
ORACLE®

JavaOne™

ORACLE®

Oracle Java for TEE
Enhanced Security for Mobile Applications

CREATE THE FUTURE

Dmitry Barinov
SecureKey

Michele Sartori
Oberthur

Eric Vétillard
Oracle

JavaOne™ ORACLE

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Speakers

- **Dmitry Barinov**
  - CTO, SecureKey
- **Michele Sartori**
  - System Security Engineer, Oberthur
- **Eric Vétillard**
  - Sr. Principal Product Manager, Oracle

Program Agenda

1. Oracle Java for TEE
2. Developing applications with the SDK
3. SecureKey: Application and return from experience
4. Oberthur: The OJTEE backend
5. Closing and Q&A

Program Agenda

1. Oracle Java for TEE
2. Developing applications with the SDK
3. SecureKey: Application and return from experience
4. Oberthur: The OJTEE backend
5. Closing and Q&A

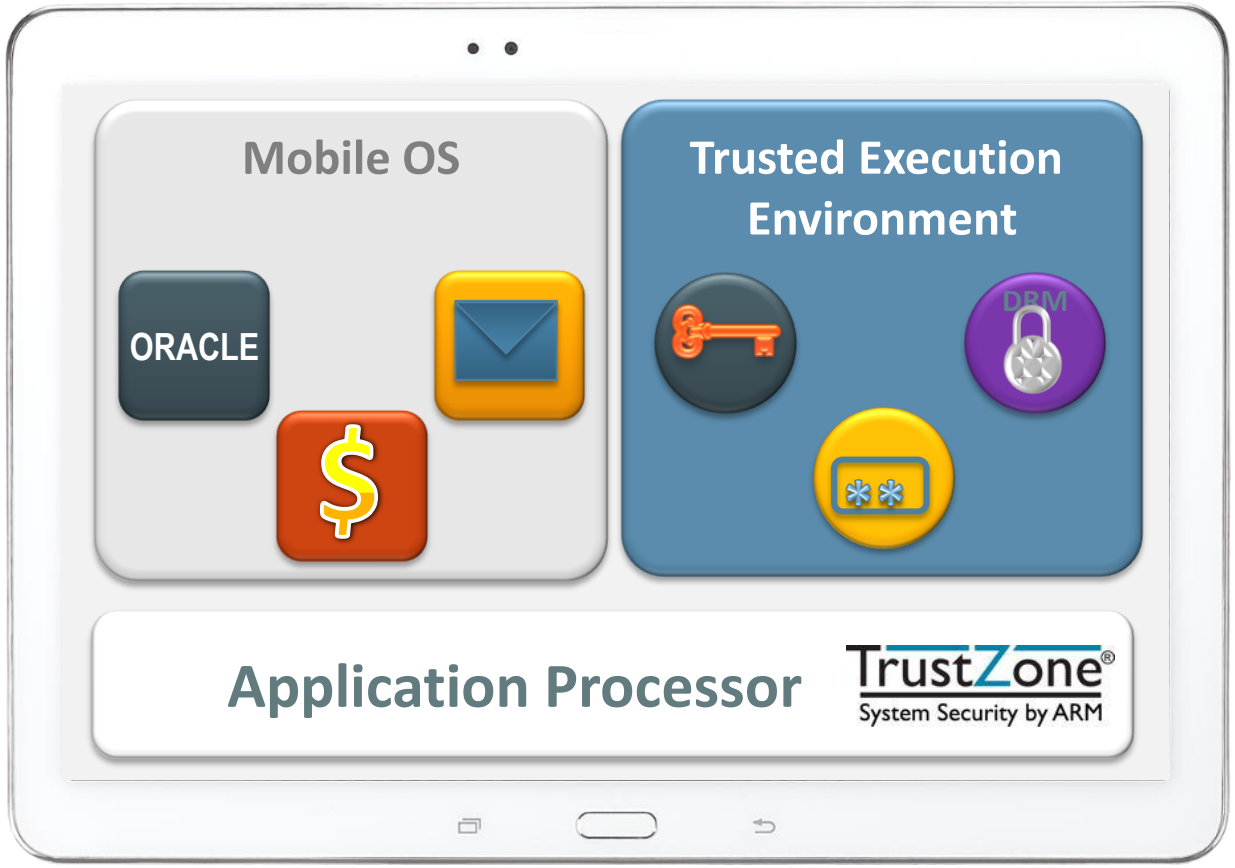
The need for mobile security

- Mobile apps have become pervasive
  - Lots of games and fun stuff
  - More and more serious stuff
    - Money, enterprise data, personal data
- Mobile apps are becoming targets
  - Phishing
  - Malicious apps
  - Attacks on stored data
- Some questions are open
  - How do I protect my user's credentials?
  - How do I get non-repudiation?
  - How do I get beyond passwords?
  - How do I protect my company's data?
- Good answers exist
  - They require a security foundation

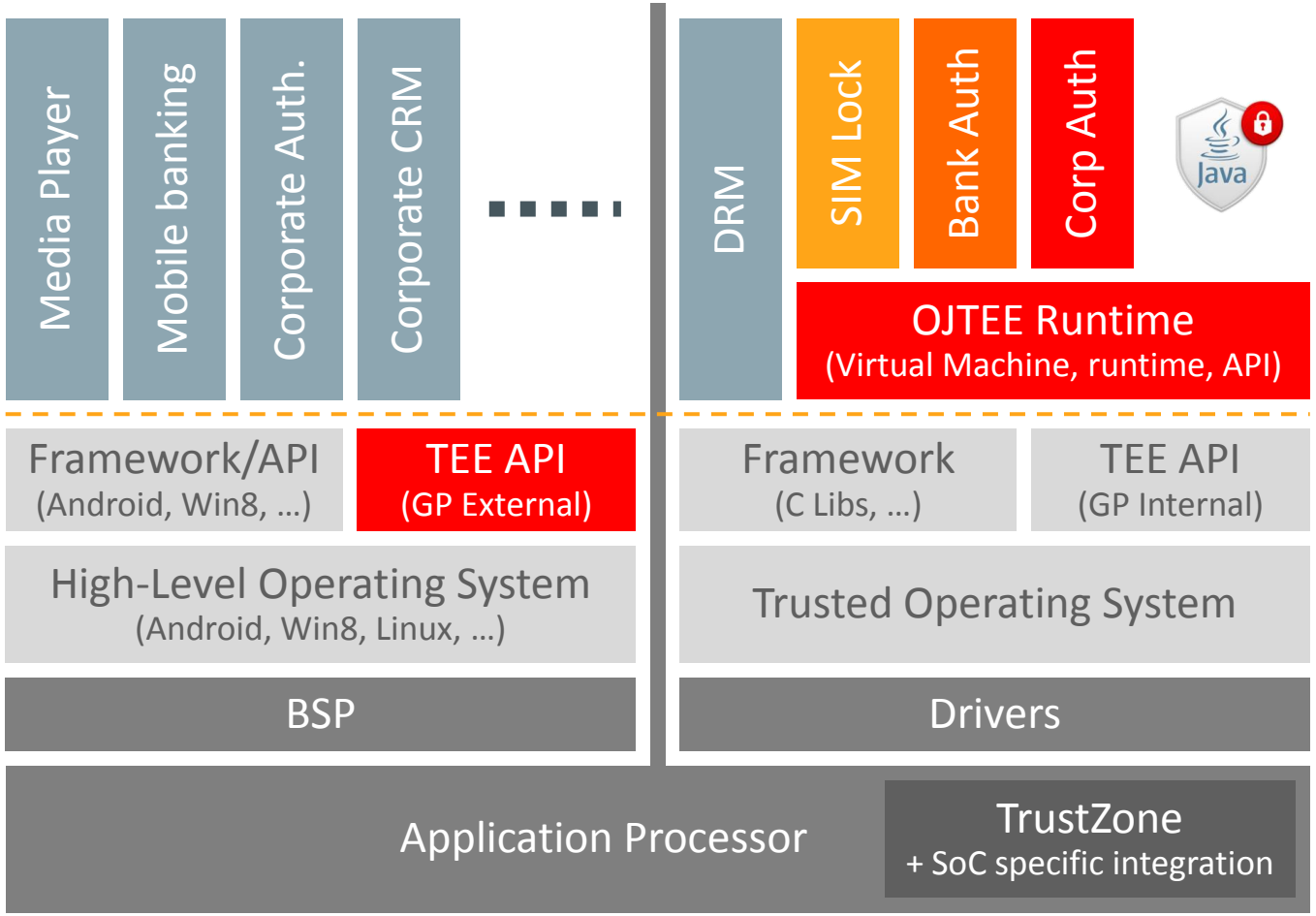
What is a TEE ?

