

Java Card PLATFORM Overview

Sebastian Hans

Senior Staff Engineer

Sun Microsystems Inc.

Agenda



Java Card 2 Platform

- Java Card 3 Platform

Smartcard basics

- Small temper resistant device
 - > 8-32 bit CPU
 - > 1-32KB RAM
 - > ROM and EEPROM up to 128KB
 - > FLASH memory can also be used
- High secure memory and CPU
- Clock and power from the terminal
- One synchronous I/O line
- Master-slave protocol based, card is the slave

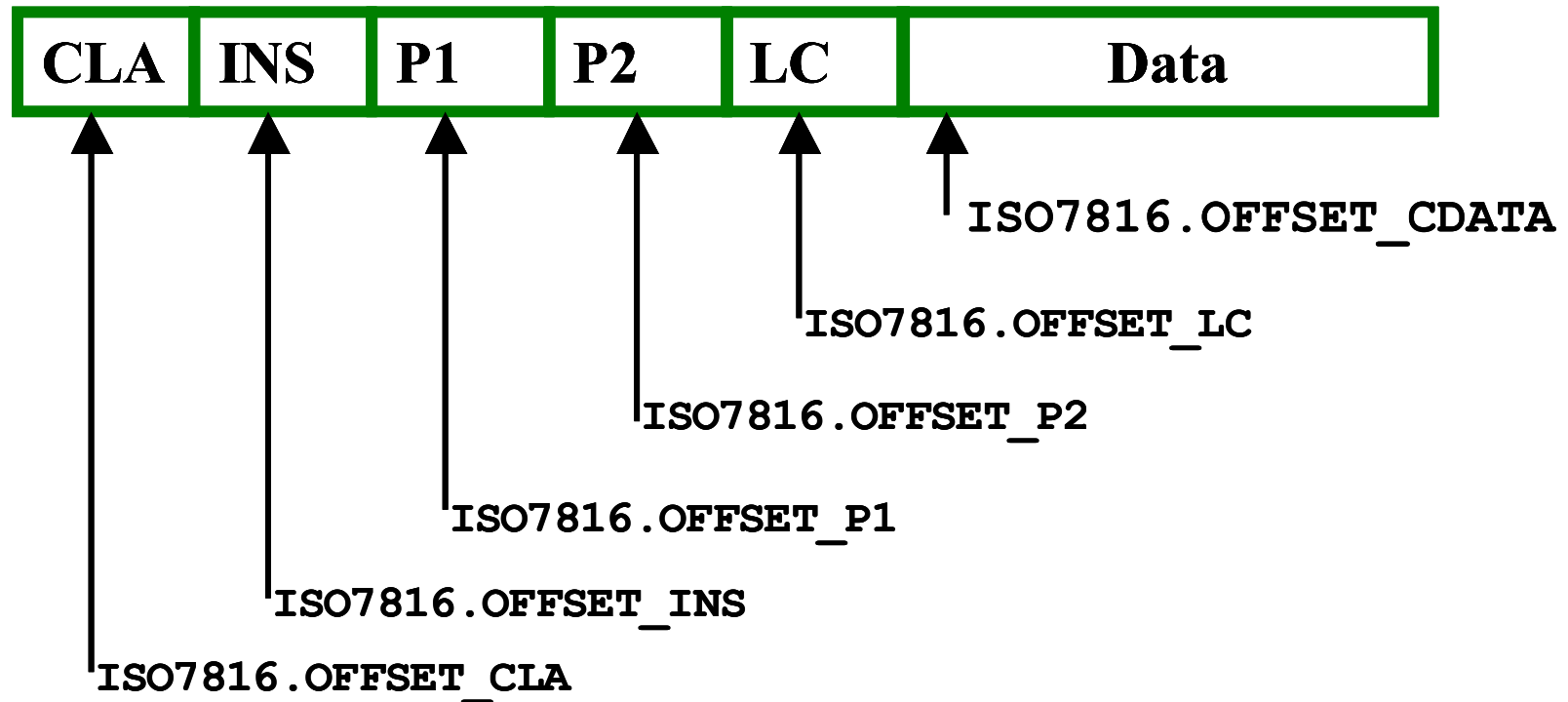
Smartcard Standards

- The Baseline for all standards and specifications are
 - > ISO 7816 series
 - > defines electrical and physical characteristics,
 - > Handshake between card and terminal
 - > transport protocols,
 - > applications protocol,
 - > File structures, Data structures (TLVs)
 - > Everything in ISO is optional
- For Telecommunications (GSM, 3GPP, 3GPP2, OMA, TETRA) ETSI is standardizing a very strict platform the SIM and UICC
- For the financial market EMV is the main specification
- ICAO defines data structures for e-passports

