Why Java is practical for modern operating systems

JNode.org

Ewout Prangsma

Contents

- Introduction
- History
- Characteristics
- Architecture
- Plugin framework
- Driver framework
- Future
- Java benefits

Introduction

- Simple to use & install operating system for personal use: written for and in Java
- Targets:
  - Modern devices
  - Desktop
  - Small servers
- Only actively developed Java OS in the open source world
  - 7 active developers, 17K downloads

History

- Original idea started in 1995
- First attempt: JBS (Java Bootable System)
  - Contained C code, did not work at all
- Second attempt: JBS2
  - Still did not work well, but was better
- Then: JNode
  - No C code anymore, Classpath classlibraries
- Went public in May '03

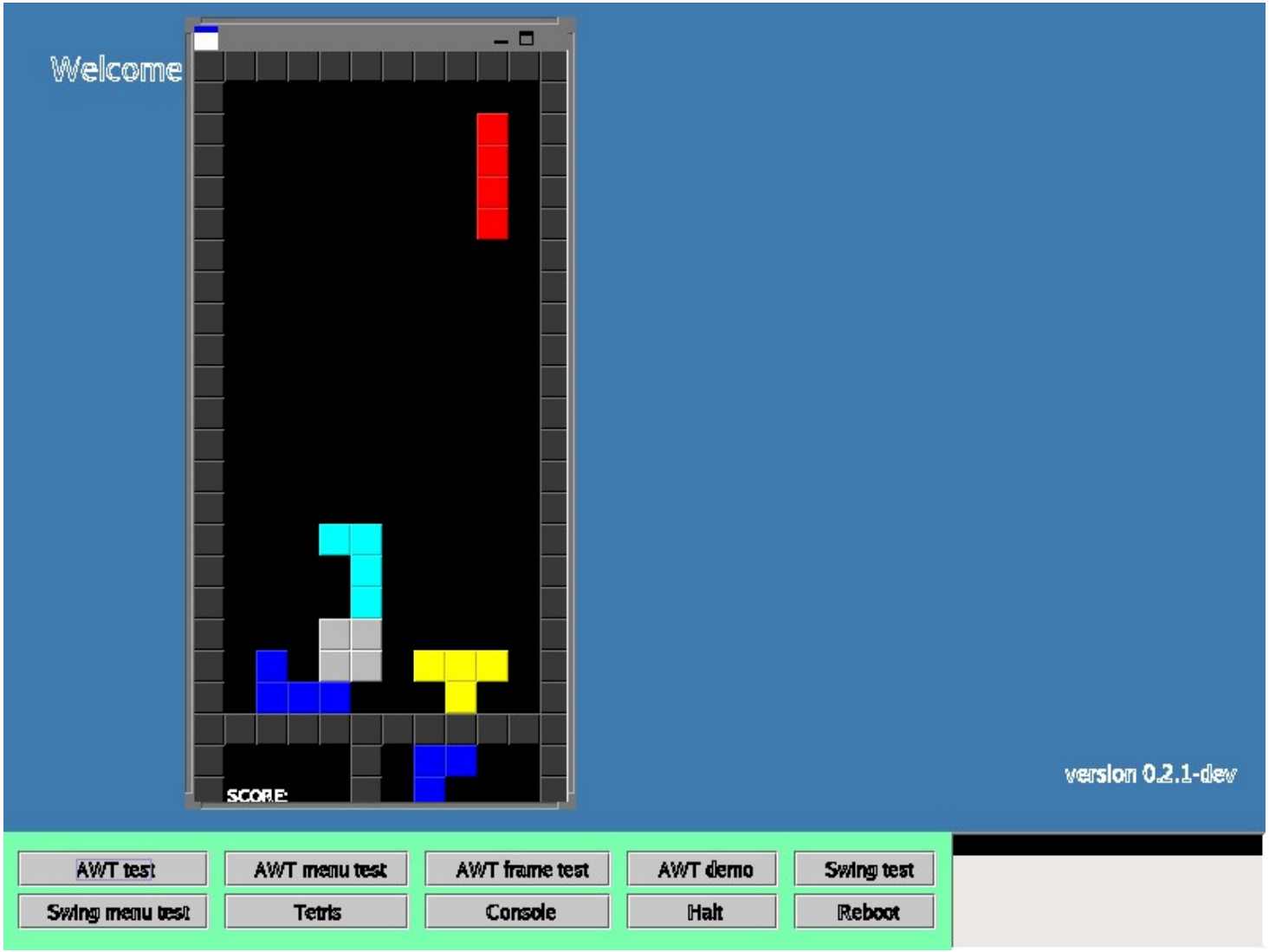
Characteristics

- All Java, minimal assembler, no C
- All java build system (almost)
- Extensible architecture
- Single flat memory address space, no virtual memory
- JVM written in Java
- All Java code is compiled on the fly, no interpreter
- Security is always on
- LGPL license

Status (1)

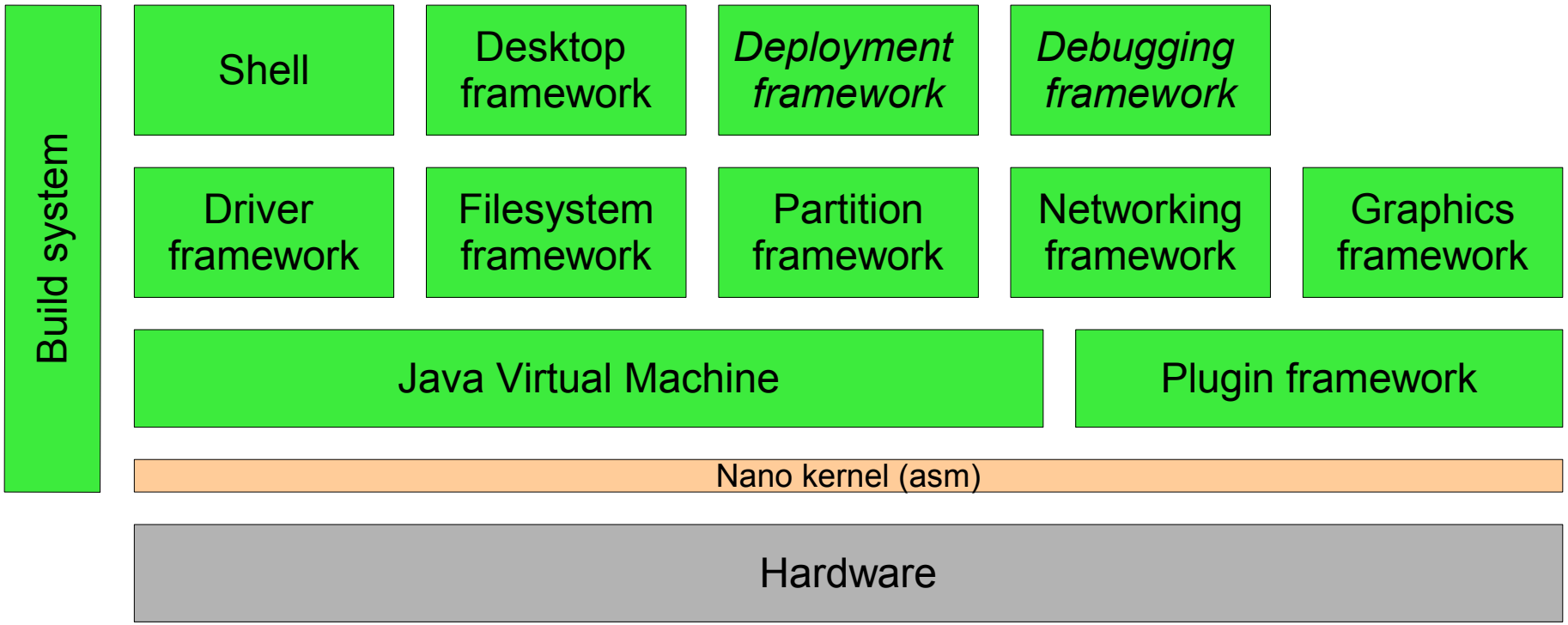
- Release 0.2.1
  - Full device driver model
  - Networking TCP/IP
  - Filesystems EXT2, FAT, NTFS, ISO9660
  - Graphics 2D
  - J2SDK 1.5 support in VM (not classlibs)
  - Simple heap managers & GC
    - MMTk based heap manager & GC in development
  - IA32 & AMD64 platform support