A Trusted App Environment within a mobile processor

- Separated from the mobile OS with a hardware-based protection
- Runs in parallel with the standard mobile OS
- Storage & management of secrets and credentials, away from logical attacks
- Secure communication channel with the backend



Oracle Java for TEE architecture



- OJTEE introduces binary-level portability for Trusted Apps (TAs)
  - Abstracts native OS & HW from Services
  - Reduced porting and integration costs
- OJTEE offers an open and flexible deployment model for Trusted Apps
  - Flexible deployment infrastructure
  - All interfaces will be standardized, allowing competition to exist at all levels
- OJTEE is a separated environment
  - Specific Java APIs and VM
  - A constrained development environment

Technical Benefit of Java TEE

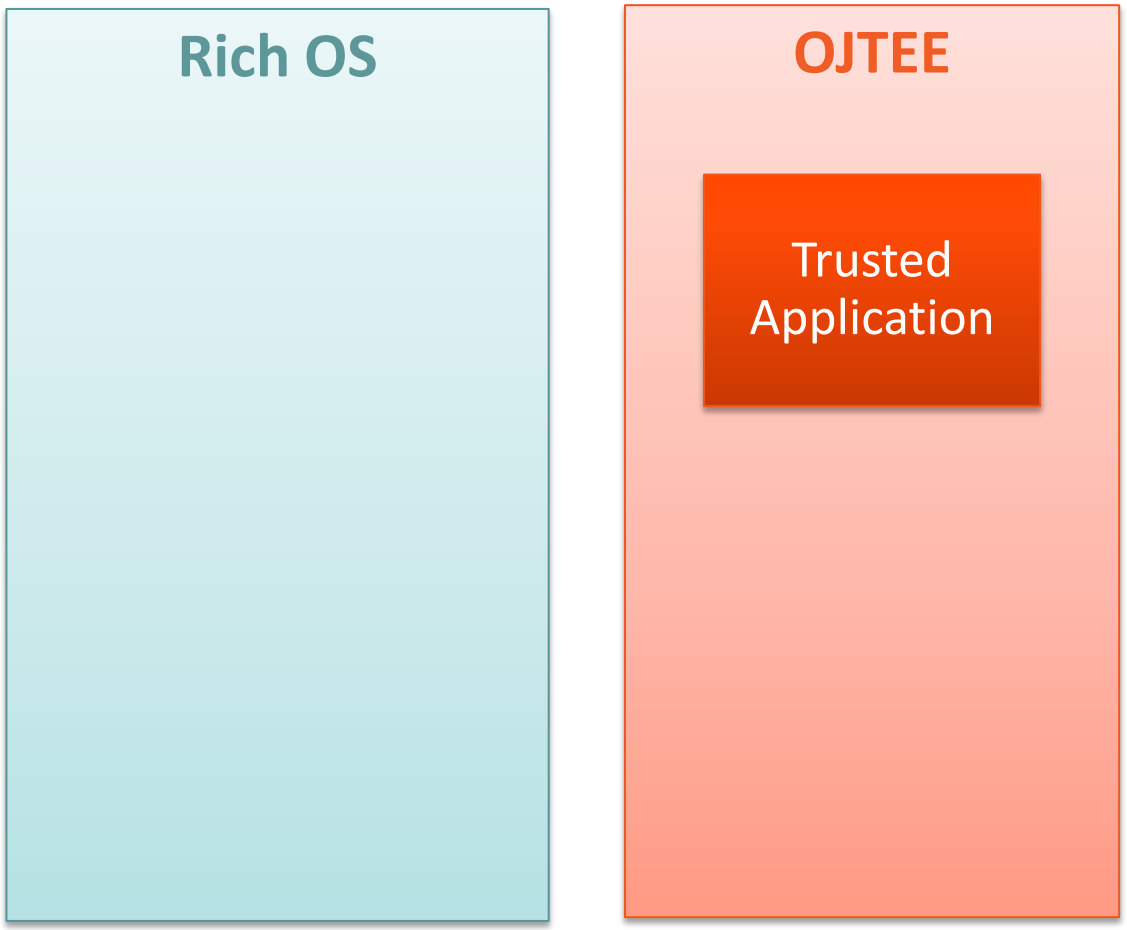
A unique blend of security and flexibility for secure mobile service

- **Portable applications** across devices and architectures
  - Open specifications, free and powerful Java development tools
  - Full interoperability guaranteed, up to binary format
  - Reduced testing and certification efforts
- **Dynamically manageable** trusted services
  - Interoperability simplifies the deployment and lifecycle management of security services
- Proven **security**
  - Shares principles with Java Card platform, deployed on billions of devices
  - Market-tested security model

Program Agenda

1. Oracle Java for TEE
2. Developing applications with the SDK
3. SecureKey: Application and return from experience
4. Oberthur: The OJTEE backend
5. Closing and Q&A

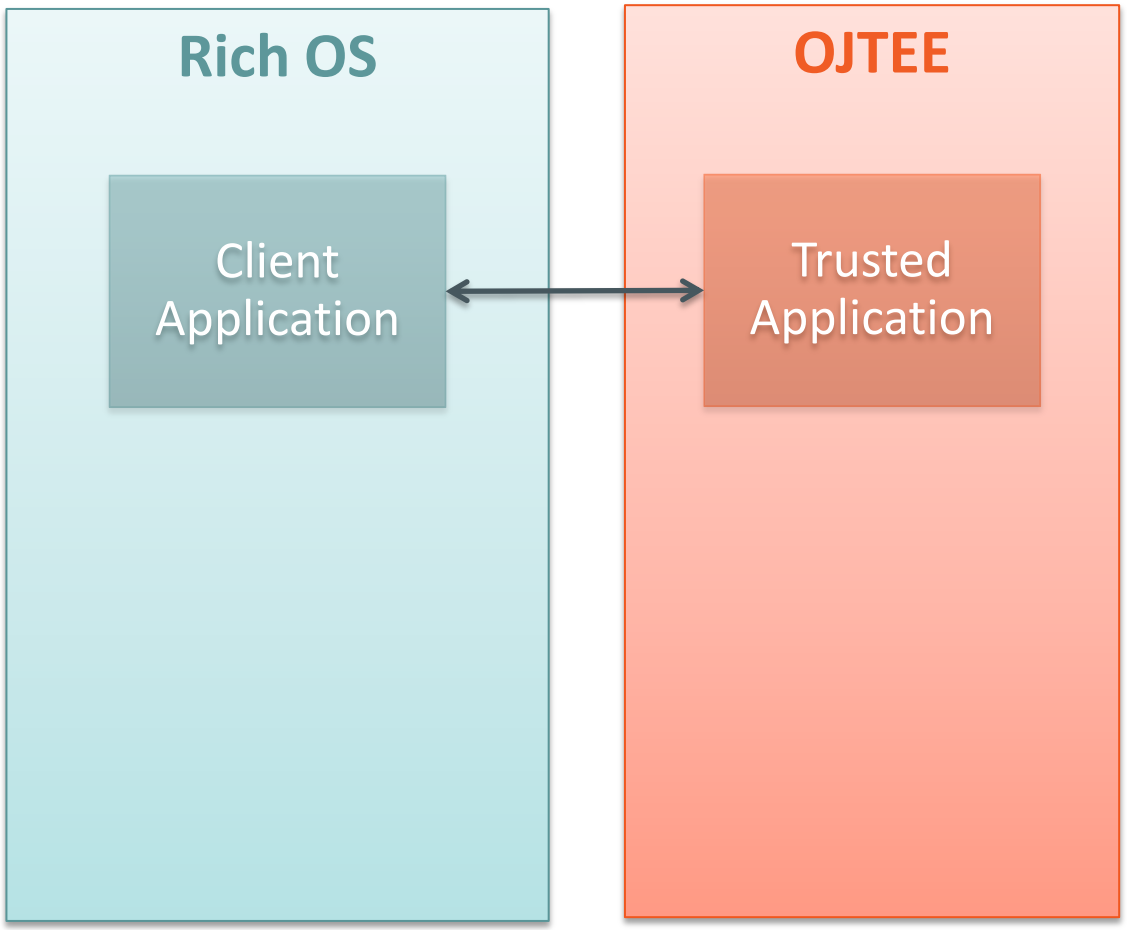
So, what do we develop?