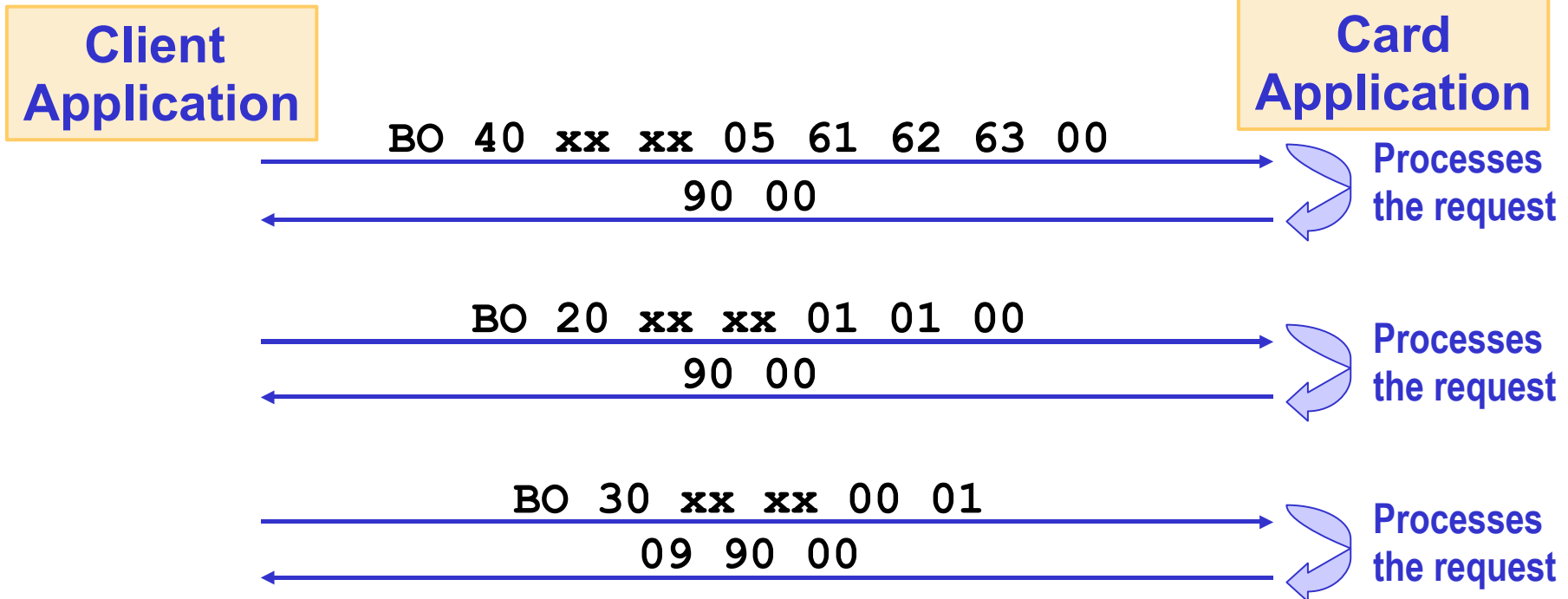
What does a card do ?

- A card is an ultimately thin Server
 - > It gets requests and processes them
 - > It never takes the initiative
- The programming model follows it
 - > It is centered around the processing of incoming requests

The APDU command



APDU exchange



Java Technology Momentum



Java™



Java Everywhere

- 3.5 Billion Java-Enabled Cards
- 1.8 Billion Java-Enabled Phones
- 7 Million Java Set-top Boxes
- 800 Million Java Desktops
- 180 Operators Deploying Java Content
- 6 Million Developers

Introduction to Java Card

- Over 3.5 Billion cards deployed to date
 - > 825M shipped in 2006
 - > 1.2B shipped in 2007
- Variety of form factors
- All market segments
 - > Telecom (SIM card)
 - > Banking (Payment card)
 - > ID (citizen/corporate card)
 - > PayTV (subscriber card)
 - > Transport, Healthcare...
- 100's of products worldwide



