Chapter-18:
Distributed Computing

Upcode Software
Engineer Team

-2023-

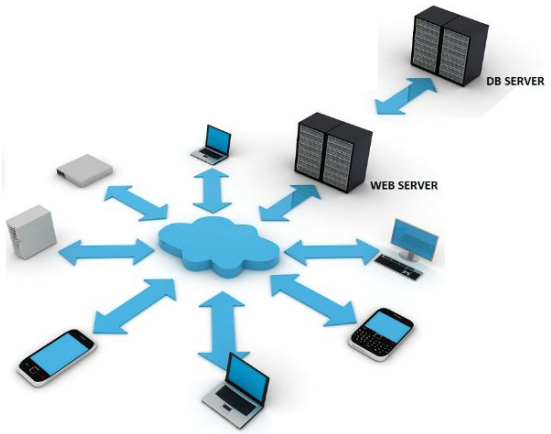
CONTENT

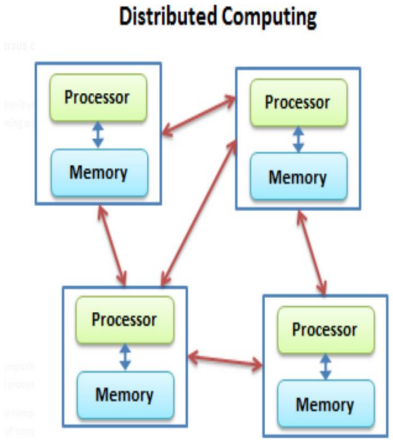
1. What is Distributed Computing ?
2. What is a RMI (Remote Method Invocation) ?
3. What is a Servlets ?
4. Source code
5. Used materials and references

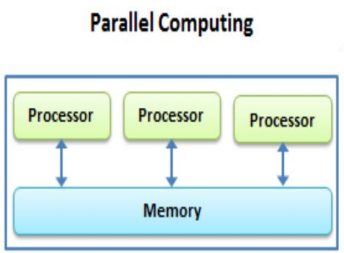
1. What is Distributed Computing ?

- **Distributed computing** is the concurrent usage of more than one connected computer to solve a problem over a network connection.

  There are several key components of a Distributed Computing System:



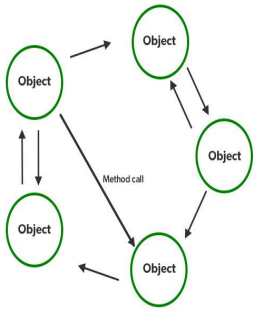


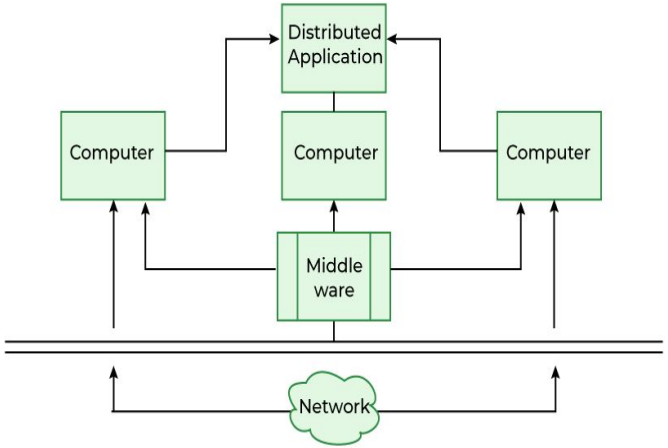


1. What is Distributed Computing ?

There are several key components of a Distributed Computing System:

- Devices or Systems
- Network
- Resource Management

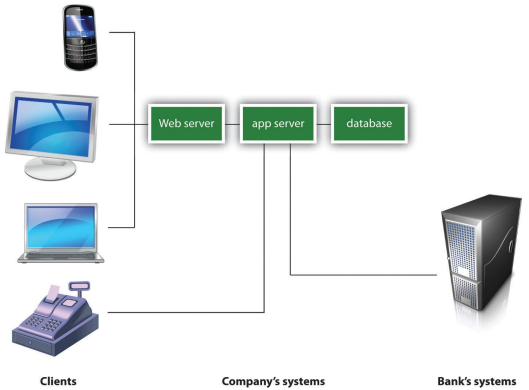


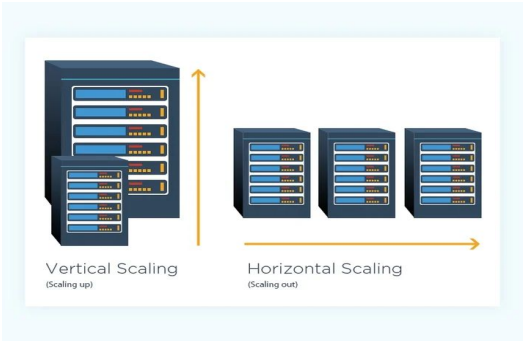


1. What is Distributed Computing ?

There are several characteristics that define a Distributed Computing System:

- Multiple Devices or Systems:
- Peer-to-Peer Architecture
- Shared Resources
- Horizontal Scaling





1. What is Distributed Computing ?

Advantages of the Distributed Computing System are:

- **Scalability**
- **Reliability**
- **Flexibility**

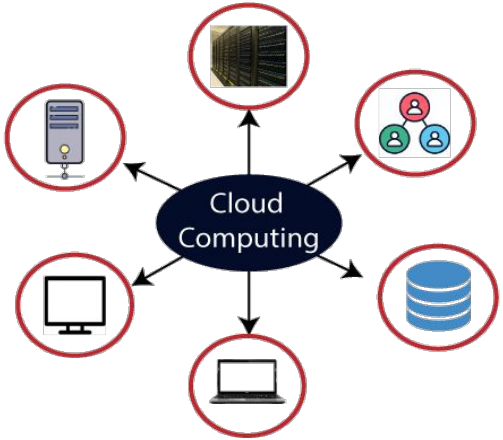
There are a few limitations to Distributed Computing System

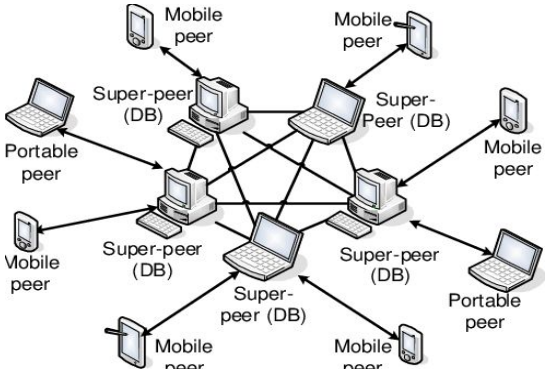
- **Complexity:**
- **Security:**
- **Performance:**

1. What is Distributed Computing ?

Distributed Computing Systems have a number of applications, including:

- **Cloud Computing**
- **Peer-to-Peer Networks**
- **Distributed Architectures**

