Ứng dụng màn hình chờ

Giảng viên hướng dẫn: TS. Trần Việt Trung

Thành viên thực hiện:

- Phạm Hoàng Đạt - 20101358
- Kiều Đức Vũ - 20102576
- Lê Xuân Hải - 20106092
- Nguyễn Hưng - 20101673

Nội dung trình bày

1. Giới thiệu đề tài và phân chia công việc
2. Công nghệ sử dụng - RMI
3. Thiết kế và triển khai
4. Kết quả
5. Demo chương trình
6. Hỏi đáp

GiỚI THIỆU ĐỀ TÀI (1)

1. Ứng dụng Xuyên màn hình
2. Yêu cầu:
    1. Chương trình được viết bằng ngôn ngữ Java
    2. Xử dụng Frame work : Game Frame work
    3. Xử dụng lập trình phân tán RMI
    4. Xử dụng Canvas để thiết kế màn hình

GiỚI THIỆU ĐỀ TÀI (2)

3. Yêu cầu chức năng:

- Các đối tượng trên màn hình có thể di chuyển giữa các máy khác nhau.
- Quản lý các máy với các kích thước màn hình khác nhau
- Có thể chọn nhiều background khác nhau
- Chức năng signin và signout vào hệ thống của từng máy

GiỚI THIỆU ĐỀ TÀI (3)

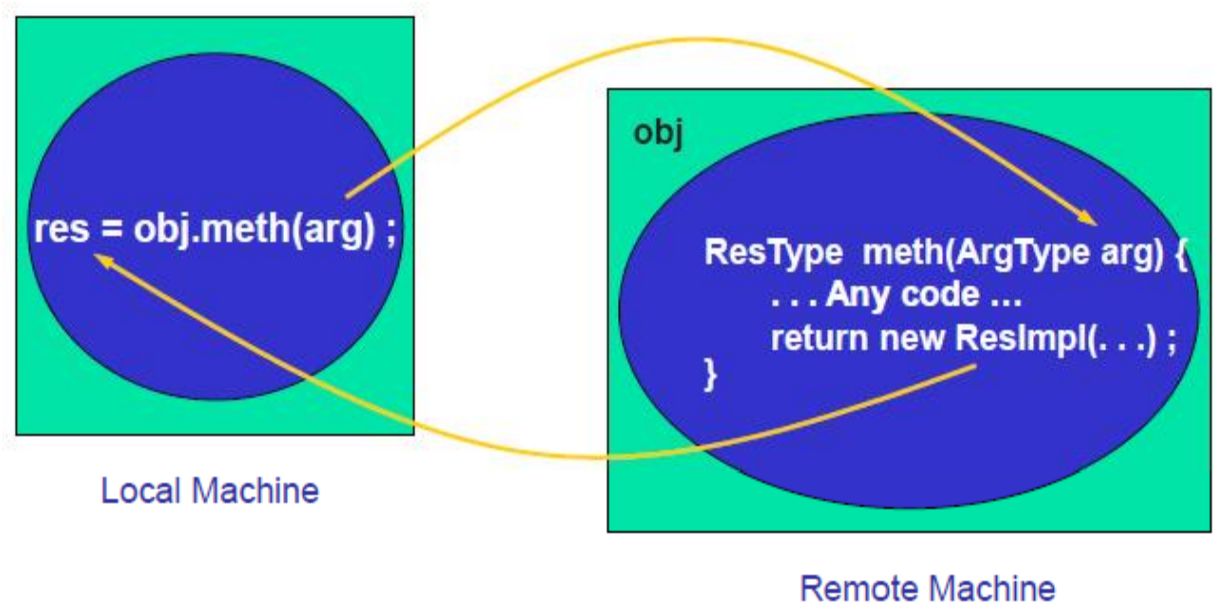
Thành viên	Công việc - Tìm hiểu
Phạm Hoàng Đạt	Giao tiếp RMI – Server – trao đổi
Lê Xuân Hải	TK giao diện – Server
Kiều Đức Vũ	Client – Nhân vật
Nguyễn Ngọc Hưng	Client – Giao diện lựa chọn chính

Giới thiệu RMI

1. RMI là gì?

2. RMI hoạt động như thế nào?

RMI là gì?


res = obj.meth(arg) ;
obj
ResType meth(ArgType arg) {
. . . Any code ...
return new ResImpl(. . .) ;
}
Local Machine
Remote Machine

Cơ chế của RMI(1)

Server:

- RMI server phải đăng kí với một **lookup service** để các client có thể tìm thấy nó.
- Nền tảng Java cung cấp một ứng dụng **rmiregistry**, chạy ở một tiến trình độc lập, cho phép đăng kí dịch vụ RMI.
- Ngay khi server được đăng kí, nó sẽ chờ các request đến từ phía các client.
- Đi cùng với một đăng kí dịch vụ là một chuỗi tên cho các client phân biệt được các dịch vụ RMI
- Các client có thể tham chiếu đến các dịch vụ trực tiếp thông qua **registry**

Cơ chế của RMI(2)

Client:

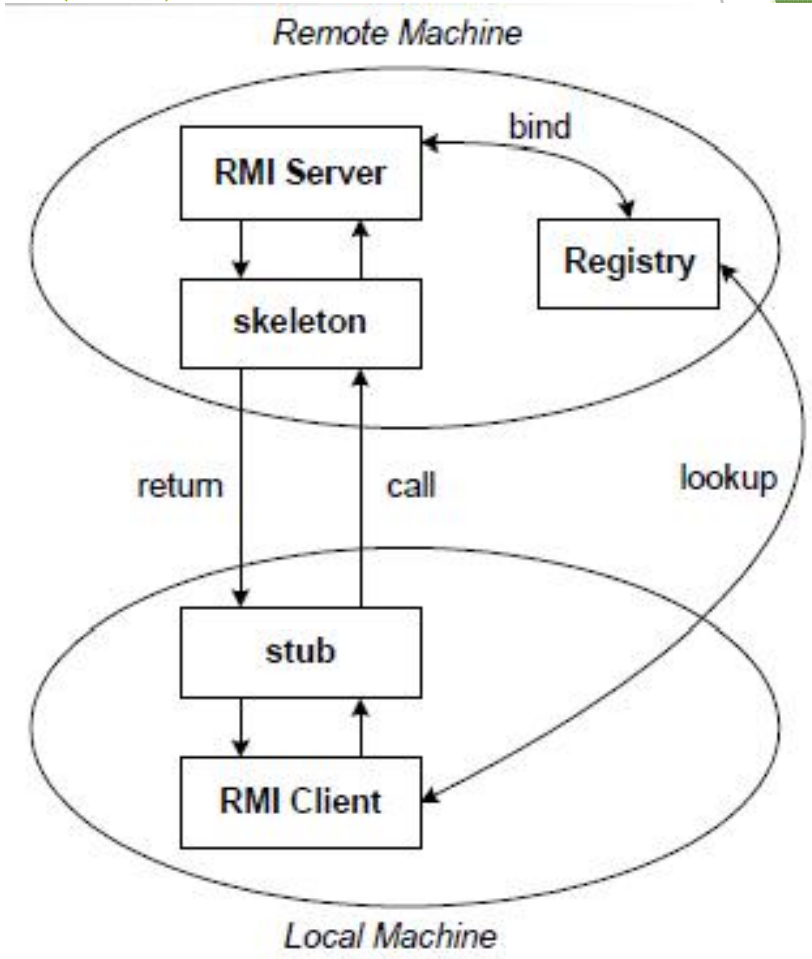
Client muốn sử dụng được các đối tượng từ xa thì phải **có một tham chiếu** đến đối tượng đó.

Đối tượng đó sẽ có được thông qua **RMI registry**.