SIM Cards



Secure Flash Memory



Passports



USB Tokens

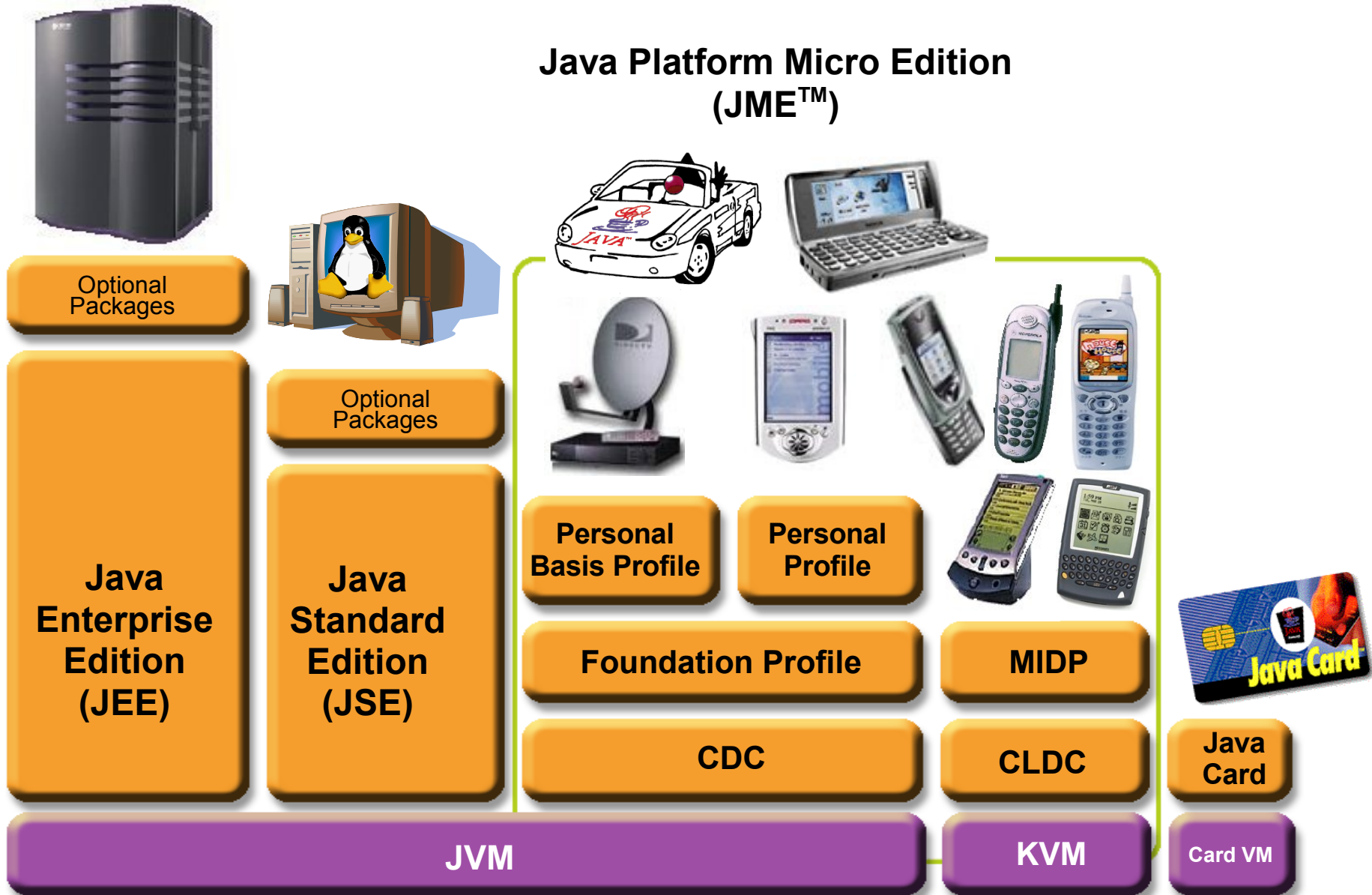


Smart Cards



Contactless

The Java™ Platform



What is a Java Card

- Java Card technology defines:
 - > A subset of the Java programming language and virtual machine definition suitable for smart card applications
 - > Core and extension Java Card API
 - > A secure multi application card runtime environment
 - > Enables post-issuance secure card application download
- Adaptable to different market needs
 - > (GSM, 3G, ID-card, Ticketing, Transport, Finance)
- All services have to be implemented as a Java Card Applet

Java Card – Historical Roadmap

- 1996** Introduction of Java Card technology
- 1997** Java Card 2.0 Technology Foundations
- 1999** Java Card 2.1 Interoperable File Format
- 2000** Java Card 2.1.1 Additional Crypto APIs
- 2002** Java Card 2.2 Next gen crypto, memory management
- 2003** Java Card 2.2.1 Enhancements for USIM
- 2004** Java Card S Entry level Fixed Function cards
- 2006** Java Card 2.2.2 ETSI and Contactless
- 2008** Java Card 3.0 “Classic” and “Connected”

