a tour of technical & business issues

presentation

Mobile App Security

Cristian TOMA

cristian.toma@ie.ase.ro – Business Card

Agenda for M-App & M-Payments Security Presentation

AGENDA in details

Section 1.1 – Mobile Networks Components & Security Features

- GSM Network Components
- 2G Network Components
- 2.5 G Network Components
- 3G Networks
- 4G Networks
- 5G Networks and IoT-NB

Section 1.2 – Technologies 4 Mobile Platforms Development & Security Issues

- Simple WAP development – *Nokia Mobile Internet*
- SIM Technologies – *SIM APDU GSM 11.11 Sample*
- JME Platform Technologies
- Symbian OS Technologies / Linux Mobile OS
- Google ANDROID OS Technologies
- Windows Mobile OS Technologies
- iPhone OS Technologies
- RIM Blackberry OS Technologies / Intel Tizen / Ubuntu Mobile
- Application Types
- Mobile Digital Rights Management
  - Models & Implementations – *Dynamic DRM Forward Lock Sample*
- *NFC – Near Field Communication*

Section 2 – Mobile Payment Systems

- Micro-payments & E-Payment Systems

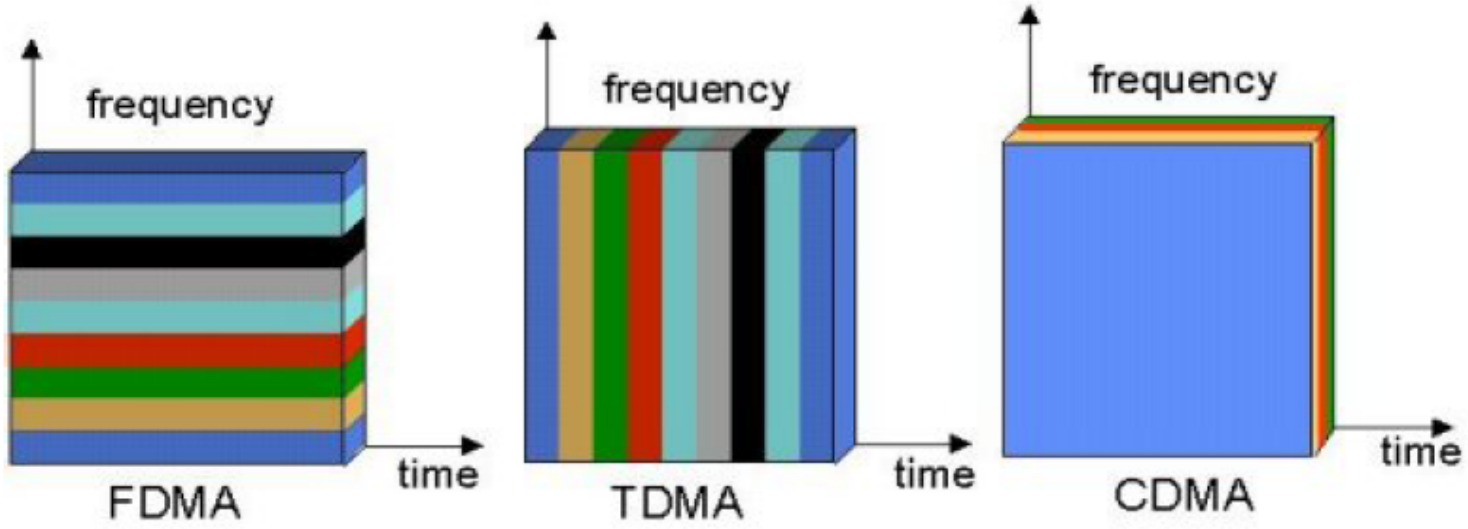
GSM Network, Mobile Device/(U)SIM App, SMS/MMS, m-DRM, 2D Barcode, NFC

Technical Security Issues Approach

GSM Subsystems

- Cellular Systems
  - GSM - Global System for Mobiles
    - DCS 900/1800/1900 - Digital Communication System
    - HSCSD - High Speed Circuit Switched Data
    - GPRS - General Packet Radio Service
    - EDGE - Enhanced Data rates for GSM/global Evolution
  - UMTS - Universal Mobile Telecommunication System
  - CDMA 2000 - Code Division Multiple Access
  - WCDMA
  - HSPDA/HSUPA
  - LTE – 4G
- Non-Cellular Systems
  - Wi-Fi - Wireless Fidelity
  - WiMAX - Worldwide Interoperability for Microwave Access

Multi-Access Radio Techniques



Courtesy of Petri Possi, UMTS World

GSM Subfamily Systems

GSM Subfamily Systems		GSM 900 P	GSM 900 E	GSM 1800	GSM 1900
Broadcast Bands [MHz]	Base Station	935÷960	921÷960	1805÷1880	1930÷1990[II]
	Mobile Station	890÷915	876÷915	1710÷1785	1850÷1910[II]
Duplex Separation [MHz]		45	45	95	80
RF Band Channels [kHz]		200	200	200	200
RF Channel Number		124	174	374	298
Network Resource Access Model		TDMA/ FDMA	TDMA/ FDMA	TDMA/ FDMA	TDMA/ FDMA
Channel on a carrier		8	8	8	8
Voice Channel Number		992	1492	2992	2384
Voice Signaling Coding		RPE / LTP	RPE / LTP	RPE / LTP	RPE / LTP
Modulation Type		GMSK	GMSK	GMSK	GMSK
Frame Duration [ms]		4,615	4,615	4,615	4,615
Maximum Distance of the Cell [km]		35	35	20	20

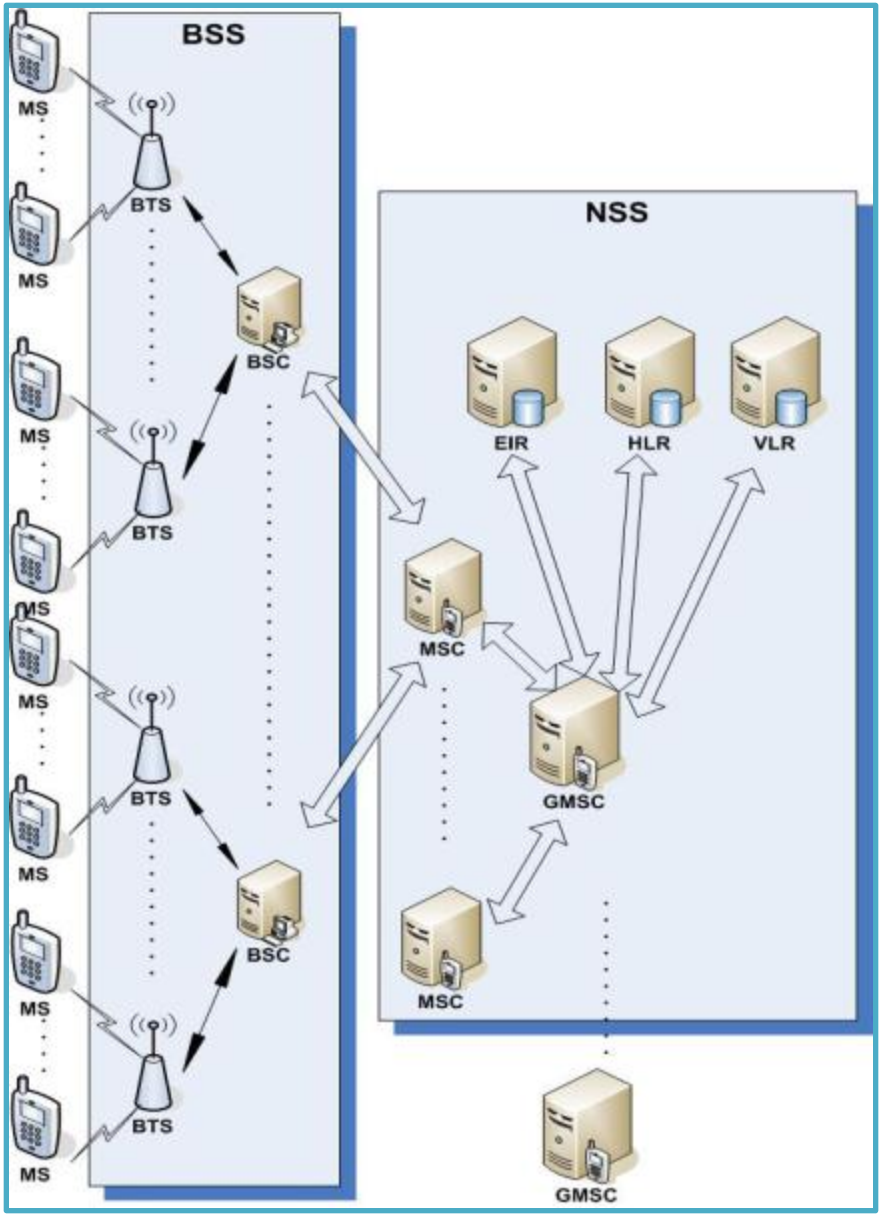
1.1 GSM Networks Overview

Main **components** to kick-off our discussion

BSS = Base Station Subsystem
BTS = Base Transceiver Station
BSC = Base Station Controller

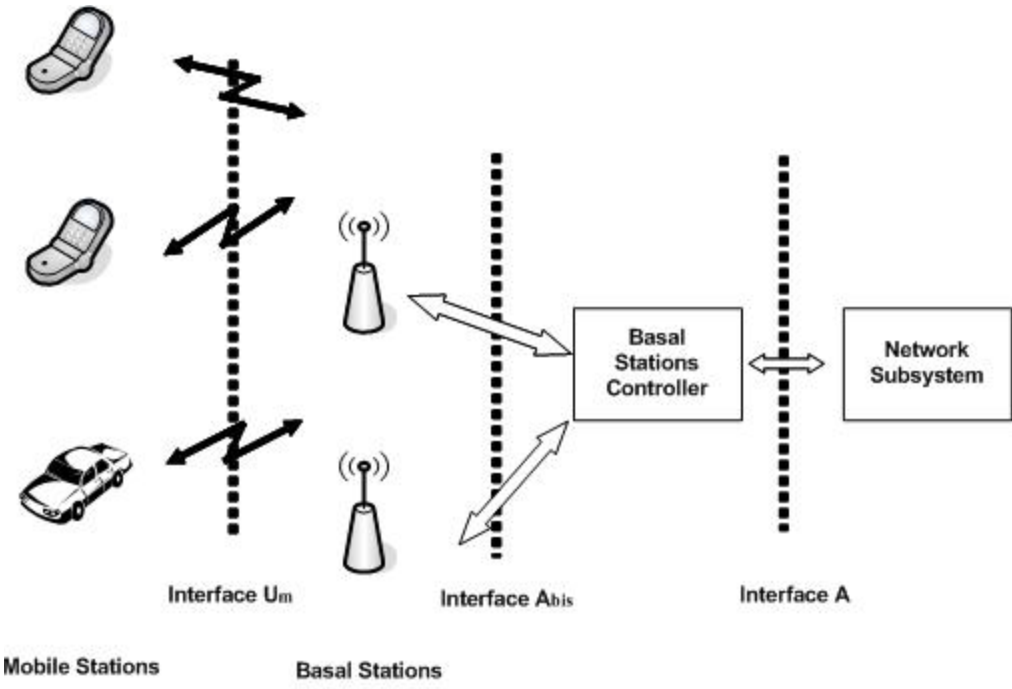
NSS = Network Sub System
MSC = Mobile Switching Center
GMSC = Gateway MSC
EIR = Equipment Identity Register
HLR = Home Location Register
VLR = Visitor Location Register

