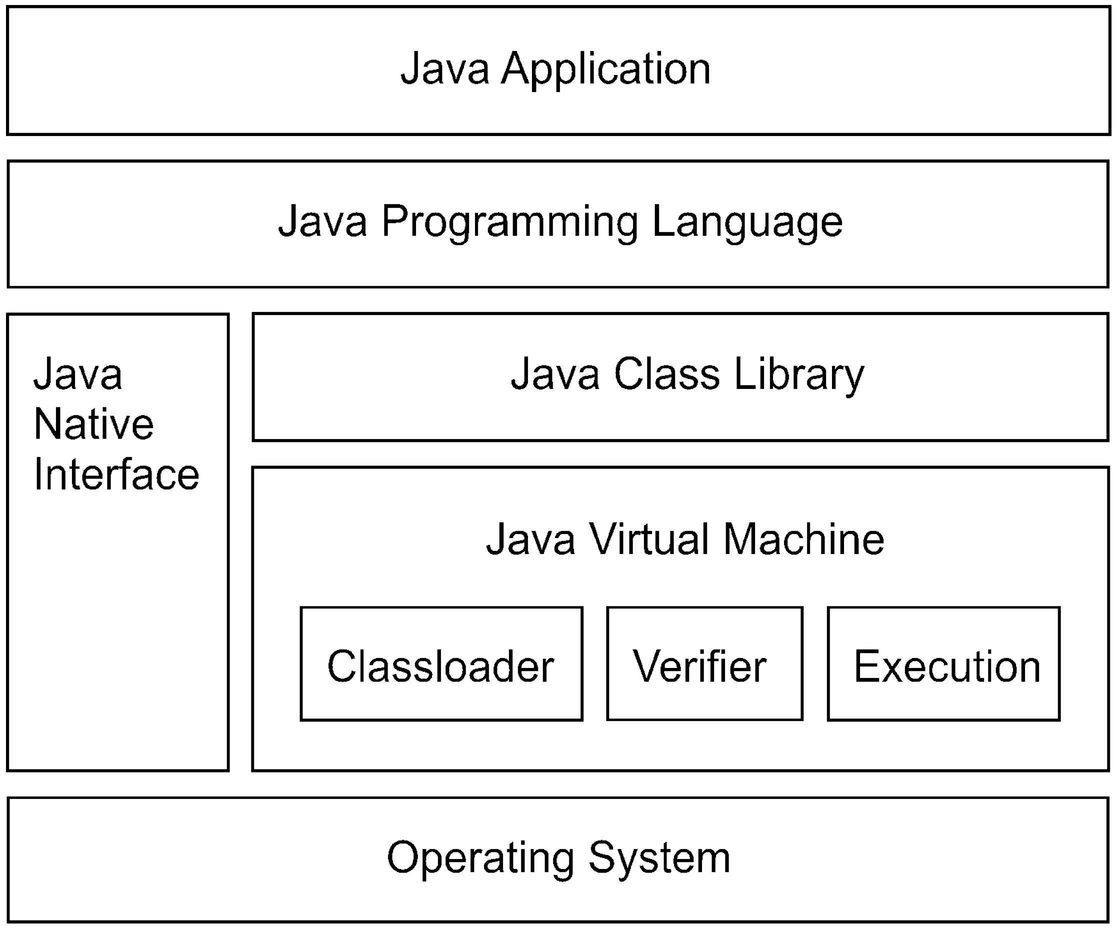
Java system overview



VM Languages

A Virtual Machine is an abstraction of the computing environment. JVM + APIs

Pro

- Platform independence
- Safer to distribute (restricts potential security attacks)
- Expressive power (programming language)
- Well documented APIs