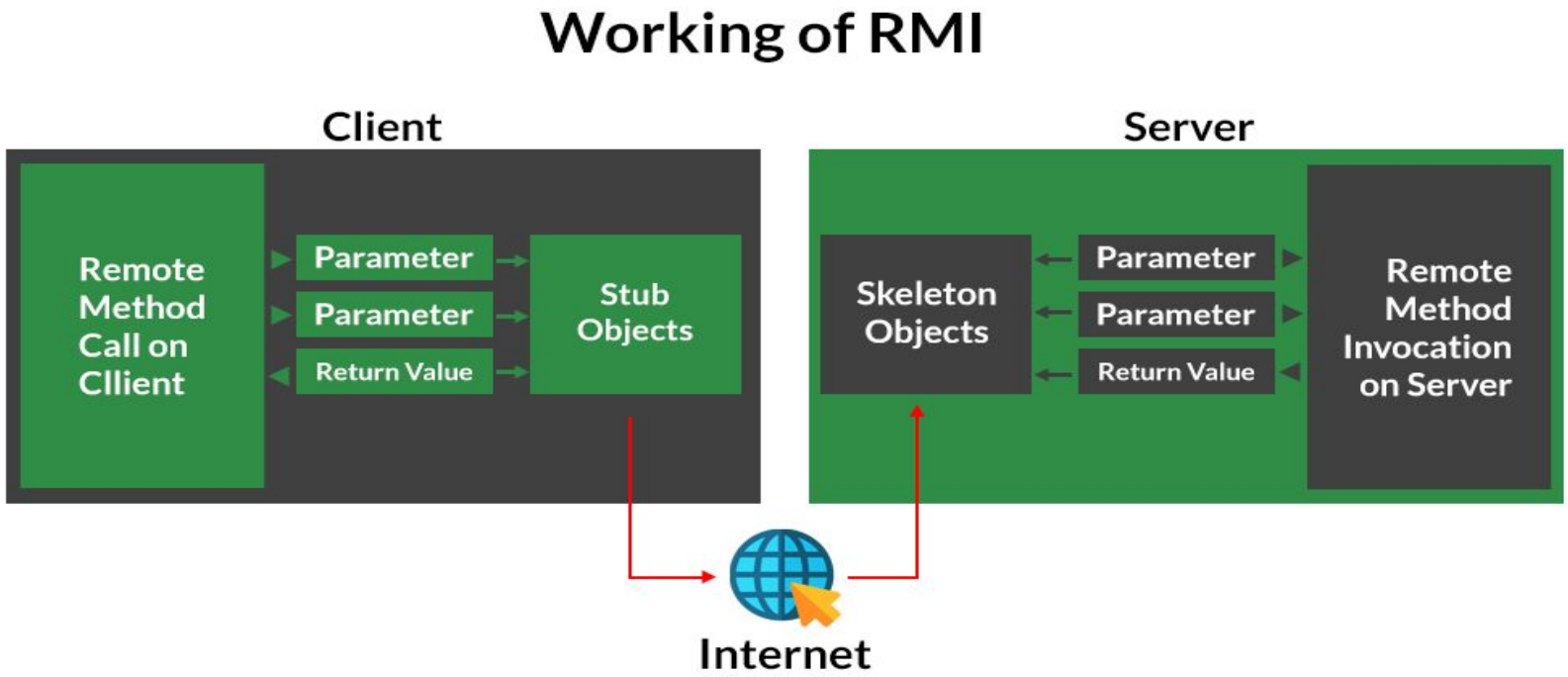




2. What is a RMI (Remote Method Invocation) ?

- **The RMI (Remote Method Invocation)** is an API that provides a mechanism to create distributed application in java.
- The RMI allows an object to invoke methods on an object running in another JVM.
- The RMI provides remote communication between the applications using two objects ***stub*** and ***skeleton***.
- RMI is used to build distributed applications;
- RMI provides remote communication between Java programs. It is provided in the package **java.rmi.**

2. What is a RMI (Remote Method Invocation) ?



2. What is a RMI (Remote Method Invocation) ?

The stub is an object, acts as a gateway for the client side.

All the outgoing requests are routed through it. It resides at the client side and represents the remote object. When the caller invokes method on the stub object, it does the following tasks:

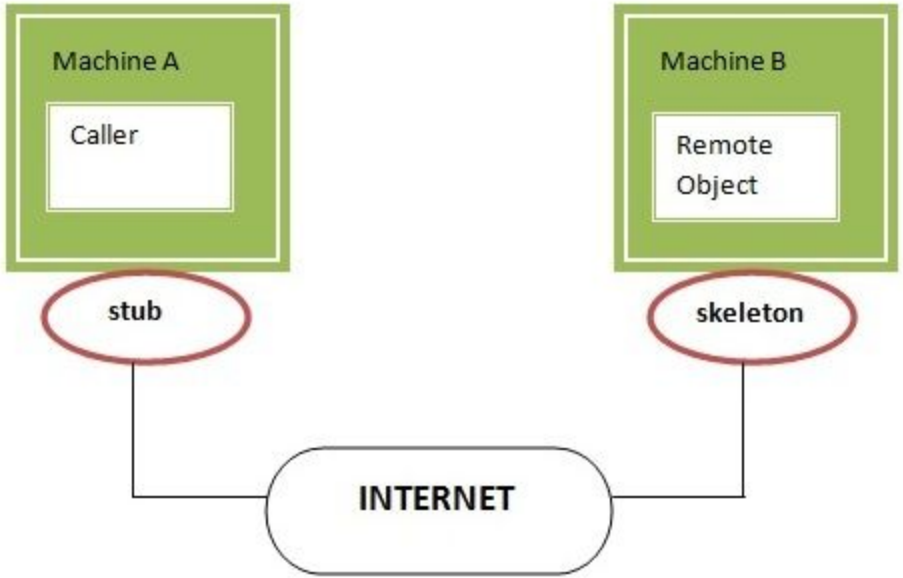
1. It initiates a connection with remote Virtual Machine (JVM),
2. It writes and transmits (marshals) the parameters to the remote Virtual Machine (JVM),
3. It waits for the result
4. It reads (unmarshals) the return value or exception, and
5. It finally, returns the value to the caller.

