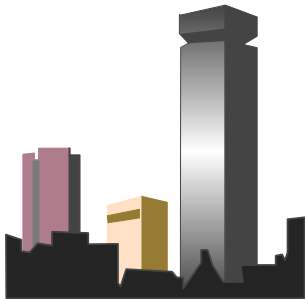

J/eXtensions for Financial Services (J/XFS) for the Java™ platform Conceptual Overview



Note



This presentation gives an overview of the upcoming J/XFS standard. All information given here was contributed by the members of the J/XFS Forum on the issues discussed as of the date of publication. It is furnished for informational purposes only and is subject to change without notice. The J/XFS Forum and its members makes no warranty, expressed or implied, with respect to this presentation. Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. All other trademarks are trademarks of their respective owners.

Mission

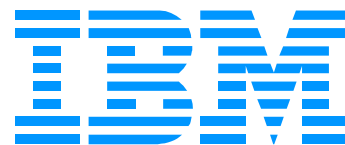


**Produce a Banking Industry standard for
a banking financial I/O device subsystem
that supports 100% pure Java applications
while leveraging existing standards.**

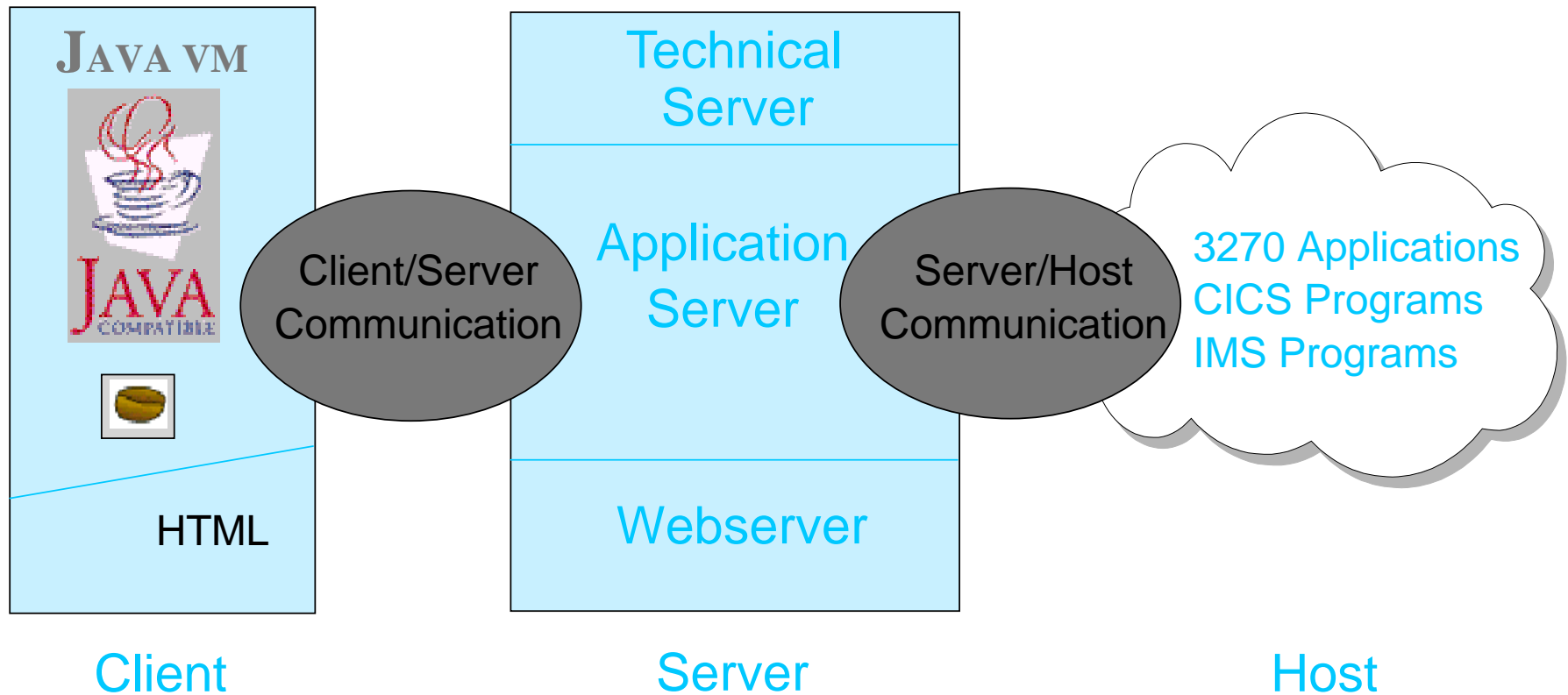
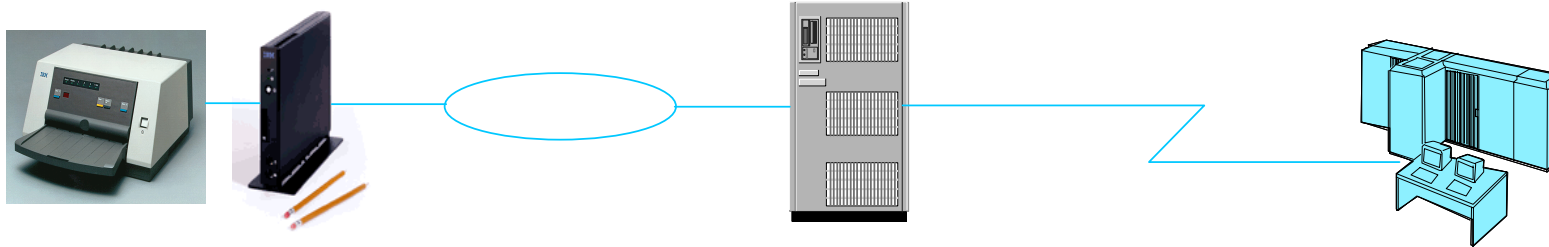
J/XFS Forum Members