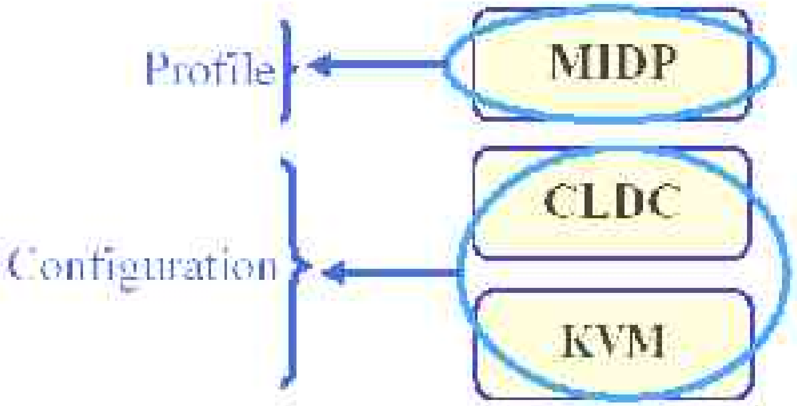
Con

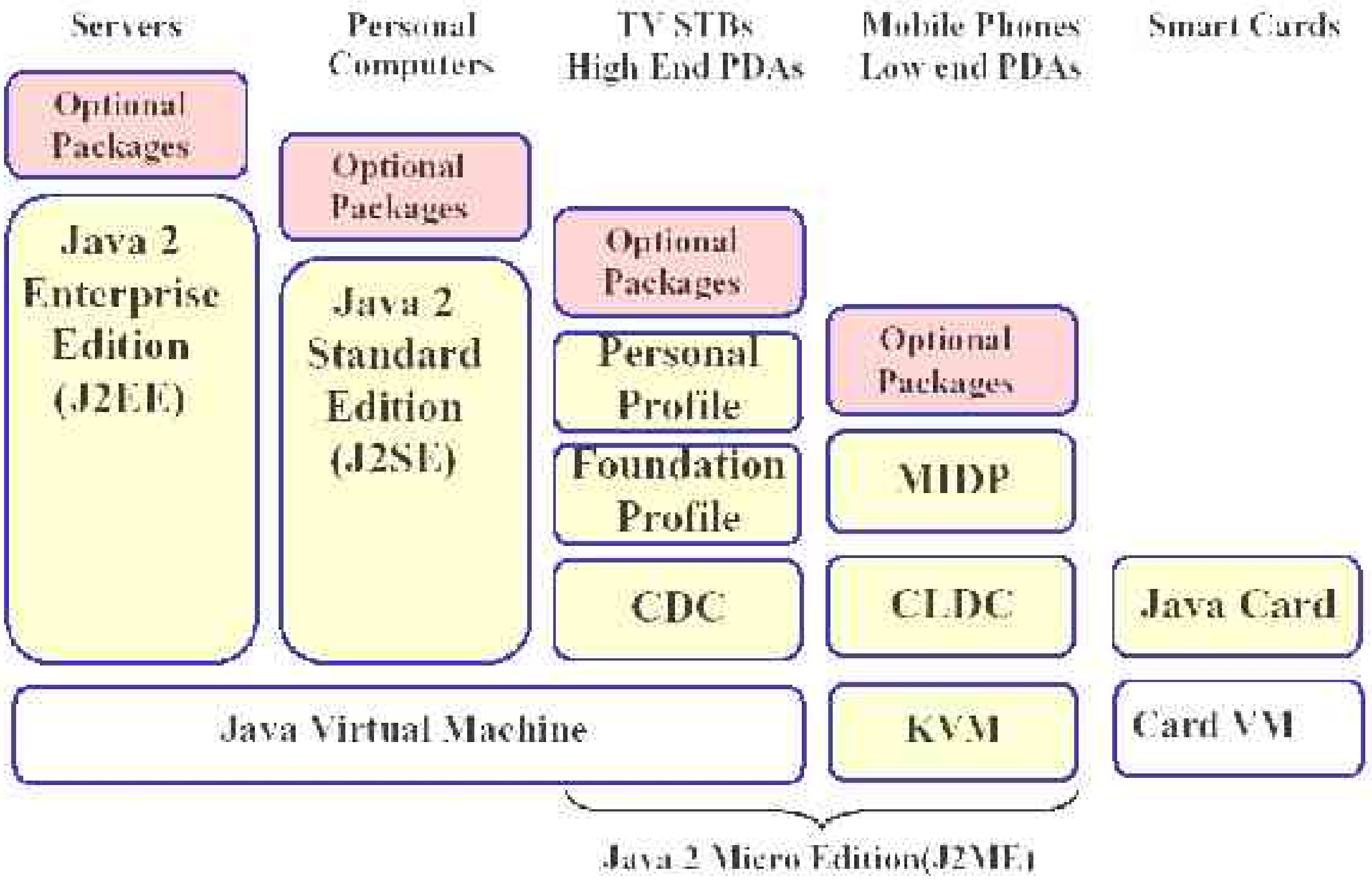
- Heavy applications (because of VM concept)
- Difficult of use (programming language)
- Less powerful than compiled languages