A Trusted Application

- Implements basic security functions
  - Protecting sensitive assets
  - Making crypto computations
  - Interacting with user thru Trusted UI
- Very limited in scope
  - Only includes sensitive operations
- Built on OJTEE
  - Using a specific Java runtime

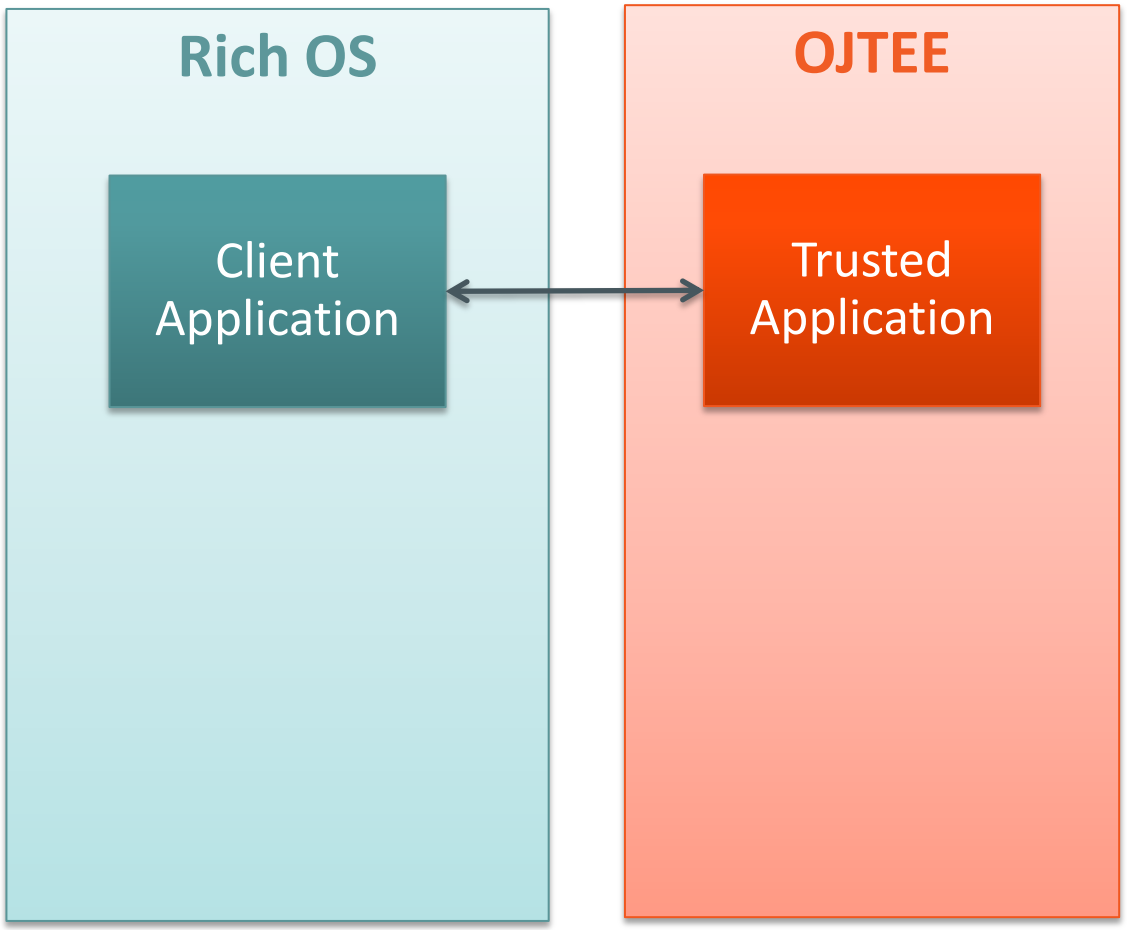
So, what do we develop?



A Service Protocol

- The interface to the TA
  - Based on a basic comms protocol
  - As defined by GlobalPlaform
- Crucial for security
  - Attack surface for the TA
  - Bad design introduces vulnerabilities
- Also important for usability
  - Practical aspects must be considered

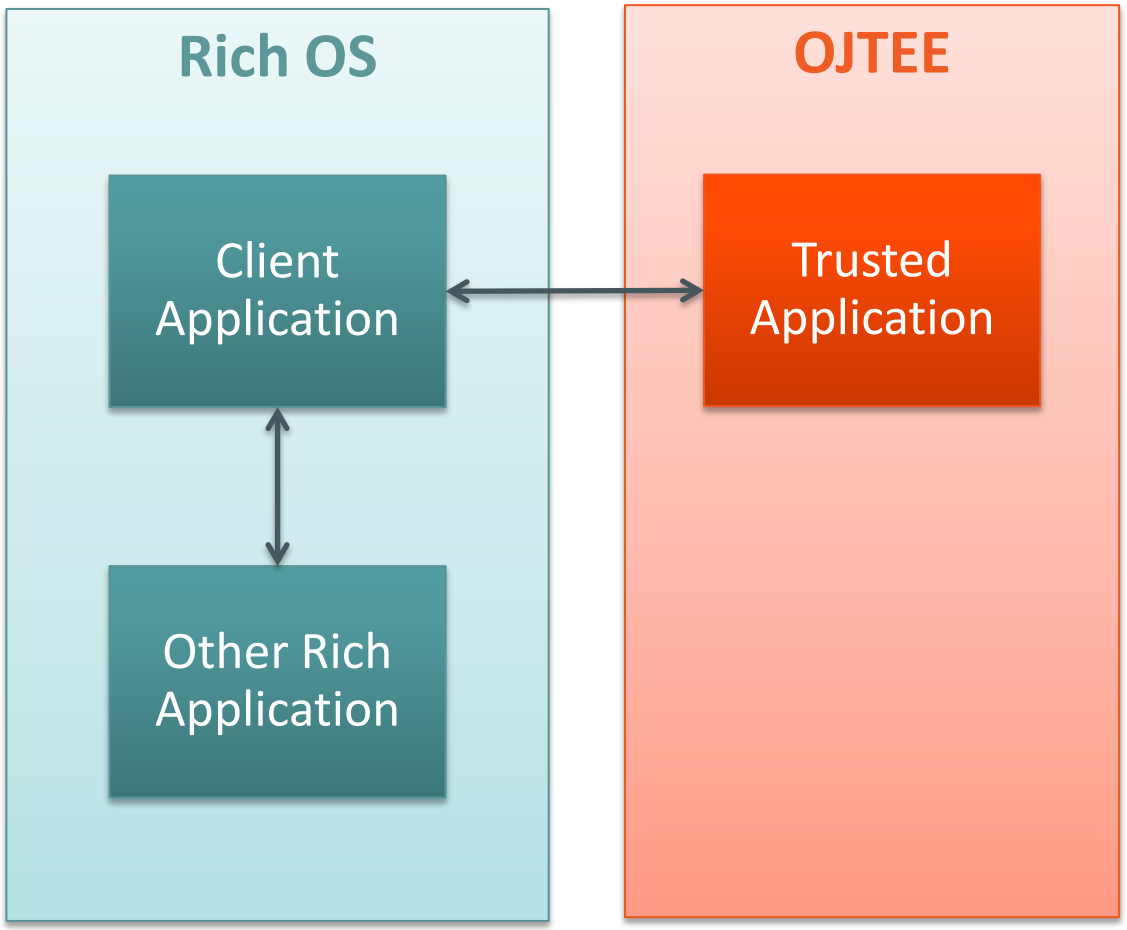
So, what do we develop?



A Client Application

- Implements the service protocol
  - From the client side
- Uses it to build a security service
  - Building on the TA's security features
  - Transforming into a practical app
- Built on a Rich OS
  - Depends on the target OS
  - For now, we support Android

So, what do we develop?



A Rich OS Service

- Making a security service available
  - Leveraging OJTEE and a TA
  - Hiding all complexity
  - Providing a high-level service
- Must be very simple
  - Targeting all mobile developers
  - Providing access to security functions

What skills are required?

Trusted Application	• Strong security skills • Oracle Java for TEE skills
Service Protocol	• Strong security skills • Previous protocol design experience
Client Application	• Some security skills • Some knowledge about Trusted Application management
Rich Application	• No specific security skills • No knowledge of Oracle Java for TEE

SDK Tools for Developers

TA Development

- An IDE plug-in
  - Ready to use with Eclipse IDE and Android Development Toolkit
  - With generation of specific binary format
  - With basic Trusted Application Management

TA Debug and Test

- Some simulators
  - Running on Windows and Android
  - Including Trusted UI simulation
- Some testing utilities
  - Client API extensions
  - Trusted UI automation

Client Development

- A client API
  - Available on Windows and Android
- A basic management API
  - To perform basic management operations

SDK Demo

JavaOne
ORACLE

Program Agenda

1. Oracle Java for TEE
2. Developing applications with the SDK
3. SecureKey: Application and return from experience
4. Oberthur: The OJTEE backend
5. Closing and Q&A

Briidge.net Connect

Protection for all your service channels on all customers touch points

Provides the best user experience AND the strongest authentication.