DeLaRue



Scenario



Objectives



- Enable the access to banking peripherals for new Java banking applications.
- Provide a migration path for current financial I/O implementations.
- Ensure co-existence between 'old' and 'new' applications at the client level.
- Support for thin clients
- Platform neutrality

Objectives ...cont'd



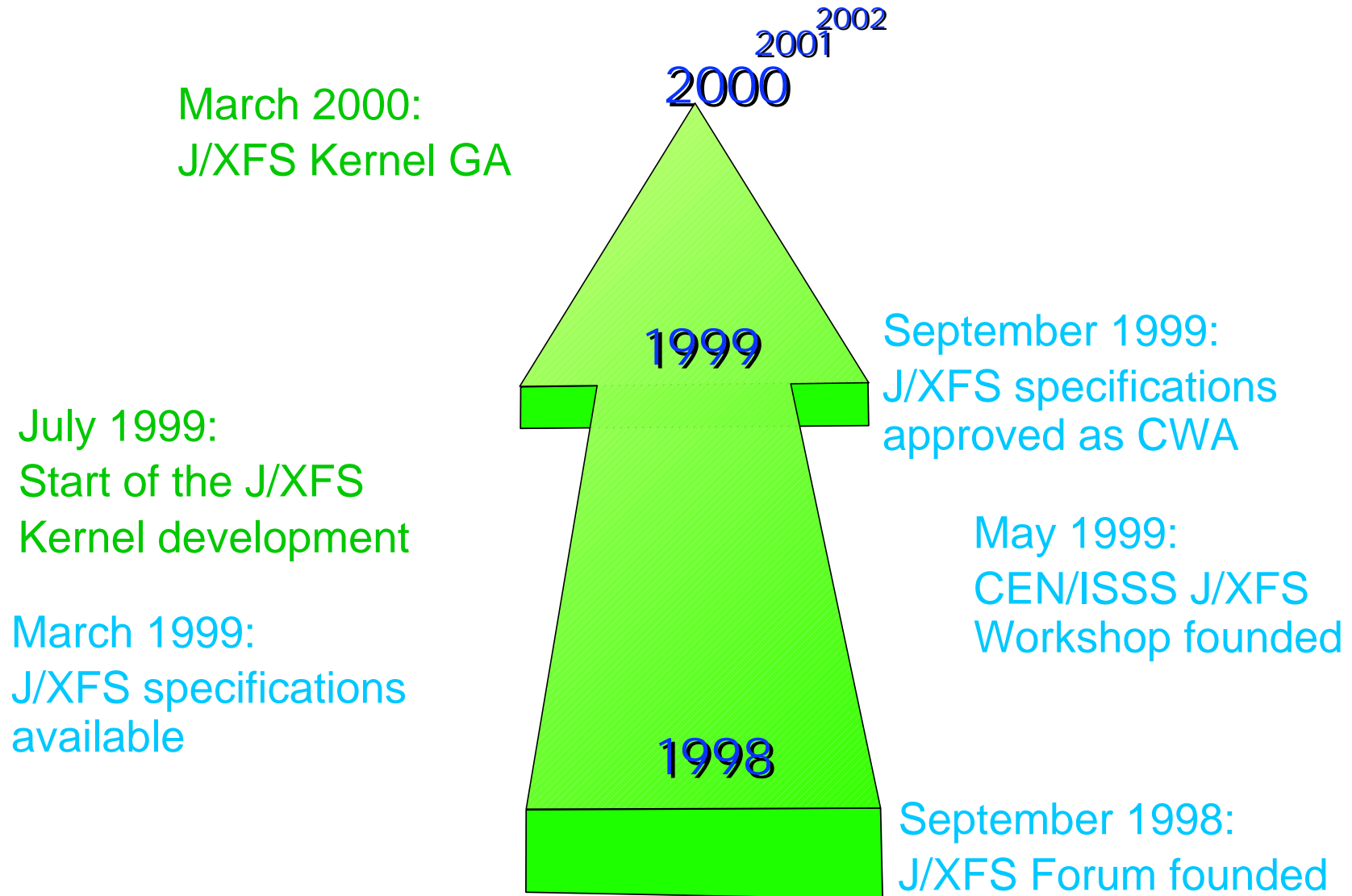
- **Protect customer investments in current infrastructure of banking devices.**
- **Enable full transparency between the application and the device level.**
- **Provide a flexible and extensible solution for the network computing paradigm.**

