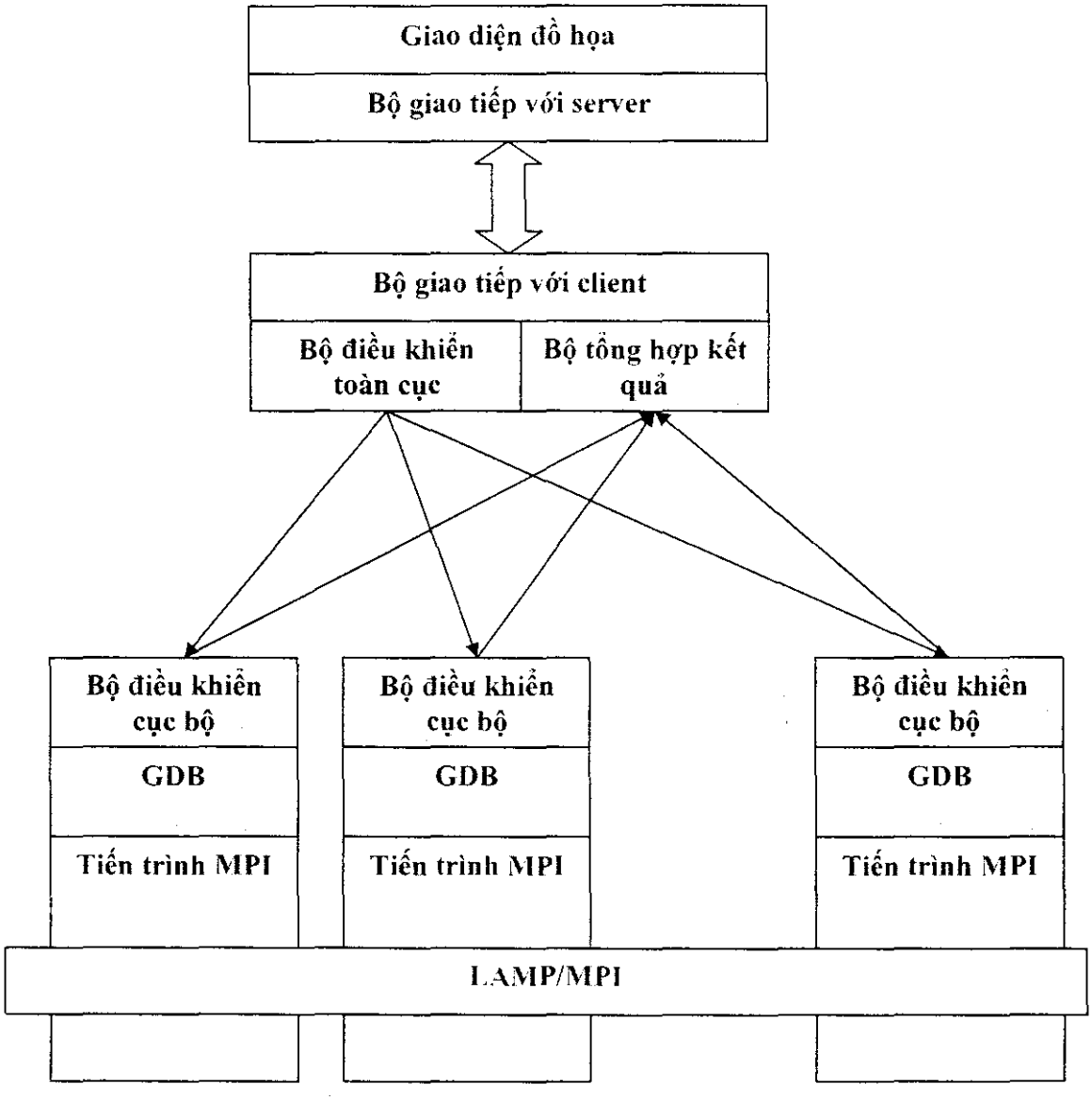


Hình 3-2 Kiến trúc module phát triển chương trình

Thành phần server cung cấp hai dịch vụ quản lý mã nguồn và biên dịch chương trình.

3.2 Module hỗ trợ gỡ rối chương trình – BKPD

Kiến trúc và các thành phần của module BKPD được mô tả trong hình vẽ sau:



Hình 3-3 Kiến trúc module hỗ trợ gỡ rối chương trình

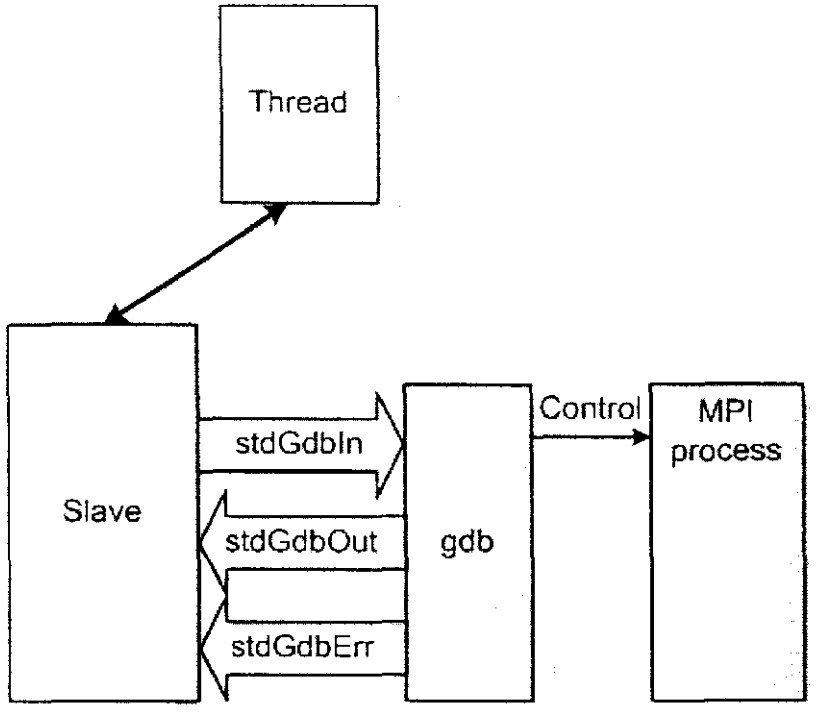
BKPD có mô hình kiến trúc theo các tầng như trên. Cụ thể là:

- DE (Debugging enviroment) : Môi trường gỡ lỗi song song. Tầng này nằm phía client, cung cấp cho người dùng khả năng tiến hành gỡ lỗi từ xa.
- Tầng truyền thông kết nối: bao gồm Client communicator, Server Daemon, và Server communicator.
    - Client và Server communicator: Là tầng đệm trung gian đảm nhận nhiệm vụ truyền thông giữa client và server. Client communicator cung cấp các hàm API cho DE để thực hiện các câu lệnh debug. Nhờ Client và Server comm, DE nhìn hệ thống phân tán của ta giống như là hệ thống cục bộ.
    - Server Daemon: Tiến trình ngầm phục vụ phía Server tính toán. Tiến trình này luôn chờ ở một cổng, mỗi khi có người dùng kết nối đến sẽ sinh ra tiến trình phục vụ.
- Master : Thành phần đảm nhận các công việc sau :
    - Khởi tạo các slave tại thời điểm ban đầu.
    - Là nơi xử lý lệnh nhận được từ người sử dụng, thực hiện lệnh, nhận kết quả trả về và xử lý kết quả (trước khi kết quả được truyền về cho người sử dụng).
- Slave : Thành phần điều khiển gdb hoạt động : truyền lệnh cho gdb, nhận kết quả trả về từ gdb và truyền cho master.
- Gdb.
- Chương trình MPI.

3.2.1 Thành phần Bộ điều khiển cục bộ

Slave là thành phần chạy trên từng nút tính toán, đảm nhận vai trò điều khiển chương trình gỡ lỗi tuần tự gdb, thực hiện gỡ lỗi trên từng tiến trình riêng biệt của chương trình song song.

Các Slave sẽ gắn gdb vào tiến trình song song, nhận lệnh gỡ lỗi từ Master, điều khiển gdb thực hiện lệnh, sau đó trả kết quả về cho Master.