- ✓ No OTP Entry
- ✓ Enables sign on and high value transfers
- ✓ Strong Assurance

Provides strong 2FA for your mobile app while making the app easier to user

- ✓ Transparent device authentication
- ✓ Fully unique Device ID
- ✓ QuickCode for easy password entry

Leverages strong credentials, secure and trusted execution environments and proximity protocols

Connect features

GOALS:

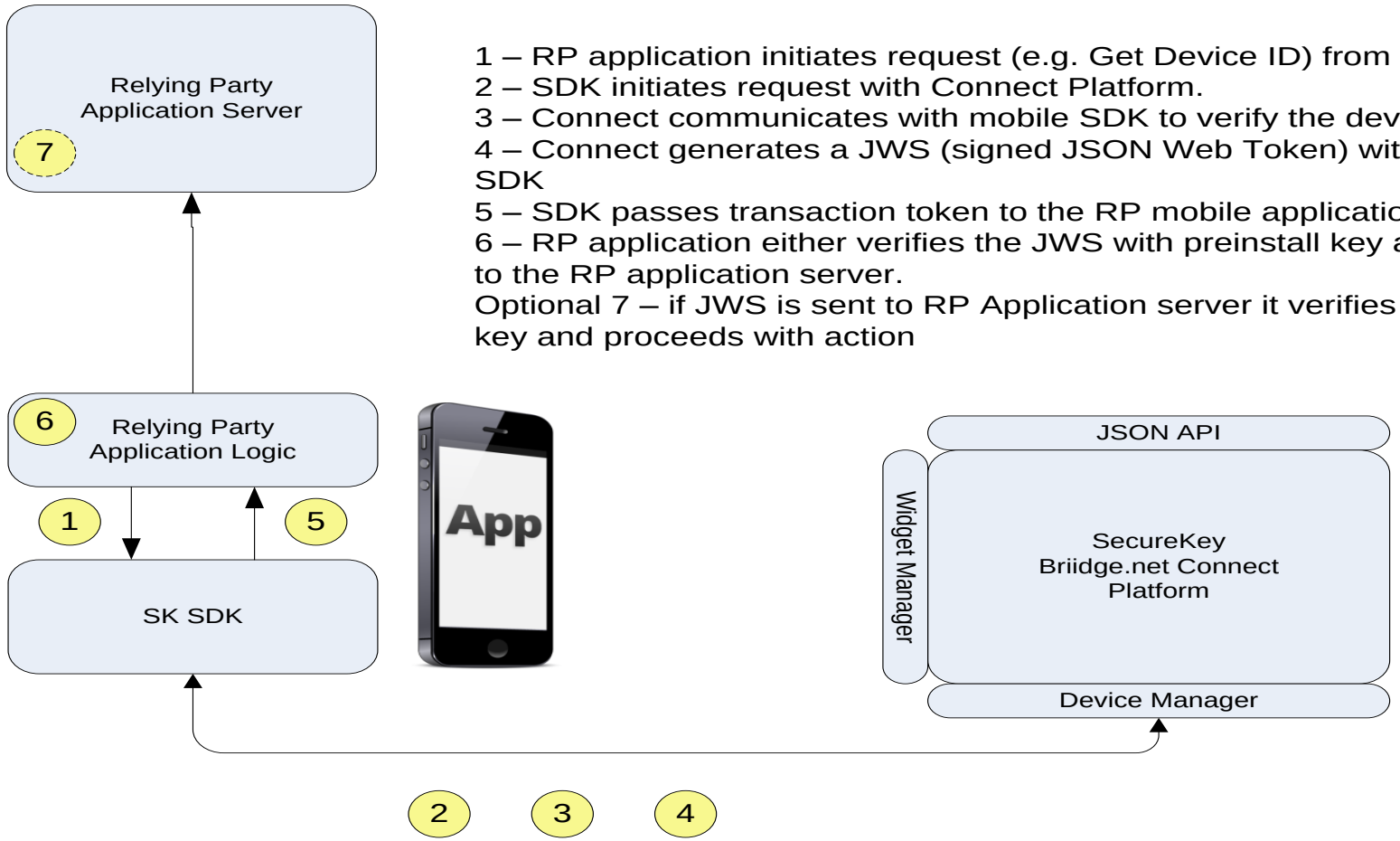
- Enable device as a strong credential or its proxy
- Provide easy integration for Relying Parties that consume assertions and attributes

- Strong anchor for device verification and User authentication.
- Server issued assurance.
- Runs on consumer devices:
  - Devices with hardware/embedded SE/TEE.
  - In software (mobile, PC).
  - In browsers.

- Additional features:
  - Biometric authentication – iOS, Android
  - Remote notification.
  - Card read and emulation.
  - Attribute management.
  - Modern protocol support (OATH, OpenID Connect, JWT).
  - On device signing

Solution Architecture:
Transaction Flow – Device verification

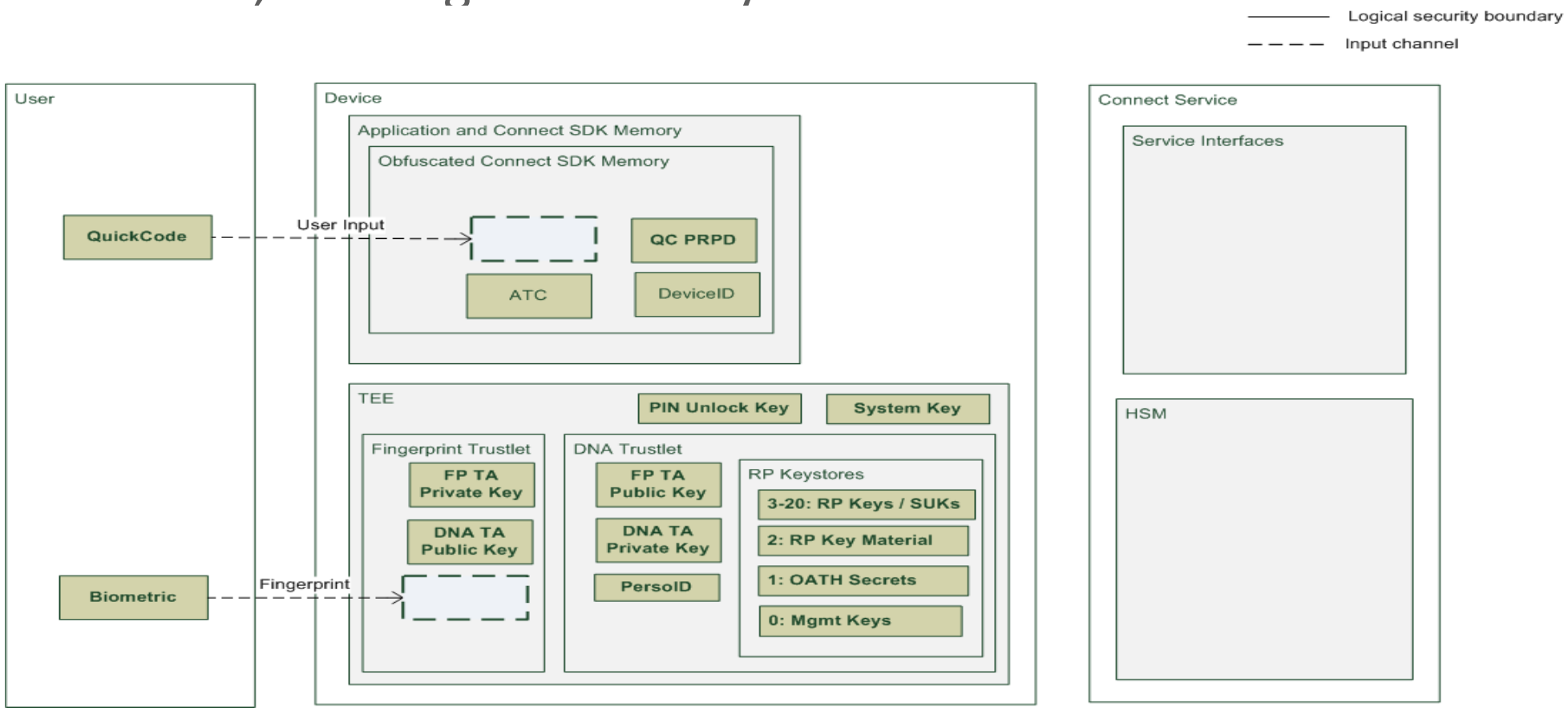
Passing Assertion to the client



1 – RP application initiates request (e.g. Get Device ID) from SDK.
2 – SDK initiates request with Connect Platform.
3 – Connect communicates with mobile SDK to verify the device credential.
4 – Connect generates a JWS (signed JSON Web Token) with assertion of verification and passes it on to SDK
5 – SDK passes transaction token to the RP mobile application.
6 – RP application either verifies the JWS with preinstall key and proceeds with action or passes the JWS to the RP application server.
Optional 7 – if JWS is sent to RP Application server it verifies the JWS with pre-exchanged verification key and proceeds with action

Security Features

- Keys, Identifiers, and Logical Security Domains



Leveraging the TEE

- Mobile apps with crypto all have same potential security issues in app sandbox. Exposure to malware is an ongoing concern.
- TEE provides secure execution environment
  - Enables secure storage of keys.
  - Requires higher privilege than other apps to access.
  - Enables gatekeeper functionality via TEE services.
- The above allows for interesting use cases:
  - Strong identity claims
  - Facilitation of payment tokens,
  - Biometric- enabled signatures.
- Simplification of development efforts (e.g. Is WBC really needed in TEE?)