Goals

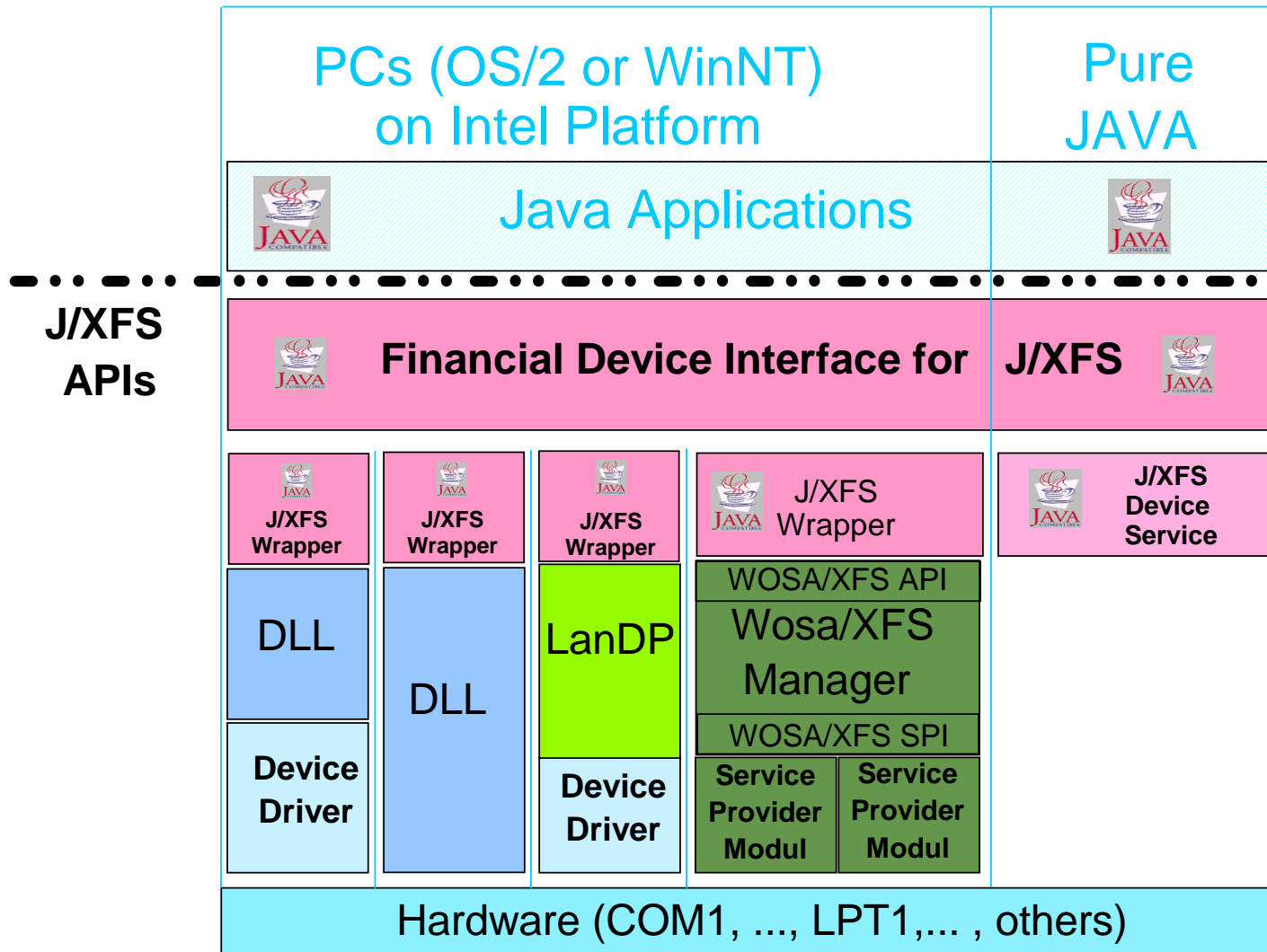


- Definition of an architecture for financial I/O device access from Java applications.
- Define a set of FIO device interfaces (APIs).
- Derive the Java APIs from existing standards (e.g. WOSA/XFS, JavaPOS).
- The J/XFS APIs are platform independent but specific to the Java programming language.

Milestones of the J/XFS project



J/XFS solution overview



Supported Devices



The J/XFS API specifications cover the following device types:

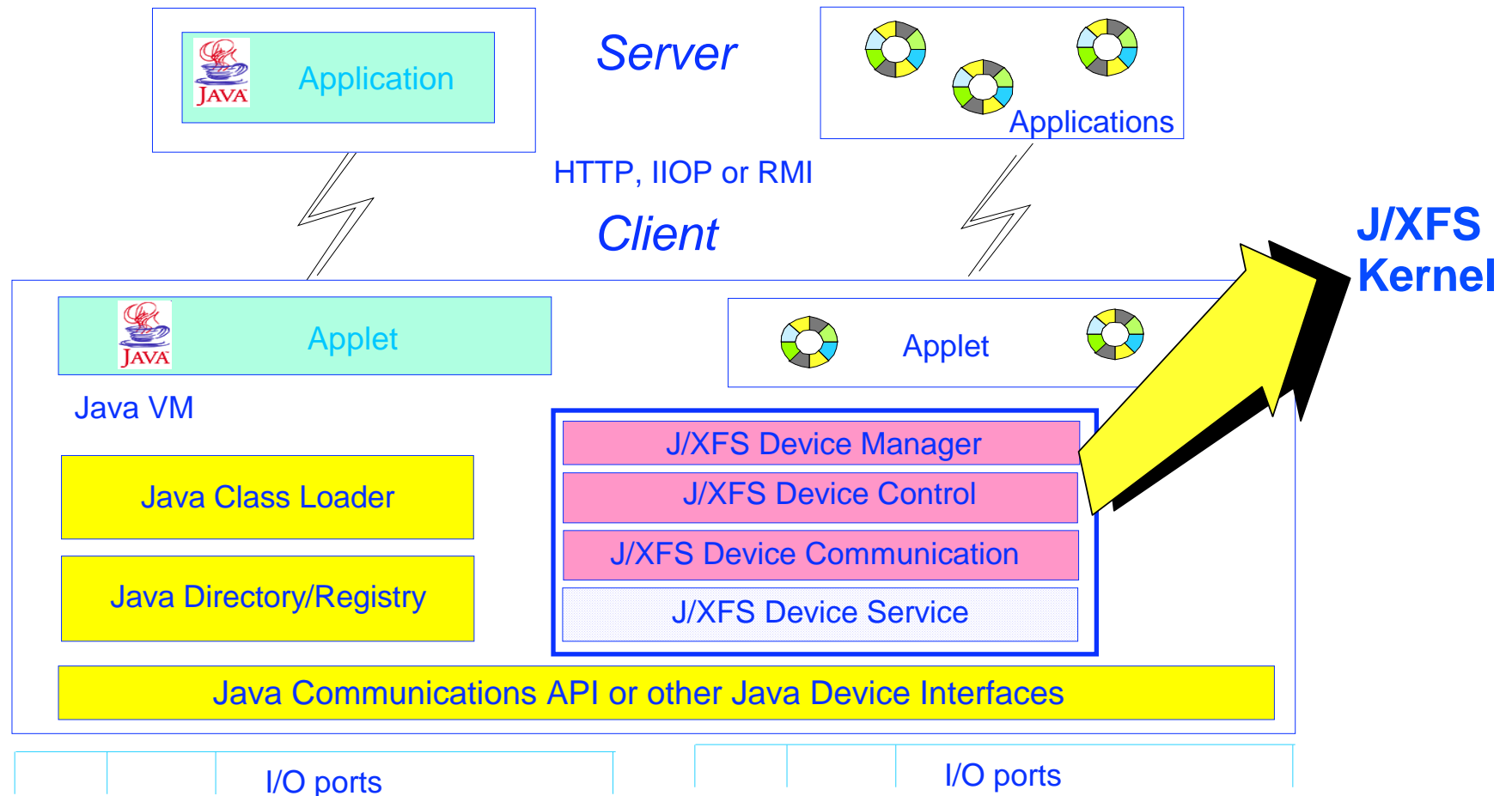
- **Printer Devices**
 - Receipt-, Journal-, Passbook-, Document-Printer, Scanner
- **Cash Dispenser/Recycler and ATM**
- **Pin Pad Device**
- **ID Card**
 - Chip Card and Magnetic Stripe Devices
- **Text I/O Device**
- **Alarm Device**
- **Depository Units**
- **Check Readers and Scanners**
- **Sensors and Indicators**
- **Cameras**

Specifications for Sorters and Counters as well as other banking devices will be covered in subsequent versions of the specifications, based on industry and customer requirements.