PLMN = Public Land Mobile Network
GSM = Global System 4 Mobiles
MS = Mobile Subscriber
MT = Mobile Terminal
ME = Mobile Equipment
SIM = Subscriber Identity Module
MT = ME + SIM; MS has a MT
MO – Message Originating vs MT – Message Terminating



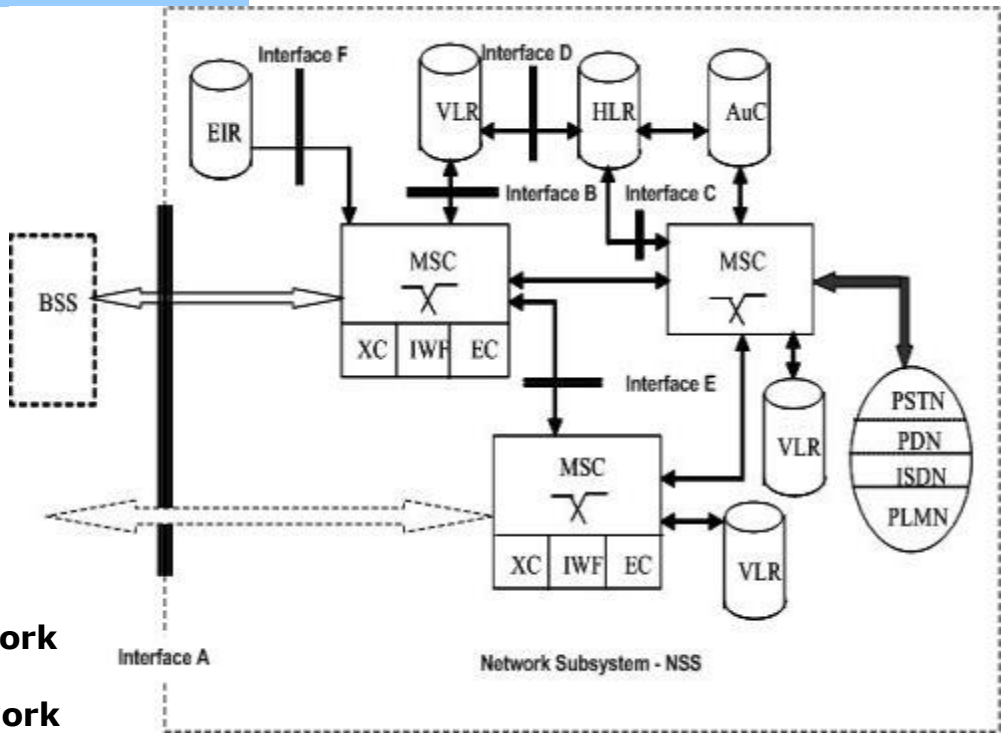
BSS – Base Station Subsystem

- **BSS = BTS + BCS**
- **BSS = Base Station Subsystem**
  - Controls the quality of the links from GSM radio interface
  - Contains BTS and BCS
- **BTS = Base Transceiver Station**
  - Controls the "antennas"
  - Maintain communication through a duplex radio channel
  - Supports configurations for:
    - Electro-magnetic power
    - Radio channel used for broadcasting
    - BSIC – Base Station Identity Code
  - **Functionalities:**
    - Message Encryption
    - Channel Coding
    - Modularization
- **BCS = Base Station Controller**
  - Administrates and controls base stations
  - Administrates radio channels
  - Voices messages coding
  - Management of localization data



NSS – Network Sub System

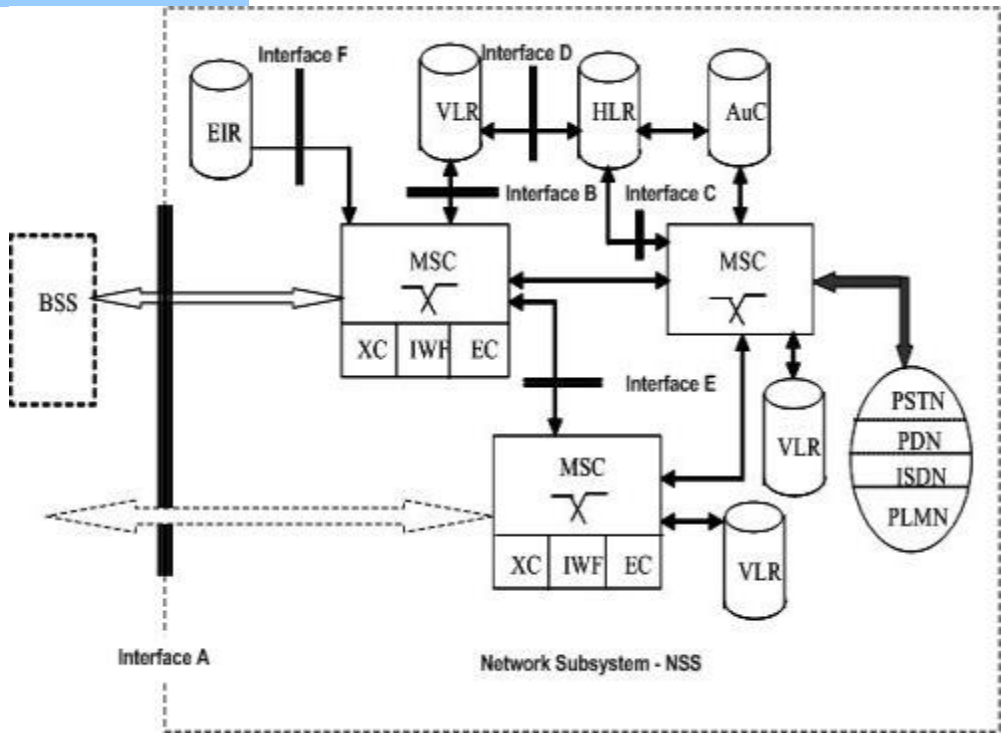
- NSS = MSC + Databases + Adapt. Eq.
- NSS = Network Sub System
  - Is formed from MSCs, Databases such as VLR, HLR, EIR and AuC and adaptation modules such as XC, IWF, EC
  - As functionalities, NSS provides:
    - Management of communication link with other mobile, land and satellite networks
    - Management of mobile subscribers from other BSCs
    - Management of the subscribers using data from AuC, EIR, VLR and HLR databases



- PSTN = Public Switched Telephony Network
- PDN = Packet Data Network
- ISDN = Integrated Services Digital Network
- PLMN = Public Land Mobile Network
- MSC = Mobile Switching Center
- XC = Transcoder
- IWF = Inter Working Function
- EC = Echo Cancellation
- HLR = Home Location Register
- VLR = Visitors Location Register
- EIR = Equipment Identity Register
- AuC = Authentication Center
- BSS = Base Station Subsystem

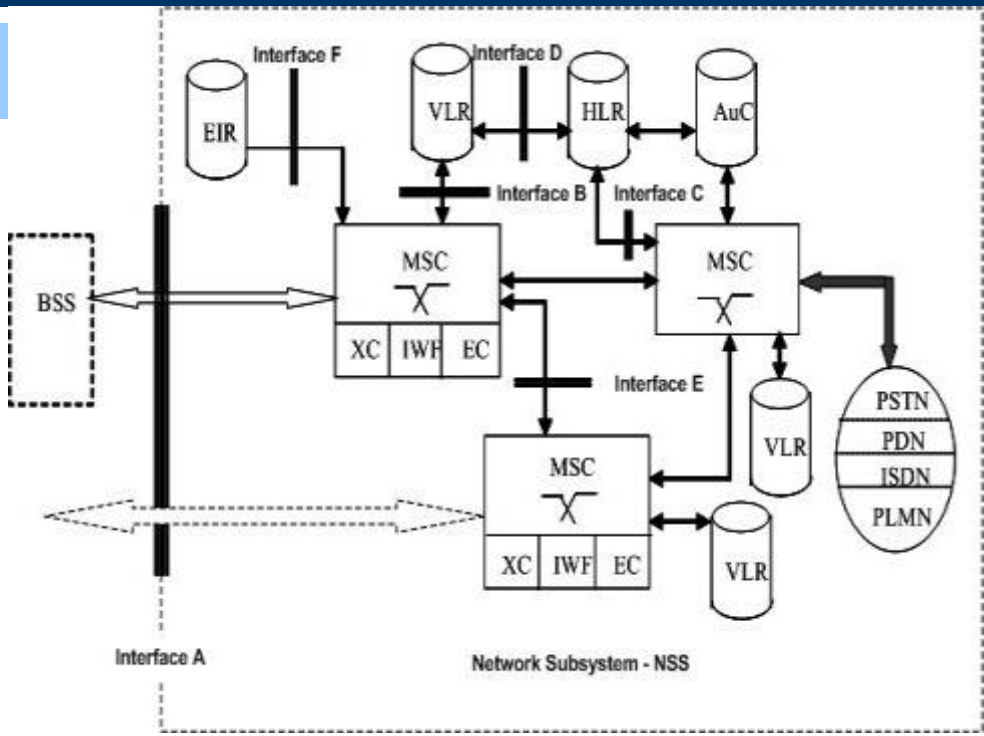
NSS – Network Sub System

- MSC = Mobile Switching Center
  - Contains switching subsystems (e.g. for PBX signaling and for communication signaling over SS7 with other MSCs) and control subsystems
  - Contains adaptation modules
    - IWF – Inter Working Function ensures the interconnectivity feature with dial-up servers, satellite networks or ISDN-PSTN PBX equipment
    - XC – Transcoder converts using padding method from digital bits of information received from BSCs into analogical signal for ISDN. IFW takes this code and send over E1 ISDN protocol.
    - EC – Echo Cancellation eliminates the delay produced by radio networking coding and decoding that it is interpreted by MS as an "echo"
  - As functionalities:
    - Do voice routing and switching procedures
    - Maintains the lists of the active subscribers – that are talking
    - Management of localization and encryption procedures started in BTSs



HLR – Home Location Register

- HLR = Home Location Register
  - Stored the subscribers parameters including the MSISDN and what kind of services has the subscriber (10 SMS, 80 minutes for 1 month)
  - Functional entities:
    - Subscriber database plus the 3 encrypting items formed together with AuC
    - Addressing number analysis
    - Database administration
    - Connectivity entity with HLR, VLR, AuC and MSC)
  - Functionalities:
    - Store the network subscriber information
    - Provides to a partner VLR information for a subscriber that it is in VLR's area
    - Collaboration with AuC for storing authentication and security parameters