Các client request đến một dịch vụ qua một **URL** có dạng

`rmi://hostname:port/servicename`

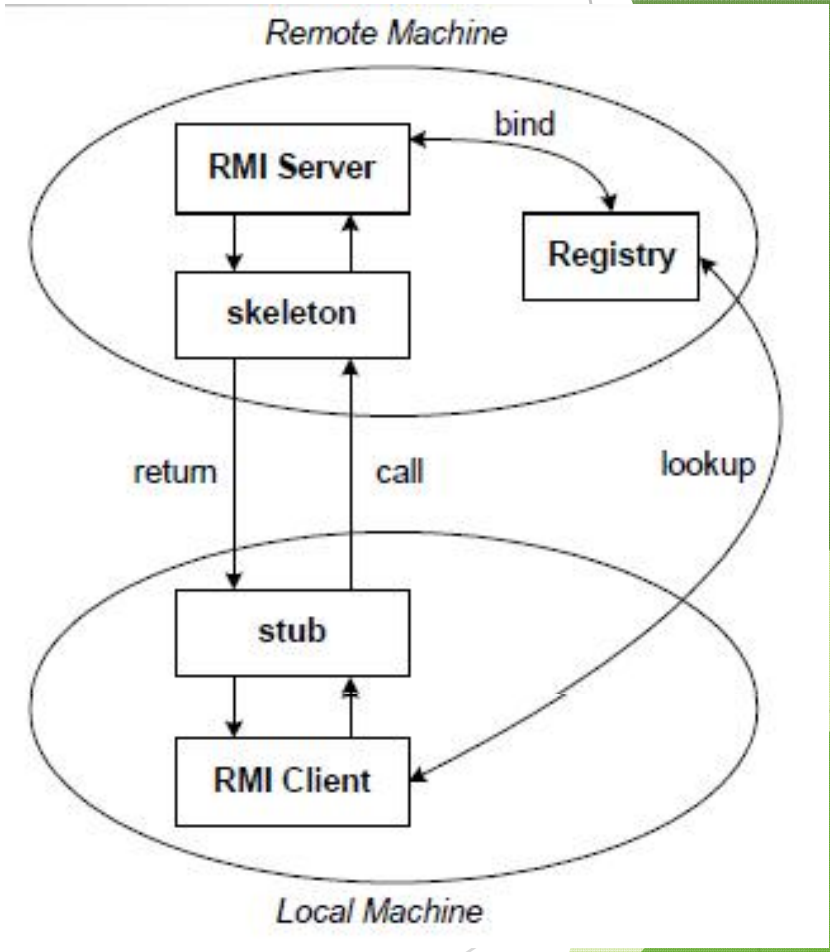
Sau khi đa nhận được đối tượng tham chiếu từ xa. Client có thể thực hiện **thao tác** với đối tượng đó như thể **đối tượng đó ở cục bộ**.



Cơ chế của RMI(3)

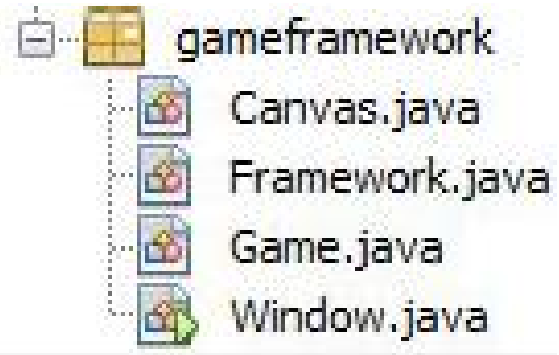
Skeleton và Stub:

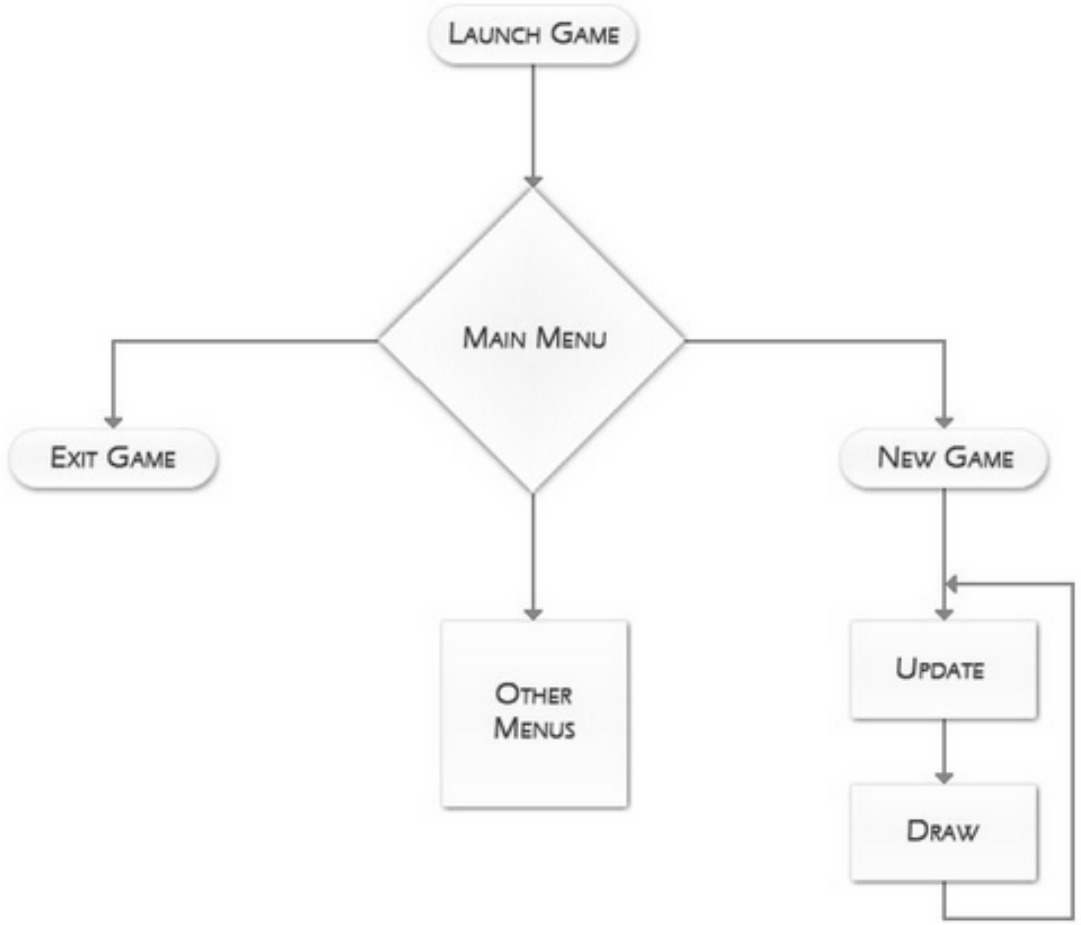
- Stub đóng vai trò như một proxy truyền tiếp các request tới RMI.
- Skeleton: skeleton triệu gọi phương thức và truyền kết quả lại cho Stub
- Chỉ là các interface, không cung cấp các cài đặt.
- Truyền thông giữa Skeleton và Stub là TCP sockets.
- Các tham số được truyền giữa Stub và Skeleton: các dữ liệu nguyên thủy và các đối tượng serializable.