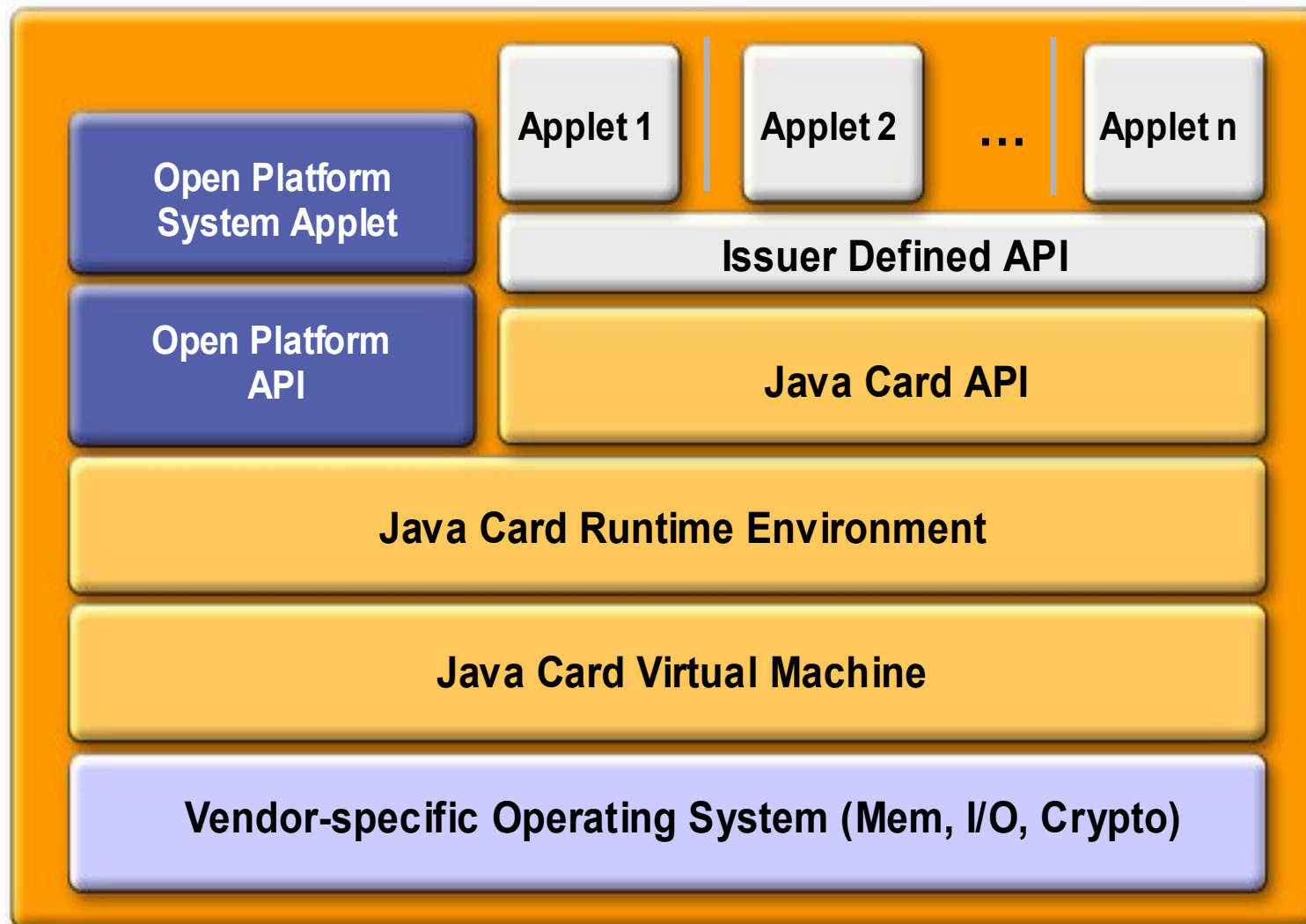
Java Card Benefits

- Object Oriented Programming
- Secure Programming Platform
- Hardware Independent
- Operating System Independent
- Multi-Application Support
- Secure Applet Loading
- Open Standard

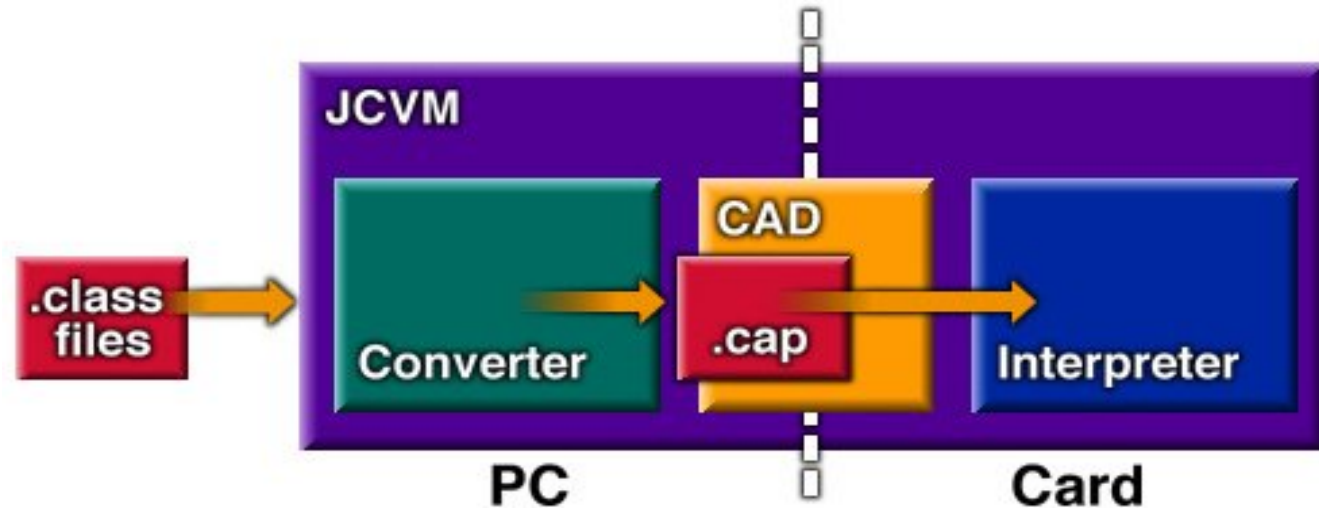
Main Java Card features

- Applications isolations by default through the firewall
- Objects sharing on a per applications base
- JCVM and JCRE are persistent and instantly available
- Single threaded VM
- Object persistent by default
- Transient objetcs
- Support for atomic transactions
- Applets run in the card under the control of the JCRE, no “main” method
- Cryptographic support (RSA, AES, ECC, DES ...)

Java Card Architecture



Split VM Architecture



- Off-card
 - Class loading, linking and name resolution
 - Bytecode verification, optimization and conversion
- On-card
 - Bytecode execution and security enforcement

Java Subset for the Java Card Platform

- Small primitive data types: boolean, byte, short
- One-dimensional arrays
- Packages, classes, interfaces, exceptions
- Inheritance, virtual methods, overloading, dynamic object creation, access scope, binding rules
- Optional: 32-bit integer “int” data type
- Optional GC

Java Card™ Runtime Environment JCRE

- Card resource management
- Communications (APDU exchange, inter-application communication)
- Applet execution (selecting and Applet, invoking process method)
- Applet security (firewall)

