PoC and Beyond— The IMS Services API (JSR 281)

Volker Bauche
Mirko Naumann

BenQ Mobile
http://www.benqmobile.com

Piotr Kessler
Stefan Svenberg

Ericsson AB
http://www.ericsson.com

Session TS-3319

Enabling the IMS Vision in Mobile Devices

Learn how JSR 281 enables creation of new multimedia services

How to combine different media and service enablers

And how to host simultaneous IP-based services

Agenda

IMS Vision and IMS Framework

Types of IMS Applications and Services

Deployment Options on Mobile Devices

IMS Functionality in Mobile Devices

Low-Level vs. High-Level APIs

JSR 281

Conclusions

Agenda

IMS Vision and IMS Framework

Types of IMS Applications and Services

Deployment Options on Mobile Devices

IMS Functionality in Mobile Devices

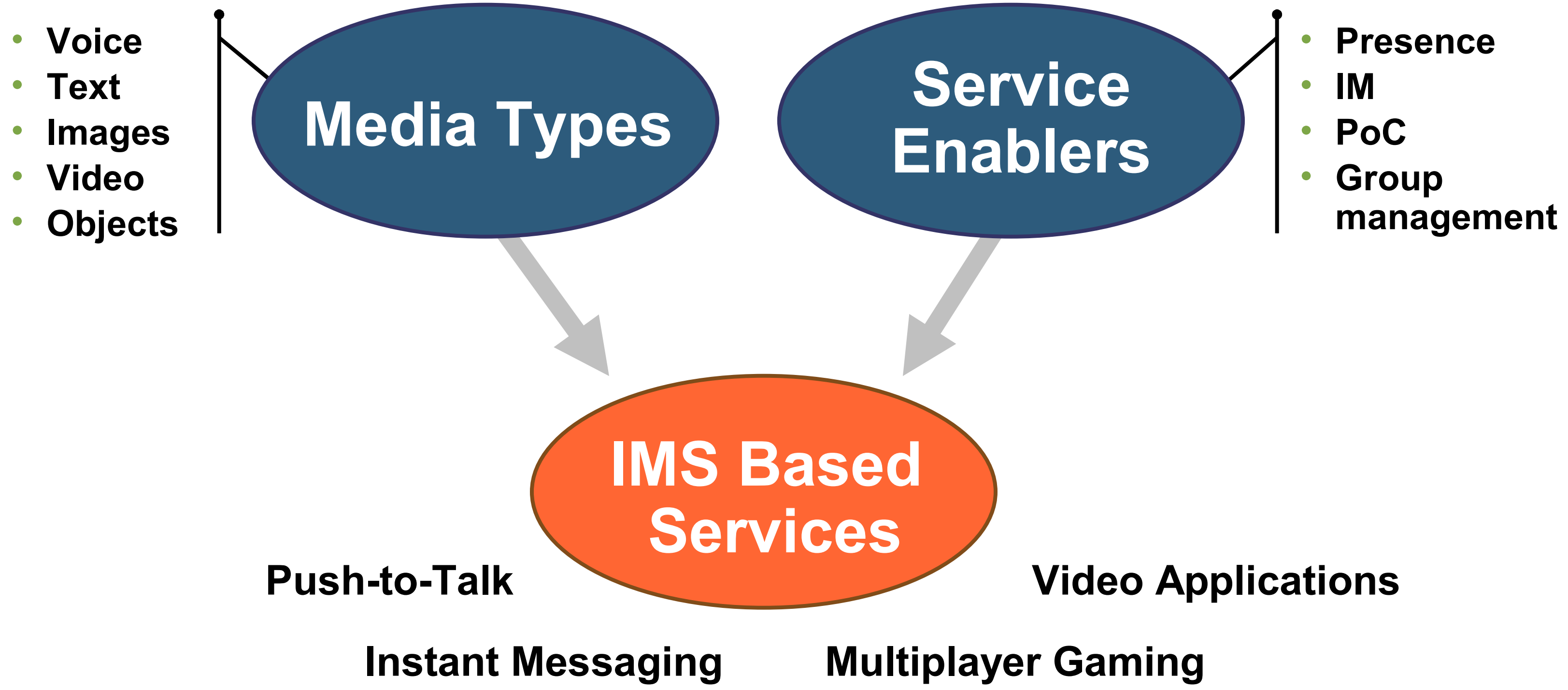
Low-Level vs. High-Level APIs

JSR 281

Conclusions

The IMS Vision

Combining different media and service enablers opens opportunities for the introduction of a host of new IP-based services