VM Languages
Java Overview

- Nowadays, trying to target all kind of computer devices
- Editions:
  - Java 2 Enterprise Edition (J2EE): for servers and enterprise computers
  - Java 2 Standard Edition (J2SE): for servers and personal computers
  - Java 2 Micro Edition (J2ME): for embedded devices, PDAs, mobile phones, and Digital television set-top boxes
  - Java Card: for smart cards

- Profile
  Requirements for a specific vertical market of devices (set of APIs)
- Configuration
  Minimum platform for a horizontal grouping of devices (VM + core APIs)

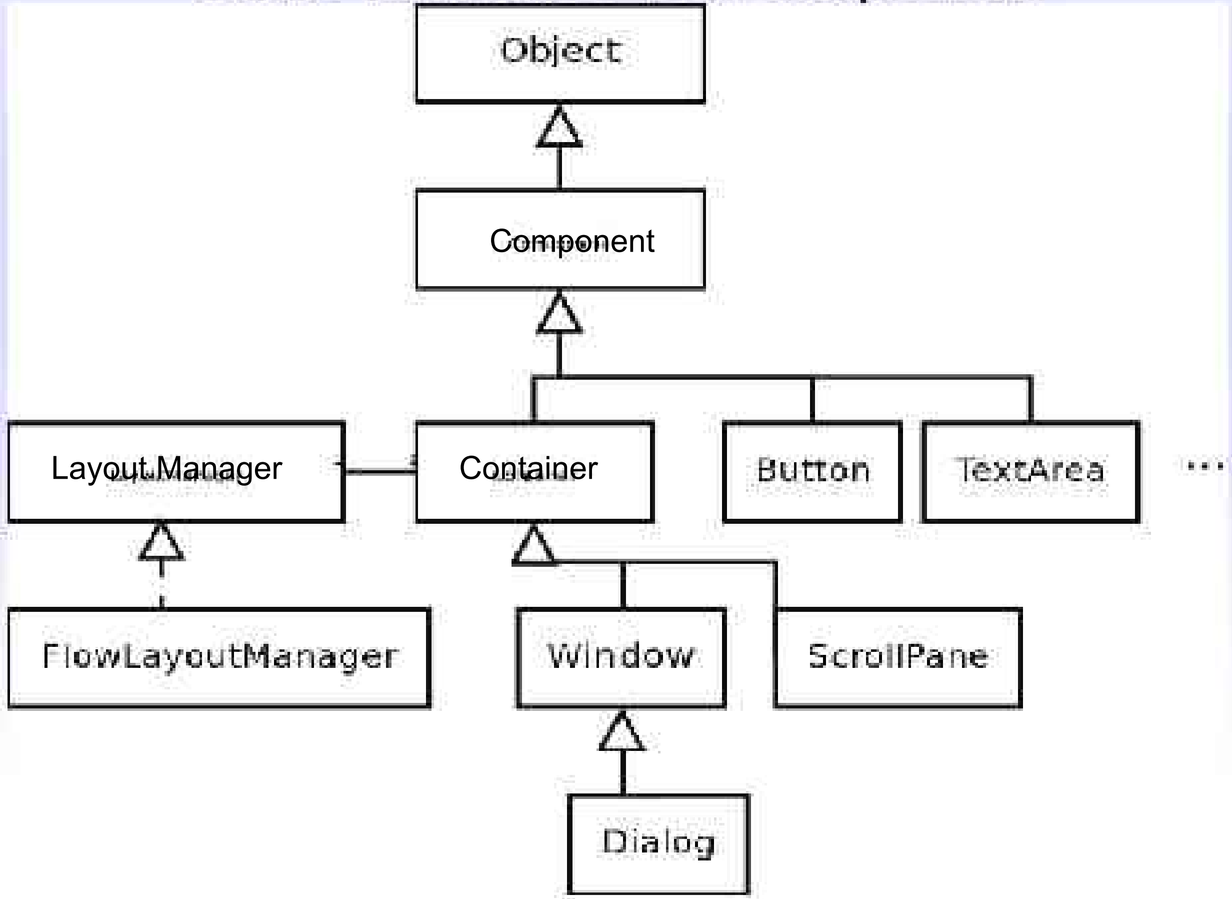



Servers
Personal Computers
TV STBs High End PDAs
Mobile Phones Low end PDAs
Smart Cards
Optional Packages
Optional Packages
Optional Packages
Optional Packages
Java 2 Enterprise Edition (J2EE)
Java 2 Standard Edition (J2SE)
Personal Profile
Foundation Profile
CDC
MIDP
CLDC
Java Card
Java Virtual Machine
KVM
Card VM
Java 2 Micro Edition(J2ME)