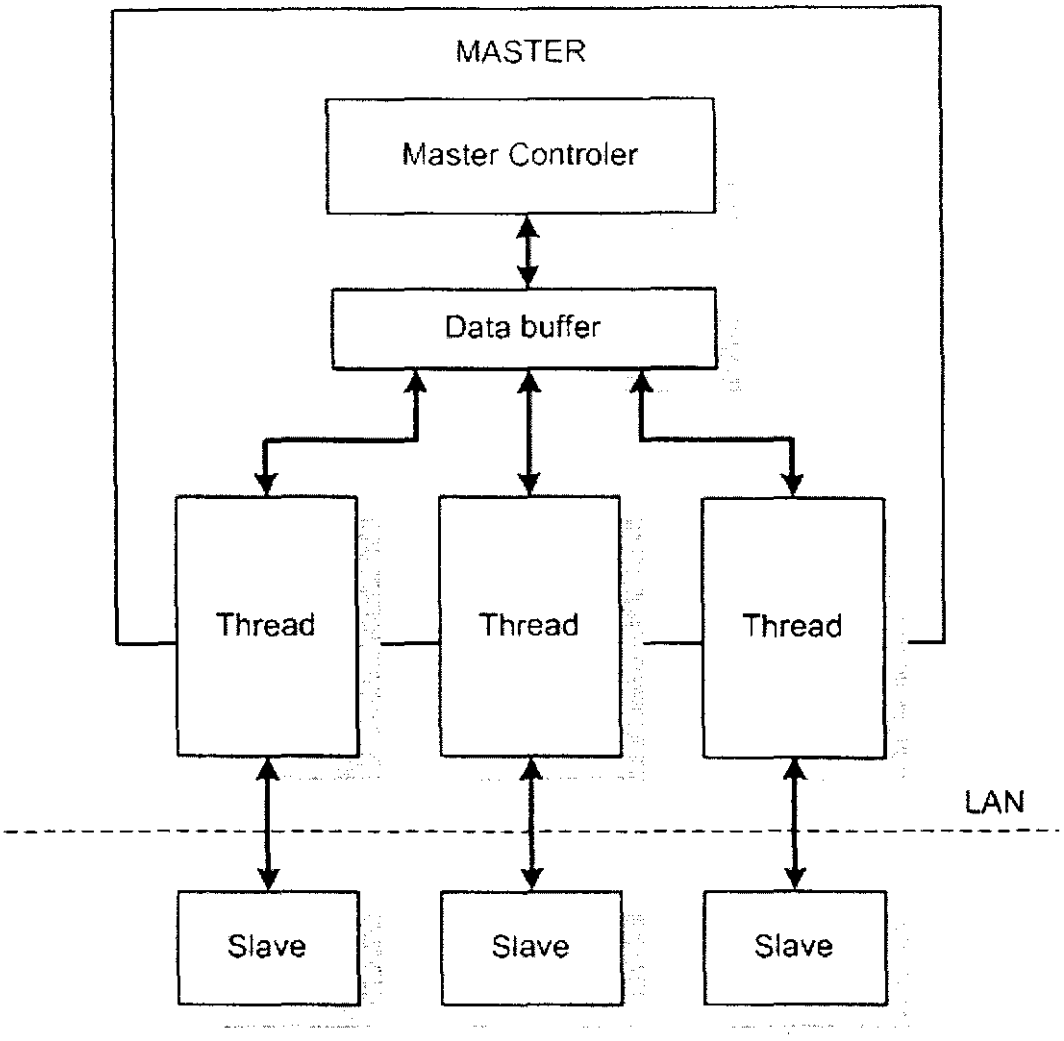
Hình 3-4 Bộ điều khiển cục bộ trong module Gỡ rối chương trình

Slave giao tiếp với gdb thông qua cơ chế đường ống pipe. Giao tiếp này bao gồm ba đường ống chuẩn sau:

- stdGdbIn: đường ống vào chuẩn của gdb.
- stdGdbOut: đường ống ra chuẩn của gdb.
- stdGdbErr: đường ống ra lỗi chuẩn của gdb.

3.2.2 Thành phần Bộ điều khiển toàn cục

Thành phần Master đảm nhiệm vai trò điều khiển các nút tính toán thực hiện câu lệnh debug của người sử dụng. Master nhận lệnh từ Server, tiền xử lý lệnh, sau đó truyền lệnh cho các slave thông qua các thread. Master nhận kết quả trả về từ các thread, tổng hợp và trả về thông báo cho Server Communicator.



Hình 3-5 Bộ điều khiển toàn cục trong module gỡ rối chương trình

Master Controler

Đây là thành phần điều khiển chính của Master. Thành phần này đảm nhận trách nhiệm:

- Xử lý câu lệnh được chuyển tới.
- Truyền câu lệnh vào bộ đệm lệnh.
- Tác động lên cờ lệnh.
- Đọc dữ liệu kết quả trả về.
- Xử lý kết quả và thông báo cho Server Communicator.

Thread

Các tiến trình chạy độc lập có vai trò giao tiếp với các slave trên các nút tính toán. Thread thực hiện các công việc sau:

- Nhận lệnh từ bộ đệm lệnh.
- Truyền lệnh cho slave của mình.
- Nhận kết quả trả về từ slave.
- Ghi kết quả vào bộ đệm kết quả.
- Tác động vào cờ kết quả.

Data Bufer

Là trung gian lưu trữ dữ liệu giao tiếp giữa Master Controler và các Thread. Data Buffer bao gồm:

- Một cờ lệnh (command flag) để Master Controler có thể báo hiệu cho các Slave khi có lệnh
- Một vùng đệm lệnh (command buffer) lưu trữ câu lệnh cần thực hiện.
- Một vùng đệm kết quả (result buffer) lưu trữ các kết quả trả về của các Slave.
- Một cờ kết quả (result flag) để báo cho Master Controler biết khi nào một Slave thực hiện xong lệnh và có kết quả trả về.

3.2.3 Bộ giao tiếp với client

Server Communicator là thành phần trung gian phía Server đảm nhận trách nhiệm giao tiếp với Client Communicator phía Client. Server Communicator nhận lệnh truyền đến từ phía client, gọi các hàm do Master cung cấp để thực hiện lệnh, sau đó trả kết quả về cho Client.

Mỗi một người sử dụng khi yêu cầu một phiên làm việc sẽ có một tiến trình Server Communicator tương ứng phía Server phục vụ.

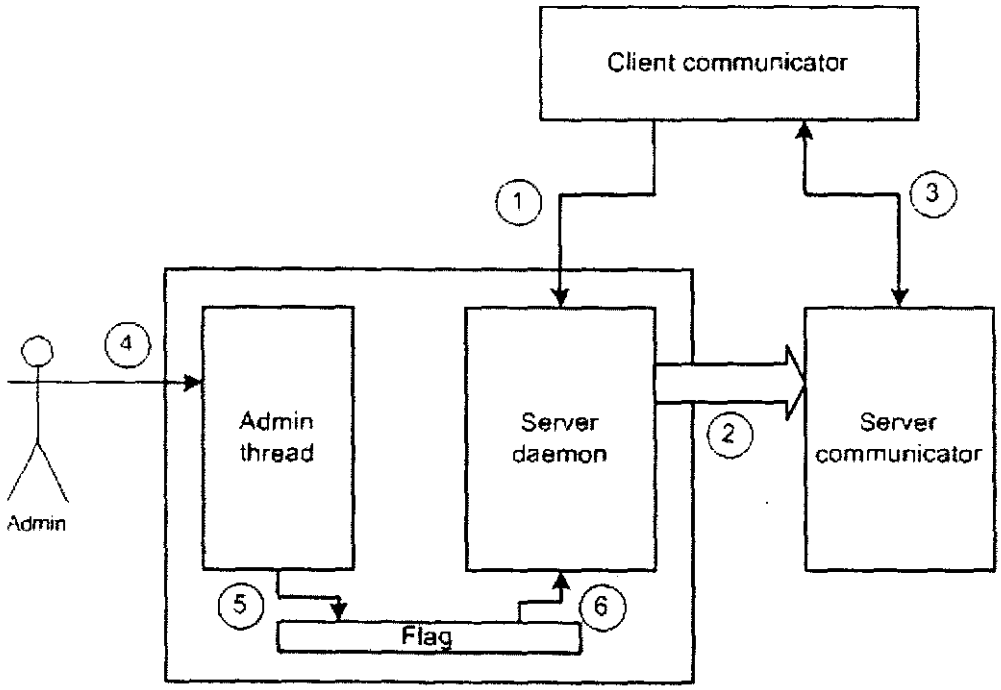
Nhờ có Server Communicator và Client Communicator, DE phía client sẽ nhìn toàn hệ thống như là một chương trình cục bộ.