- Voice
- Text
- Images
- Video
- Objects

Media Types

Service Enablers

- Presence
- IM
- PoC
- Group management

IMS Based Services

Push-to-Talk

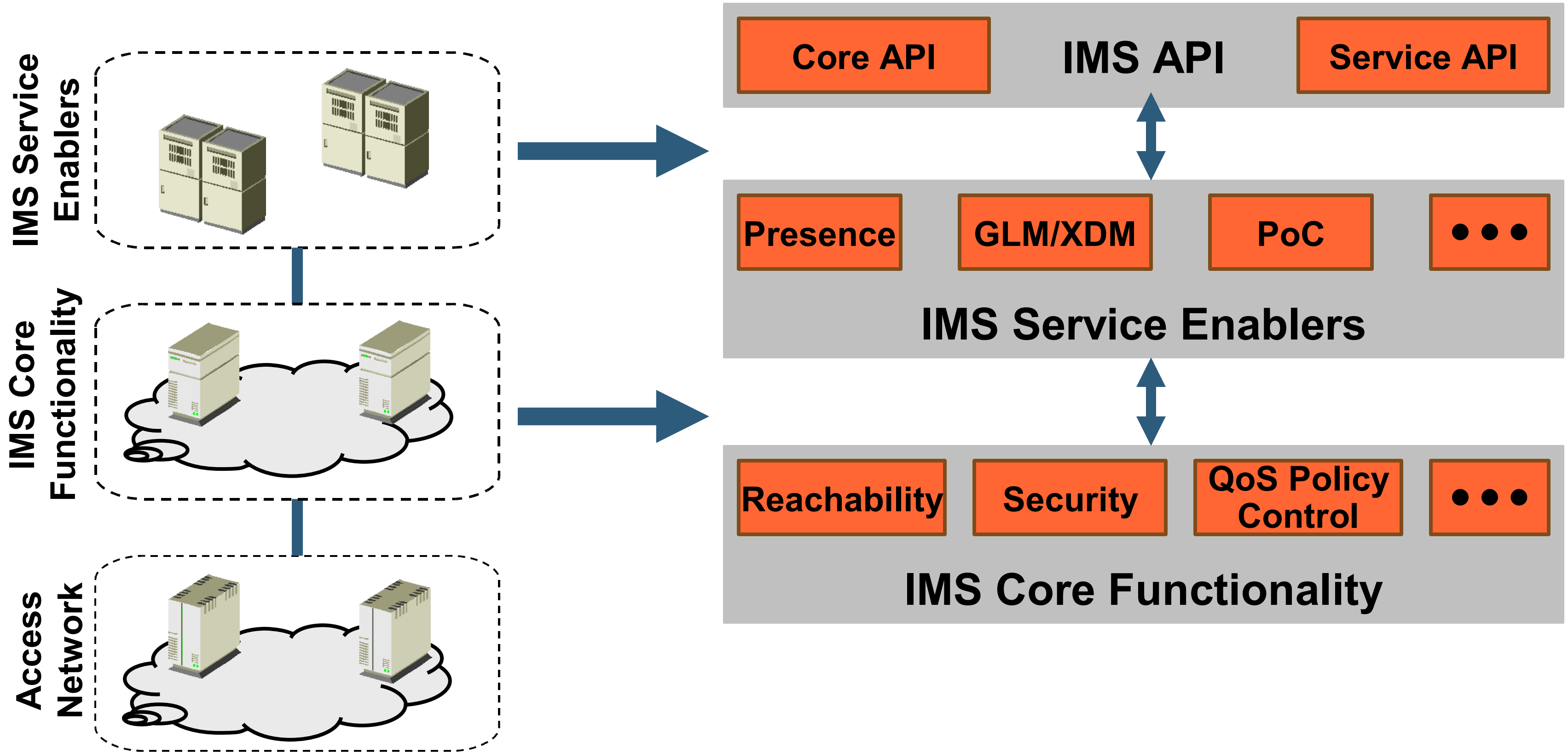
Video Applications

Instant Messaging

Multiplayer Gaming

IMS Framework

The IMS consolidates common functionality and supports standardized service enablers; the actual services are built on top of this framework



