PROPOSAL DEFENSE

On

Java Virtual Machine (JVM) for Subset of Java

OVERVIEW

- Introduction
- Literature Review
- Methodology
- System Design
- Timeline

INTRODUCTION

BACKGROUND

- Runs program compiled to java bytecode.
- Platform-independent runtime environment for programs.
- Includes garbage collection and dynamic class loading.
- Allow to focus on application development.

WHY THIS PROJECT ?

OBJECTIVES

Features to be supported :

- primitive data types: int, float, boolean, character and such
- control flow and conditionals
- object oriented features: class, inheritance and method overriding
- static methods and variables
- exception handling and stack tracing
- garbage collection and memory management

OBJECTIVES

Things not planned to implement :

- Multithreading
- Reflection
- Annotation
- Just In Time (JIT) compilation

LITERATURE REVIEW

RELATED WORKS

Codename One

HotSpot

Eclipse OpenJ9

GraalVM

RELATED THEORY

- **Process Virtual Machines:**

  Emulate a complete system's runtime environment for processes.

- **Java Bytecode:**

  Intermediate code compiled from Java source.

- **Interpreter:**

  Executes Java bytecode instruction by instruction.

- **Garbage Collection:**

  Manages, reclaims, and optimizes memory allocation.

METHODOLOGY

FEASIBILITY STUDY

- Technically Feasible

- Economically Feasible

- Time Feasible

REQUIREMENT ANALYSIS

- **Functional Requirements :**

  -Class Loading

  -Bytecode Interpretation

  -Method Execution

  -Memory Management

  -Exception Handling

REQUIREMENT ANALYSIS

- **Non-Functional Requirements :**

  -Performance

  -Scalability

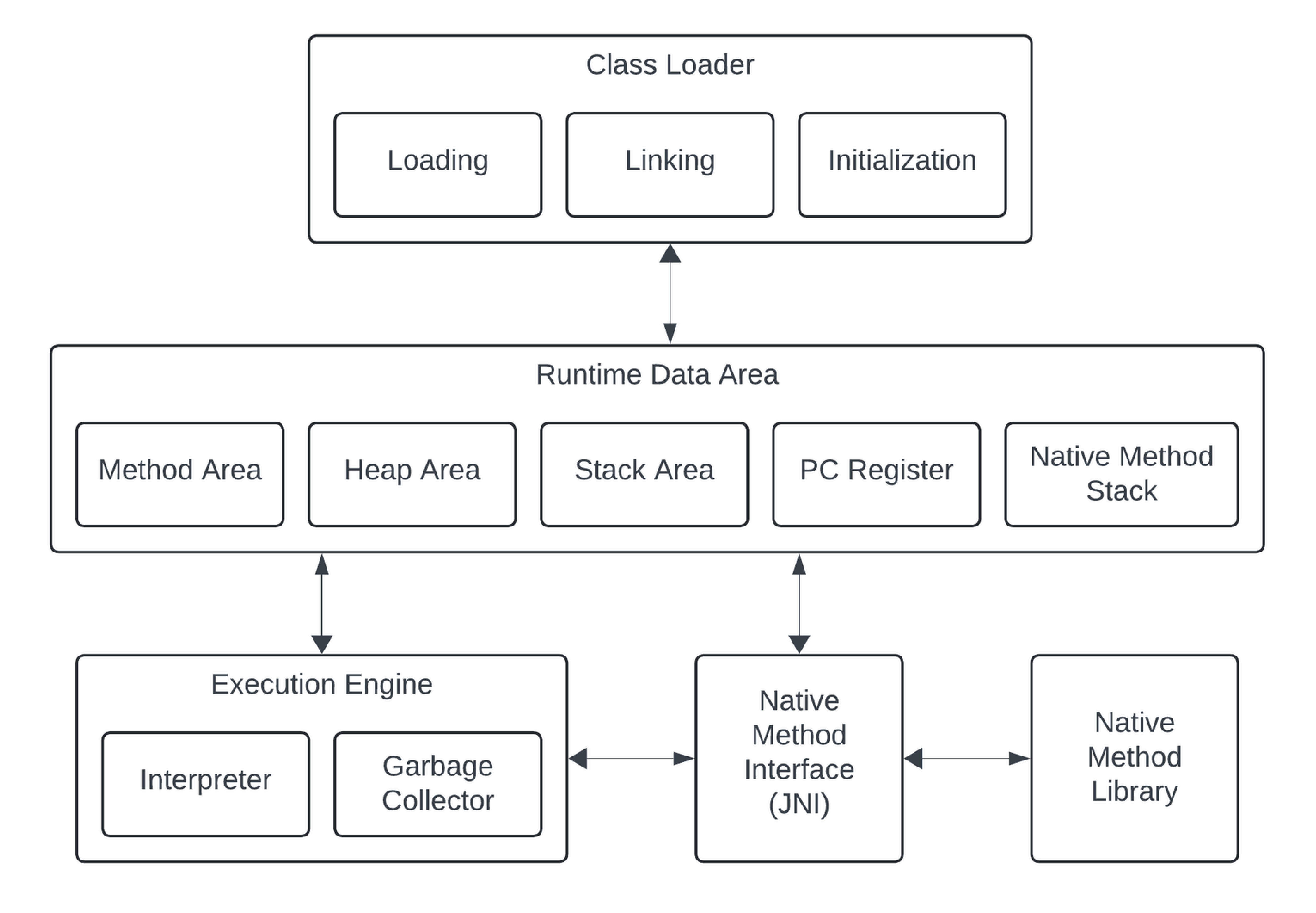
  -Portability

  -Extensibility

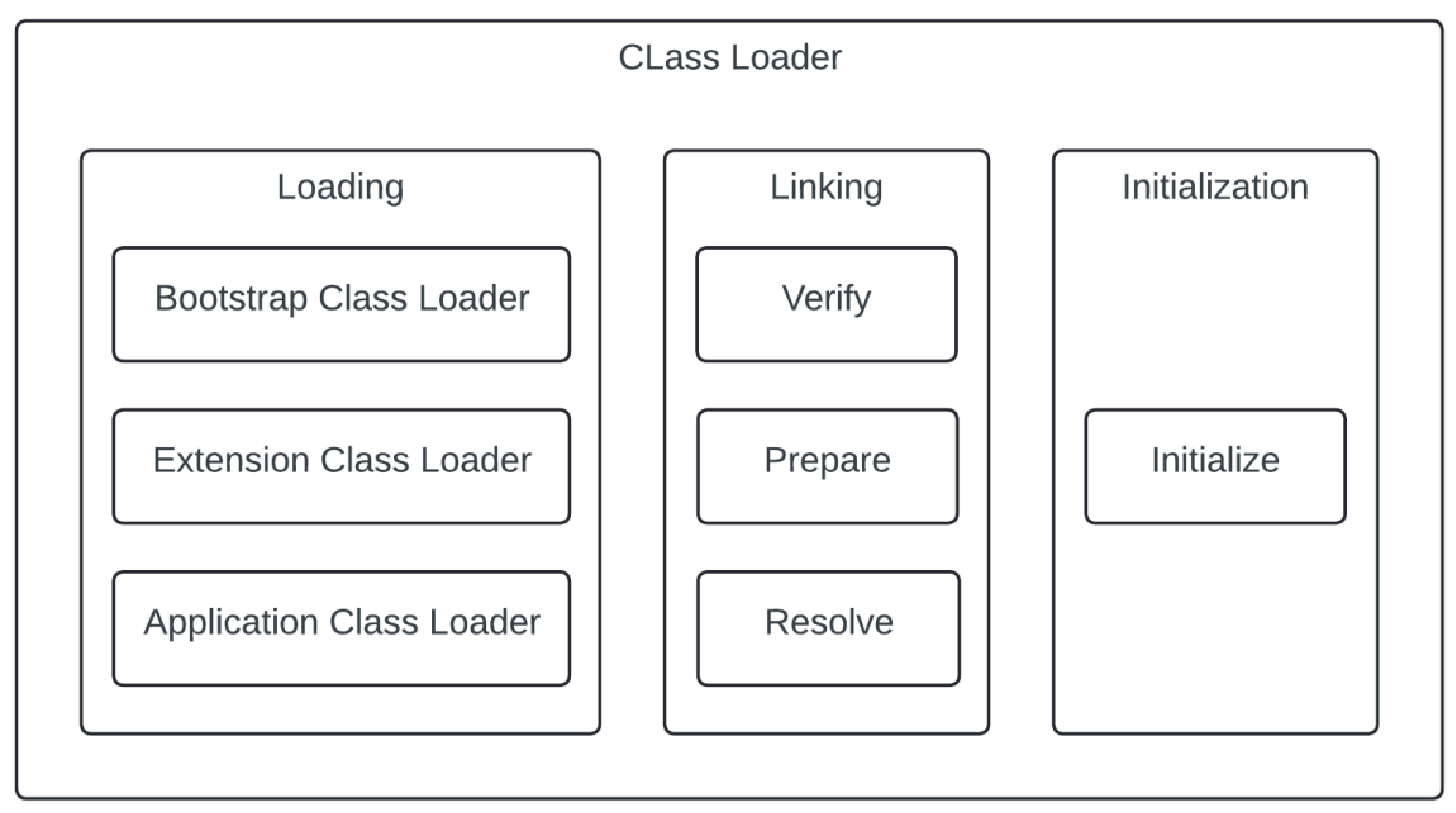
  -Maintainability

SYSTEM DESIGN

PROPOSED ARCHITECTURE



CLASS LOADER



CLass Loader

Loading

Bootstrap Class Loader

Extension Class Loader

Application Class Loader

Linking

Verify

Prepare

Resolve

Initialization

Initialize

- **Loading :**

takes the binary representation (bytecode)