3.2.4 Thành phần Server Daemon

Server Daemon là tiến trình ngầm chờ nhận kết nối của người dùng. Mỗi khi có người dùng mới, Server Daemon sẽ sinh ra một tiến trình mới phục vụ.

Server Daemon cũng sinh ra một thread riêng cho phép người quản trị có khả năng can thiệp vào quá trình hoạt động của mình như xem số người sử dụng, khoá và mở khoá server...

Server Daemon hiện nay mới chỉ là một mô đun đơn giản, trong tương lai có thể phát triển hơn hoặc có thể sẽ bị thay thế bởi server chung của cả hệ thống BKCluster.



Hình 3-6 Thành phần Server Daemon trong module gỡ rối chương trình

Trong thành phần Server Daemon bao gồm ba thành phần chính:

- Server Daemon: là thành phần chính, tiến trình ngầm chờ và phục vụ người sử dụng.
- Admin thread: là thread chạy song song, cho phép admin quản trị hoạt động của Server.

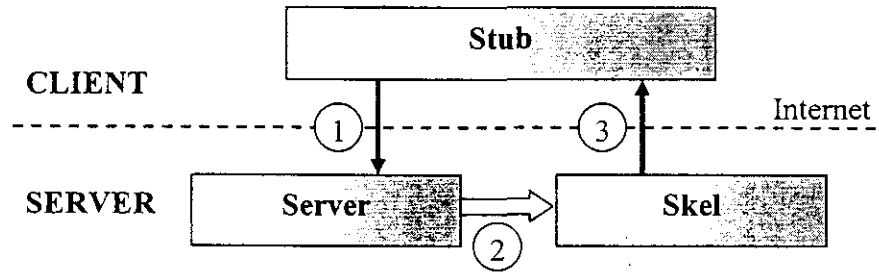
- Flag: các cờ để báo hiệu các lệnh của người dùng đến Server daemon.

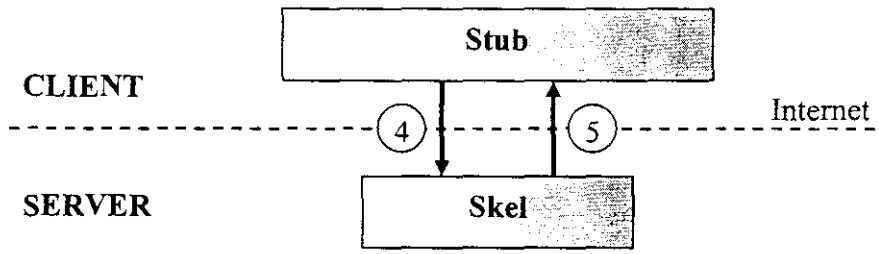
3.2.5 Giao thức

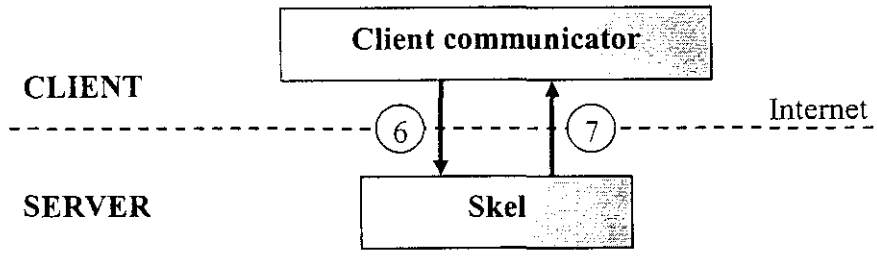
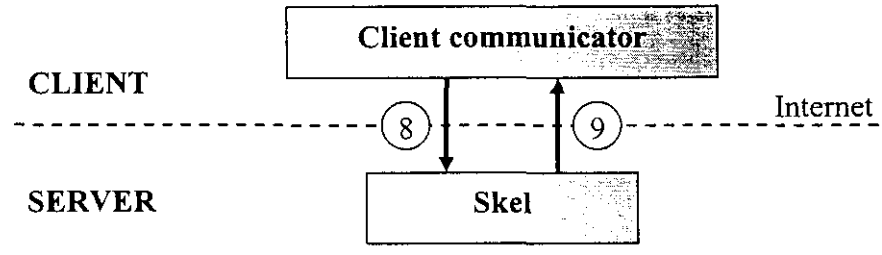
Trong các chương trình ứng dụng mạng, việc truyền thông qua mạng (mạng Internet, mạng LAN) cần có một giao thức cụ thể hợp lý, đồng thời cần có cơ chế đóng gói dữ liệu để truyền đi sao cho hiệu quả.

Trong BKPD có hai giao tiếp cần xây dựng giao thức : giao tiếp giữa Client communicator với Server và Skel, và giao tiếp giữa Master và Slave.

3.2.5.1 Giao thức giữa Client communicator với Server và Skel

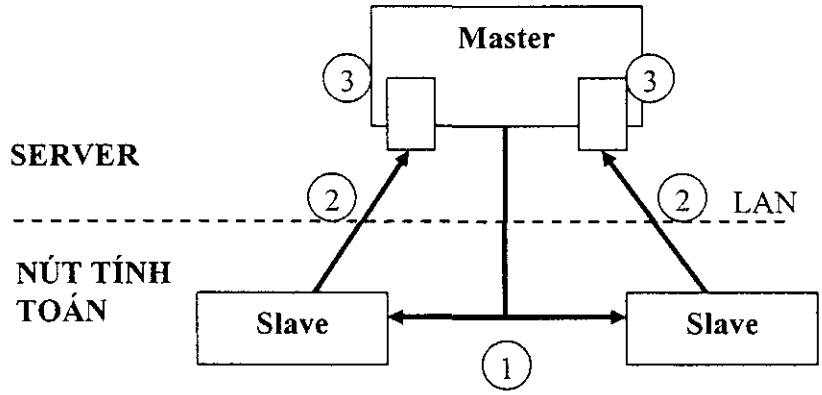
<table>
<tr><th>Pha</th><th>STT</th><th>Mô tả</th><th>Dữ liệu</th></tr>
<tr><td rowspan="3">Kết nối</td><td colspan="3"> 
</td></tr>
<tr><td>1.</td><td>Client communicator truyền thông tin cho Server, yêu cầu tạo một phiên làm việc mới.</td><td>Xâu string xác định định danh người dùng: user ID</td></tr>
<tr><td>2.</td><td>Server sinh ra một tiến trình, gán hàm chạy của Skel cho tiến trình đó.</td><td>Tham số Server truyền cho Skel là định danh socket giao tiếp với Client communicator</td></tr>
</table>

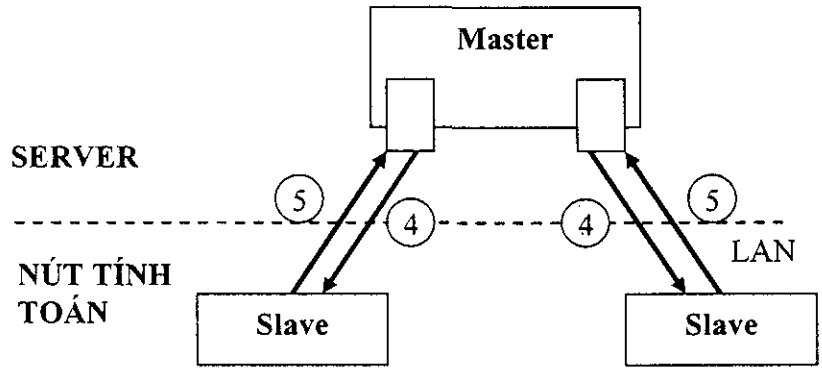
	3.	Skel truyền thông tin về cho Client communicator xác nhận một phiên làm việc mới sẵn sàng.	Dữ liệu truyền từ Skel cho Client communicator là một số nguyên: chính là tổng số nút tính toán trong môi trường tính toán song song tại trung tâm.
Thiết lập			
	4.	Client communicator truyền cho Skel thông tin về chương trình song song muốn chạy và số nút muốn sử dụng.	Xâu string có dang: "so_nut&ten_chuong_trinh"
	5.	Sau khi khởi tạo xong môi trường, Skel truyền thông báo thông tin về quá trình khởi tạo cho Client communicator.	Dữ liệu truyền về là mã lỗi trong quá trình khởi tạo môi trường debug. Mã lỗi = 0 nếu khởi tạo thành công.