Lessons Learned

- Code size limitation for a package is 64K, not yet documented.
- Debugging within the libraries is not available
- Logging for debug purpose is not available in TA.
- More than one package in same project is not supported, not yet documented
- Calling a TEE api within an inner class gives verifier error, after moving the class out works fine.
- Documentation about the important build related settings is in build.xml and jtee-ant.xml files.
- EA3 has all new development settings, setting up the custom built process is bit tricky
- EA3 added an Android simulator, provides real time working scenario
- TEE Internal API - Method required to retrieve Try counter in the Authenticator interface.

Program Agenda

1. Oracle Java for TEE
2. Developing applications with the SDK
3. SecureKey: Application and return from experience
4. Oberthur: The OJTEE backend
5. Closing and Q&A

OT Offers End-to-end Solutions

oberthur TECHNOLOGIES
THE M COMPANY

DIGITAL IDENTITY

SMART TRANSACTIONS

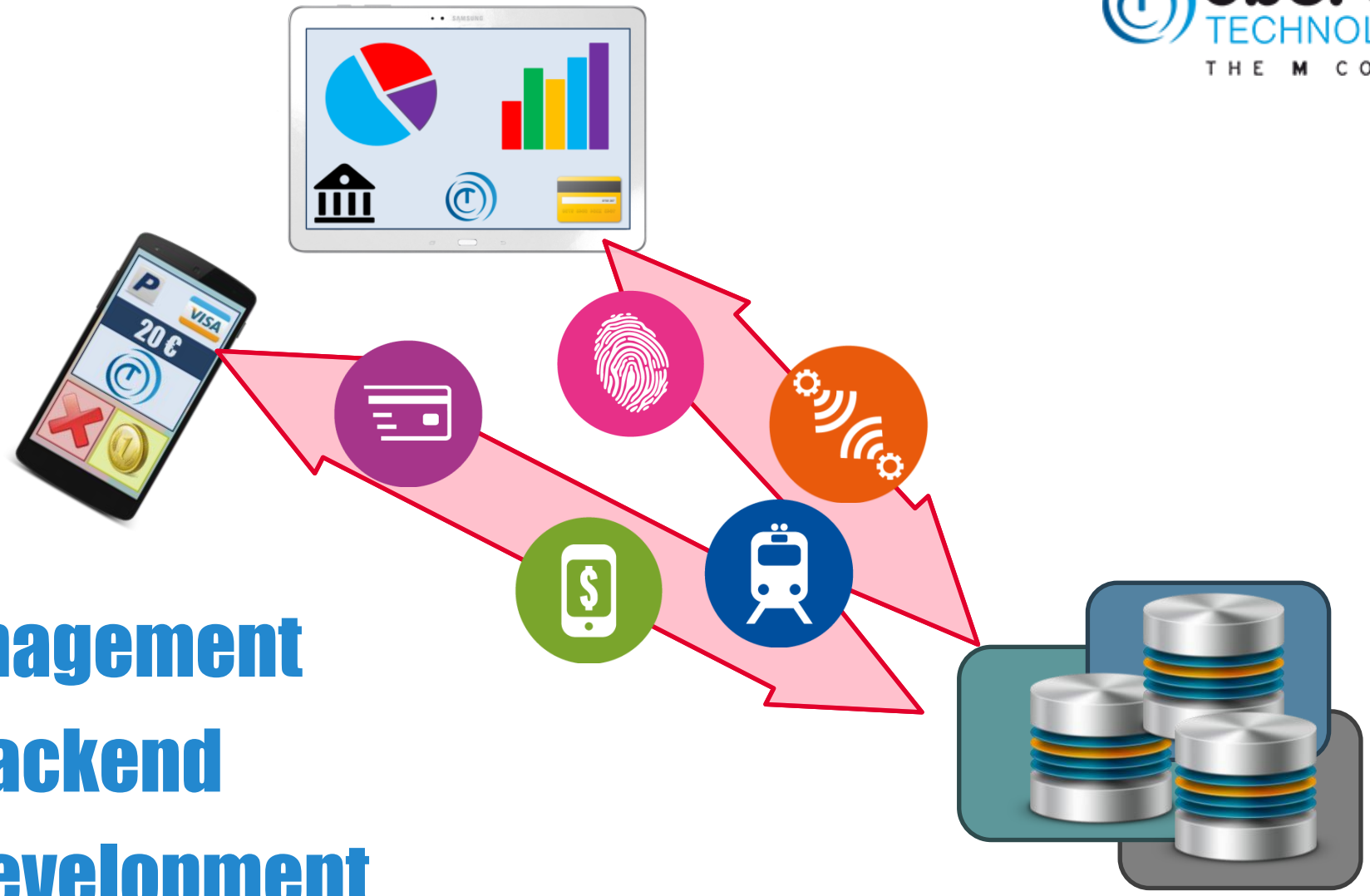
TRANSPORT AND ACCESS CONTROL

MACHINE-TO-MACHINE

MOBILE FINANCIAL SERVICES

JavaOne ORACLE

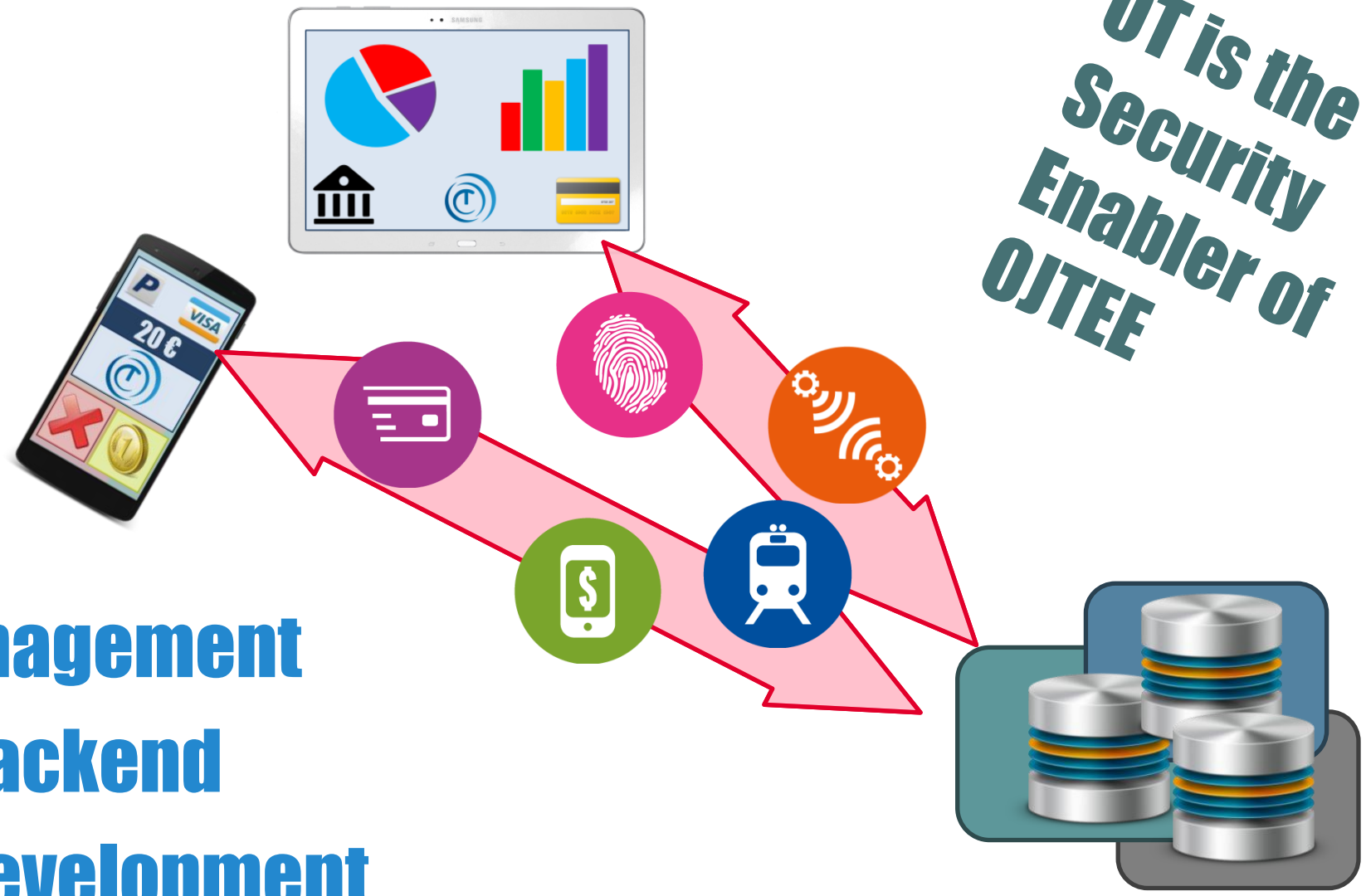
oberthur TECHNOLOGIES
THE M COMPANY



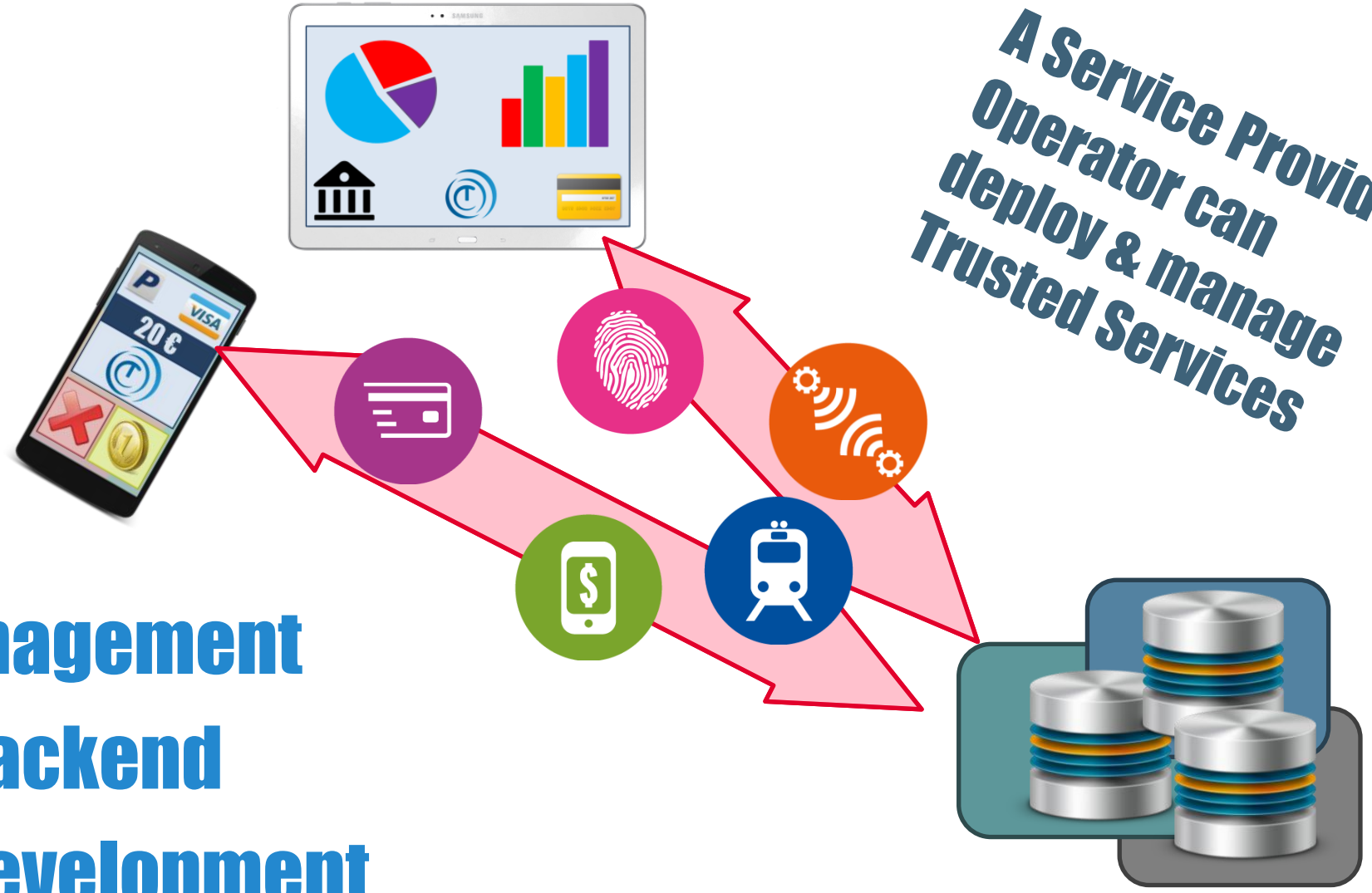
- Device Management
- Services Backend
- Services Development