- The J/XFS solution in a pure Java environment will run on every platform where a Java Virtual Machine is present without requiring any OS specific banking pre-requisites (e.g. financial I/O subsystems).
- Support for the Java Comm API or other Java Device Interfaces within the JavaVirtual Machine of the various operating systems (e.g. OS/2, WindowsNT, Unix etc.) is needed.
- Availability of a J/XFS Kernel implementation consisting of
 - J/XFS Device Manager.
 - J/XFS Device Control.
 - J/XFS Device Communication.
 - J/XFS Server with Basic Configuration Tool
- Availability of a J/XFS compliant Device Service for the particular banking device, to be provided by the manufacturer.

J/XFS Client Overview



J/XFS Kernel Implementation



Implementation of the full J/XFS standard:

- full J/XFS API support
- remote device access for device sharing via network
- exchangeable communication layer (TCP/IP)
- Server-based repository
- Administration component

Written in 100% pure Java



Primary OS platforms: WinNT, OS/2 and Linux

Supported scenarios:

- standalone systems
- PC-based LAN environments
- Network Computer (Applets; only JavaVM plus network connection, no local storage)

Channels

- Retail, Self Service, Call Center, Homebanking

J/XFS Kernel Deliverables



▶ Server package

- ▶ contains all serverside parts incl. Basic Configuration Tool and Repository

▶ Client package

- ▶ contains all clientside infrastructure, the Device Controls, and the communication classes

▶ J/XFS Development Kit

- ▶ Sample Device Service, Sample application, Detailed usage descriptions for application programmers and Device Service programmers