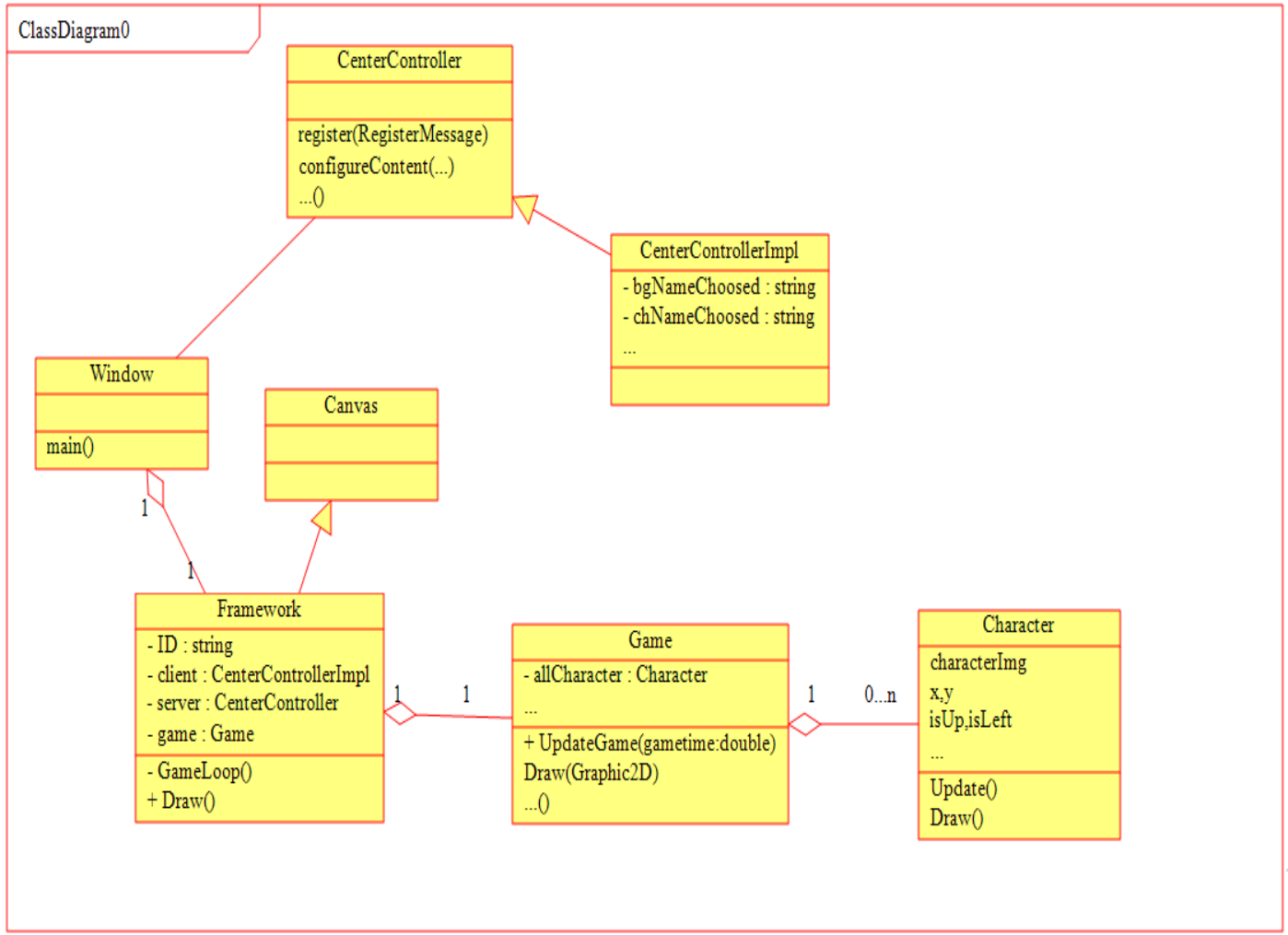
THIẾT KẾ CHƯƠNG TRÌNH

Framework