-Bootstrap Class Loader

loads the standard Java packages (java.lang, java.net)

-Extension Class Loader

loads the extensions of standard Java libraries

-Application Class Loader

final class loader which loads the files present on the classpath.

- **Linking :**

  Combines the different elements and dependencies of the program

  -Verification

  Checks the structural correctness of the .class file

  -Preparation

  Allocates memory for the static fields and initializes the default values.

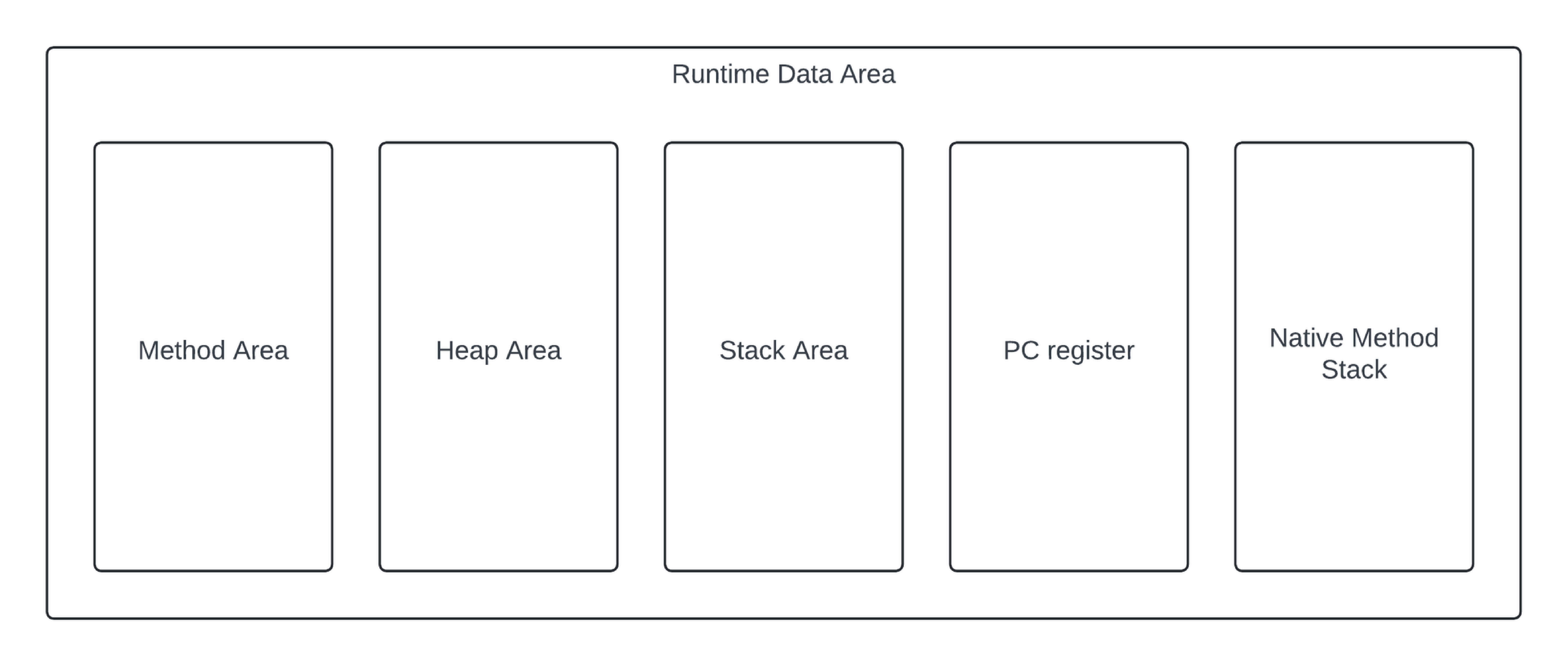
  -Resolution

  Symbolic references are replaced with direct references.

- **Initialization :**

  Involves executing the initialization method .