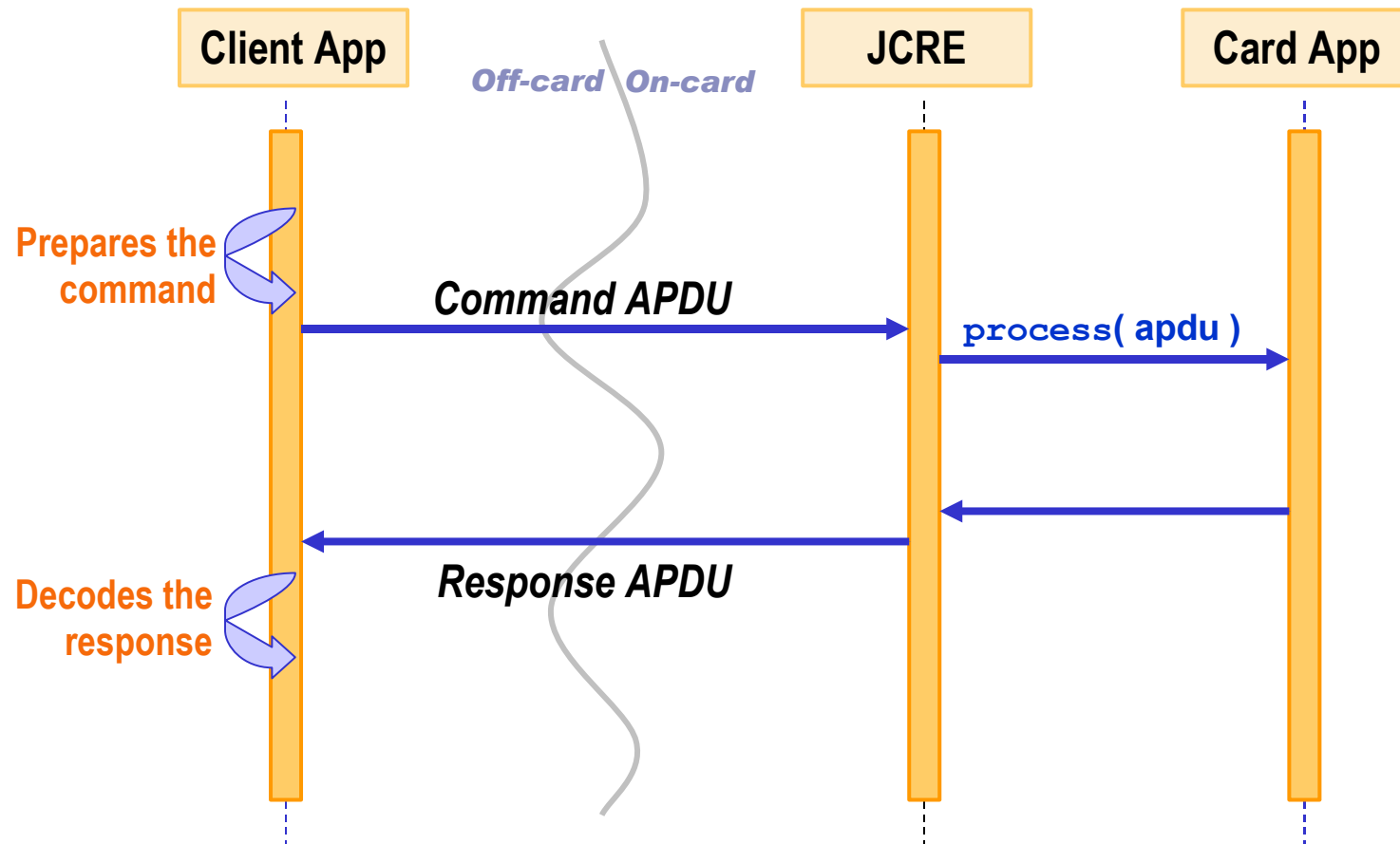
**Performs the tasks of
an operating system**

Java Card™ Runtime Features

- Persistent and transient objects
- Atomic operations and transactions
- Applet firewall and sharing mechanisms

**Java Card VM and Java Card RE
run for the whole card lifetime!**

JC application Sequence Diagram



JC application modell

- A JC applications is always a subclass of the Applet class from `javacard.framework`
- Applets class provides entry points to select and deselect the application, install it and receive APDUs from the terminal
- Reacts to APDU's send from the JCRE to the process method
- Only one active applications at a time
- Several applet can be selected at the same time but can not work in parallel

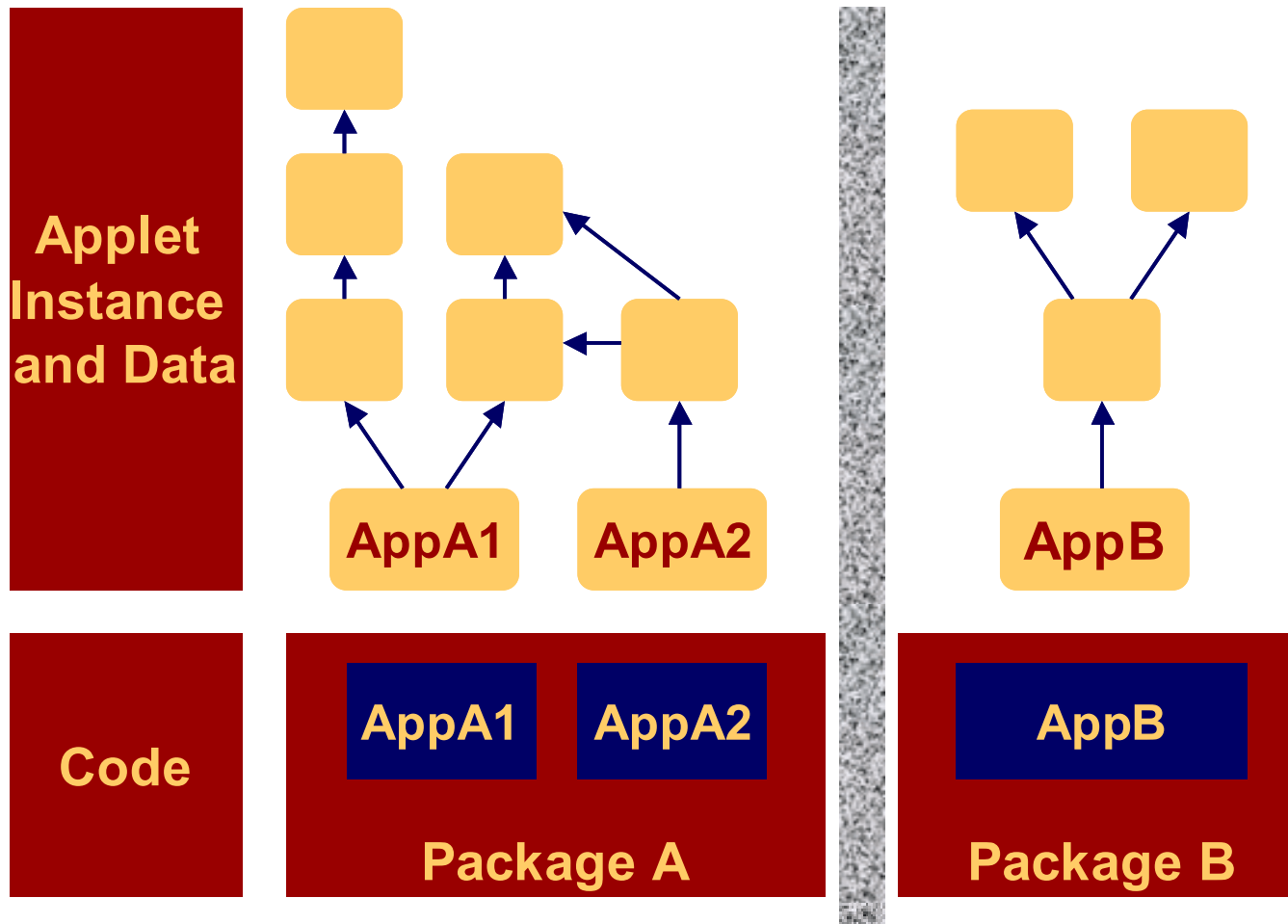
Why a Firewall ?