2. What is a RMI (Remote Method Invocation) ?

The skeleton is an object, acts as a gateway for the server side object. All the incoming requests are routed through it. When the skeleton receives the incoming request, it does the following tasks:

1. It reads the parameter for the remote method
2. It invokes the method on the actual remote object, and
3. It writes and transmits (marshals) the result to the caller.

2. What is a RMI (Remote Method Invocation) ?

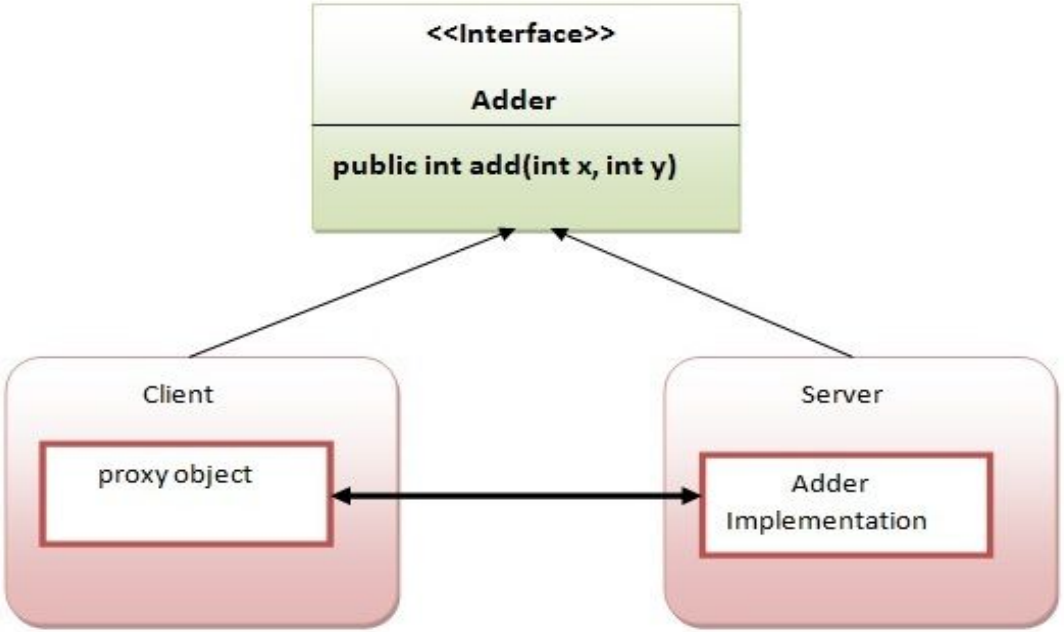


2. What is a RMI (Remote Method Invocation) ?

The is given the 6 steps to write the RMI program.