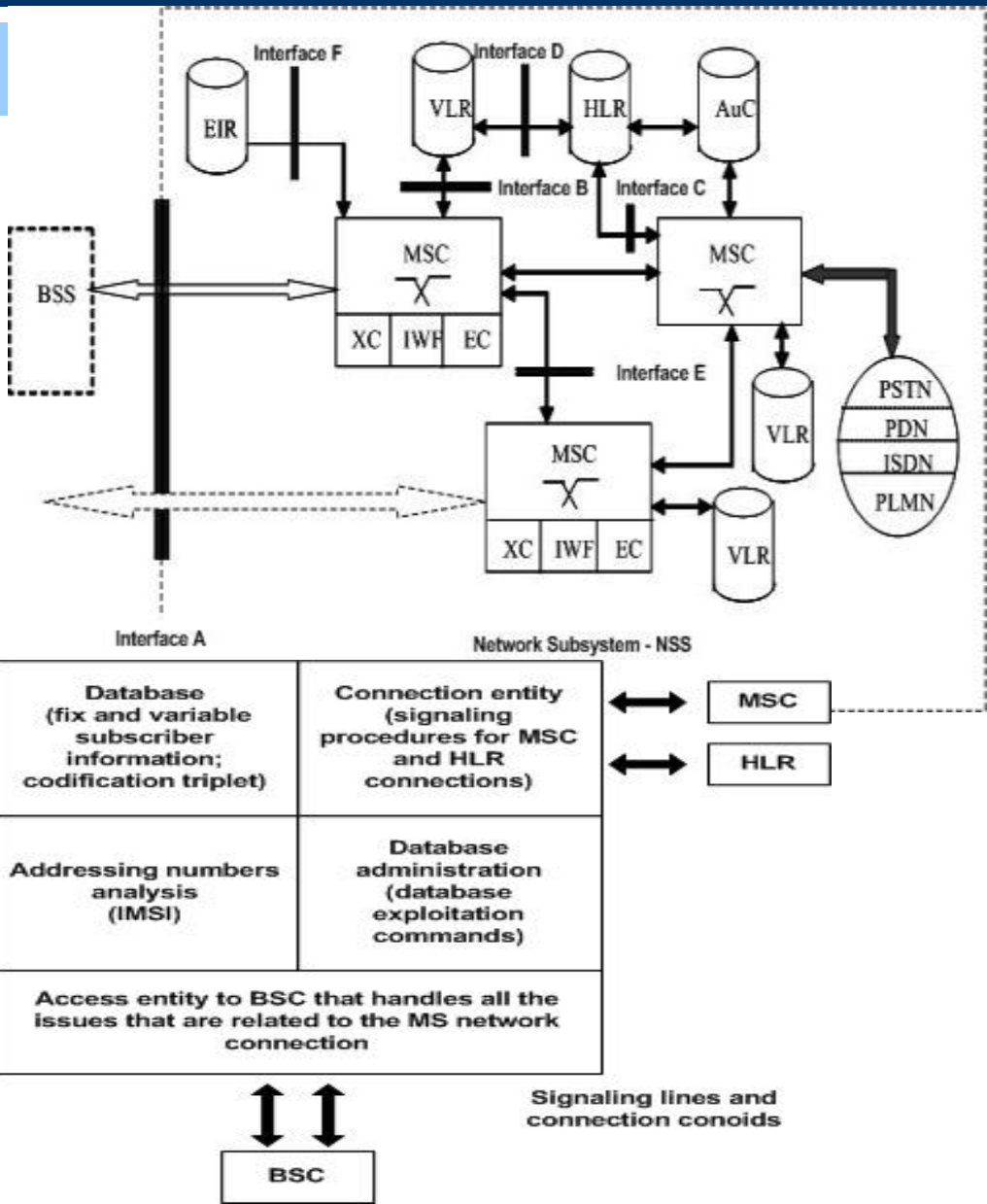


Database (fix and variable subscriber information; codification triplet)	Connection entity (signaling procedures for GSMC, VLR and AuC connections)	↔ GSMC ↔ VLR ↔ AuC
Addressing numbers analysis (IMSI, MSISDN)	Database administration (database exploitation commands)	

Section 1 – Mobile Networks Components & Security Features

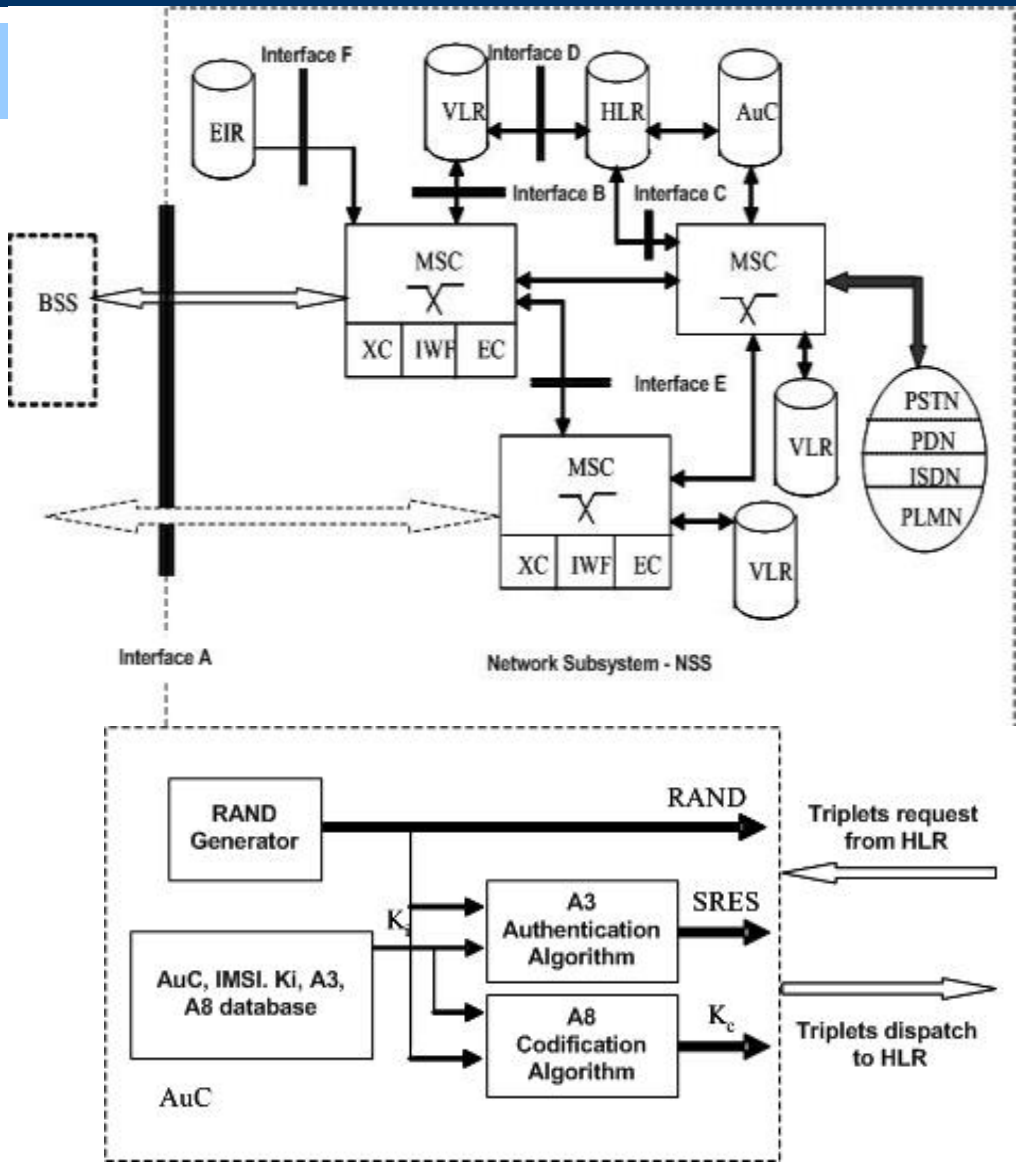
VLR & EIR

- VLR = Visitors Location Register
- It is a mirroring database of HLR for temporary subscribers of another VLR area.
  - Functional entities:
    - Databases with subscriber information and security parameters
    - Addressing number analysis (IMSI)
    - Database administration
    - Connectivity entity with HLR, VLR, AuC and MSC)
    - Connectivity entity with BSCs
  - Functionalities:
    - Update of the mobile station localization and this update value is sent to the HLR (connection with HLR)
    - Provides to roaming number, MSRN (connection with HLR)
    - Send necessary parameters to HLR in order to provide proper codification triplet (connection with HLR)
    - Management connections, mobility and radio resources (through connection with BSC)
- EIR = Equipment Identity Register
  - Centralized database with IMEI (unique number for each provider-device) for each mobile device



AuC – Authentication Center

- AuC = Authentication Center
  - Functionalities and responsibilities:
    - Authorization process for the subscriber access into mobile network
    - For encrypting transmission on radio path
    - For assign of the temporary identity - TMSI (Temporary Mobile Subscriber)
  - Authentication process of the mobile subscriber:
  - The mobile station through "Net Attach" is sending only the SIM's IMSI to the HLR
  - Through "Send Auth. Info" message (contains SIM's IMSI) a triplets request is sent to the AuC from HLR (all HLR, AuC and SIM suppose to have same Ki)
  - The AuC generates a response that contains:
    - RAND (random number – challenge)
    - use stored identity key – Ki from AuC corresponding to the received IMEI for obtaining Kc – encryption key through A8 Codification algorithm
    - SRES – Signed RESponse generated through A3 Auth. Algorithm with RAND and Ki as input
  - The HLR received the triplets and send to the mobile only RAND
  - The mobile device must be enable using Ki from SIM and A3 and A8 algorithms to reconstruct Kc and SRES, then sends to the HLR via MSC or SGSN (in GPRS only) the SRES.
  - If SRES received by HLR from mobile device is the same with SRES received from AuC then Authentication is done and for ENCRYPTION will be used Kc with A5



GSM Security – in a simple glance

GSM security algorithms are used to provide authentication and radio link privacy to users on a GSM network.

GSM uses three different security algorithms called A3, A5, and A8. In practice, A3 and A8 are generally implemented together (known as A3/A8).

An A3/A8 algorithm is implemented in Subscriber Identity Module (SIM) cards and in GSM network Authentication Centres. It is used to authenticate the customer and generate a key for encrypting voice and data traffic, as defined in 3GPP TS 43.020 (03.20 before Rel-4). Development of A3 and A8 algorithms is considered a matter for individual GSM network operators, although example implementations are available.

An A5 encryption algorithm scrambles the user's voice and data traffic between the handset and the base station to provide privacy. An A5 algorithm is implemented in both the handset and the base station subsystem (BSS).

1.1 GSM Networks Overview

Mobile Station Components

- Microprocessor
- EEPROM - Memory
- Operating System
- Java or .NET Virtual Machine
- X509 Certificates
- Mobile Apps & games
- IMEI = International Mobile Station Equipment Identity