- Provides isolation between applications
 - > In addition to the Java™ programming language rules
- Required because of persistence
- Operates dynamically at run-time
 - > Objects are “owned” by applications

The Firewall Is Flexible

- System objects are handled specifically
 - > Some access constraints are relaxed
JCRE entry point objects
- Isolation is at the package level
 - > Several applets can be in the same context
- Applets can explicitly share objects
 - > `javacard.framework.Shareable`

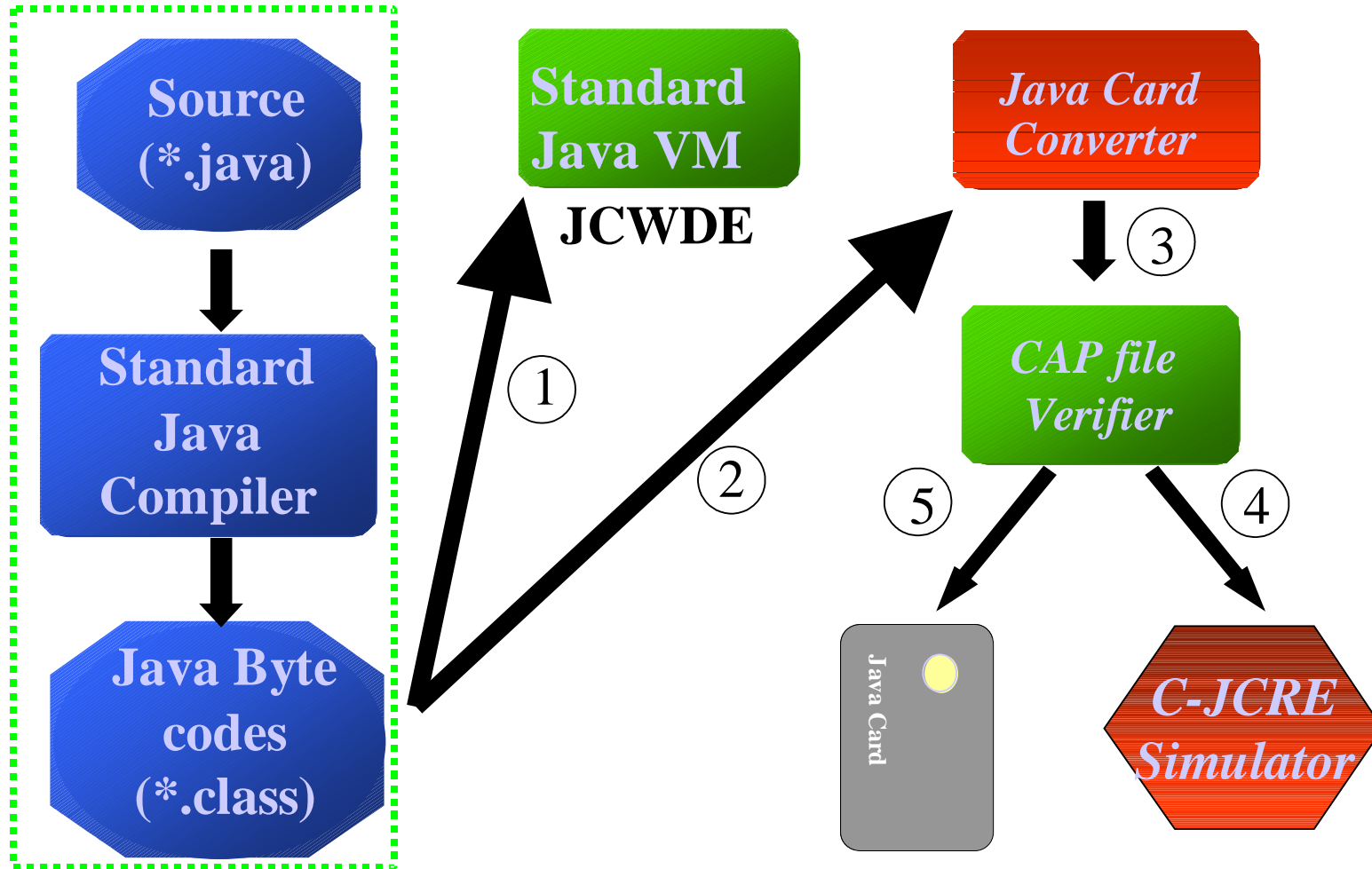
Firewall Granularity



Java Card API packages

- Java API Packages
 - > package `java.lang`
- Java Card specific packages
 - > package `javacard.framework`
 - > package `javacard.security`
 - > package `javacardx.crypto`