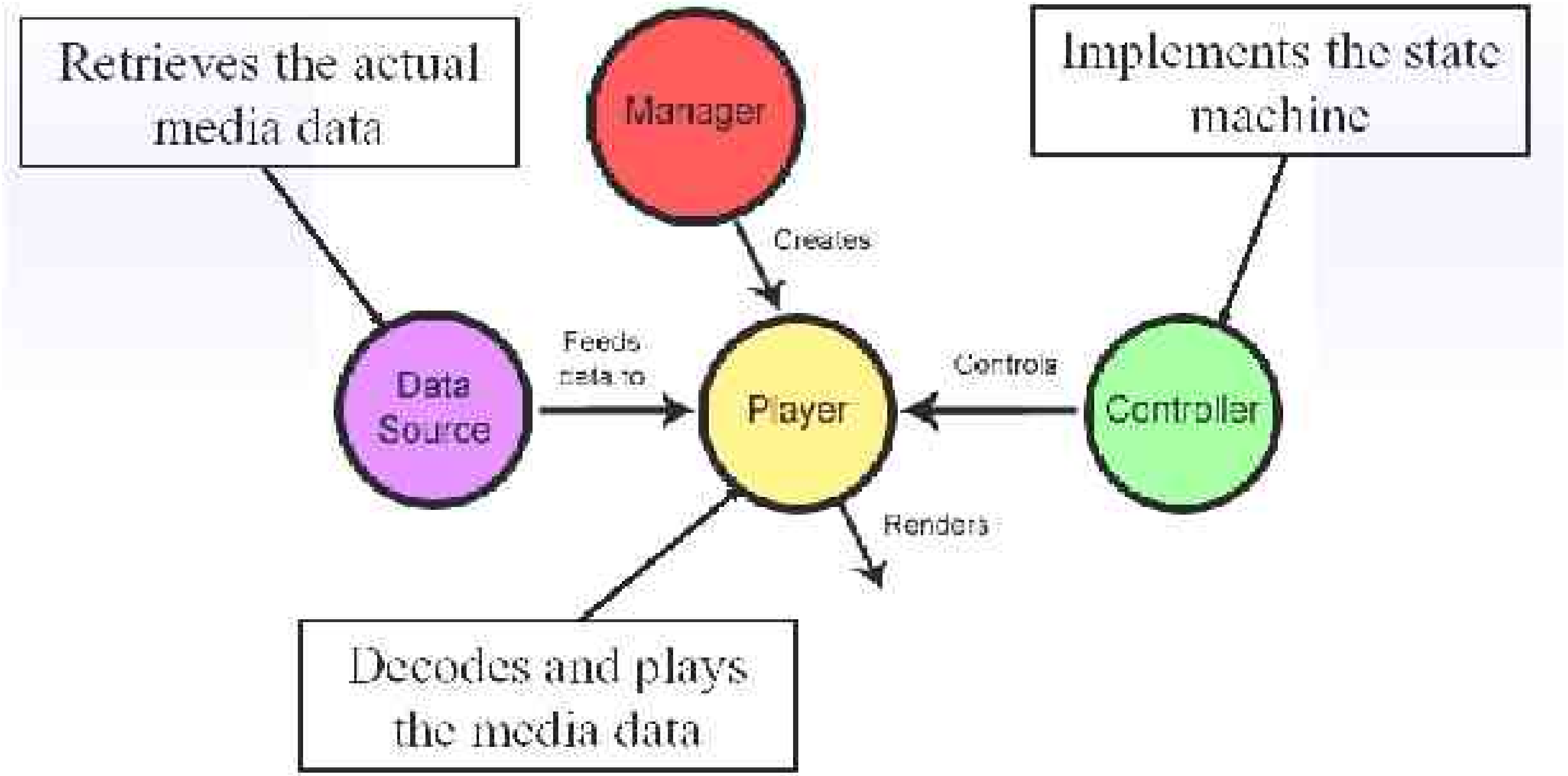
VM Languages
Multimedia

- User interface development (AWT/Swing)
  - Layout: Grid, North-South-East-West, Flow
  - Set of Widgets: Button, TextArea
  - User Interaction: awt.ui.* (Mouse, Keyboard...)
- Video/Audio and Synchronization (JMF)
  - Manager, Player, Data Source, and Controller
- 3D Graphics
  - Java3D
  - Java wrappers for OpenGL
- Different Devices
  - Television: MHP/OCAP/ACAP/ARIB -> GEM
  - Handheld: MIDP