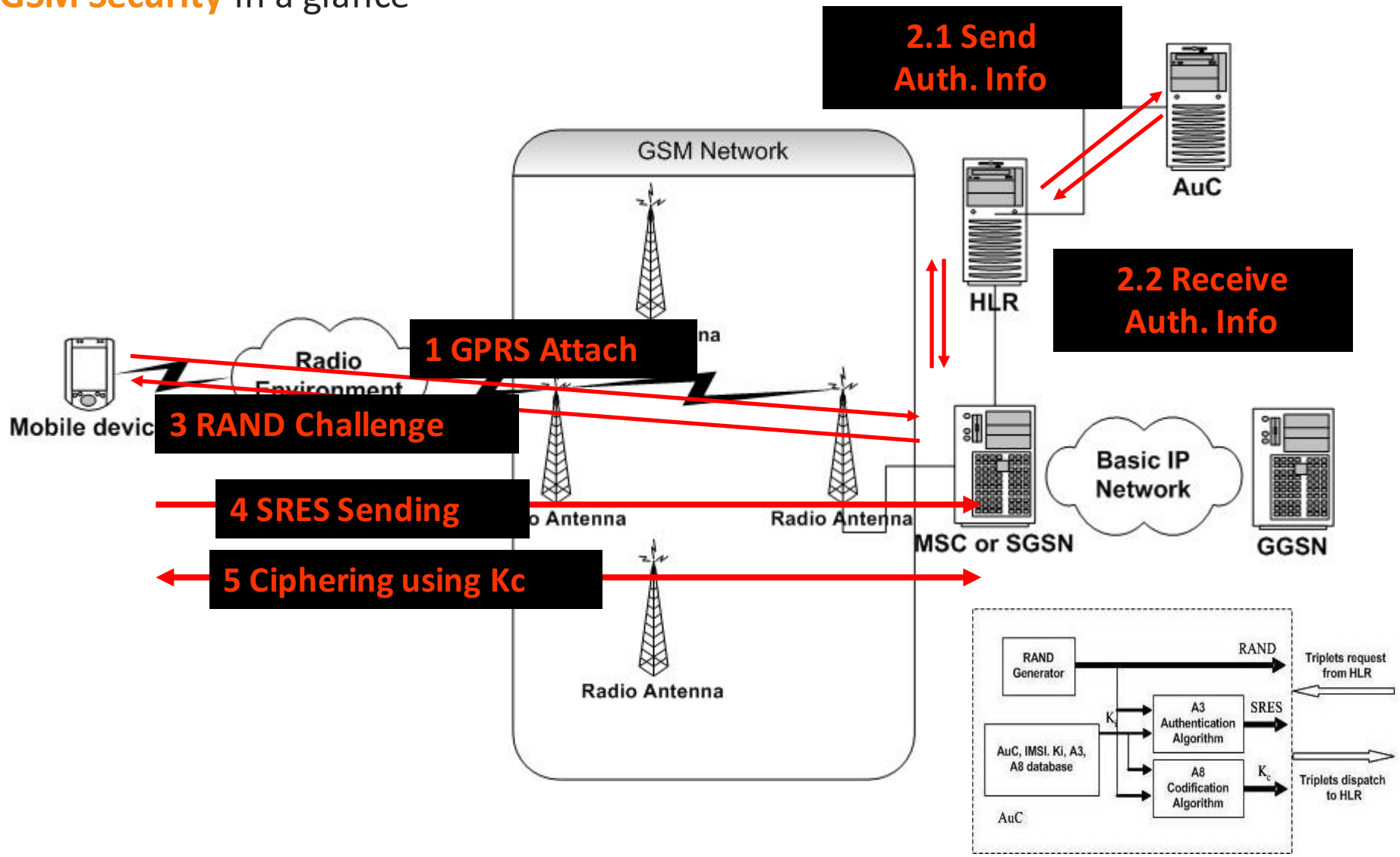
SIM = Subscriber Identity Module

- Microprocessor
- EEPROM – Memory
- Operating System
- SIM Toolkit Application
- Java Card VM
- Encrypting keys 4 communication
- IMSI = International Mobile Subscriber Identity

MT = Mobile Terminal (MSISDN stored in HLR)

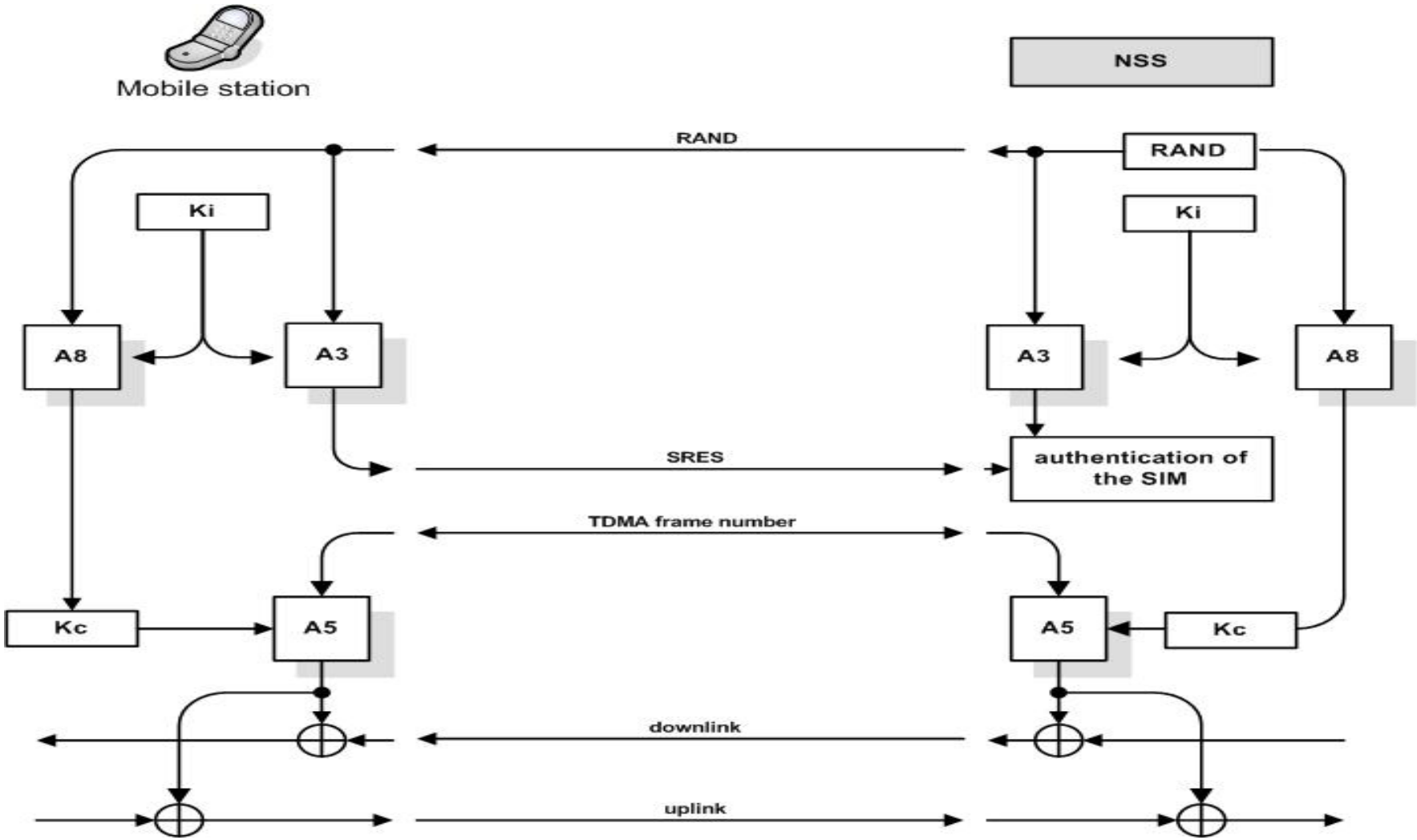
1.1 GSM Networks Overview

GSM Security in a glance

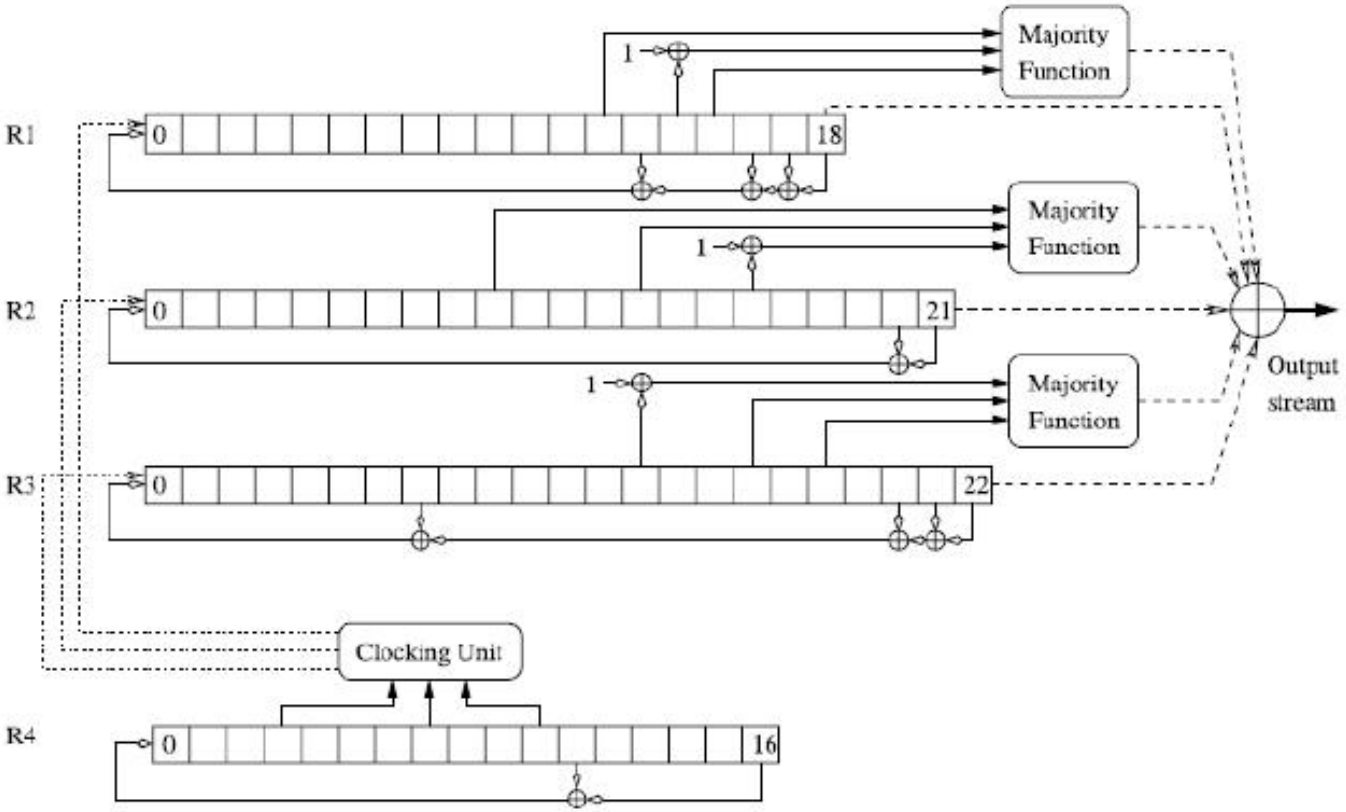


1.1 GSM Networks Overview

GSM Security Details



GSM Security – A5 Algorithm



During each cycle two or three of registers R1, R2, and R3 are clocked by a clocking unit, based on the value of three bits of R4: R4[3], R4[7], and R4[10]. The clocking unit performs a majority function on the bits: **maj(a; b; c) = ab ⊕ ac ⊕ bc** where a; b; c are three bits from R4. Then, the registers are clocked as follows: R1 is clocked if and only if R4[10] agrees with the majority. R2 is clocked if and only if R4[3] agrees with the majority. R3 is clocked if and only if R4[7] agrees with the majority. After these clockings, R4 is clocked, and an output bit is generated from the values of R1, R2, and R3, by XOR-ing their rightmost bits to three majority values, one of each register.