RUNTIME DATA AREA



Runtime Data Area

Method Area

Heap Area

Stack Area

PC register

Native Method Stack

- **Method Area**

  Holds class-level data and method information.

- **Heap Area**

  Holds objects and instance variables .

- **Stack Area**

  -Local variables

  Local variables with corresponding values are stored in an array.

  -Operand Stack

  LIFO operand stack for runtime intermediate operations.

  -Frame Data

  Holds method symbols and catch block information.

- PC Registers

  -Holds the address of the currently executing instruction.

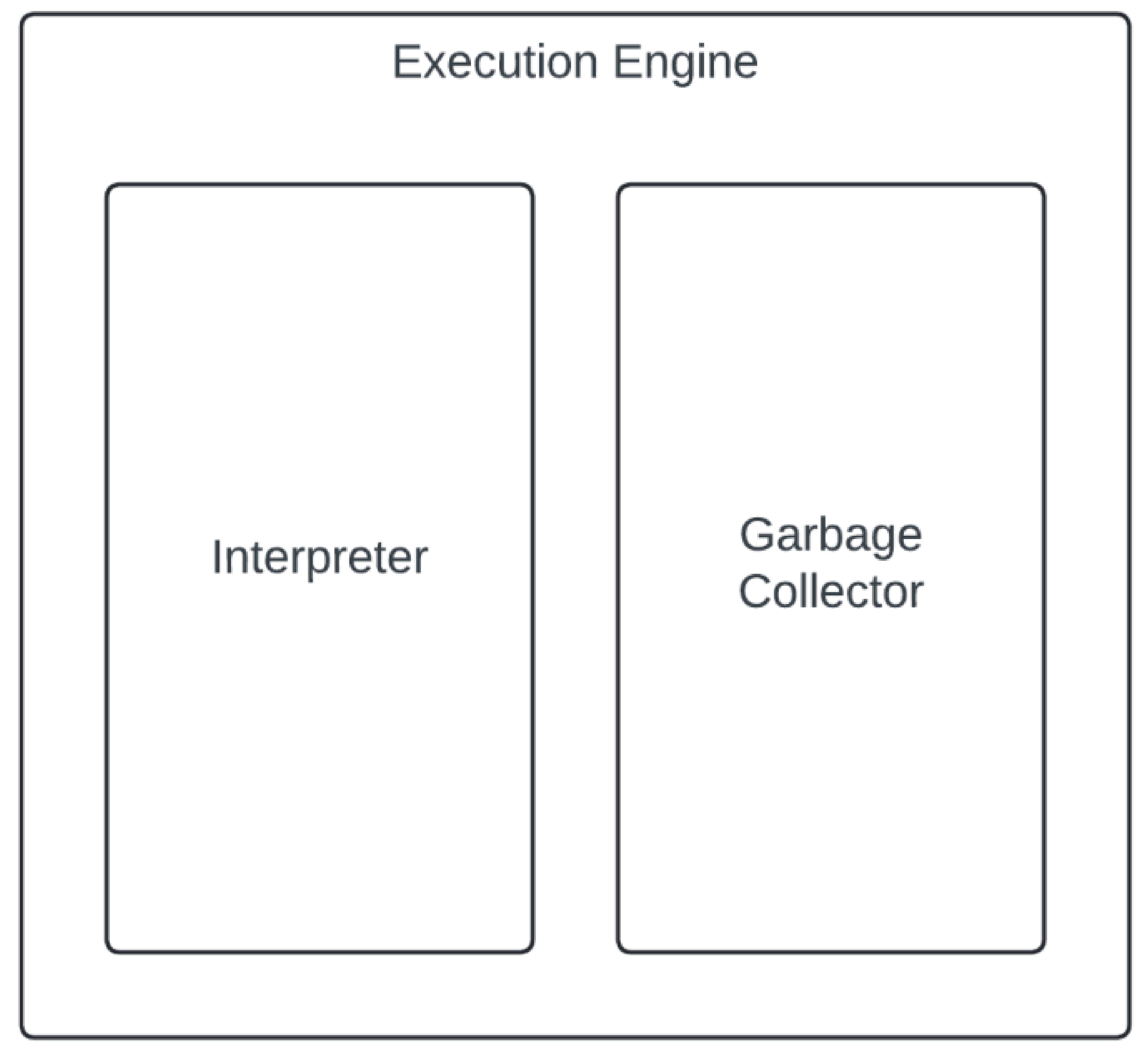
  -Updated with next instruction.

- Native Method Stack

  -Support native methods.

  -Methods are written in a language other than the Java.

EXECUTION ENGINE



- **Interpreter**

  Reads and executes the bytecode instructions line by line.