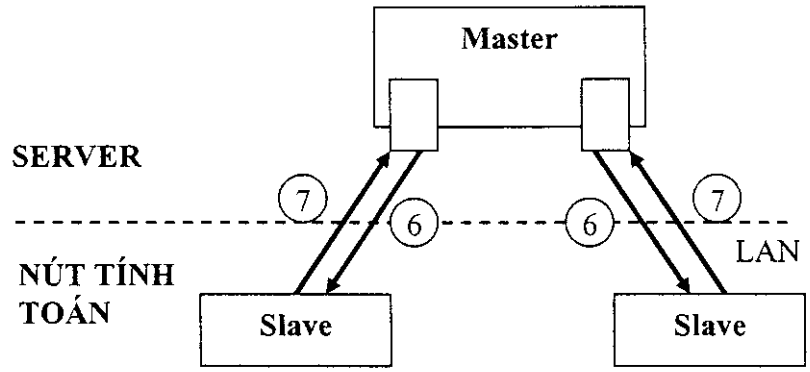
<table>
<tr><td rowspan="3">Phục vụ</td><td colspan="3"> 
</td></tr>
<tr><td>6.</td><td>Client communicator truyền cho Skel câu lệnh debug người sử dụng yêu cầu.</td><td>Câu lệnh có dạng:
“lenh –n ds_nut”</td></tr>
<tr><td>7.</td><td>Skel sau khi thực hiện xong lệnh sẽ truyền kết quả về cho Client communicator.</td><td>Kết quả có dạng:
ma_loi&ma_ket_qua&ma_lenh&ket_qua
Trong đó kết quả có dạng :
ket_qua1@ket_qua2@....</td></tr>
<tr><td rowspan="3">Kết thúc</td><td colspan="3"> 
</td></tr>
<tr><td>8.</td><td>Client communicator truyền lệnh kết thúc của người dùng đến cho Skel.</td><td>Lệnh có dạng:
“q –n a”</td></tr>
</table>

	9.	Skel truyền về cho Client communicator xâu xác nhận rồi kết thúc.	Xâu bất kỳ.

3.2.5.2 Giao thức giữa Master và Slave

Pha	STT	Mô tả	Dữ liệu
Thiết lập			
	1.	Master triệu gọi các Slave.	Master gọi Slave thông qua câu lệnh: Mpirun –np nnodes Slave mpi_prog port Trong đó: - nnodes: số nút chạy Slave - mpi_prog: file chạy chương trình song song. Yêu cầu được dịch với tuỳ chọn –g. - port: cổng để Slave kết nối đến Master.

	2.	Slave sau khi khởi tạo các tiến trình gdb thành công sẽ gửi thông báo xác nhận về cho Master đến cổng đã được truyền qua dòng lệnh mpirun.	Dữ liệu truyền là một xâu chứa rank của nút tính toán chạy Slave tương ứng. VD máy có rank=1 thì xâu truyền đi là "1".
	3.	Master mỗi khi nhận được một tín hiệu xác nhận từ Slave sẽ tạo ra một thread đọc lập đảm nhiệm giao tiếp với Slave đó trong suốt phiên làm việc.	
Phục vụ			
	4.	Các thread chạy trong Master truyền cho Slave câu lệnh đã được xử lý.	Xâu lệnh chính là phần "lenh" trong câu lệnh "lenh –n ds_nut" mà Client communicator truyền cho Skel.

	5.	Slave sau khi điều khiển gdb thực hiện xong lệnh sẽ truyền kết quả về cho Master (thông qua các thread)..	Dữ liệu truyền về là xâu dữ liệu trả về của gdb.
Kết thúc			
	6.	Master thông qua các thread truyền cho Slave lệnh kết thúc	Câu lệnh có dạng: "q"
	7.	Slave sau khi kết thúc xong gdb, sẽ gửi thông báo xác nhận về Master sau đó tự kết thúc.	Thông báo xác nhận là một xâu bất kỳ ("Bye")

CHƯƠNG 4
HỆ THỐNG THỰC THI TÍNH TOÁN

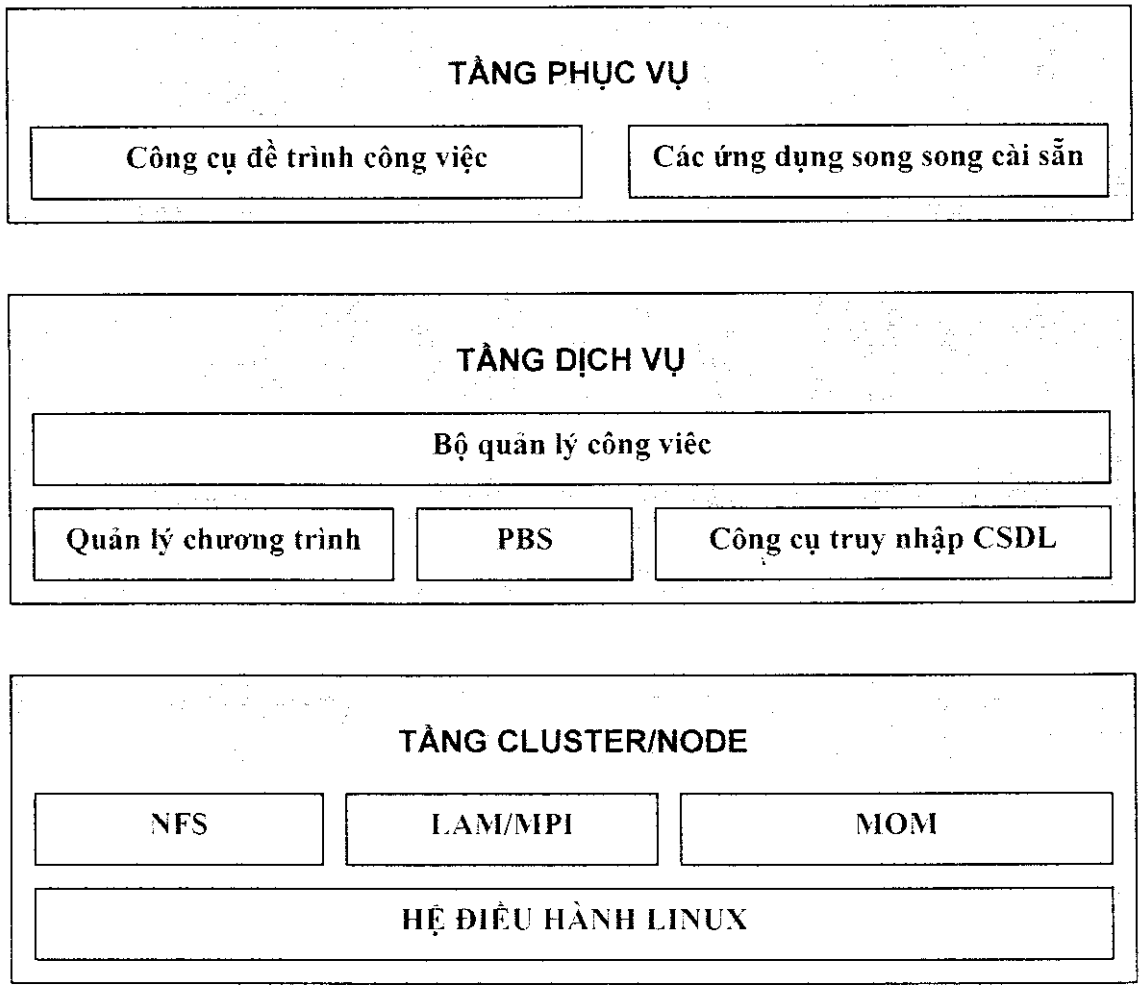
Việc thực thi một chương trình toán toán song song đòi hỏi thêm khá nhiều tham số ví dụ như số tiến trình cho một tác vụ, số nút thực thi (trên thực tế, người ta cố gắng sử dụng số nút bằng chính số tiền trình được yêu cầu). Việc xác định đực một bộ tham số tối ưu có ảnh hưởng nhất định tới hiệu quả tính toán của riêng tác vụ được yêu cầu cũng như hiệu năng chung của cả hệ thống. Để tăng cường phạm vi phục vụ của hệ thống tính toán, chúng tôi cho phép người dùng khai thác năng lực tính toán của hệ thống theo hai hình thức:

- Người dùng sử dụng sẵn một số ứng dụng tính toán đã được cài đặt sẵn trên hệ thống. Trong trường hợp này, thao tác là rất đơn giản, người dùng chỉ cần kích hoạt chương trình bằng lệnh
- Người dùng đệ trình một công việc dưới dạng một file thực thi. Ngoài việc, chỉ đinh file thực thi, người dùng còn phải xác lập một số tham số cần thiết.