Agenda

IMS Vision and IMS Framework

Types of IMS Applications and Services

Deployment Options on Mobile Devices

IMS Functionality in Mobile Devices

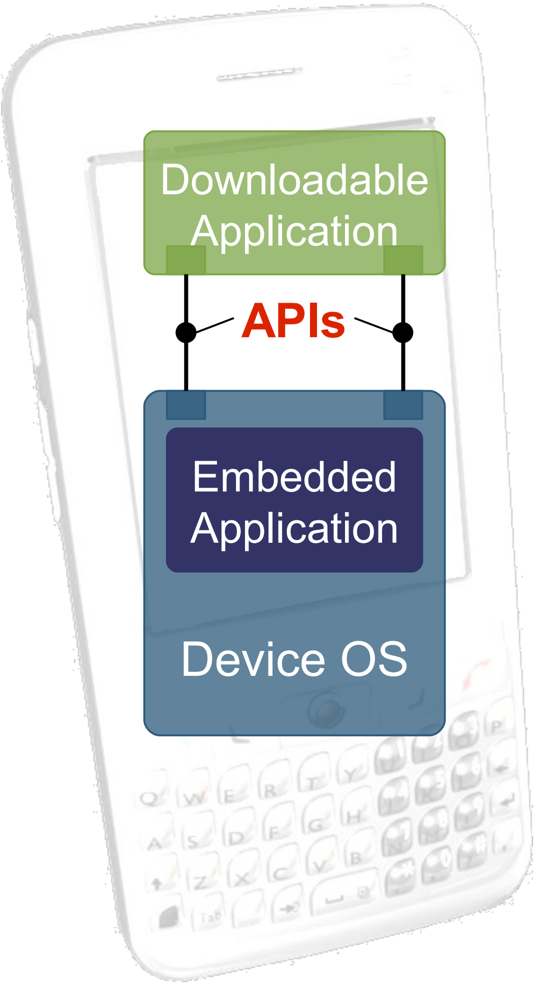
Low-Level vs. High-Level APIs

JSR 281

Conclusions

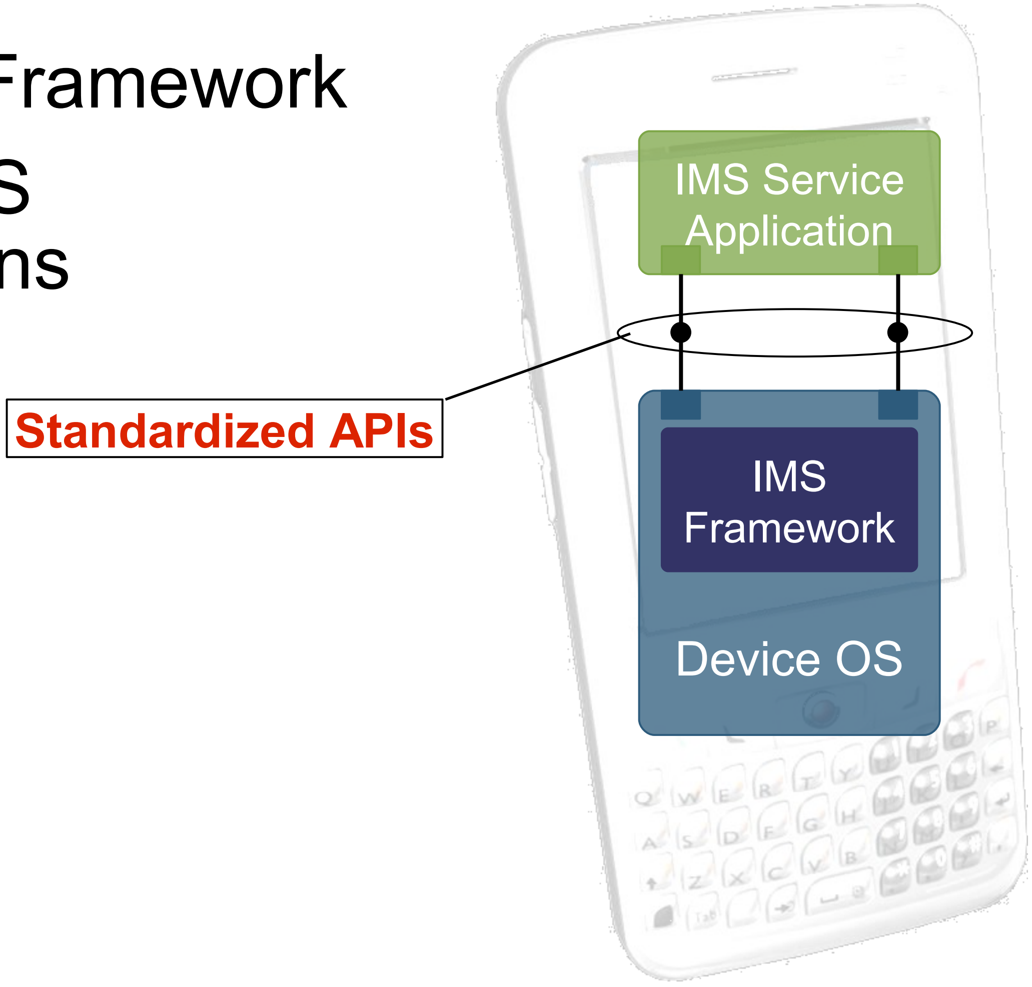
Types of IMS Applications

- ‘Embedded’ applications
  - Applications are pre-installed into the device
  - Tight integration with the device’s Operating System (OS)
- Downloadable applications
  - Applications are installed after device leaves the manufacturing process
  - Required: Open and–preferably–standardized Application Programming Interfaces (APIs)