GSM Security – A5 Algorithm

1. Set R1 = R2 = R3 = R4 = 0.
2. For i = 0 to 63 - Clock all four registers

$$R1[0] \leftarrow R1[0] \oplus K_c[i];\ R2[0] \leftarrow R2[0] \oplus K_c[i];\ R3[0] \leftarrow R3[0] \oplus K_c[i];\ R4[0] \leftarrow R4[0] \oplus K_c[i].$$

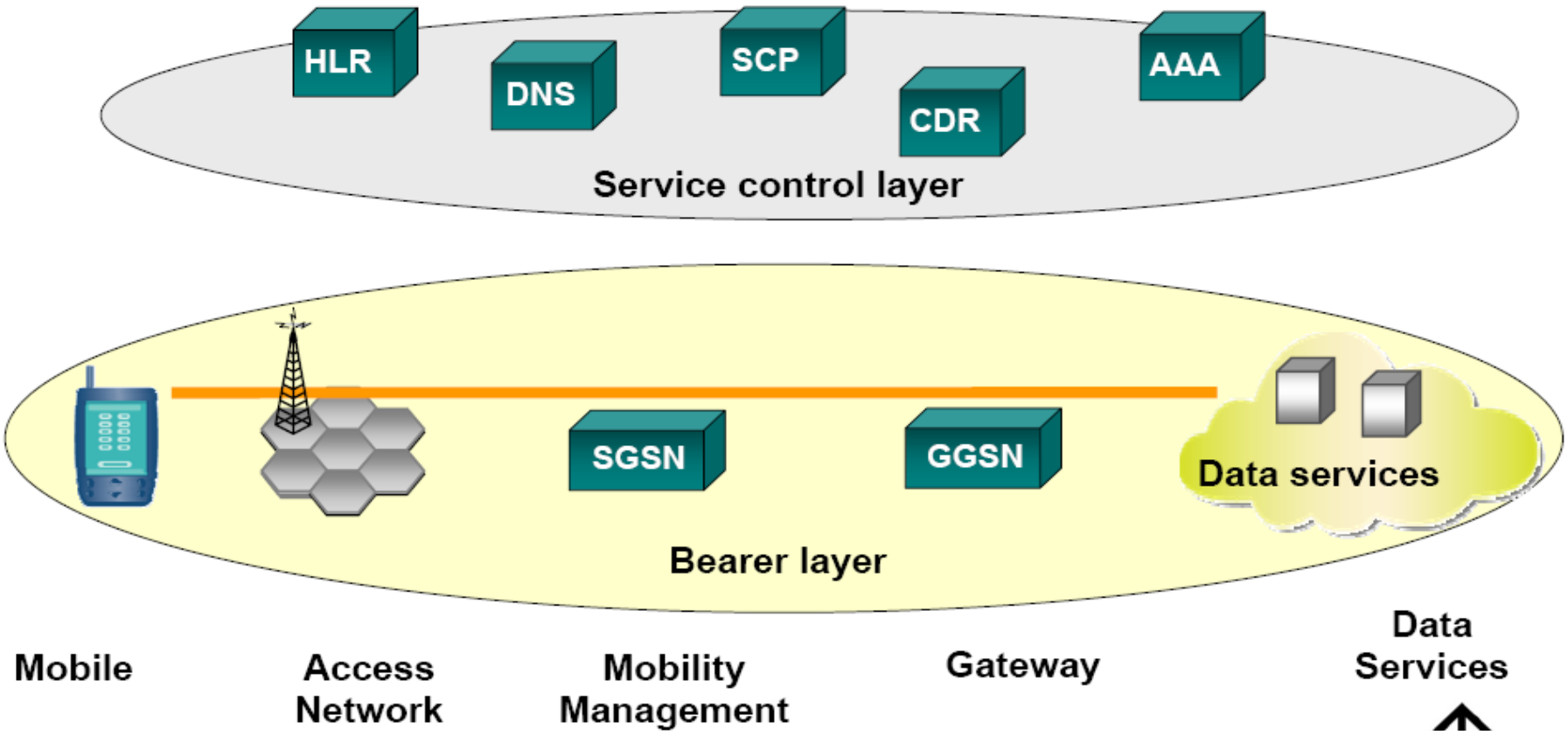
3. For i = 0 to 21 - Clock all four registers

$$R1[0] \leftarrow R1[0] \oplus f[i];\ R2[0] \leftarrow R2[0] \oplus f[i];\ R3[0] \leftarrow R3[0] \oplus f[i];\ R4[0] \leftarrow R4[0] \oplus f[i].$$

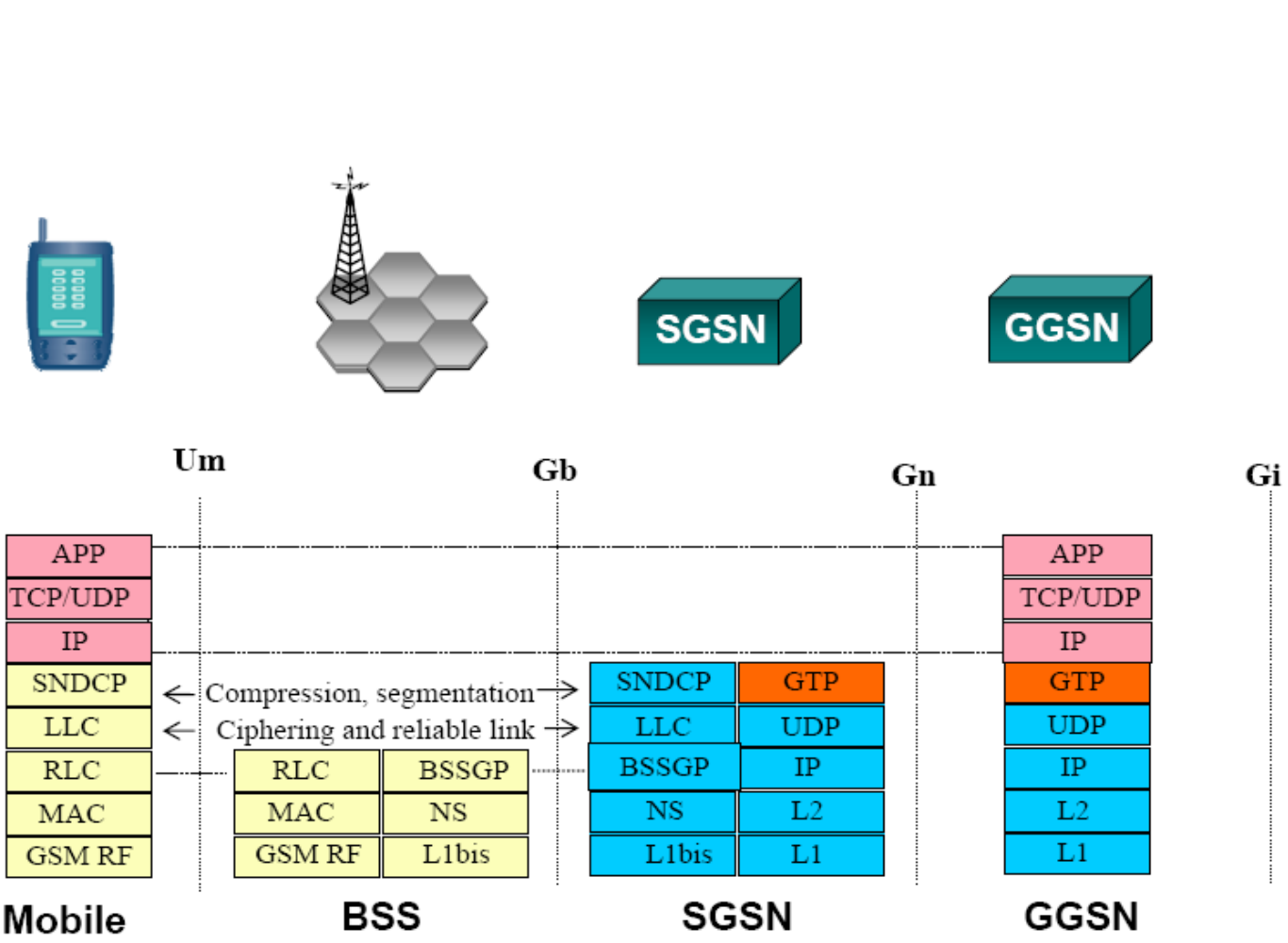
4. Set the bits

$$R1[15] \leftarrow 1, R2[16] \leftarrow 1, R3[18] \leftarrow 1, R4[10] \leftarrow 1$$

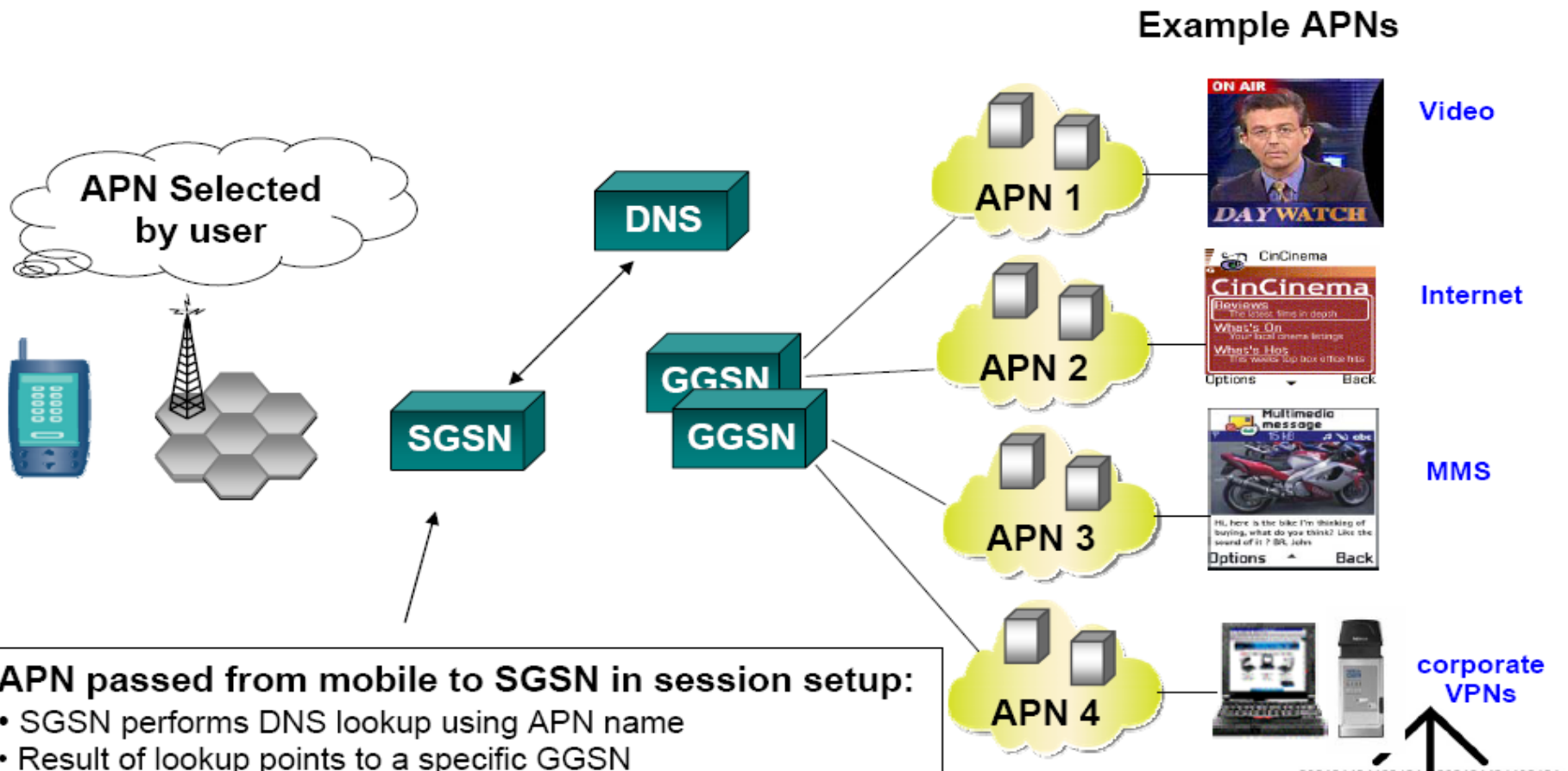
GPRS Bearer Service Elements



GPRS Bearer Service Protocol Stacks Mapping



GPRS Bearer Services – APNs



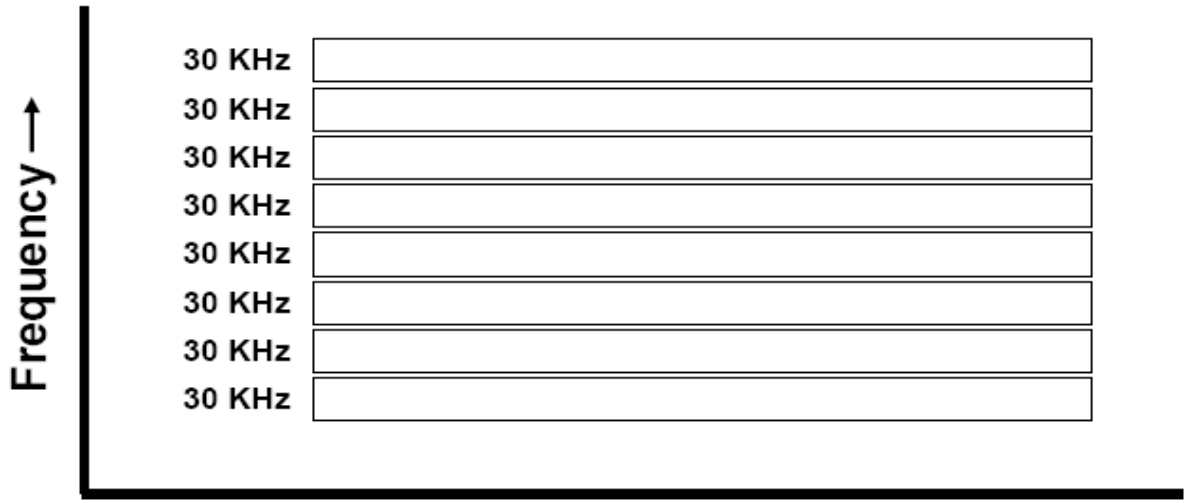
APN passed from mobile to SGSN in session setup:

- SGSN performs DNS lookup using APN name
- Result of lookup points to a specific GGSN

1G Access for Network Resource

1G — Separate Frequencies

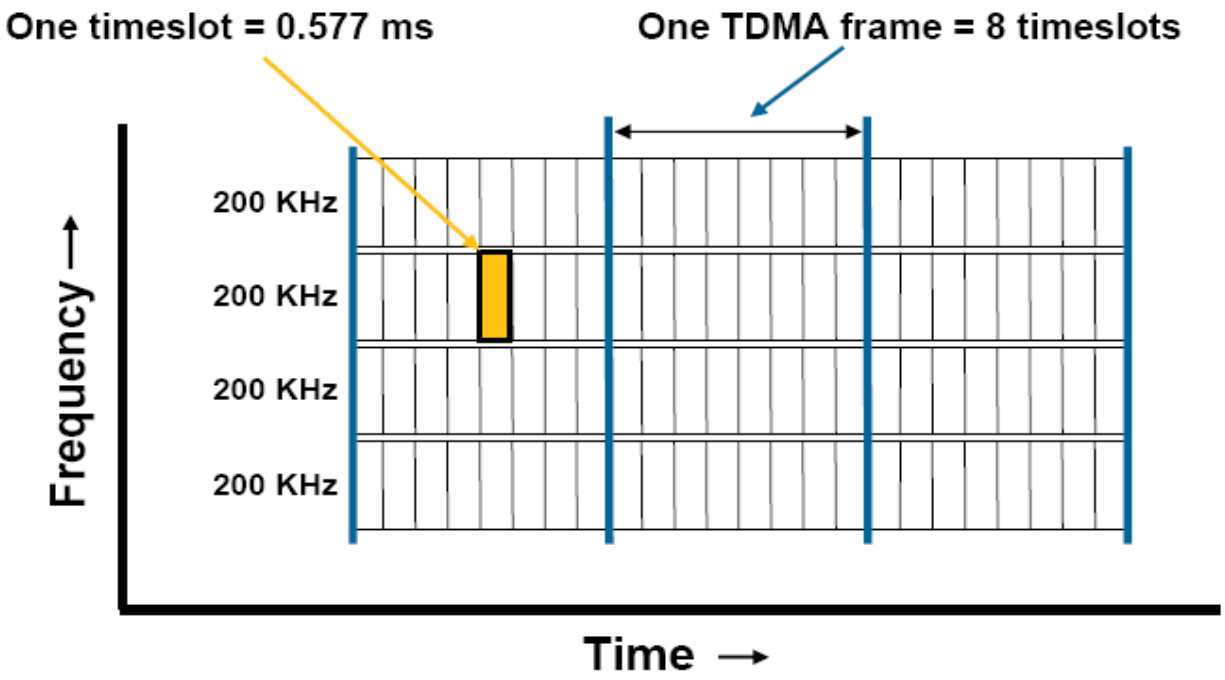
FDMA — Frequency Division Multiple Access