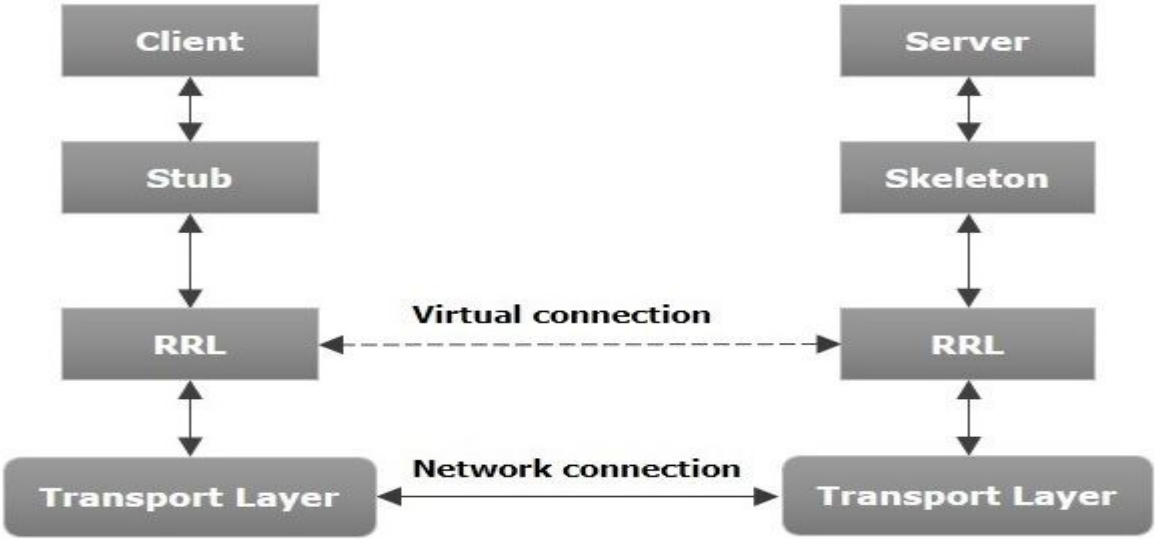
1. Create the remote interface
2. Provide the implementation of the remote interface
3. Compile the implementation class and create the stub and skeleton objects using the rmic tool
4. Start the registry service by rmiregistry tool
5. Create and start the remote application
6. Create and start the client application

2. What is a RMI (Remote Method Invocation) ?



2. What is a RMI (Remote Method Invocation) ?

The following diagram shows the **architecture of an RMI** application.



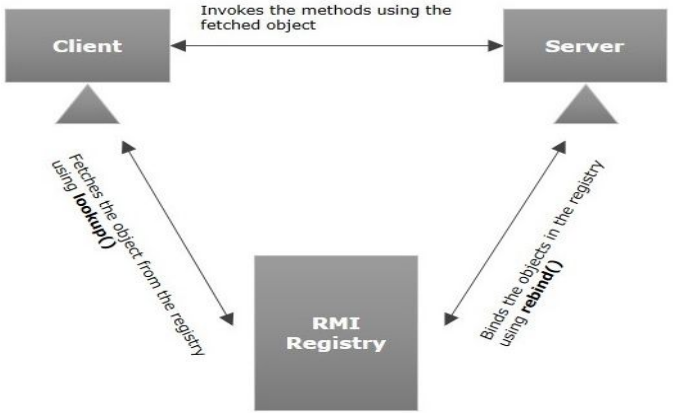
2. What is a RMI (Remote Method Invocation) ?

- **Transport Layer** – This layer connects the client and the server. It manages the existing connection and also sets up new connections.
- **Stub** – A stub is a representation (proxy) of the remote object at client. It resides in the client system; it acts as a gateway for the client program.
- **Skeleton** – This is the object which resides on the server side. stub communicates with this skeleton to pass request to the remote object.
- **RRL(Remote Reference Layer)** – It is the layer which manages the references made by the client to the remote object.

2. What is a RMI (Remote Method Invocation) ?

RMI registry is a namespace on which all server objects are placed. Each time the server creates an object, it registers this object with the RMIregistry (using **bind()** or **reBind()** methods). These are registered using a unique name known as **bind name**.

To invoke a remote object, the client needs a reference of that object. At that time, the client fetches the object from the registry using its bind name (using **lookup()** method).



3. What is a Servlets ?

Servlet is a server-side Java program module that handles client requests and implements the *servlet* interface.

3. What is a Servlets ?

- The entire life cycle of a servlet is managed by the *Servlet container* which uses the ***javax.servlet.***
- *Servlet* interface to understand the Servlet object and manage it.

  The Servlet life cycle mainly goes through four stages:
  - Loading a Servlet
  - Initializing the Servlet
  - Request handling
  - Destroying the servlet