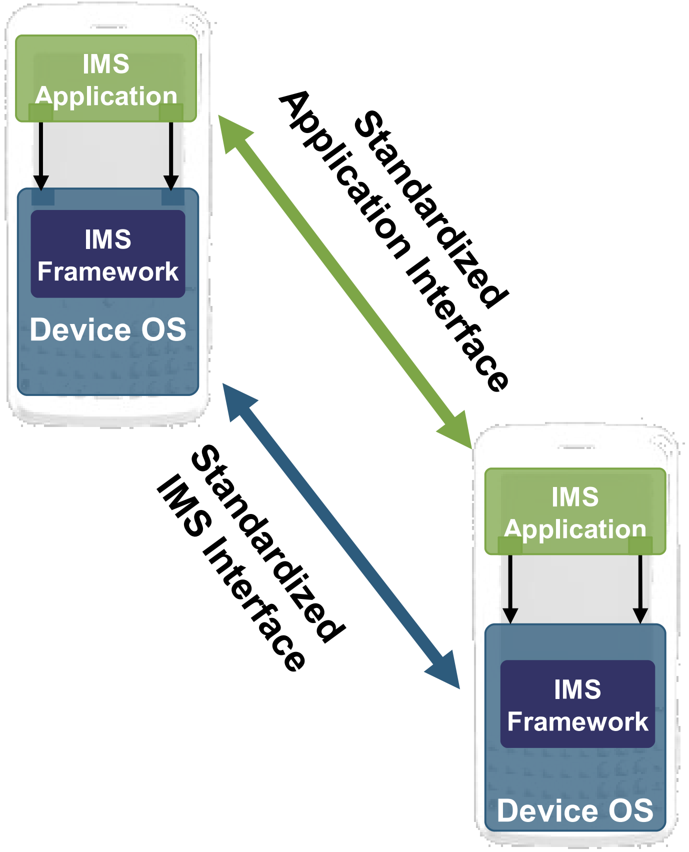
JSR 281: IMS Services API— Merging Both Advantages

- ‘Embedded’ IMS Framework
- Downloadable IMS Service applications



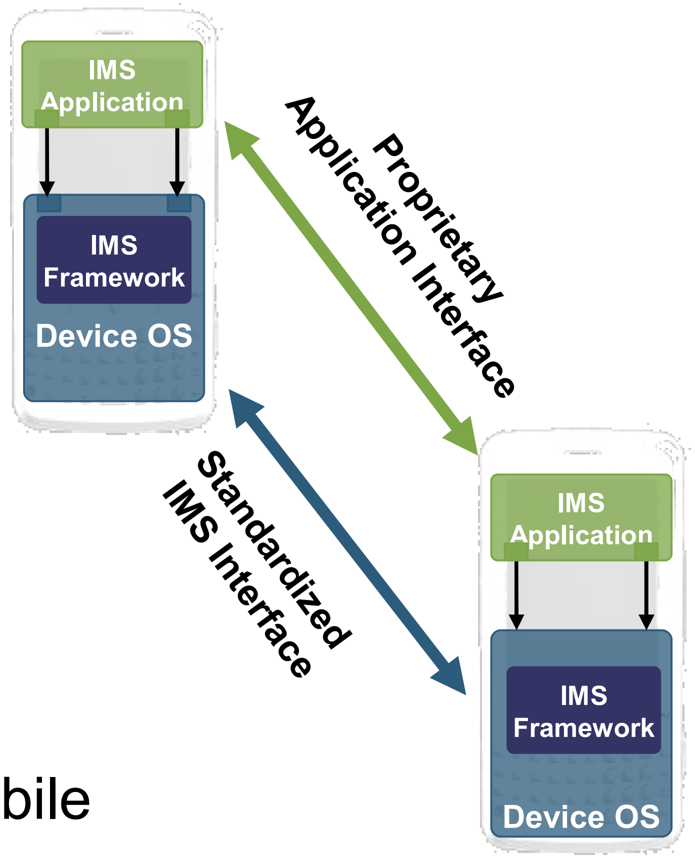
Standardized IMS Services

- All aspects of the service are standardized
  - User-related features
  - Architecture
  - Protocol handling
  - Application level interface
- Example
  - OMA PoC



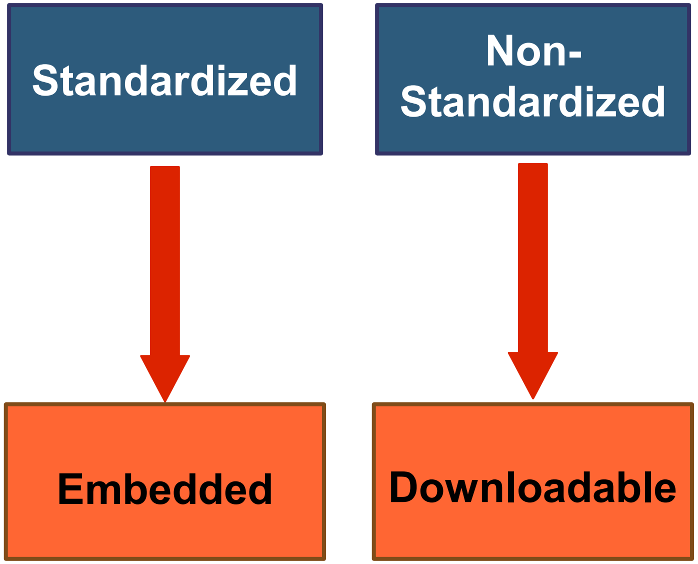
‘Non Standardized’ IMS Services

- Combination of standardized key IMS service enablers
  - PoC
  - Instant messaging
  - Group and list management
- And application defined multimedia sessions
- Application level interfaces ‘application-specific’
  - A.k.a. ‘proprietary’
  - e.g., user plane from mobile to mobile