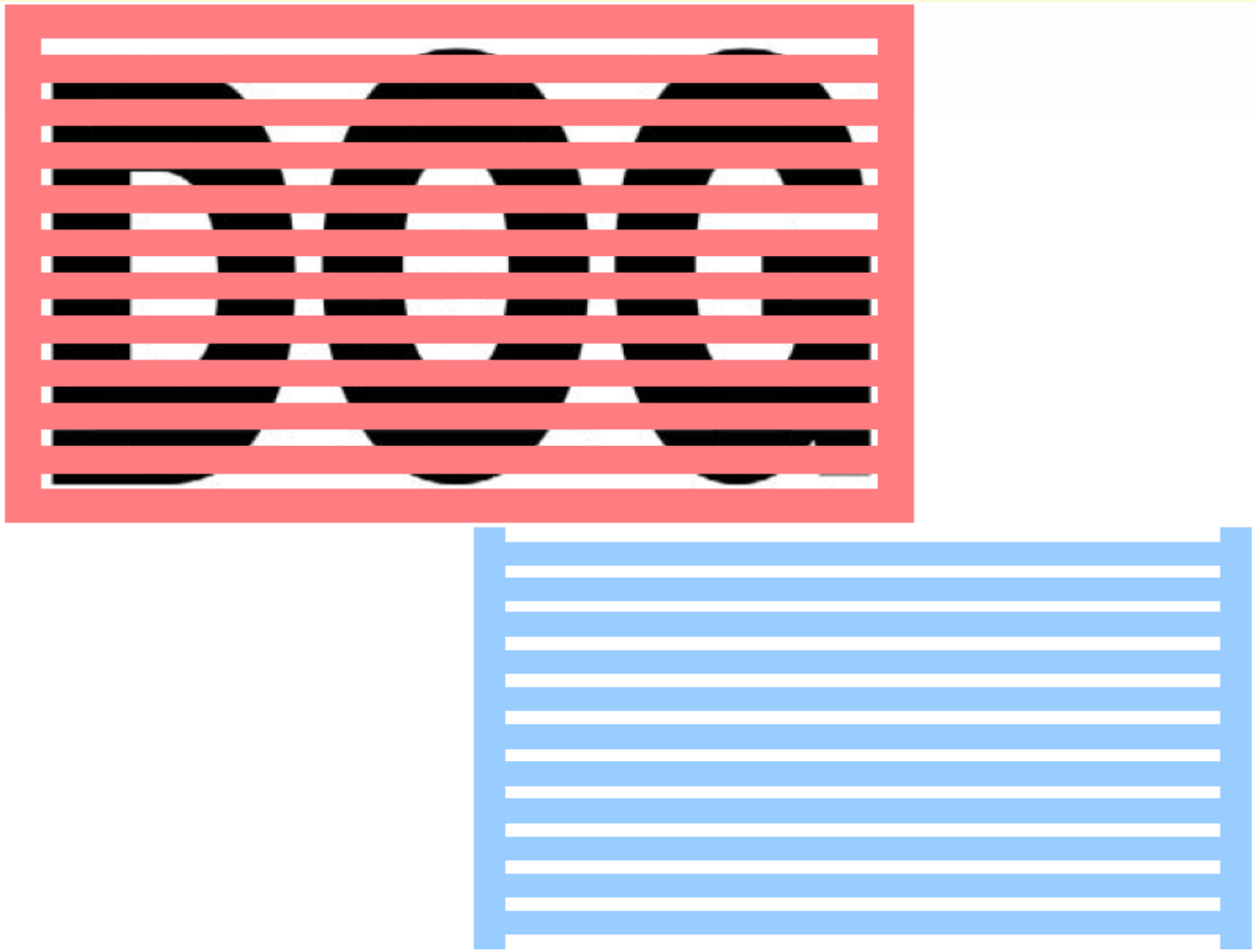
2G Access for Network Resource

2G — TDMA

Time Division Multiple Access

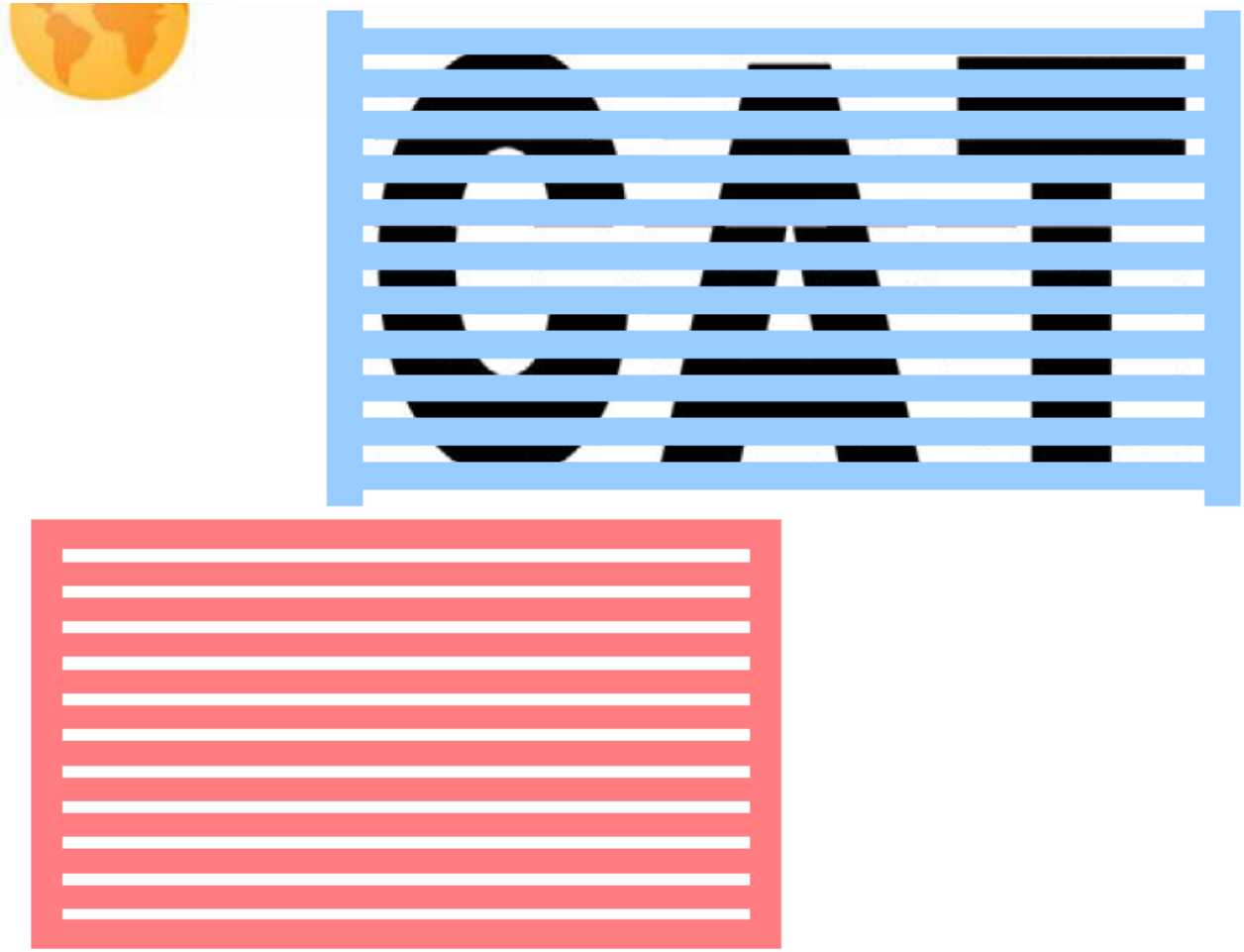


3G Access for Network Resource - WCDMA



Courtesy of Suresh Goyal & Rich Howard

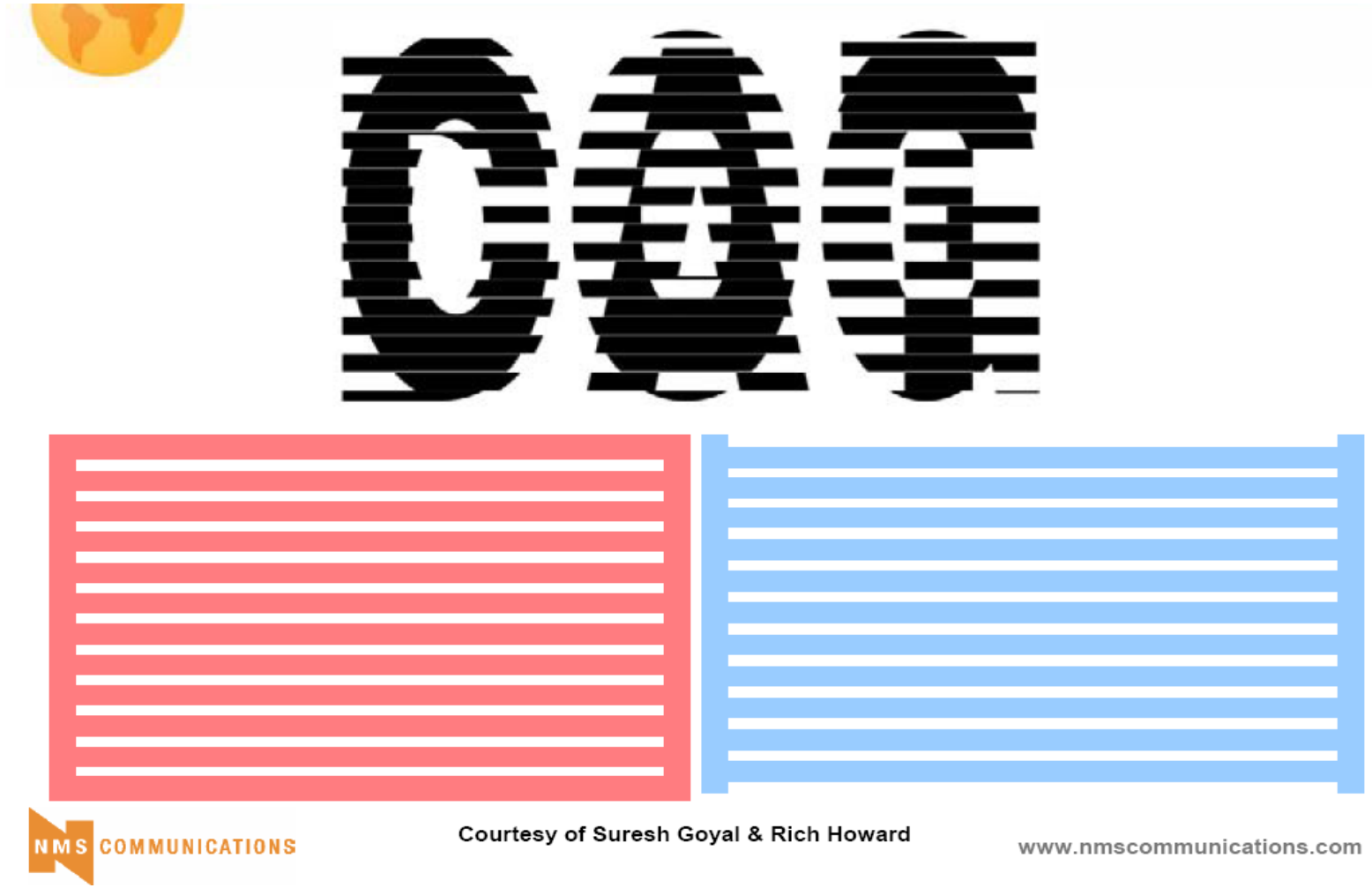
3G Access for Network Resource - WCDMA



NMS COMMUNICATIONS

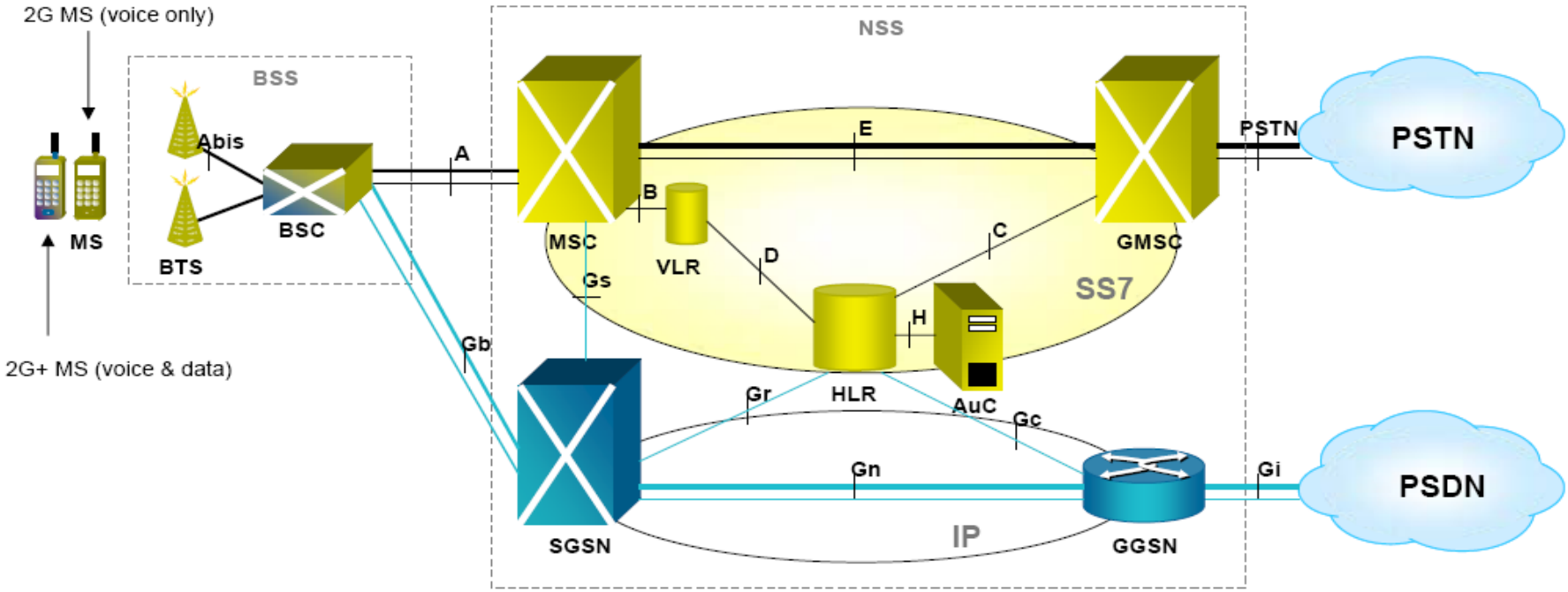
Courtesy of Suresh Goyal & Rich Howard

3G Access for Network Resource - WCDMA



1.1 GSM Networks Overview

GSM 2.5G: GPRS



BSS — Base Station System

BTS — Base Transceiver Station

BSC — Base Station Controller

NSS — Network Sub-System

MSC — Mobile-service Switching Controller

VLR — Visitor Location Register

HLR — Home Location Register

AuC — Authentication Server

GMSC — Gateway MSC

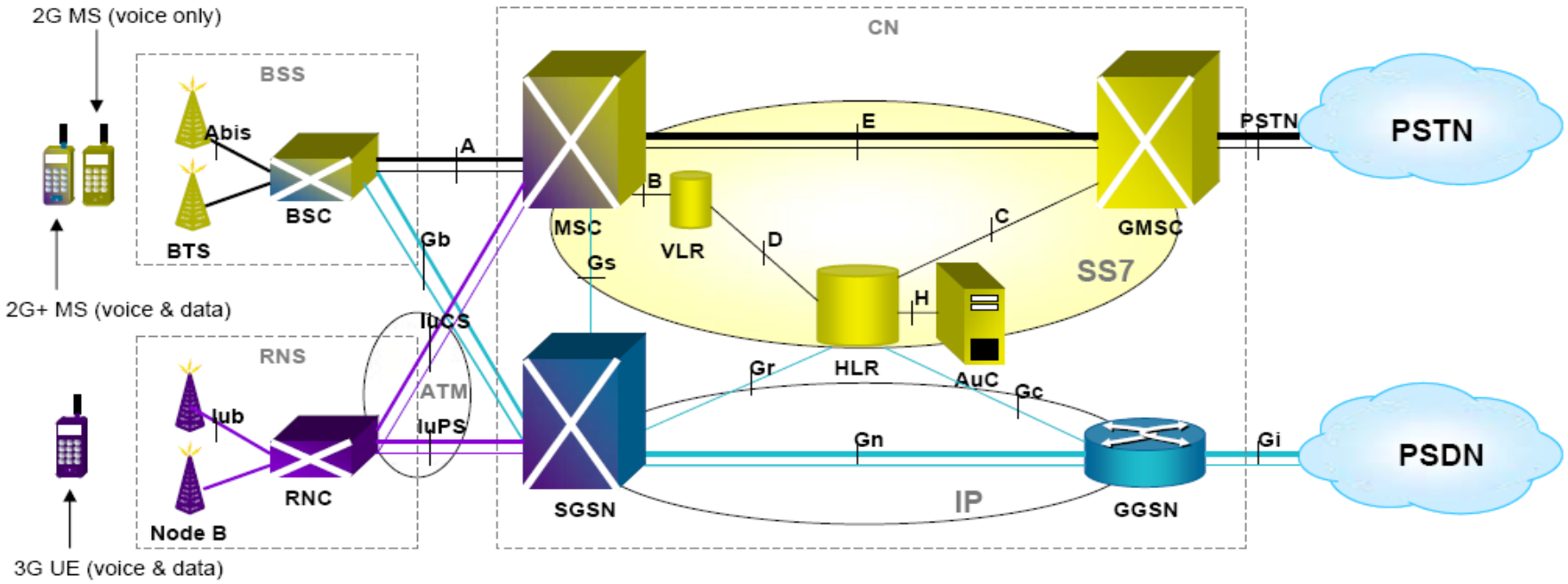
SGSN — Serving GPRS Support Node