VM Languages
User Interface Development

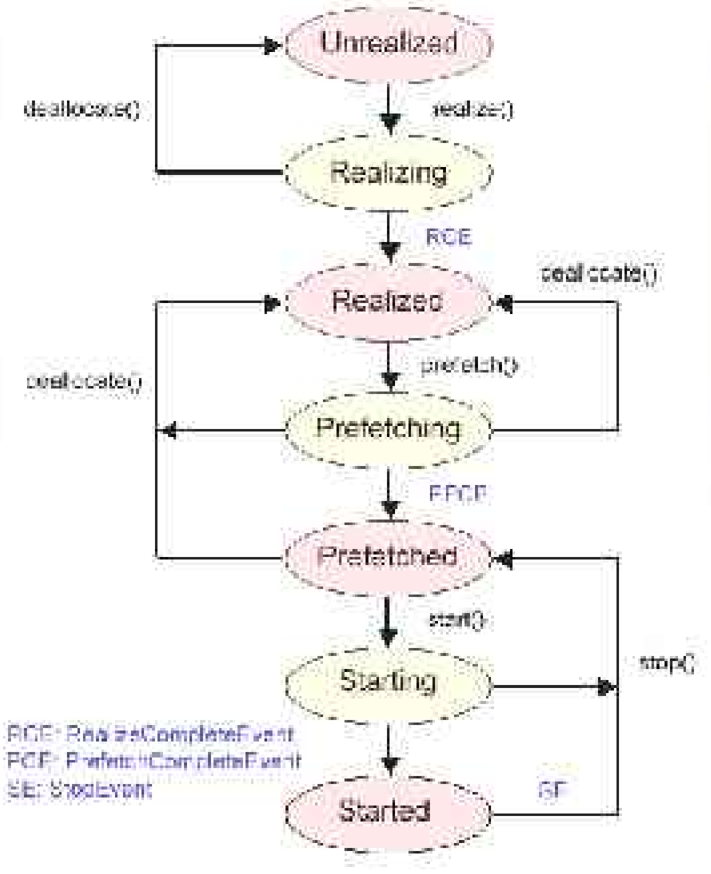


VM Languages
JMF (1/2)


Retrieves the actual media data
Manager
Implements the state machine
Creates
Feeds data to
Data Source
Player
Controls
Controller
Renders
Decodes and plays the media data

VM Languages
JMF (2/2)

- Unrealised: when it does not have all the information to acquire the needed resources
- Realised: when it has all the information to acquire the needed resources
- Prefetched: when it has all the needed resources, and has already prefetched enough media data to start playing immediately
- Started: when it is actually playing the media