Mapping IMS Application Types With Service Types

- Standardized services
  - Embedded == higher integration, optimization and testing effort, longer lead times
    - Realized through service enablers
- Non-standardized services
  - Downloadable == integration through open API
    - Open OS without IMS framework == higher testing effort, shorter lead times
    - JSR 281 standardized Java API == lower testing effort, shorter lead times



Agenda

IMS Vision and IMS Framework

Types of IMS Applications and Services

Deployment Options on Mobile Devices

IMS Functionality in Mobile Devices

Low-Level vs. High-Level APIs

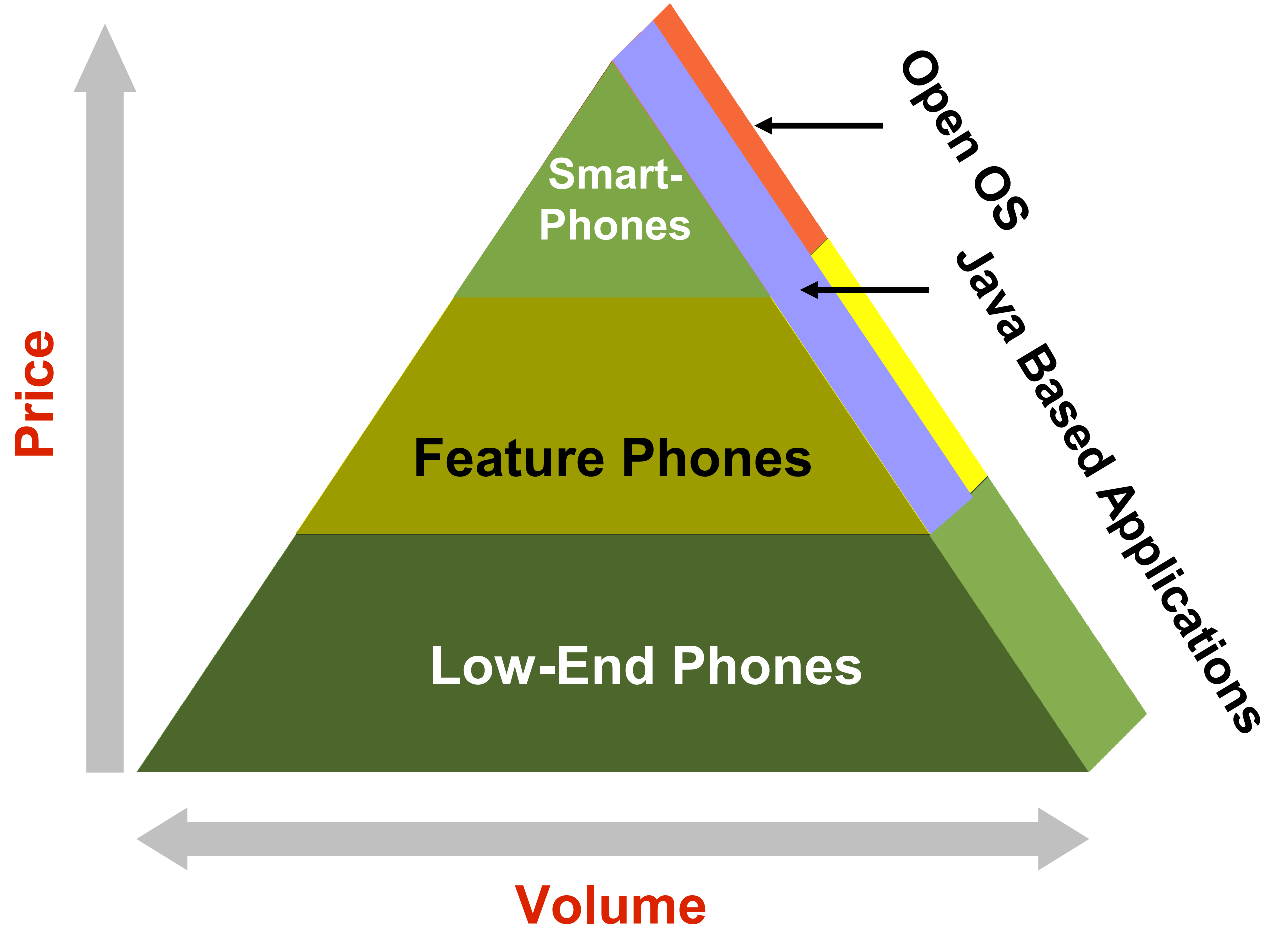
JSR 281

Conclusions

Platforms for Deploying IMS Applications (1)

- Downloadable applications on
  - Open OS devices
  - Devices with open application environment, e.g., Java application environment
- Open OS devices are suitable, but
  - Inherently limited by volume to address the mass market
  - Predicted market volume in 2008 less than 25% of worldwide shipped devices
- Services based on Java can address the mass market sought by operators