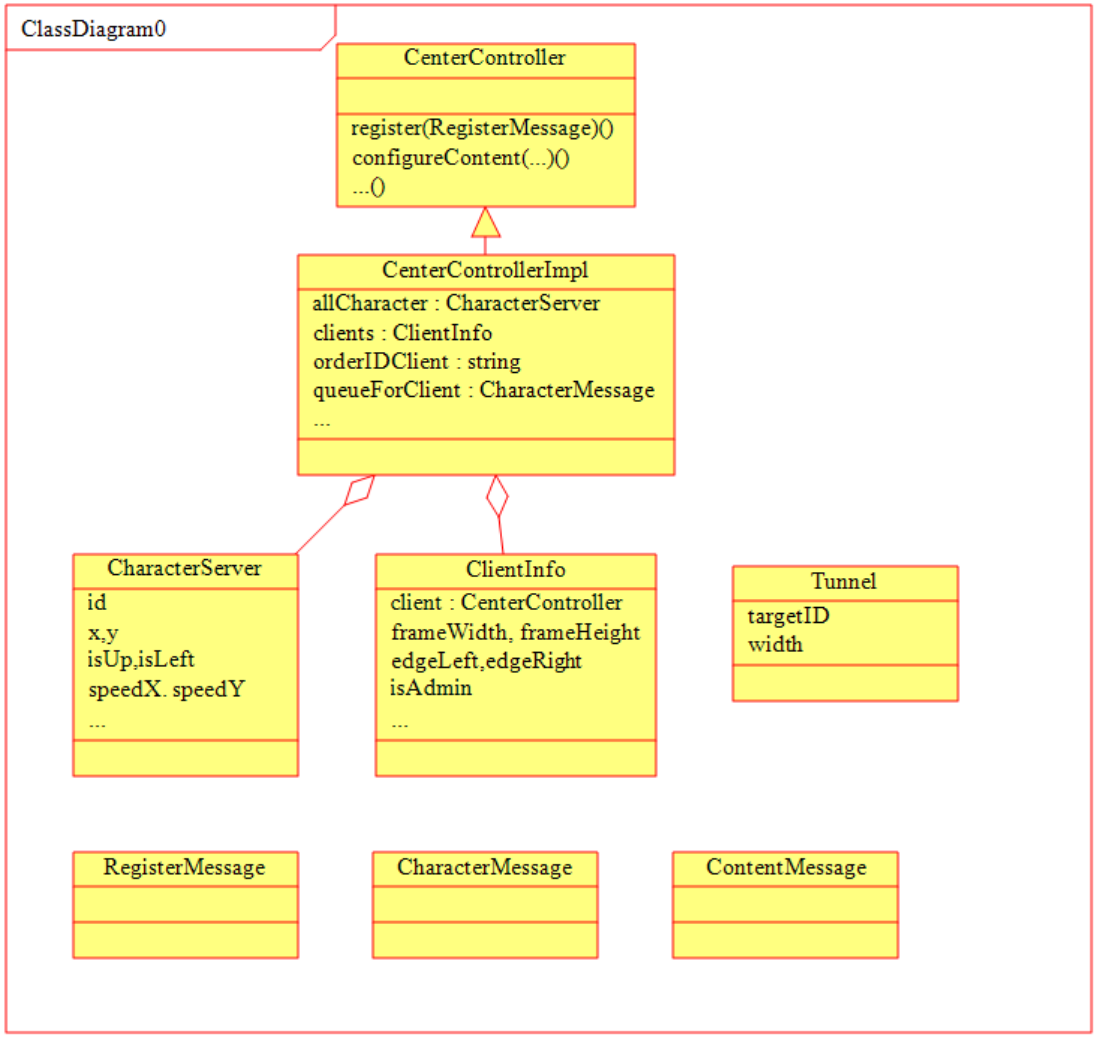




THIẾT KẾ CHƯƠNG TRÌNH