Status (2)

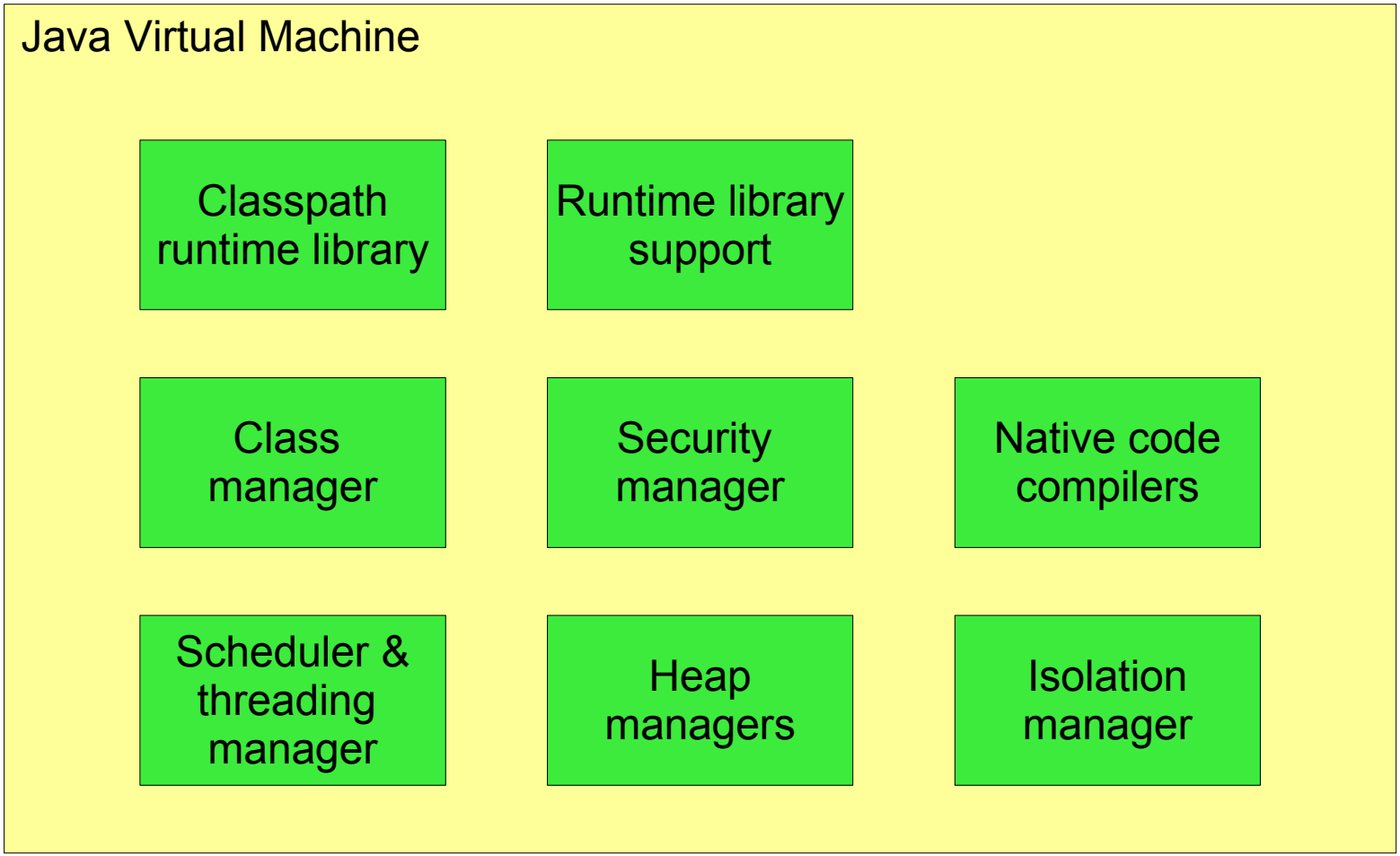


Playing Tetris on JNode

Architecture (1)