Platforms for Deploying IMS Applications (2)



Agenda

IMS Vision and IMS Framework

Types of IMS Applications and Services

Deployment Options on Mobile Devices

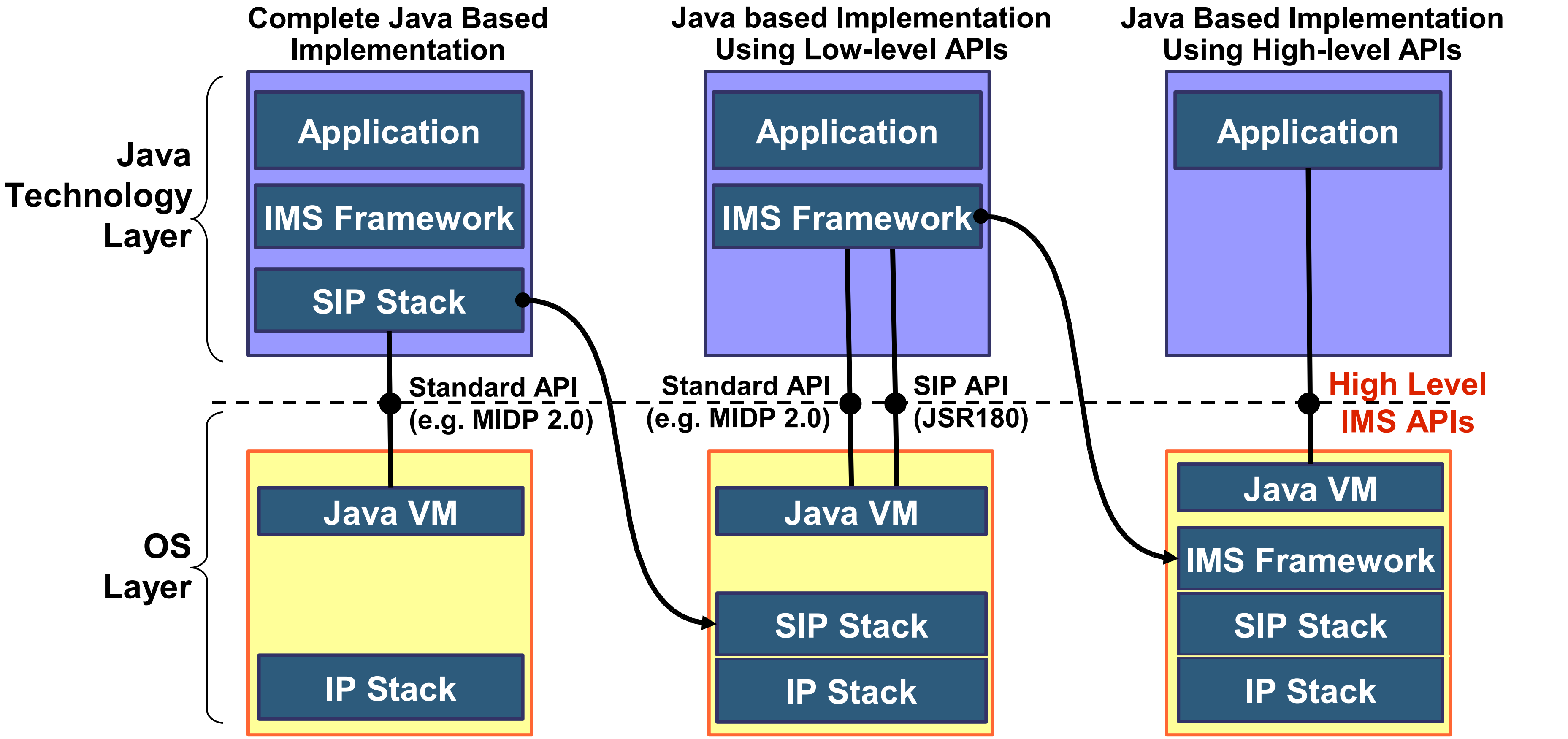
IMS Functionality in Mobile Devices

Low-Level vs. High-Level APIs

JSR 281

Conclusions

Java Technology-Based IMS/SIP Functionality in Feature Phones



- 👍 Usage of Standard APIs
- 👎 Code Size/Performance Issues

- 👍 Standards Available
- 👎 Requires IMS Aware SIP Implementation in the Core

- 👍 Easy, Straight-forward Application Development

Agenda

IMS Vision and IMS Framework

Types of IMS Applications and Services

Deployment Options on Mobile Devices

IMS Functionality in Mobile Devices

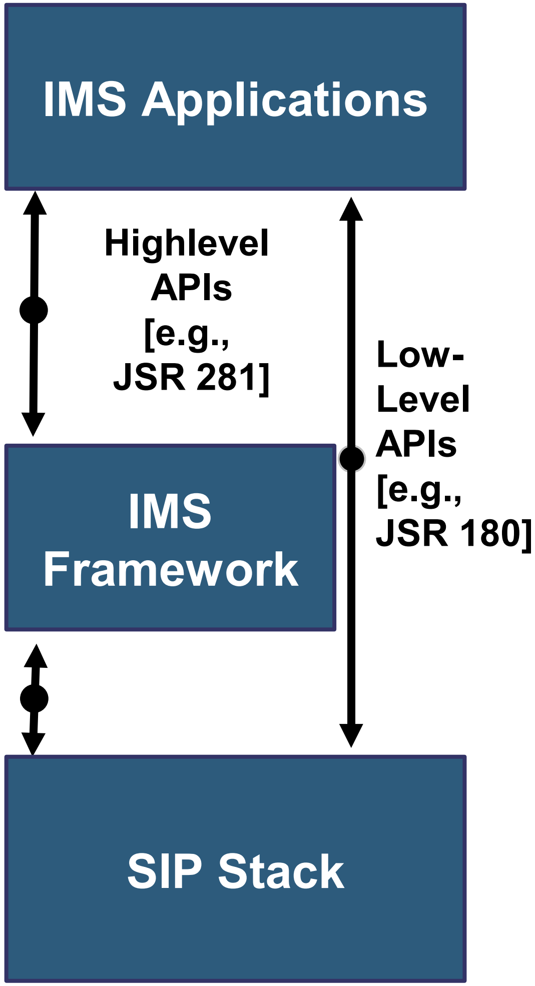
Low-Level vs. High-Level APIs

JSR 281

Conclusions

Lowlevel vs. Highlevel APIs

- Key issue: What type of APIs?
  - Low-level APIs require deep knowledge of SIP
  - Highlevel APIs relate to service enabler functionality and must be implemented IMS aware
- JSR 281 meets this requirement
  - Specification of PoC, Group Management and IMS core functionality
  - Consensus on APIs for Instant Messaging and Presence


IMS Applications
Highlevel APIs [e.g., JSR 281]
Low-Level APIs [e.g., JSR 180]
IMS Framework
SIP Stack

DEMO

A Fancy IMS e2e Service

Agenda

IMS Vision and IMS Framework

Types of IMS Applications and Services

Deployment Options on Mobile Devices

IMS Functionality in Mobile Devices

Low-Level vs. High-Level APIs

JSR 281

Conclusions

Standardized Way Forward: JSR 281

IMS Services API

- API for Client application development for Java ME devices: CDC/CLDC

- Abstracts IMS technology through API
  - Generic IMS API
  - IMS Services API
- Brings standardized IMS Client Service Creation toolbox for Java Development Community
- Lead by
  - Ericsson (Piotr Kessler and Stefan Svenberg)
  - BenQ (Volker Bauche and Mirko Naumann)