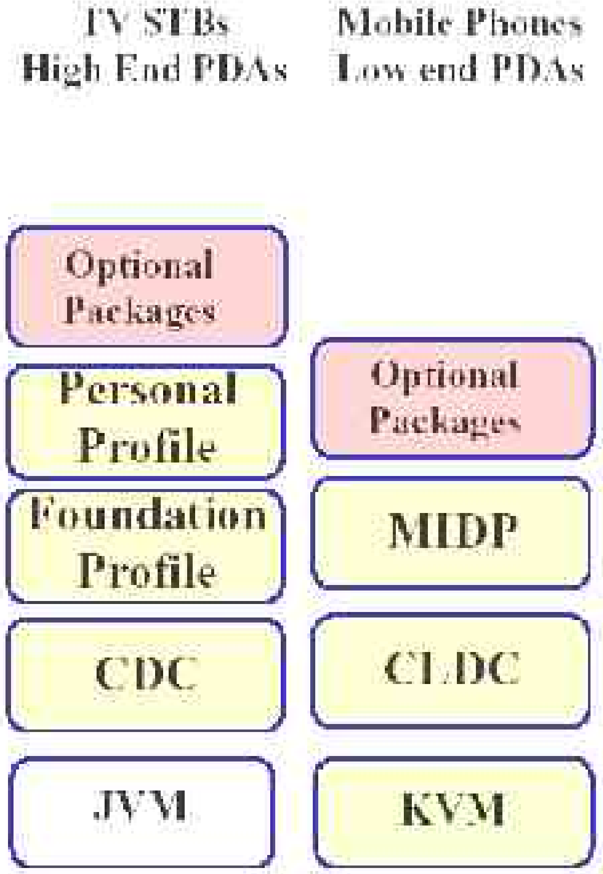


VM Languages
3D Graphics

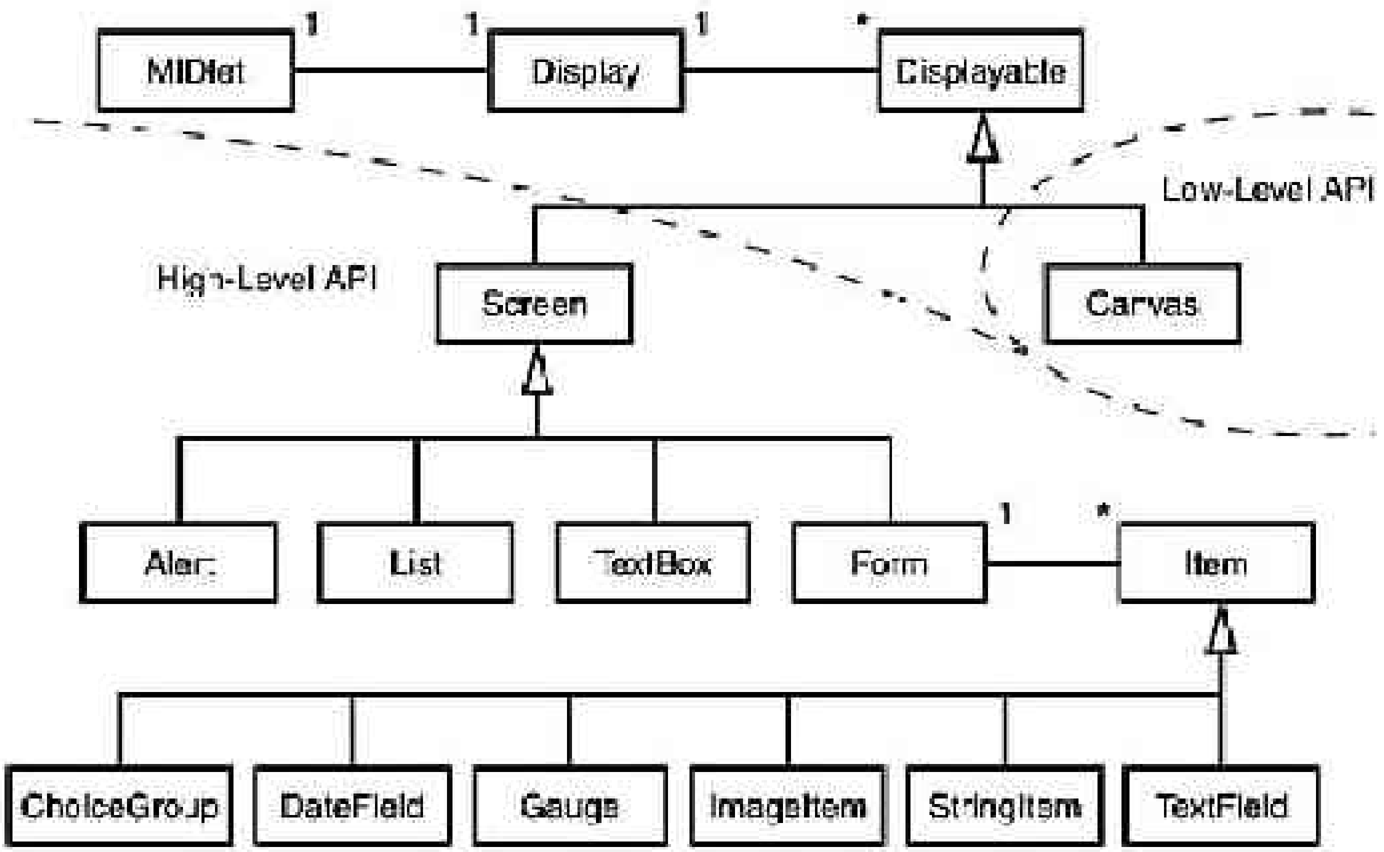
- Java3D
  - Completely new API for stand-alone 3D graphics applications
  - Can use any underlying architecture (Direct-X, OpenGL...)
  - It might not be the most efficient approach
  - Developers have to learn a new API
- Java wrappers of OpenGL
  - Functionality from OpenGL
  - Developers knows the API already
  - Only wrappers: uses Java Native Interface (JNI)
  - Much intercommunication between layers (Java -> C)
  - API is not standardised yet (Java Specification Requests)
    - JSR 231: OpenGL
    - JSR 239: OpenGL ES