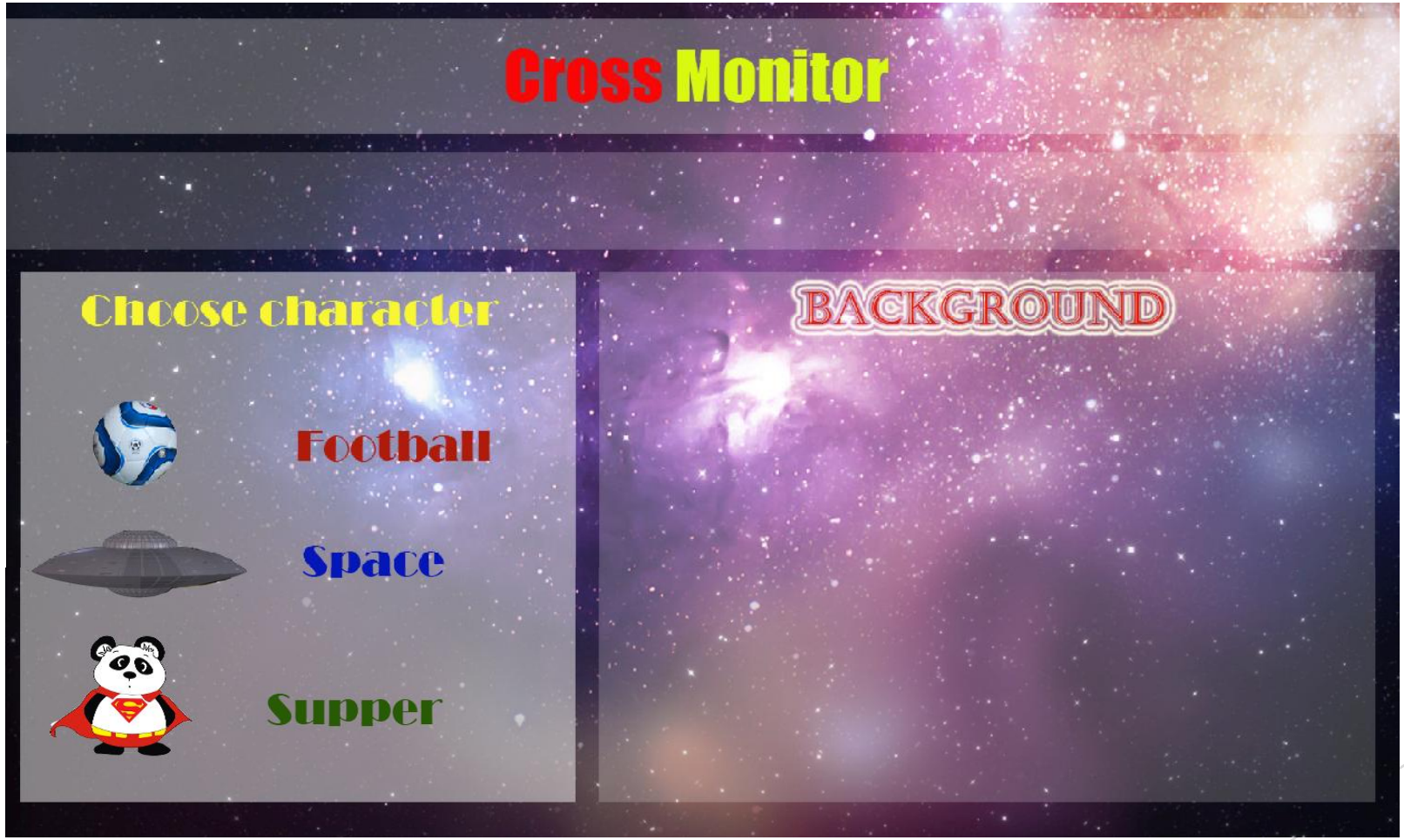


THIẾT KẾ CHƯƠNG TRÌNH



THIẾT KẾ GIAO DiỆN(1)