Trong trường hợp thứ nhất, người dùng khai thác hệ thống thông qua một ứng dụng song song đã cài đặt sẵn. Dưới hình thức này, người dùng không phân biệt được đây là một ứng dụng song song hay tuần tự. Họ cũng không phân biệt được minh đang khai thác hệ thống BKluster hay một máy tính thông thường. Để có được điều này, các chương trình đã được phát triển và thiết lập các tham số phù hợp với hệ thống. Ưu điểm của các chương trình này là sự tiện lợi và hiệu quả tính toán do được phát triển dựa trên đặc thù hệ thống. Tuy nhiên, các chương trình này thường mang tính tổng quát, giải quyết một lớp bài toán hoặc liên quan tới một lĩnh vực rộng. Do vậy, không thể nào đáp ứng hoàn toàn như cầu của tất cả lớp người dùng. Nhưng có thể nói đây lại chính là hướng phát triển mở rộng dịch vụ tính toán cho hệ thống BKluster.

Hướng tiếp cận thứ hai trong việc khai thác năng lực tính toán nhằm tăng cường

tính linh hoạt. Nó cho phép người dùng tự phát triển một chương trình tính toán theo mô hình truyền thông điệp. Sau đó họ có thể yêu cầu chạy chương trình này trên hệ thống BKluster nhằm khai thác năng lực tính toán rất mạnh của hệ thống. Rõ ràng, đây là hướng tiếp cận dành cho những người chuyên nghiệp, có những hiểu biết nhất định về tính toán song song, các hệ thống phân cụm nói chung. Người dùng đề xuất yêu cầu tính toán thông qua một công cụ có tên là Job Submit Tool. Chúng tôi thiết kế công cụ này cũng nhằm đơn giản hóa thao tác đề xuất yêu cầu tính toán theo hình thức này. Tuy nhiên, để tối ưu, người dùng sẽ phải tự xác lập một số tham số.

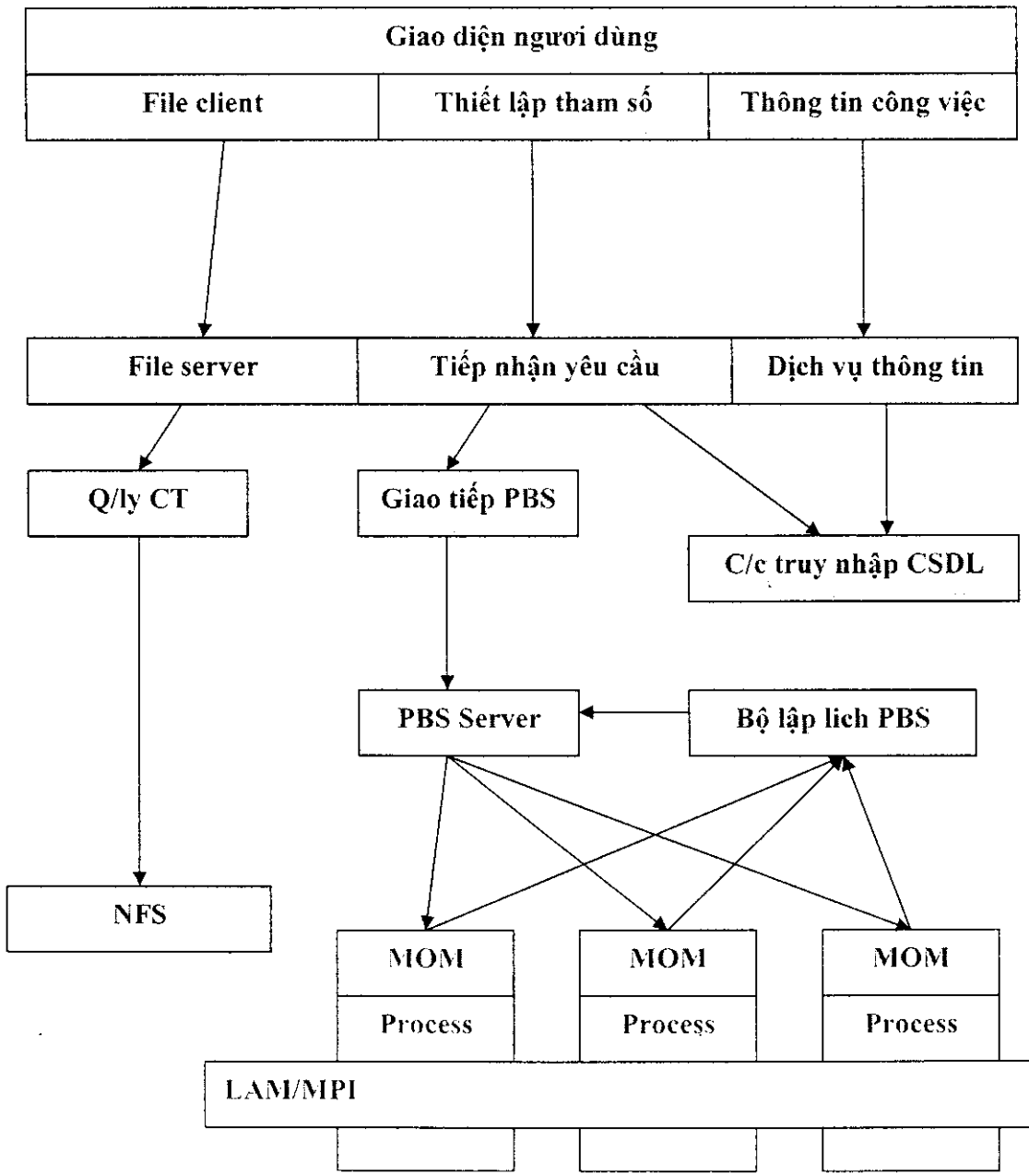


Hình 4-1 Hệ thống thực thi tính toán

Như vậy, người dùng sẽ khai thác hệ thống thông qua một môi trường tính toán. Môi trường này ngoài việc cho phép kích hoạt các chương trình tính toán có sẵn còn có một công cụ hỗ trợ đệ trình công việc. Môi trường này sẽ trao đổi với bộ

quản lý công việc để xác lịch trình làm việc chính xác cho tác vụ được yêu cầu. Hệ thống xếp lịch được xây dựng dựa trên PBS (Portable Batch Server)

Công cụ đệ trình công việc là một bộ công cụ nhỏ trong môi trường tính toán. Công cụ này cho phép người dùng đệ trình một công việc cũng như theo dõi hiện trạng thực hiện của các công việc đã đệ trình. Một đệ trình công việc gồm 2 phần lớn: file thực thi tính toán và bản mô tả các tham số. Các thành phần tham gia vào các tác vụ của bộ công cụ này được mô tả trong hình vẽ sau:

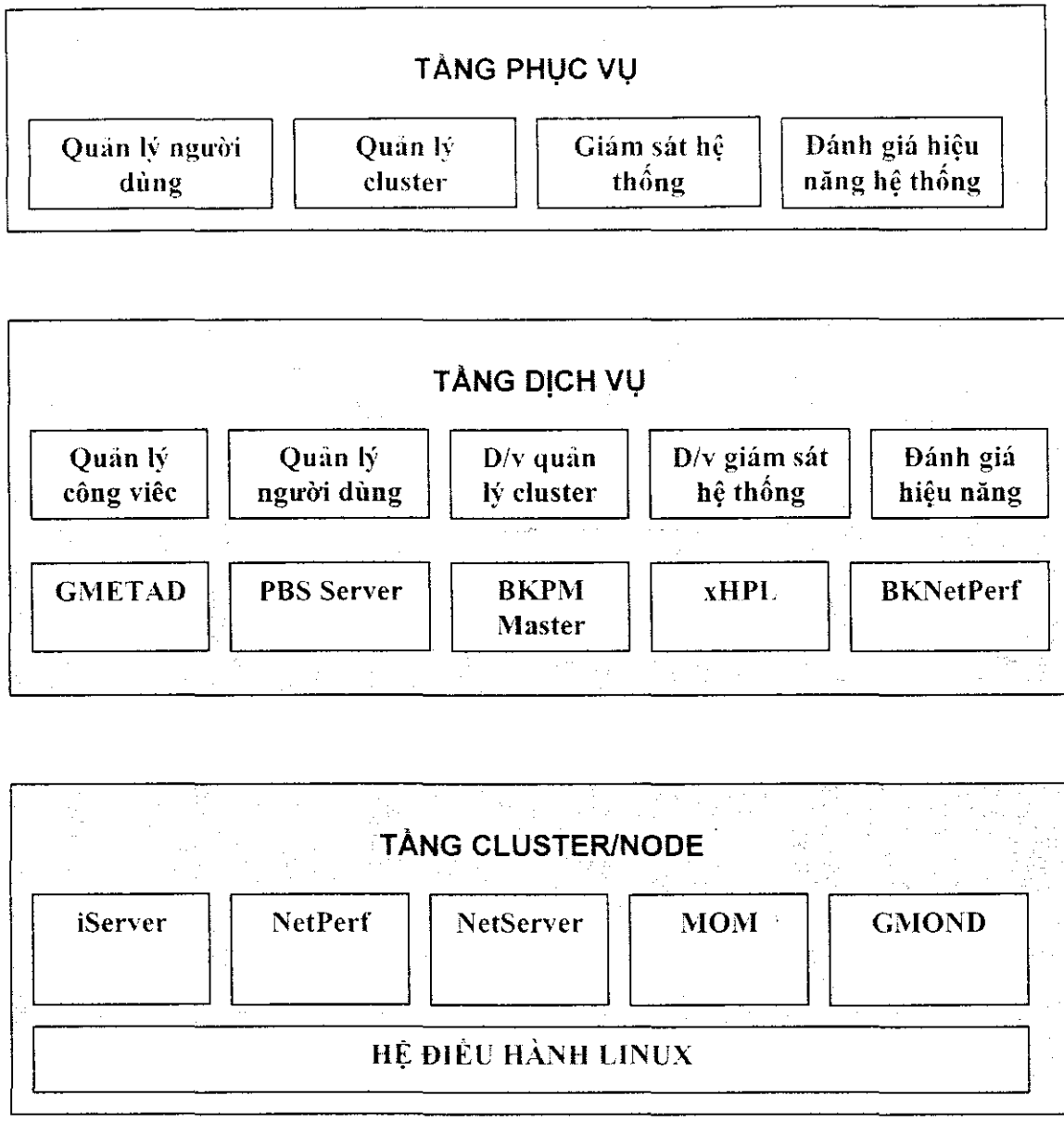


Hình 4-2 Công cụ đệ trình công việc

CHƯƠNG 5
HỆ THỐNG GIÁM SÁT VÀ QUẢN TRỊ

Hệ thống con giám sát và quản trị là hệ thống con phức tạp nhất. Nó gồm nhiều các thành phần với nhiều dạng chức năng khác nhau. Xét từ góc độ các nhóm chức năng, hệ thống lại có thể được chia thành các module tương ứng với các nhóm chức năng chính:

- Module quản lý các gói phần mềm trên các nút tính toán.
- Module quản lý cấu hình cluster: Module này có hai nhiệm vụ chính là quản lý các thành phần vật lý của cluster bao gồm các nút tính toán, máy chủ và hệ thống lưu trữ, giám sát và quản lý các tham số của hệ thống mà cụ thể là các hàng đợi công việc, bộ lập lịch.
- Module quản lý người dùng: quản lý người dùng của BKluster và mối quan hệ với người dùng LINUX trên các nút tính toán.
- Module giám sát hoạt động hệ thống: Module này cho phép hiển thị trực quan các thông tin liên quan tới tải hiện tải của từng nút cũng như thông lượng trao đổi thông tin giữa các nút.
- Module đánh giá hiệu năng hệ thống cho phép đánh giá năng lực tính toán của hệ thống.



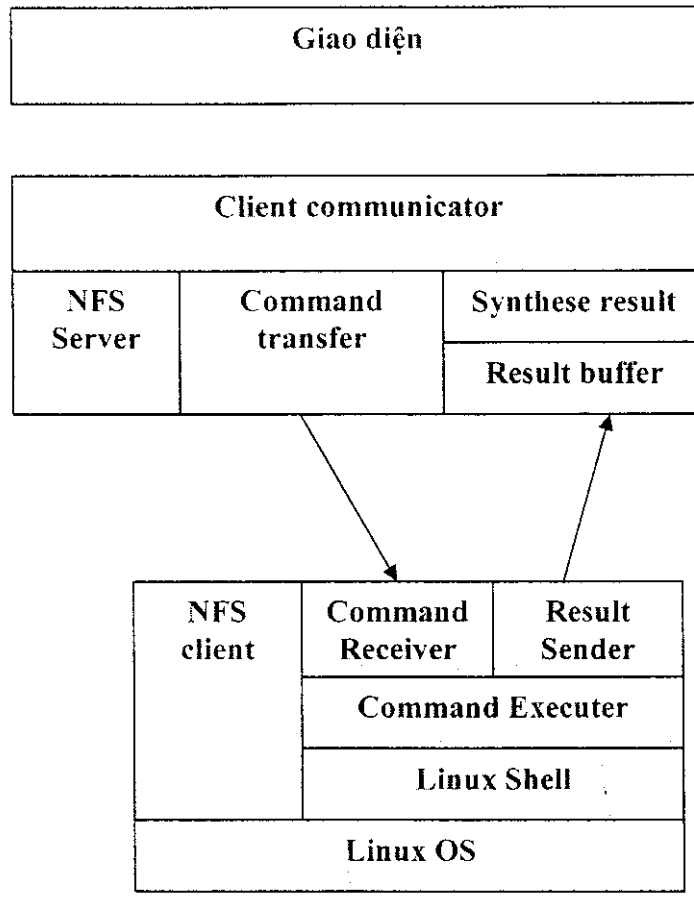
Hình 5-1 Hệ thống giám sát và quản trị

5.1 Module quản lý các gói phần mềm

Module quản lý các gói phần mềm gồm hai thành phần chính

- bkcluster-package-manager chạy trên máy chủ, cung cấp giao diện tương tác cho người sử dụng. Thành phần này có nhiệm vụ nhận lệnh từ người sử dụng, rồi truyền yêu cầu đến các iserver trên các máy trạm để thực thi.
- iserver chạy trên các máy trạm, có vai trò nhận lệnh từ máy chủ, thực thi

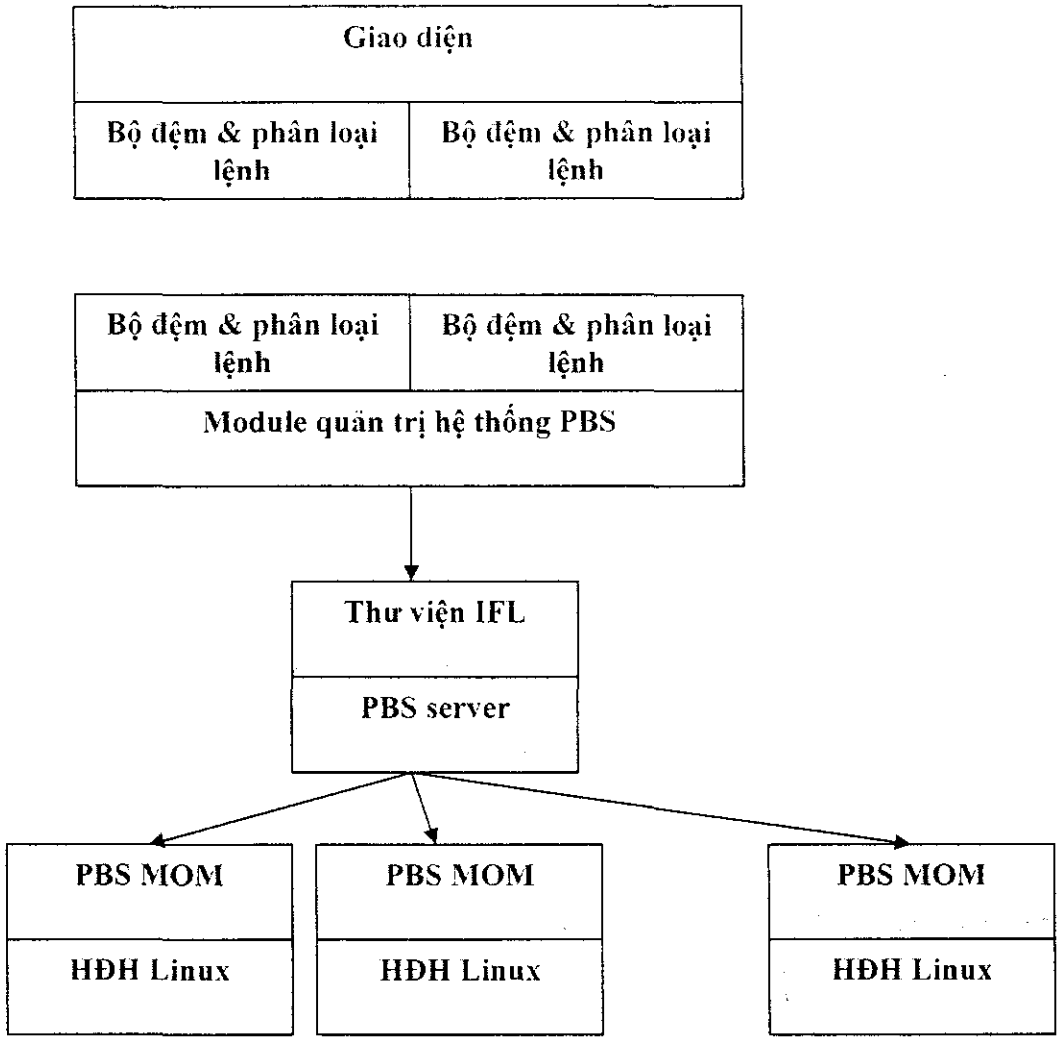
các yêu cầu tương ứng, rồi truyền kết quả thực hiện lại cho máy chủ hiển thị. iserver phải được thực thi ở chế độ nền, được gọi ngay khi các máy trạm khởi động, sẵn sàng nhận yêu cầu từ máy chủ.