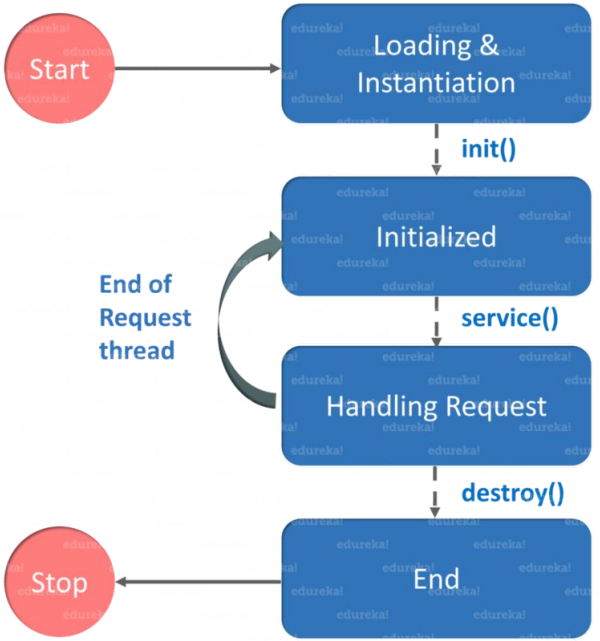
edureka!

http://

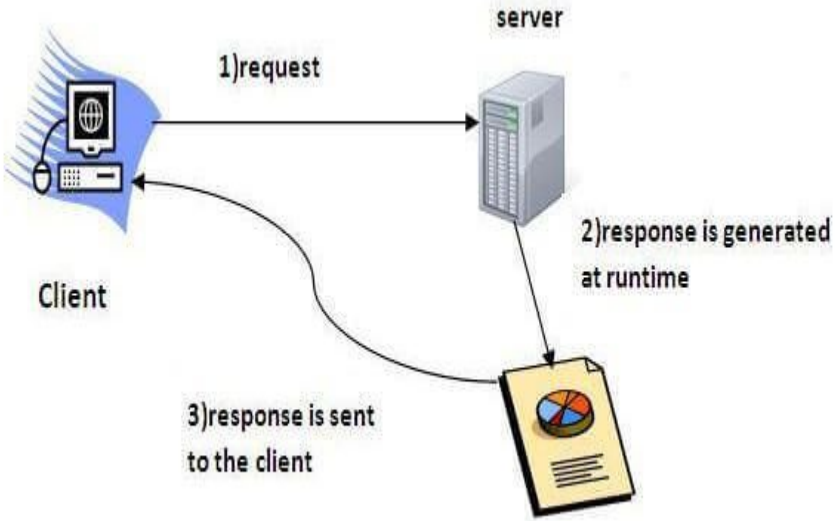
3. What is a Servlets ?

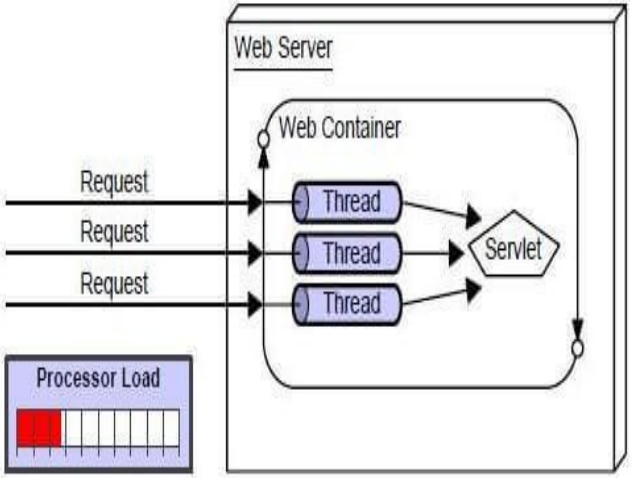
The Servlet life cycle mainly goes through four stages:

- Loading a Servlet
- Initializing the Servlet
- Request handling
- Destroying the servlet



3. What is a Servlets ?





3. What is a Servlets ?

There are many advantages of Servlet over CGI.
The web container creates threads for handling the multiple requests to the Servlet.
Threads have many benefits over the Processes such as they share a common memory area, lightweight, cost of communication between the threads are low.
The advantages of Servlet are as follows:

- **Better performance:** because it creates a thread for each request, not process.
- **Portability**: because it uses Java language.
- **Robust**: JVM manages Servlets, so we don't need to worry about the memory leak, garbage collection, etc.
- **Secure**: because it uses java language.

Reference Resources?

1. Head First JAVA (book)
2. Object and class in java in www.javatpoint.com
3. Classes and Objects in Java in www.geeksforgeeks.org
4. Difference between object and class in www.javatpoint.com
5. Difference Between Class and Object in OOPs in www.guru99.com

Thank you!

Presented by Sanjar