Architecture (2)



Plugin framework (1)

- Everything is contained in a plugin
  - code, resources
  - even JVM & the plugin framework itself
- Plugins can:
  - be loaded, unloaded & reloaded (at runtime)
  - depend on other plugins
  - provide well known extension points
  - connect to well known extension points

Plugin framework (2)

- Plugins are:
  - described by a descriptor
    - descriptor also contains license info
  - JAR files
  - inspired by Eclipse plugins

Plugin framework (3)

```
<plugin id="org.jnode.driver" name="JNode Driver Framework" version="@VERSION@"
        provider-name="JNode.org" license-name="lgpl" class="org.jnode.driver.DriverPlugin">

    <requires>
        <import plugin="org.jnode.work"/>
    </requires>

    <runtime>
        <library name="jnode-core.jar">
            <export name="org.jnode.driver.*"/>
            <export name="org.jnode.driver.util.*"/>
        </library>
    </runtime>

    <extension-point id="finders" name="System device finders"/>
    <extension-point id="mappers" name="Device to Driver mappers"/>

    <extension point="org.jnode.security.permissions">
        <permission class="java.util.PropertyPermission" name="jnode.cmdline"/>
    </extension>
</plugin>
```

General info

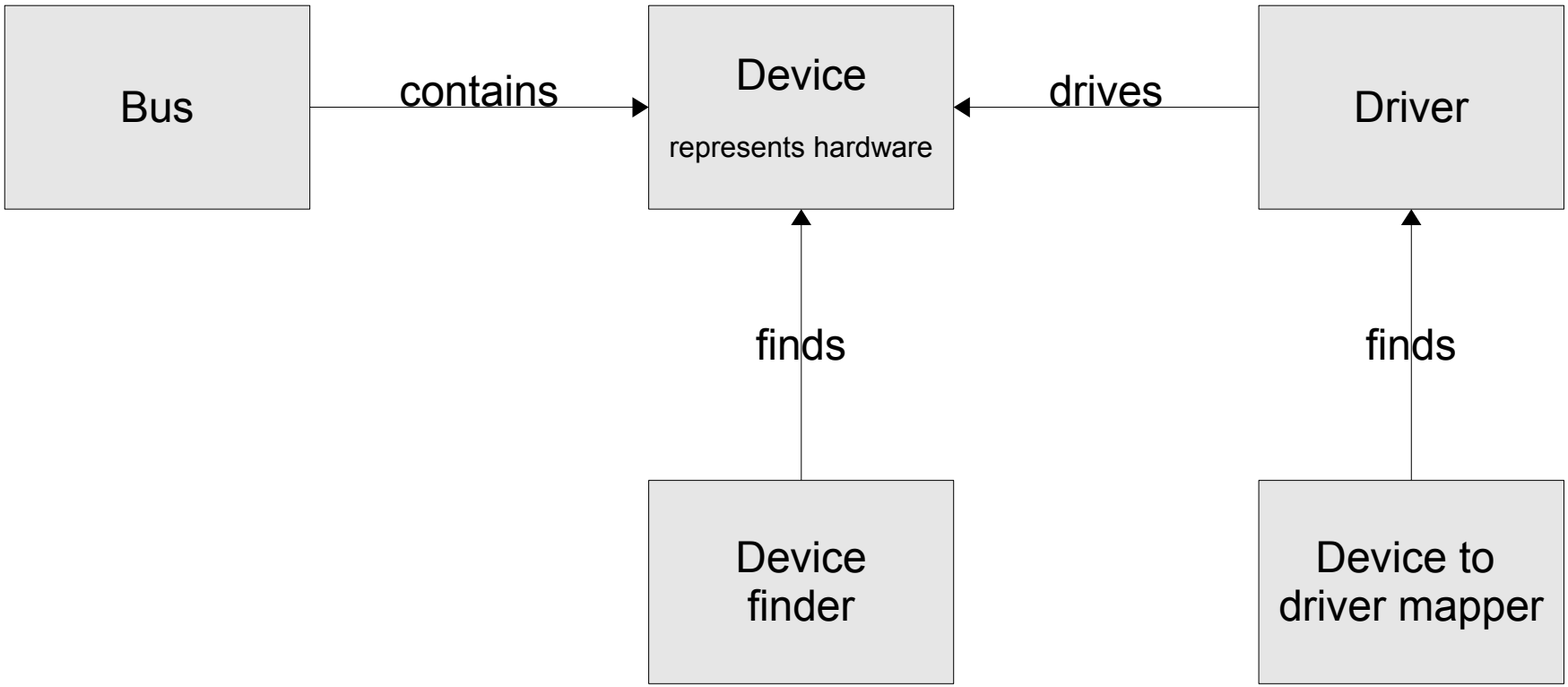
Dependencies

Code & resources

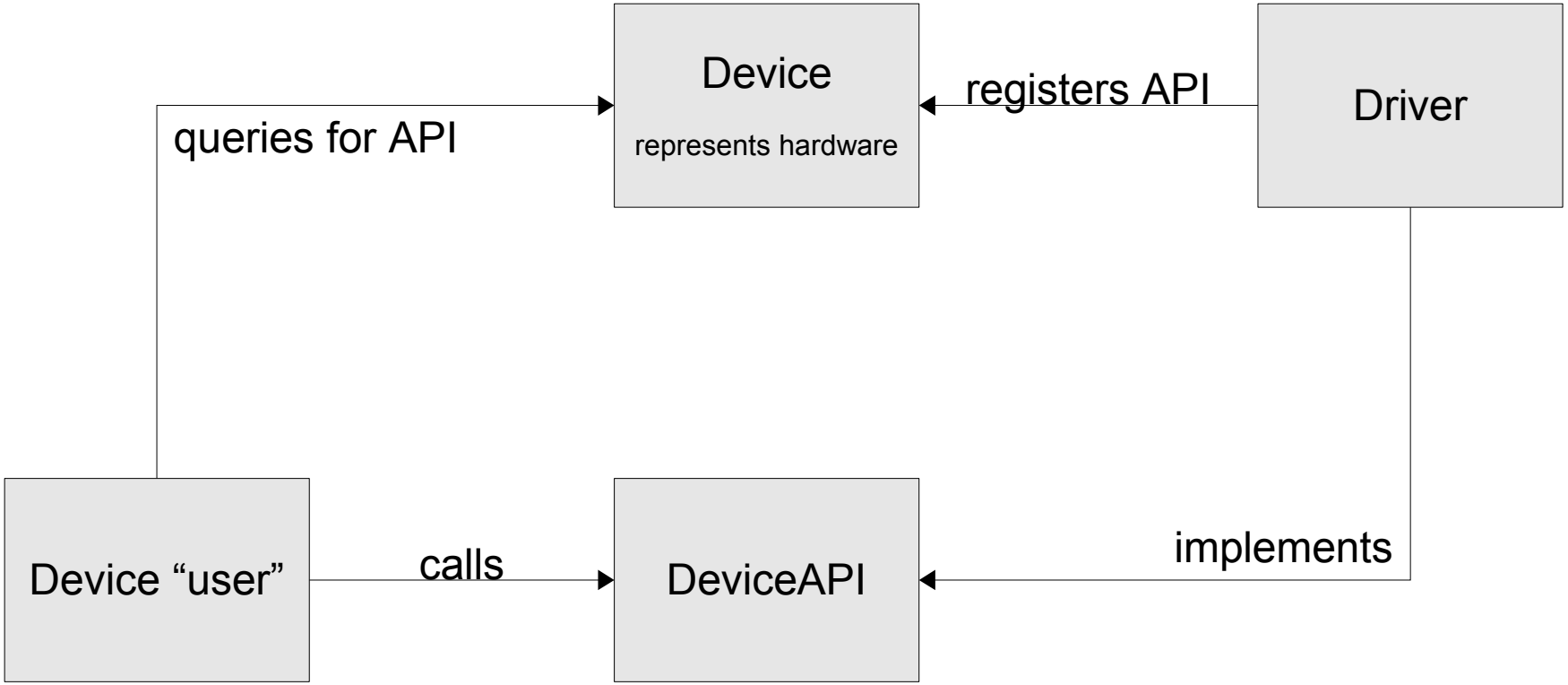
Well known extension points

Connection to well known extension point

Driver framework (1)