V1.0 – Màn hình Cài đặt



THIẾT KẾ GIAO DiỆN(2)

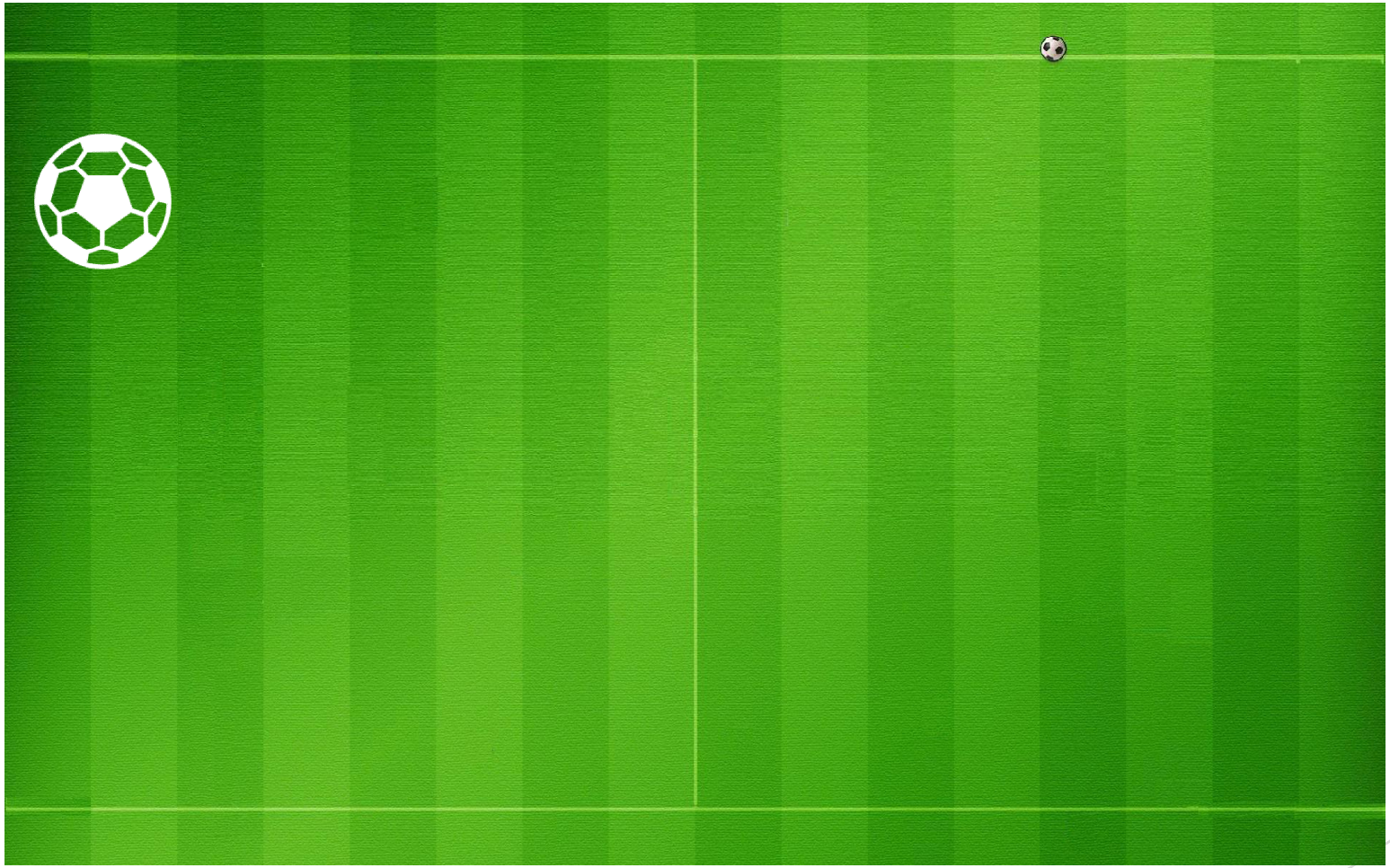
V2.1 – màn hình cài đặt

THIẾT KẾ GIAO DIỆN(3)

màn hình của các máy Client

THIẾT KẾ GIAO DIỆN(4)