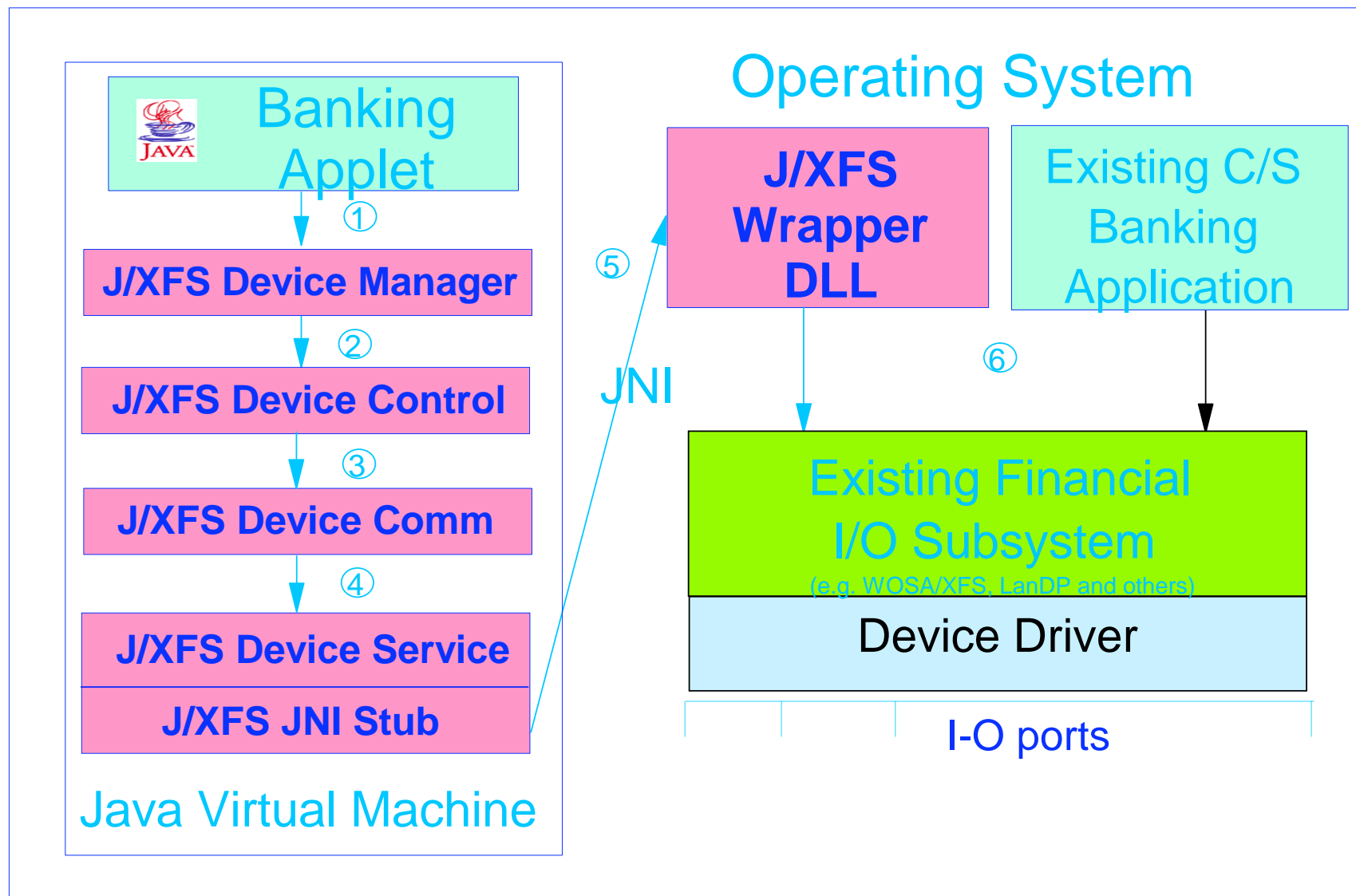
▶ GA March 28, 2000

Pragmatic Approach for the Wrapping Solution



- J/XFS API-conformed wrappers running in a JVM will map the calls from a Java banking application to an existing I/O subsystem on that particular operating system platform.
- These wrapper solutions will be implemented within the scope of actual projects:
 - WOSA/XFS
 - LanDP (product feature of V5)
 - other (customer specific, ISVs, etc.)
- JDK 1.1.6 functionality will be used.
- Access to native code through the Java Native Interface (JNI).
- Leveraging existing hard- and software implementations and therefore protecting customer investments while providing an evolutionary way to start with new Java applications.

J/XFS Wrapping Solution



J/XFS Wrapping Solution-Explanation



1. A Java application or applet requests a Device Control from the J/XFS Device Manager.
2. Based on the configuration information which is available upon initial load, the J/XFS Device Manager loads the appropriate Device Control for the requested peripheral.
3. The J/XFS Device Control passes all requests to a corresponding J/XFS Device Service.
4. The J/XFS Device Communication layer will only be used if a connection to a remote client is needed.
5. The initialized J/XFS DeviceService is a shadow bean which behaves as a Device Service towards the J/XFS Kernel, and uses a JNI stub to pass requests through the Java Native Interface to a corresponding J/XFS Wrapper DLL, which runs under the control of the operating system.
6. This Wrapper DLL maps the J/XFS call to the appropriate I/O subsystem call, which is expected from the existing implementation running under the control and using the hardware device interface of the operating system.

Application Compliance



Application compliance means, that a Java based banking application can make a J/XFS API call to a specific kind of banking device and the J/XFS implementation as well as the appropriate Device Service will understand the call correctly and all defined functions and behaviour as defined in the J/XFS specifications are supported. The application itself must not care about the real hardware access, the transparency is achieved by the J/XFS Device Service in conjunction with the J/XFS implementation.