JavaOne ORACLE

OT is the Security Enabler of OJTEE

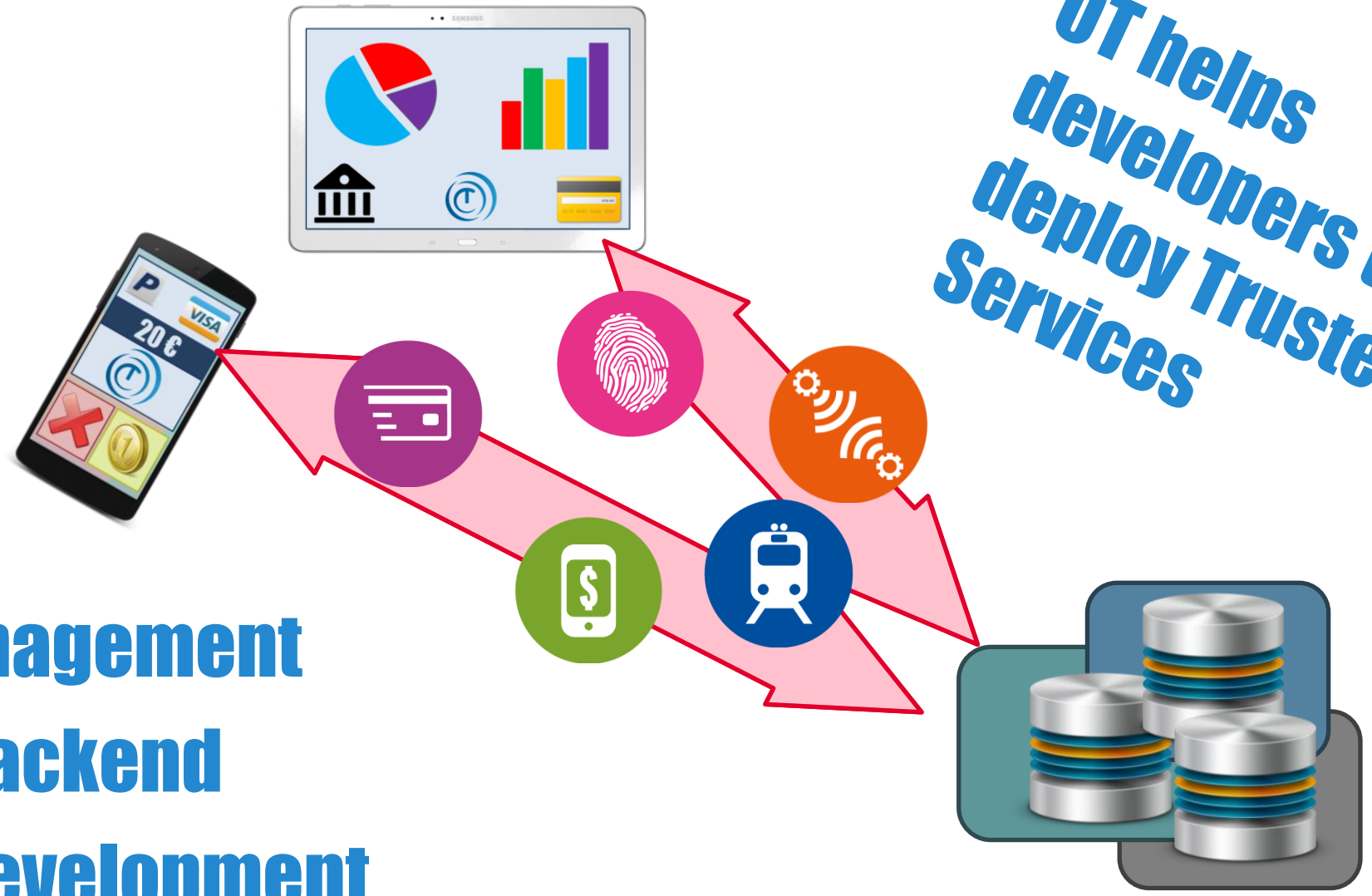


- Device Management
- Services Backend
- Services Development



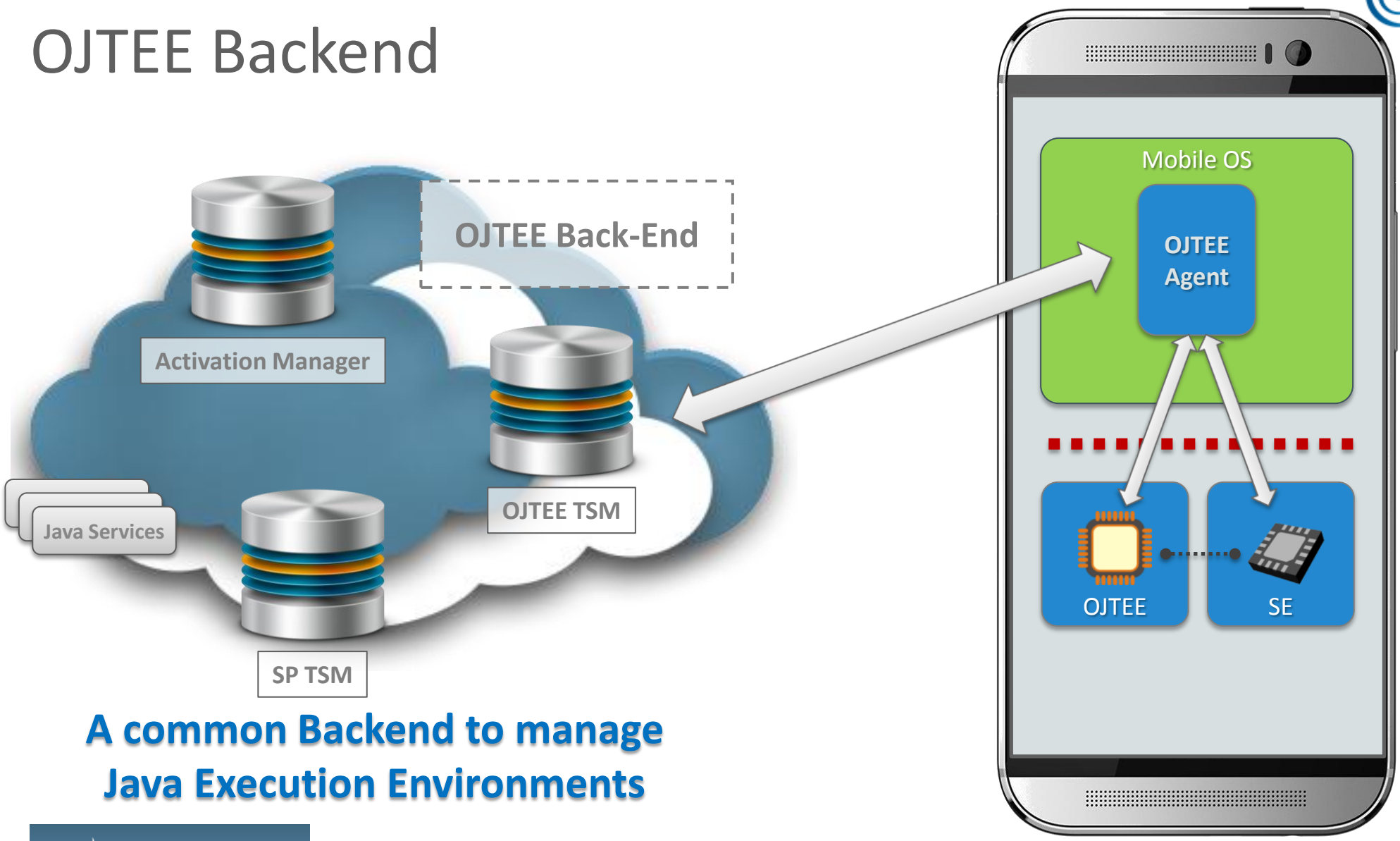
- Device Management
- Services Backend
- Services Development