- **Garbage Collector**

  Reclaims unused memory, removing unreferenced objects and freeing heap space.

  -Mark

  Identifies the unused objects in memory.

  -Sweep

  Removes the objects identified during the Mark phase.

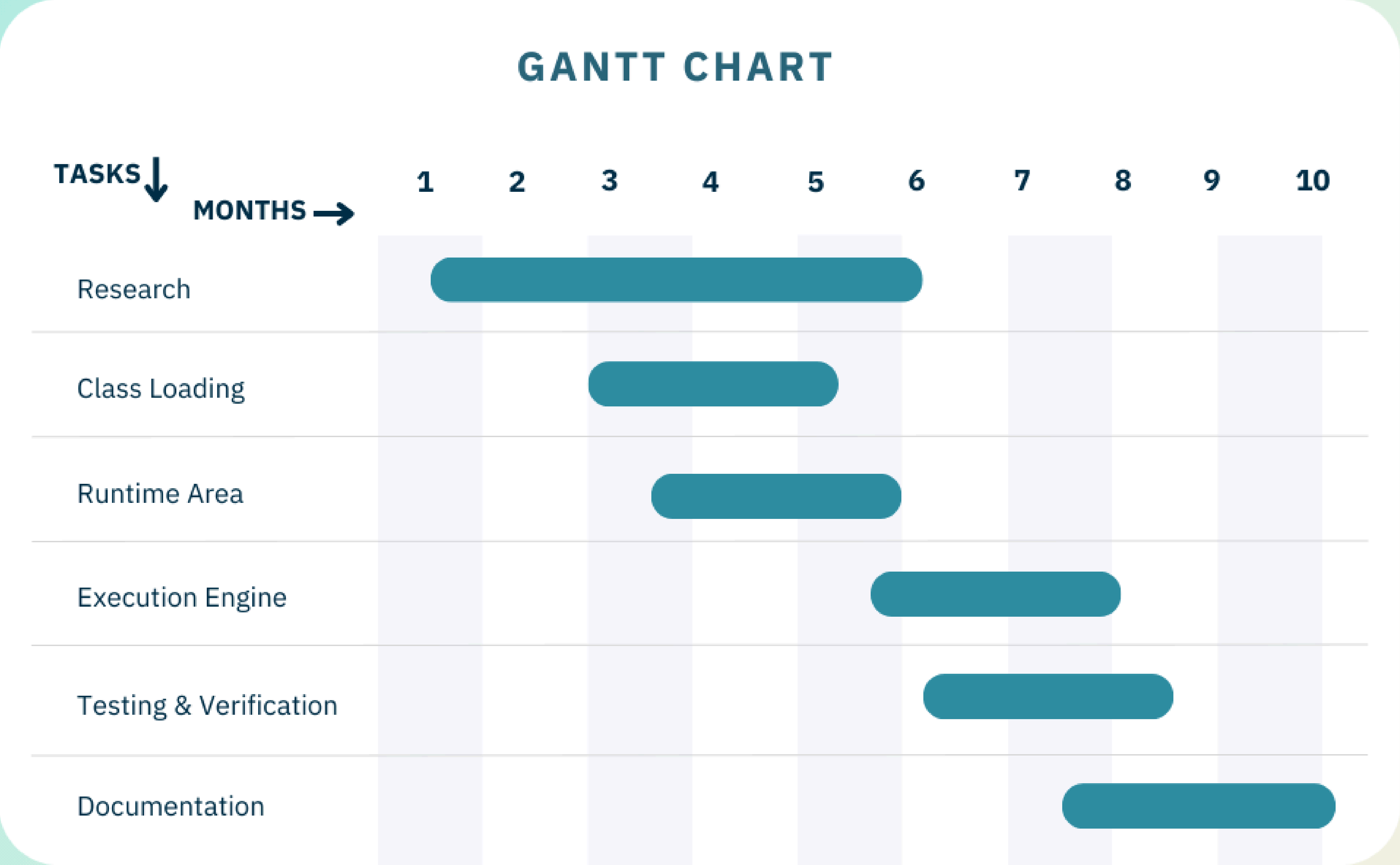
JAVA NATIVE INTERFACE

Bridge for permitting the supporting packages for other programming languages.

NATIVE METHOD LIBRARIES

Libraries that are written in other programming languages.
Usually in the form of .dll or .so files.

TIMELINE


GANTT CHART
TASKS
MONTHS
1
2
3
4
5
6
7
8
9
10
Research
Class Loading
Runtime Area
Execution Engine
Testing & Verification
Documentation

THANK YOU