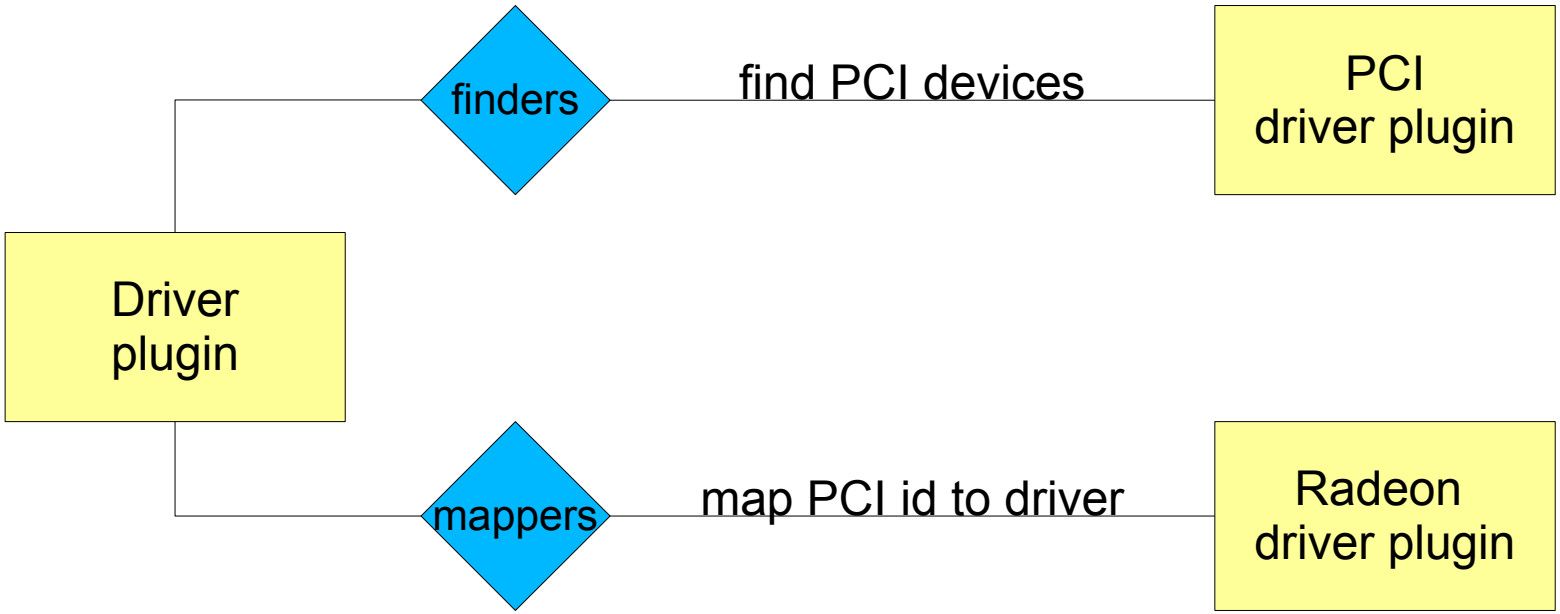


Driver framework (2)