Hình 5-2 Kiến trúc module quản lý gói phần mềm

5.2 Module quản lý cấu hình cluster

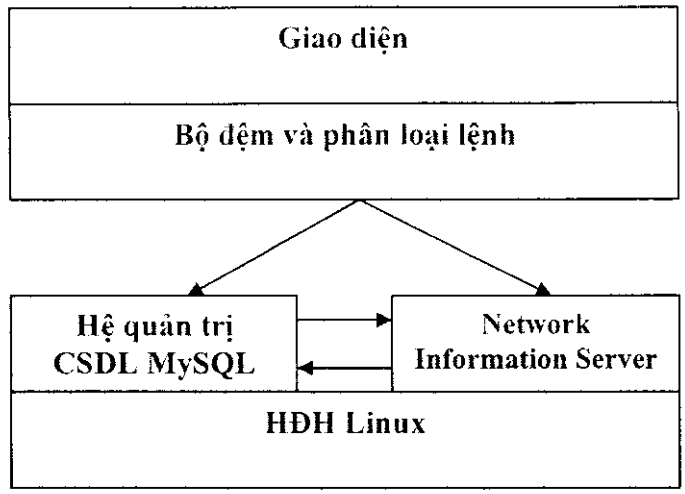
Module quản lý cấu hình của cluster được xây dưng dựa trên PBS, thực hiện việc quản lý: số nút tính toán , thực trạng từng nút tính toán, cấu hình PBS server.



Hình 5-3 Kiến trúc module quản lý cluster

5.3 Module quản lý người dùng

Một trong những nhiệm vụ chính của công cụ quản trị là quản lý người dùng tách biệt khỏi hệ điều hành để tránh các nguy cơ an toàn thông tin(quản lý người dùng chung với cơ sở dữ liệu người dùng của hệ điều hành là cách quản lý của hầu hết các hệ thống tính toán song song phân cụm), để đảm bảo quản lý người dùng hiệu quả người dùng hệ thống được phân ra làm ba lớp người dùng với những quyền tương ứng: người quản trị, thành viên và khách.



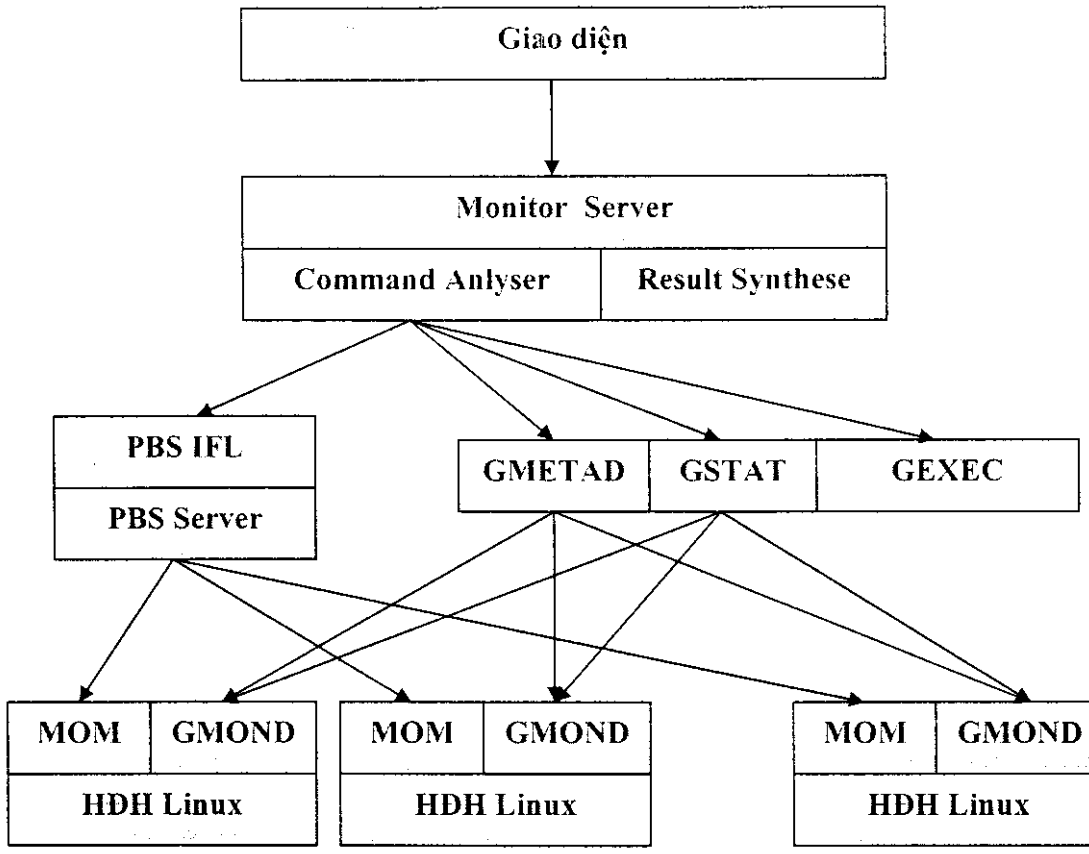
Hình 5-4 Kiến trúc module quản lý người dùng

5.4 Module giám sát hoạt động hệ thống

Đối với một hệ thống tính toán, thực trạng khai thác hệ thống được thể hiện qua tải hệ thống và thực trạng năng lực tính toán hiện tại. Đối với BKluster, chúng tôi quan tâm tới các yếu tố sau:

- Số lượng công việc đang được thực thi và trong hàng đợi
- Sử dụng tài nguyên tính toán trên các nút tính toán
- Thông lượng trao đổi thông tin giữa các nút

Chúng tôi sử dụng PBS và Ganglia để có thể nắm bắt được đầy đủ thông tin liên quan tới 3 vấn đề trên. Các thông tin này được cập nhật liên tục theo một chu kỳ định trước và hiển thị một cách trực quan dưới các dạng biểu đồ.