JSR 281 EG Members

vodafone

中国移动通信 CHINA MOBILE 移 动 信 息 专 家

Sony Ericsson

BenQ

Sun microsystems

orange

cingular raising the bar

NOKIA Connecting People

Lucent Technologies Bell Labs Innovations

IBM

france telecom

ERICSSON

PHILIPS

SIEMENS

esmertec

T · · Mobile · Get more from life

SAMSUNG

LG Life's Good

TELECOM ITALIA

SiRF

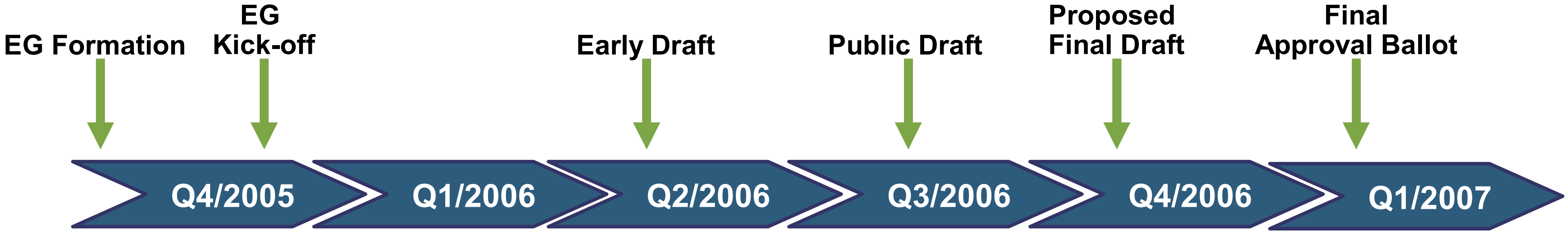
Panasonic ideas for life

openera™

NEC Empowered by Innovation

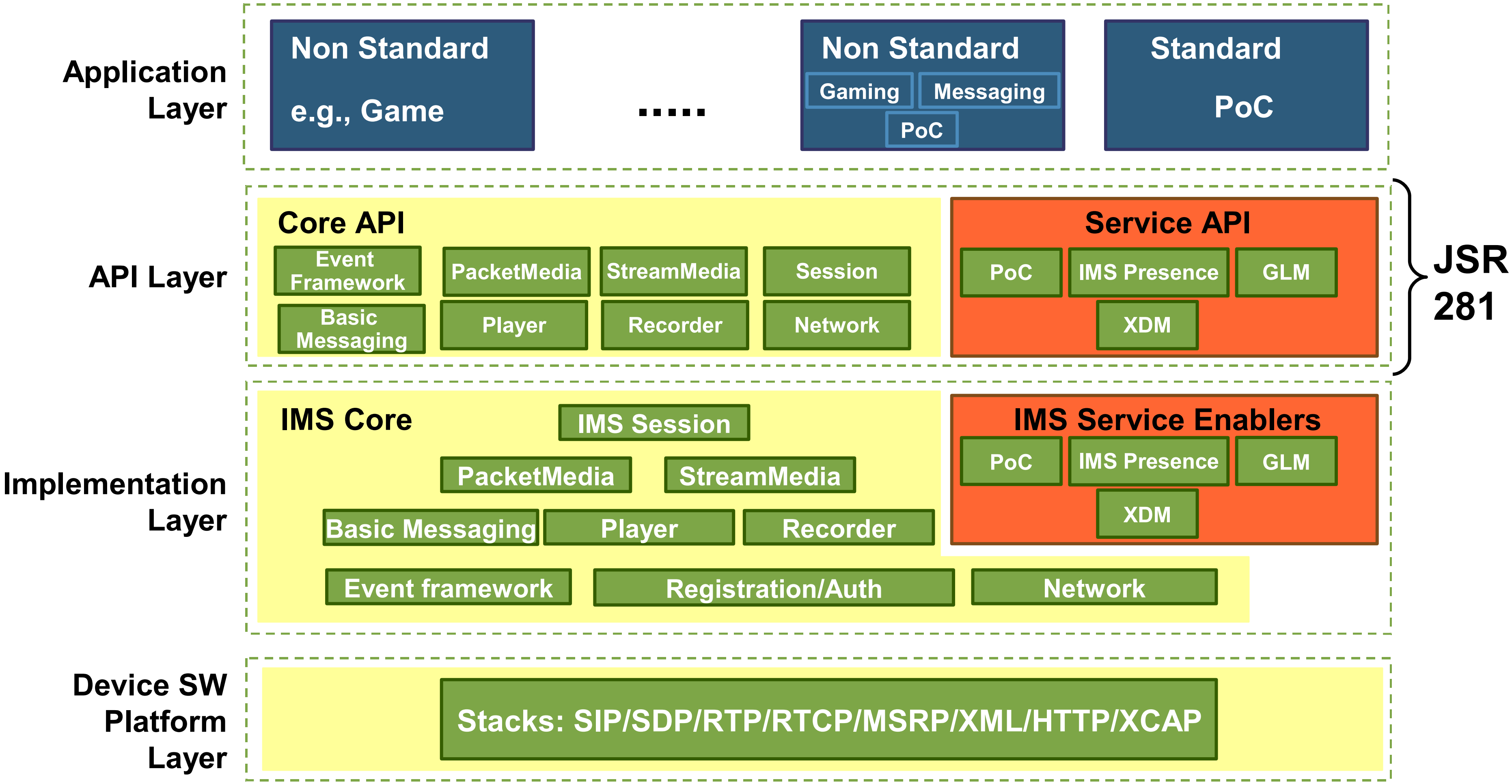
QUALPHONE

JSR 281 Main Milestones



- Early draft: Q2 2006
  - Requirements and spec developed for first public review
  - Reference implementation work started
- Proposed final draft: Q4 2006
  - Specification ready
  - RI and TCK assumed ready, still possibility to finalize
- Final approval ballot: Q1 2007
  - TCK and RI made available for licensing

JSR 281 Architectural Concept



Agenda

IMS Vision and IMS Framework

Types of IMS Applications and Services

Deployment Options on Mobile Devices

IMS Functionality in Mobile Devices

Low-Level vs. High-Level APIs

JSR 281

Conclusions

Summary

- JSR 281 must become an inherent part of any IMS-enabled device
  - Interoperability
- Consistent API required across platforms → JSR 281
  - Interoperability and IMS Portability
- Highlevel IMS APIs must be further developed and promoted to ‘incubate’ the developer community
  - Common set of IMS service enablers for future Interoperability and IMS Portability

For More Information

- **TS-3234** IMS Client Platform and IMS E2E
- OMA Push to Talk Over Cellular V1.0
- OMA XML Document Management V1.0
- OMA Presence Simple V1.0
- http://www.openmobilealliance.org
- http://www.3gpp.org
- http://jcp.org/en/jsr/detail?id=281

Q&A

PoC and Beyond— The IMS Services API (JSR-281)

Volker Bauche
Mirko Naumann

BenQ Mobile
http://www.benqmobile.com

Piotr Kessler
Stefan Svenberg

Ericsson AB
http://www.ericsson.com

Session TS-3319