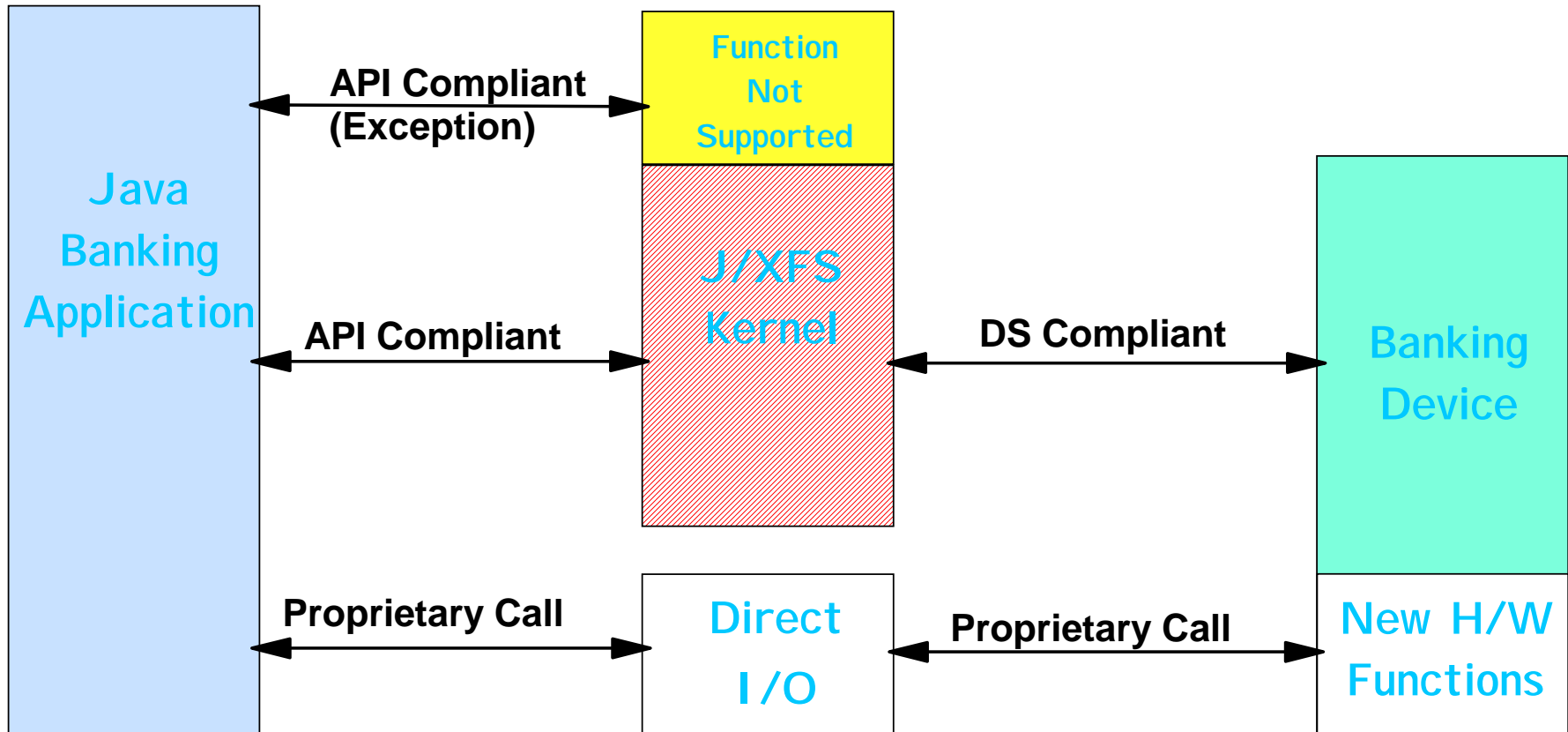
Device Compliance



Device compliance means, that a J/XFS Device Service written in Java is able to understand the appropriate J/XFS API calls correctly and to support all functions and behaviour as defined in the J/XFS specifications. For a Device Service to be J/XFS compliant, the vendor has to support this functionality, if the interface for that device type is defined in the J/XFS specifications.

The Device Service itself will handle the real hardware access in a transparent manner to the J/XFS implementation and the Java banking application, regardless of the specific OS platform the JVM runs on.

Compliance At A Glance



More Information



For more information see the
web:

<http://www.jxfs.com>