Hình 5-5 Kiến trúc module giám sát hệ thống

5.5 Module đánh giá hiệu năng hệ thống

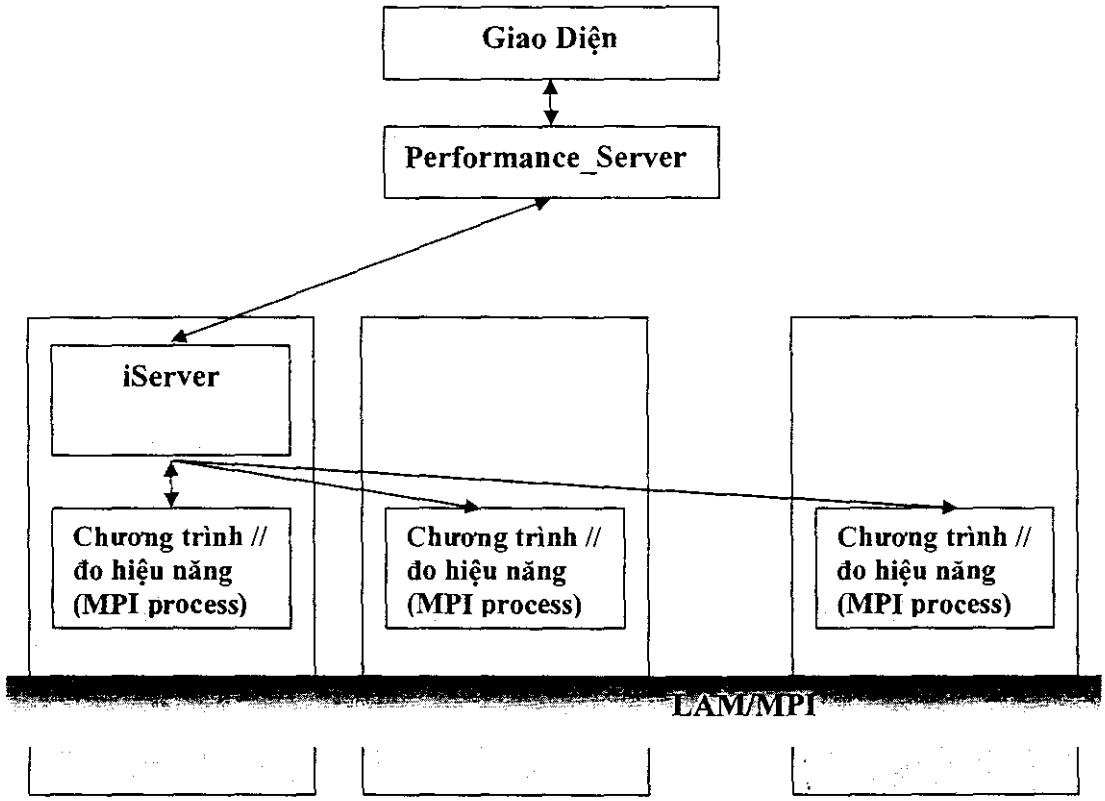
Năng lực tính toán của một hệ thống phân cụm được cấu thành từ nhiều yếu tố, trong đó, quan trọng nhất là năng lực tính toán trên từng nút và khả năng trao đổi thông tin của hệ thống kết nối. Năng lực tính toán này cũng được lượng hóa theo đơn vị số phép tính trên một giây: FLOPS, GFLOPS, ...Thực tế sử dụng, khai thác hệ thống, chúng tôi nhận thấy tác động rất lớn của thông lượng hệ thống kết nối các nút đối với năng lực tính toán. Do vậy, ngoài công cụ cho phép lượng hóa năng lực tính toán của hệ thống, BKluster cũng có công cụ cho phép đánh giá năng lực truyền thông trong hệ thống.

5.5.1 Thành phần đánh giá năng lực tính toán của hệ thống

Công cụ đánh giá năng lực tính toán của hệ thống cho phép biên dịch và chạy một số chương trình đánh giá hiệu năng thông dụng trên Thế Giới như: xHPL, NPB,

NETPIPE, ... trên một số nút tính toán xác định. Các chương trình đánh giá hiệu năng tính toán đều có dạng một chương trình song song, thực hiện một số lượng lớn phép toán số thực dấu phẩy động, có khả năng biên dịch theo một số chuẩn thông dụng, đặc biệt là chuẩn mpi.

Cấu trúc của công cụ đánh giá hiệu năng tính toán được mô tả trong hình vẽ sau:



Hình 5-6 Kiến trúc thành phần đánh giá hiệu năng tính toán hệ thống

Quá trình đánh giá hiệu năng tính toán được thực hiện qua những bước như sau:

- Người quản trị thông qua thành phần giao diện xác định các thông số như: tên phần mềm đo hiệu năng, tổng số nút trong cụm cần đo hiệu năng, ...
- Performance_Server gửi các thông số trên đến iServer trên một nút bất kỳ trông số tổng các nút cần đánh giá hiệu năng
- iServer thực hiện các việc sau:
  - Biên dịch chương trình đánh giá hiệu năng (Source code và file thực

thi sau khi biên dịch của các chương trình đánh giá hiệu năng được đặt trên bộ nhớ dùng chung của BKluster)

- Chạy chương trình đánh giá hiệu năng dưới dạng một chương trình song song viết theo chuẩn MPI trong môi trường LAM
- Gửi kết quả về Performance_Server

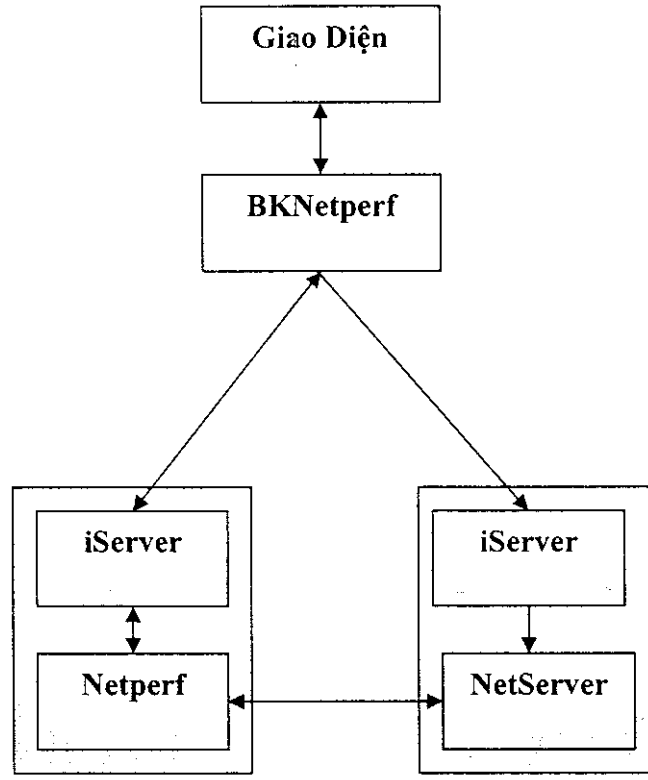
5.5.2 Thành phần đánh giá năng lực truyền thông trong hệ thống

Công cụ đánh giá năng lực truyền thông hệ thống được phát triển dựa trên phần mềm mã nguồn mở Netperf. Netperf bao gồm 2 module Netserver và Netperf chạy trên 2 máy tính khác nhau. Module Netperf sẽ gửi các gói tin đến module Netserver, dựa vào kết quả phản hồi của Netserver sẽ tính được tốc độ truyền thông trong mạng nối giữa hai máy tính.

Cấu trúc công cụ đánh giá năng lực truyền thông được mô tả trong hình vẽ sau. Các module Netperf, NetServer và iServer đã được cài đặt sẵn trên mọi node tính toán

Quá trình đánh giá năng lực truyền thông được thực hiện qua những bước như sau:

- Người quản trị thông qua thành phần giao diện xác định các thông số như: kiểu gói tin, kích thước gói tin, tên máy nguồn, tên máy đích, ...
- BKNetperf gửi các thông số trên đến iServer trên máy nguồn và máy đích,
- iServer trên máy đích khởi động chương trình NetServer
- iServer trên máy nguồn gọi chương trình Netperf, thực hiện truyền tin theo các thông số đã xác định
- iServer trên máy nguồn trả kết quả về cho BKNetperf



Hình 5-7 Thành phần đánh giá năng lực truyền thông hệ thống

Tài liệu tham khảo

[1]. **Thomas Sterling**, *"An Introduction to PC Clusters for High Performance Computing", California Institute of Technology and NASA Jet Propulsion Laboratory, USA*

[2]. **Albeaus Bayucan Robert L. Henderson Casimir LesiakBhroam Mann Tom ProettDave Tweten**, "Portable Batch System External Reference Specification". *MRJ Technology Solutions*

[3]. **Albeaus Bayucan Robert L. Henderson Casimir LesiakBhroam Mann Tom ProettDave Tweten**, "Portable Batch System Internal Design Specification". *MRJ Technology Solutions*

[4]. **Albeaus Bayucan Robert L. Henderson Casimir LesiakBhroam Mann Tom ProettDave Tweten,** "Portable Batch System Administrator Guide". *MRJ Technology Solutions*

[5]. **Albeaus Bayucan Robert L. Henderson Casimir LesiakBhroam Mann Tom ProettDave Tweten,** "Portable Batch System user guide". *MRJ Technology Solutions*

[6]. **PACS Training Group**, "Introduction to MPI", *University of Illinois.*

[7]. LAM/MPI Forum, "www.lam-mpi.org ".

[8]. **The LAM/MPI Team**, "LAM/MPI Documents", *Open Source Lab*

[9]. **Nicolas J. Nevin**, "The XMPI API and trace file format", *January 29, 1997.*

[10]. "UC Berkeley Grid" *http://www.millennium.berkeley.edu/ganglia/*