màn hình của các máy Client



Kết quả đạt được

Đã đạt được

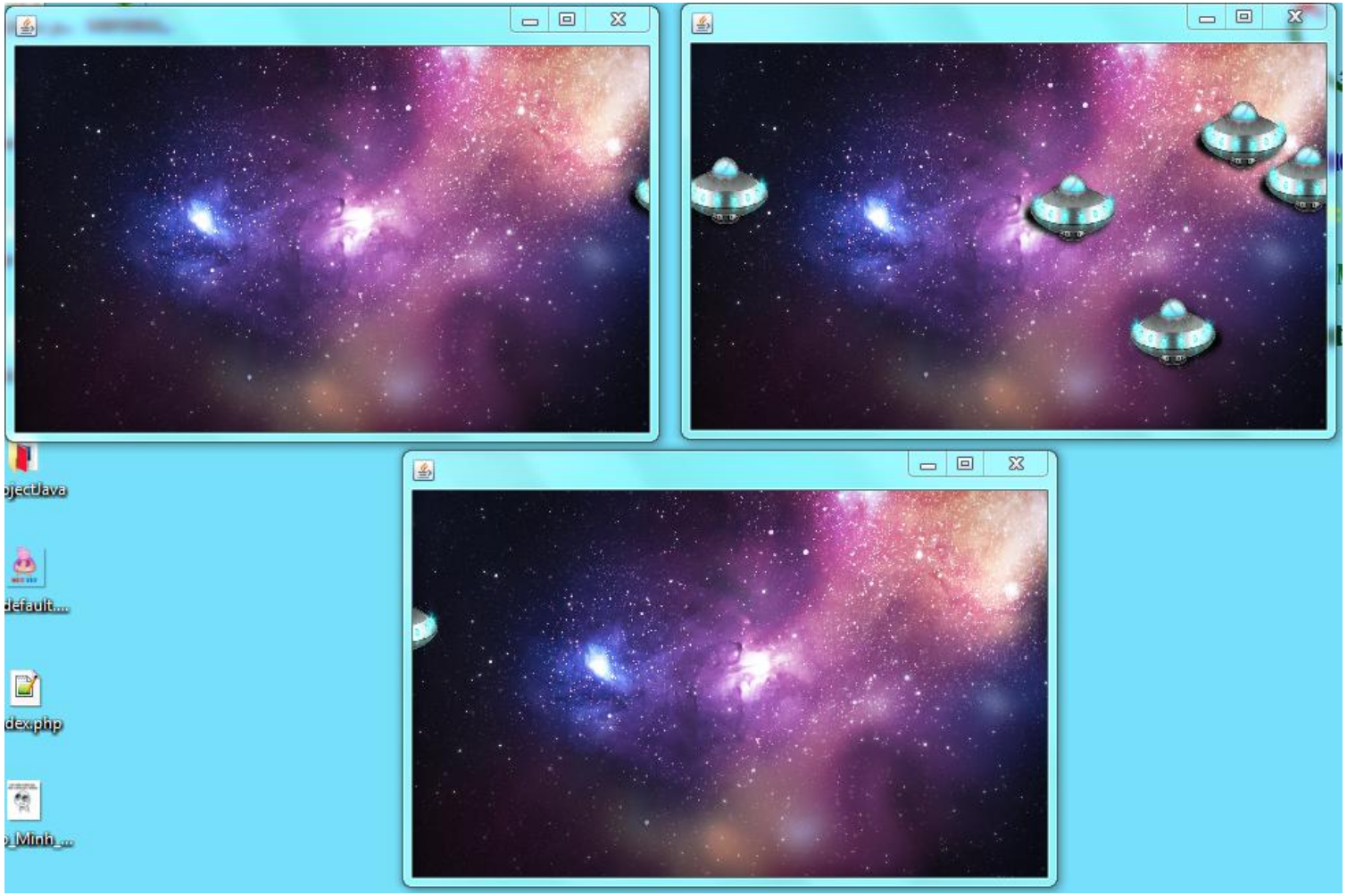
- Chương trình đã chạy thành công đúng với chức năng đặt ra
- Cài đặt được giao thức RMI
- Các thành viên trong nhóm cùng hỗ trợ nhau làm việc, nâng cao được kỹ năng trong việc thực hành làm việc với ngôn ngữ Java

Chưa đạt được

- Chương trình vẫn còn đơn giản, chưa hoàn thiện
- Chưa sử dụng hết khả năng của RMI (sử dụng RMI callback)
- Chưa mô phỏng được chạy trên các máy vật lý độc lập

Demo chương trình(1)

Demo chương trình(2)



Demo chương trình(3)

HỎI ĐÁP