OT helps developers to deploy Trusted Services



- Device Management
- Services Backend
- Services Development

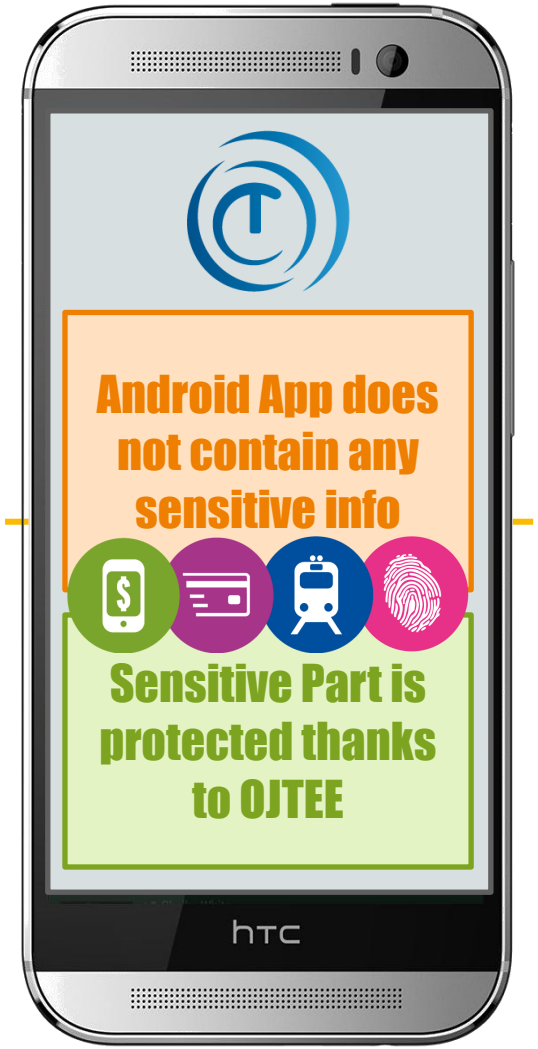
OJTEE Backend



A common Backend to manage Java Execution Environments

The Developer Perspective

Trusted Application Development

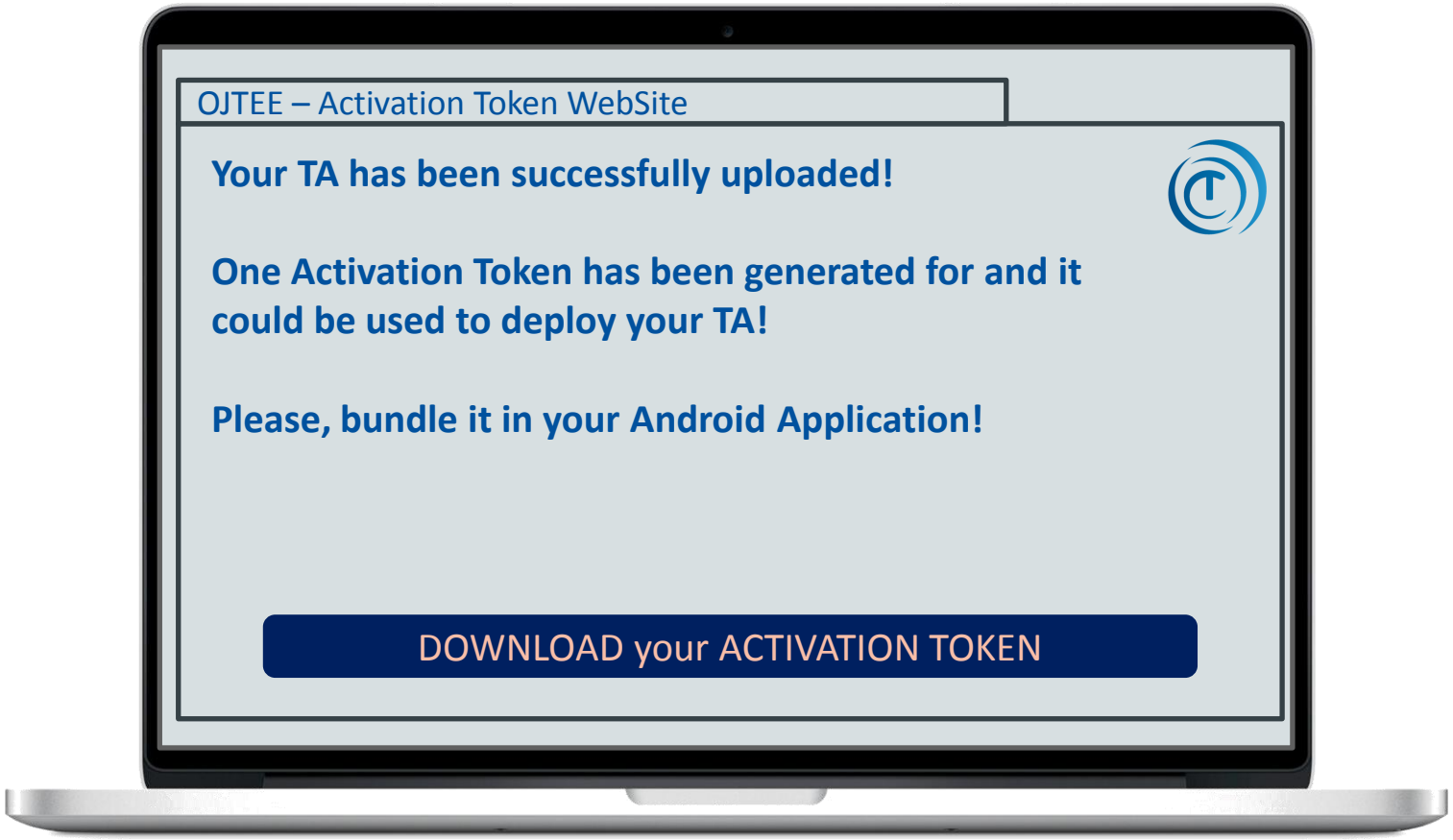


- Configuration files
- Images / Videos
- General data

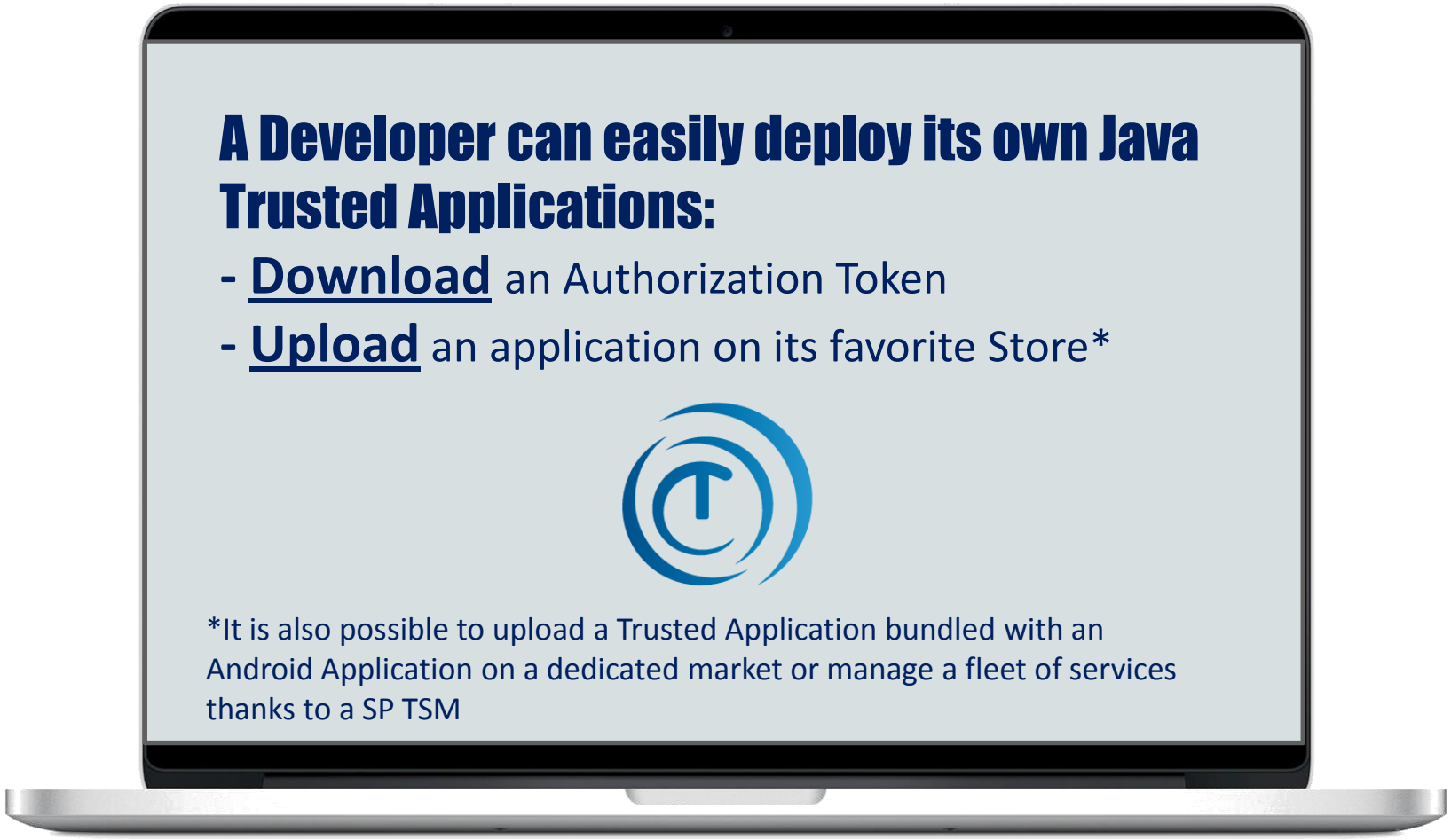
- Trusted User Interface
- Secure Storage of Sensitive data
- Cryptographic algorithms

Trusted Application Verification

Trusted Application Activation Token



Trusted Service Activation and Deployment



OJTEE Backend Control Panel

The User Perspective

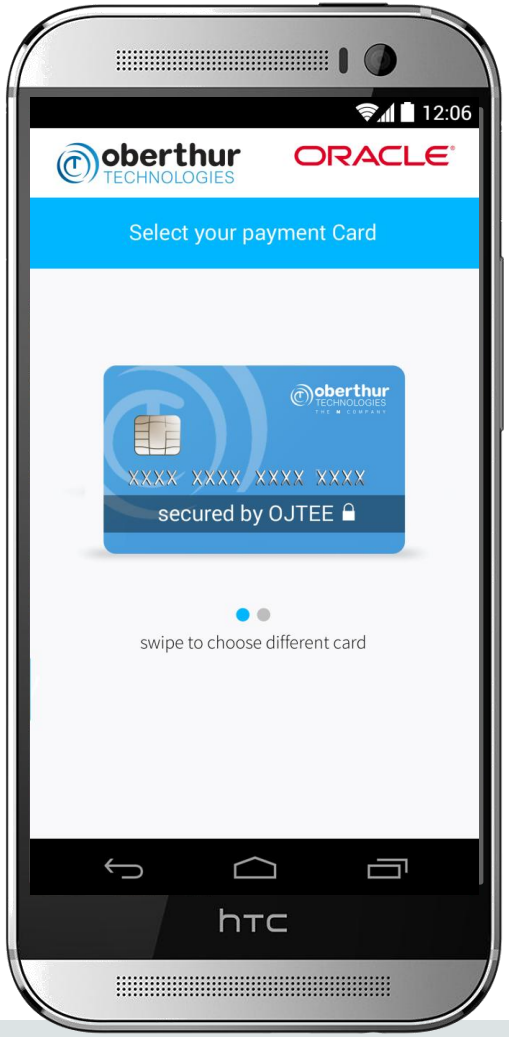
Trusted Application Download

- Trusted Services are bundled with a classical Mobile Application
- The installation process is completely transparent for the end user

User Authentication and Trusted Input

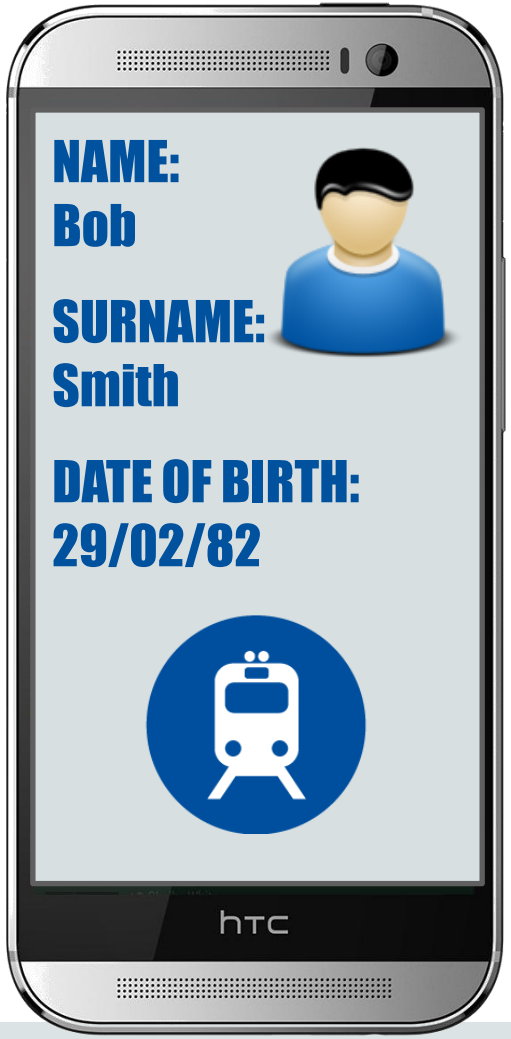
- Thanks to OJTEE, malware applications cannot intercept user inputs
- Thanks to OJTEE backend, the developer can finally trust the link between user and device

Trusted Storage



- Thanks to OJTEE, card information is stored in a secure manner
- Thanks to OJTEE Backend, card credentials can be remotely and dynamically changed

Trusted Display



- Thanks to OJTEE, information displayed on the screen can be trusted.
- Thanks to OJTEE backend, User Information can be pushed directly to a target device.

Program Agenda

1. Oracle Java for TEE
2. Developing applications with the SDK
3. SecureKey: Application and return from experience
4. Oberthur: The OJTEE backend
5. Closing and Q&A

Project Status & Call to Action

Oracle Java for TEE

- Oracle Java for TEE is under development at Oracle
  - Undergoing Beta testing with selected partners
- Objective: release progressively in the upcoming months
  - Platform details and development tools initially
  - Devices and deployment infrastructure through 2015
- Today: Come see our demos and chat with us
  - Learn about Java Card on OTN
  - Register interest for SDK and Device availability

Getting more information

At JavaOne

- CON3165. Oracle Java for TEE: Bringing Trust to Mobile Devices
  - Thursday, Oct. 2, 1:00PM - 2:00PM – Hilton – Yosemite B/C
- Oracle Java for TEE demo
  - On the DemoGrounds

Questions?

JavaOne
ORACLE

Hardware and Software
Engineered to Work Together

JavaOne™

ORACLE®


ORACLE®