VM Languages J2ME

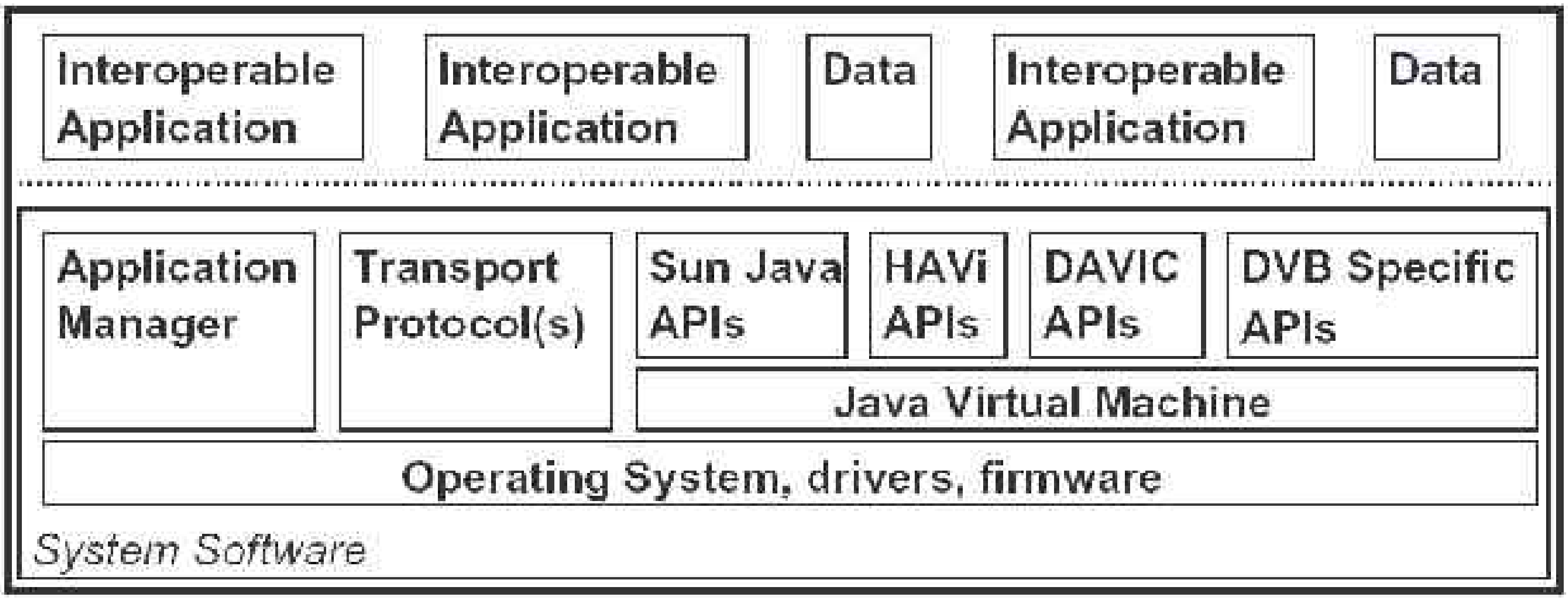
- Defines two Configurations:
  - CDC: High end consumer devices
    - RAM Java Memory: around 2MB
    - ROM Java Memory: around 2.5MB
  - CLDC: Low end consumer devices
    - Processor:16 bit/16 MHz or higher
    - Java total memory: 160-512 KB
- CDC (Connected Device)
  - Personal Profile
    - Adds support for lightweight AWT
  - Foundation Profile
    - Basic application APIs (no GUI)
- CLDC (Connected Limited Device)
  - Mobile Information Device Profile (MDIP)
    - Application APIs + GUI APIs



VM Languages Handheld



VM Languages
Television



VM Languages
Summary

Supported Media Types		
	Text, Graphics	AWT
	Video, Audio	JMF
Arrangement of the signs		
	Spatial	AWT
	Temporal	Java Threads
Interaction		AWT Events
Different Devices		
	Handheld	MIDP
	Television	GEM