Applet Development Path



Latest Java Card Specification – 2.2.2

Focus on Contactless and ID

- 2.2.1 maintenance
 - > RMI-related bug in VM spec
 - > Correct CRC32 bug in Checksum class
 - > Utility APIs for TLV, short, int
- Contactless Enhancements
 - > Extended length APDU support
 - > Memory access API
 - > Contactless crypto performance enhancement
 - > Multiple Interfaces management
 - > BCD Utility API
- Crypto and Security
 - > Additional Crypto algorithms
 - > HMAC-MD5, HMAC-SHA1, SHA-256, Korean Seed
 - > Signature w/msg recovery
 - > Partial message digest
 - > Incorporation of Biometrics API
- Standards alignment
 - > 20 Logical Channels support

Agenda



- Java Card 2 Platform

Java Card 3 Platform

Java Card 3.0 Specifications

- **Launched March 31 2008**
- Two stand-alone “Editions” for Java Card 3.0 specifications
- **Connected Edition**
 - > Includes all new network-oriented features
- **Classic Edition**
 - > Leverages the existing Java Card 2.x platform architecture
 - > For the more resource-constrained devices
- Both Editions are backward compatible with previous versions and share key security features

“Classic” Edition Features

- Traditional split VM
 - > resource efficient, 16-bit on-card VM
 - > off-card conversion for applet size optimization : CAP files
 - > on-card or off-card byte code verification
 - > on-demand Garbage Collection
- “Classic” Java Card APIs
 - > Incremental extension of Java Card 2.2.2 platform framework
- APDU-based communication
 - > Contact or contactless



“Connected” Edition features

- Embedded web server with Java Servlet API support
 - > Service static and dynamic content via HTTP(s)
- Multi threaded environment
- Concurrent communication over USB, ISO, contactless
- Client & Server communication
- Full backward compatibility