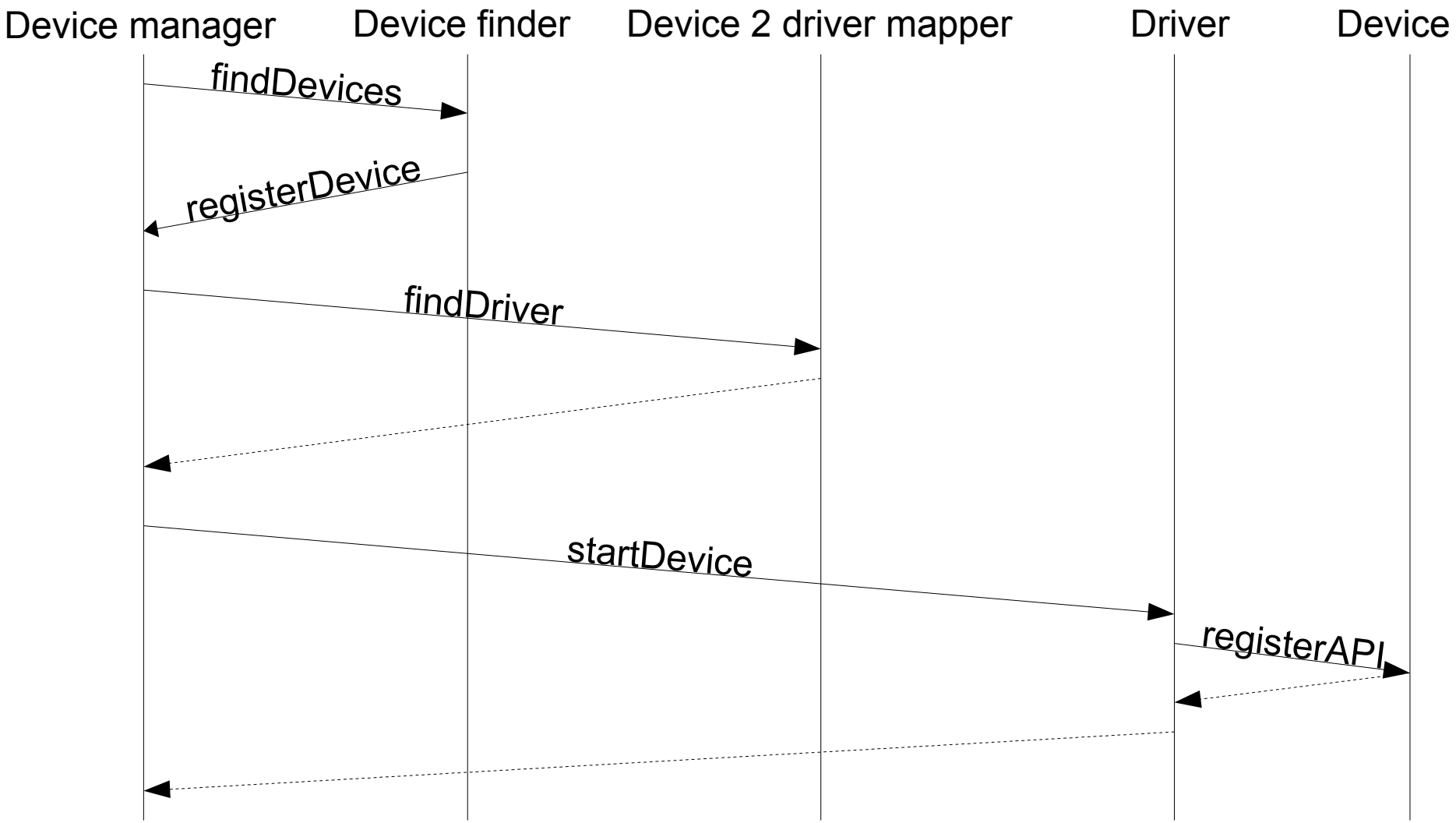


Driver framework (3)

Example: *Radeon Graphicscard driver*



Driver framework (4)



Future

- Short term:
  - Improved JVM performance (mm, compilers)
  - Deployment framework
  - Improved graphics
- Long term:
  - Simple to use desktop environment
    - Fully document oriented instead of app. oriented
  - Java powered servers
    - e.g. Cooperation with ApacheDS

Java benefits

- Dynamic linking
- Type safe language (even more in J2SDK 1.5)
- Security
  - Security manager
  - No uncontrolled memory access
- Great development tools:
  - Eclipse, Ant

We need your help!

- Don't be scared of by the codebase
  - Most of it is classpath libraries
- Ask questions

- Visit http://www.jnode.org
- Contact me: epr@jnode.org

Questions