Java Card 3.0 Features – Specifications

Java Card 3.0 Specifications

• Connected Products

- > Network-oriented
- > High-speed interface
- > Larger memory



• Classic Products

- > Traditional card architecture
- > APDU – based
- > Constrained memory

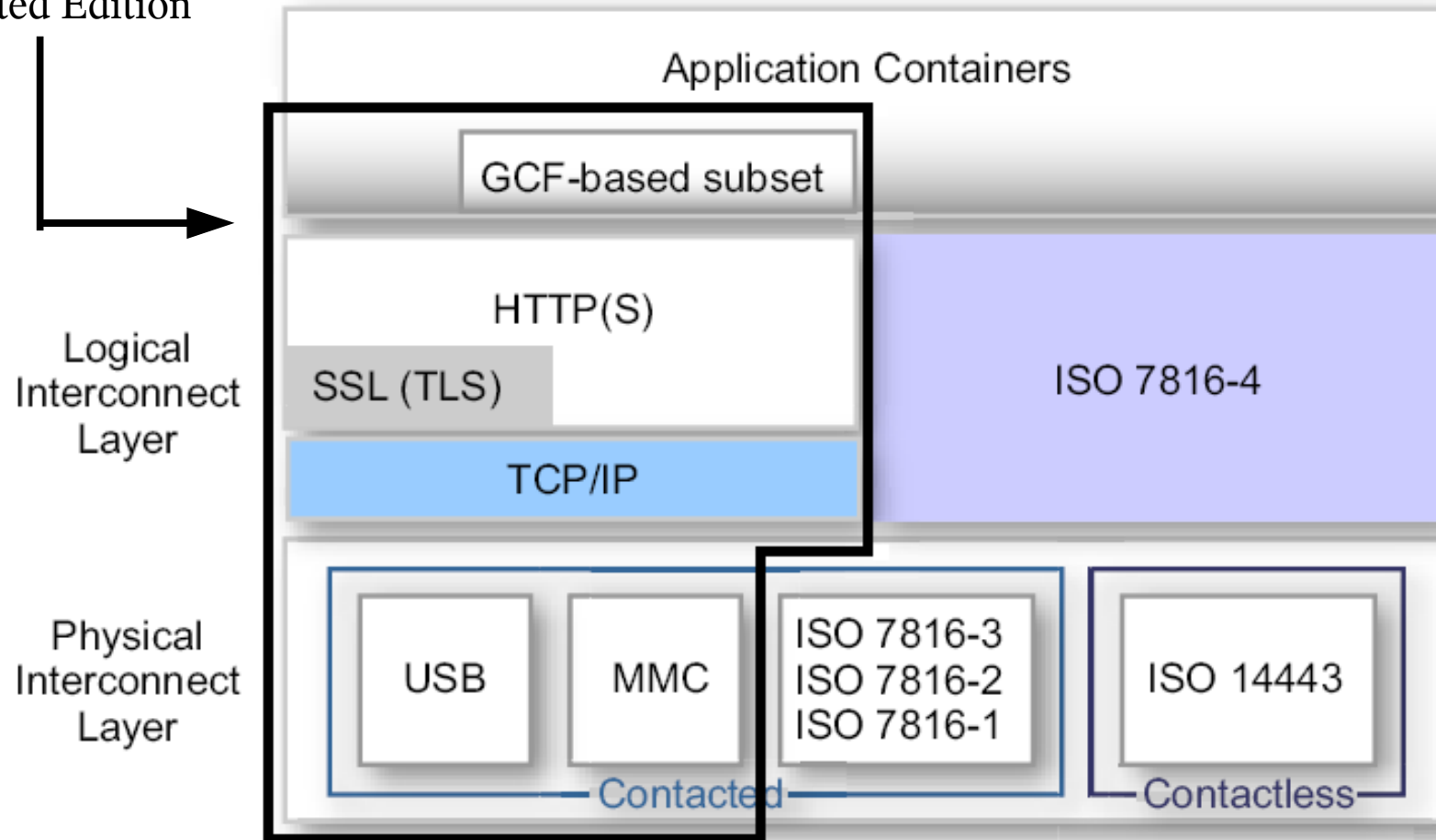


<ul style="list-style-type: none"> > HTTP Webserver > Generic Comm. Framework > Client mode 	<ul style="list-style-type: none"> > Java Card Security Features > Cryptography > Backward compatibility
<ul style="list-style-type: none"> > String, char, long > Multi-dim. arrays, collections > Event Framework 	
<ul style="list-style-type: none"> > 32 bit, KVM-level VM > Concurrent app execution > .class loading, automatic GC 	
<ul style="list-style-type: none"> > APDU-based communication 	
<ul style="list-style-type: none"> > Incremental evolution of the Java Card FW 	
<ul style="list-style-type: none"> > 16 bit, JC 2.x-level VM > Off-card conversion > Single threaded 	

Java Card 3.0

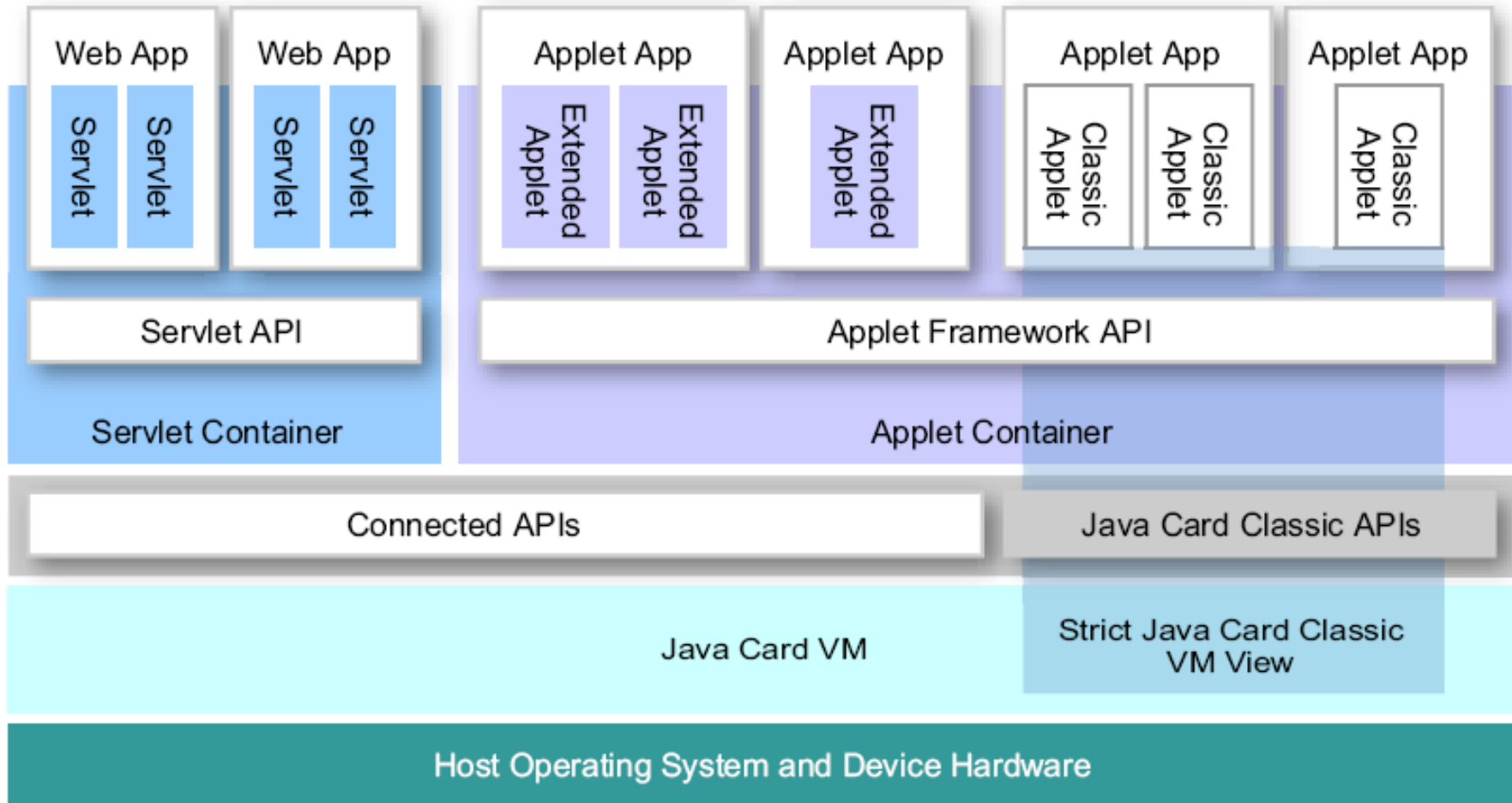
Connectivity Layers and Protocol Stack

New In Java Card 3.0
Connected Edition



Java Card 3.0

High Level Architecture



Thank You

Sebastian Hans
sebastian.hans@sun.com