GGSN — Gateway GPRS Support Node

GPRS — General Packet Radio Service

1.1 GSM Networks Overview

GSM 3G: first UMTS 99



BSS — Base Station System

BTS — Base Transceiver Station

BSC — Base Station Controller

RNS — Radio Network System

RNC — Radio Network Controller

CN — Core Network

MSC — Mobile-service Switching Controller

VLR — Visitor Location Register

HLR — Home Location Register

AuC — Authentication Server

GMSC — Gateway MSC

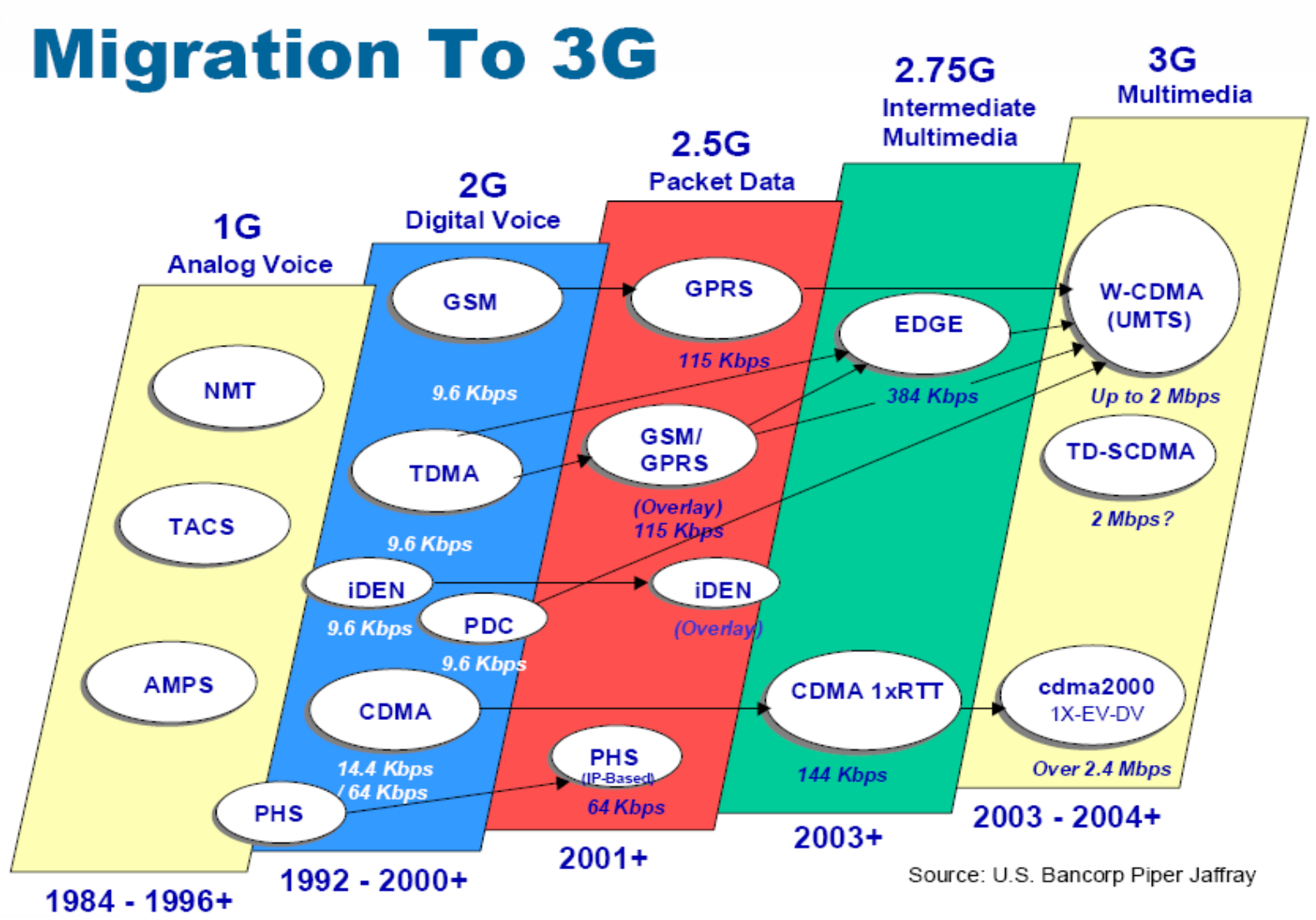
SGSN — Serving GPRS Support Node

GGSN — Gateway GPRS Support Node

UMTS — Universal Mobile Telecommunication System

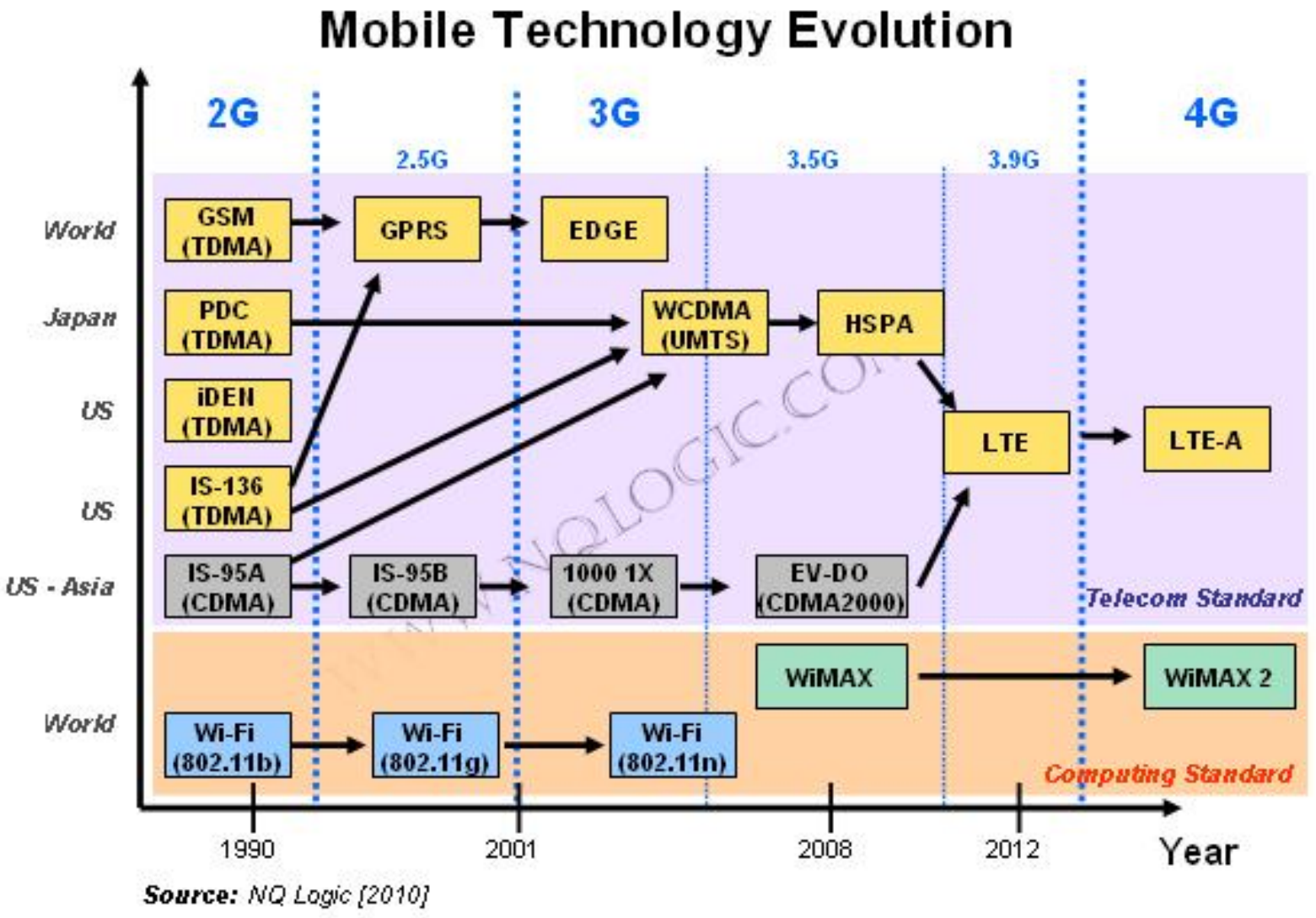
1.1 GSM Networks Overview

GSM Migration to 3G



1.1 GSM Networks Overview

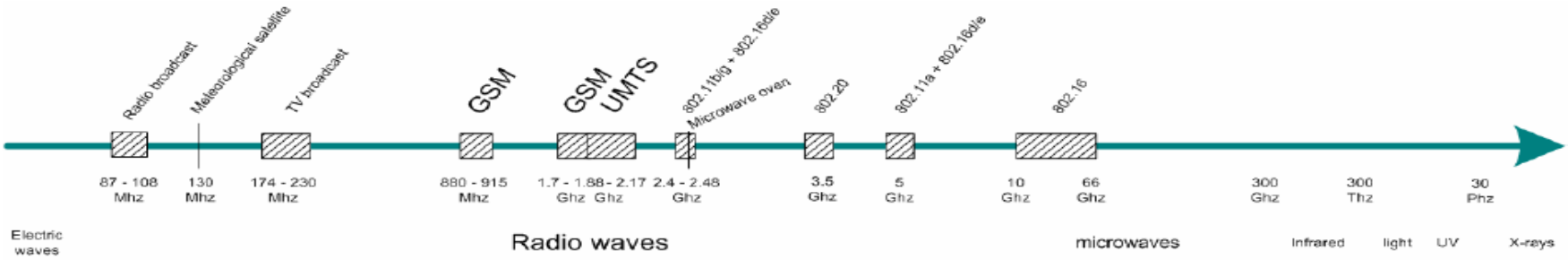
Wireless 2G-4G Evolution

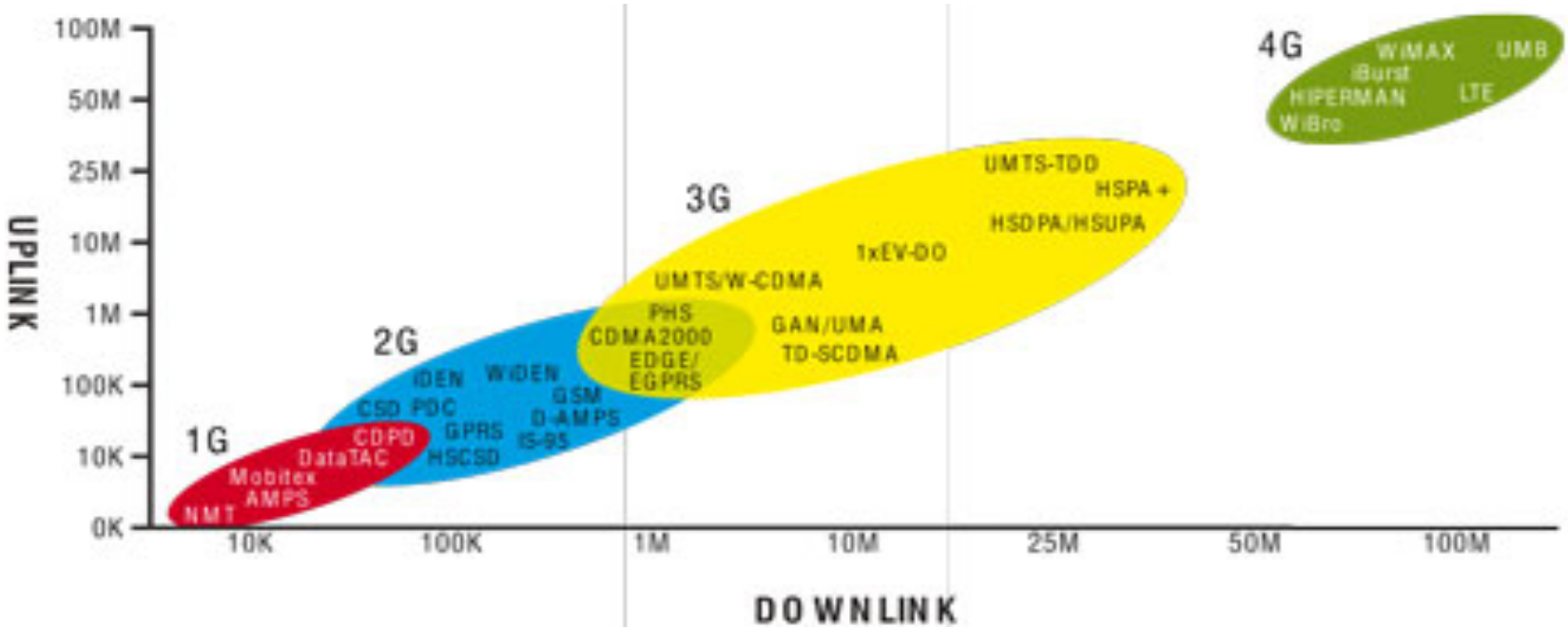


Source: *NQ Logic [2010]*

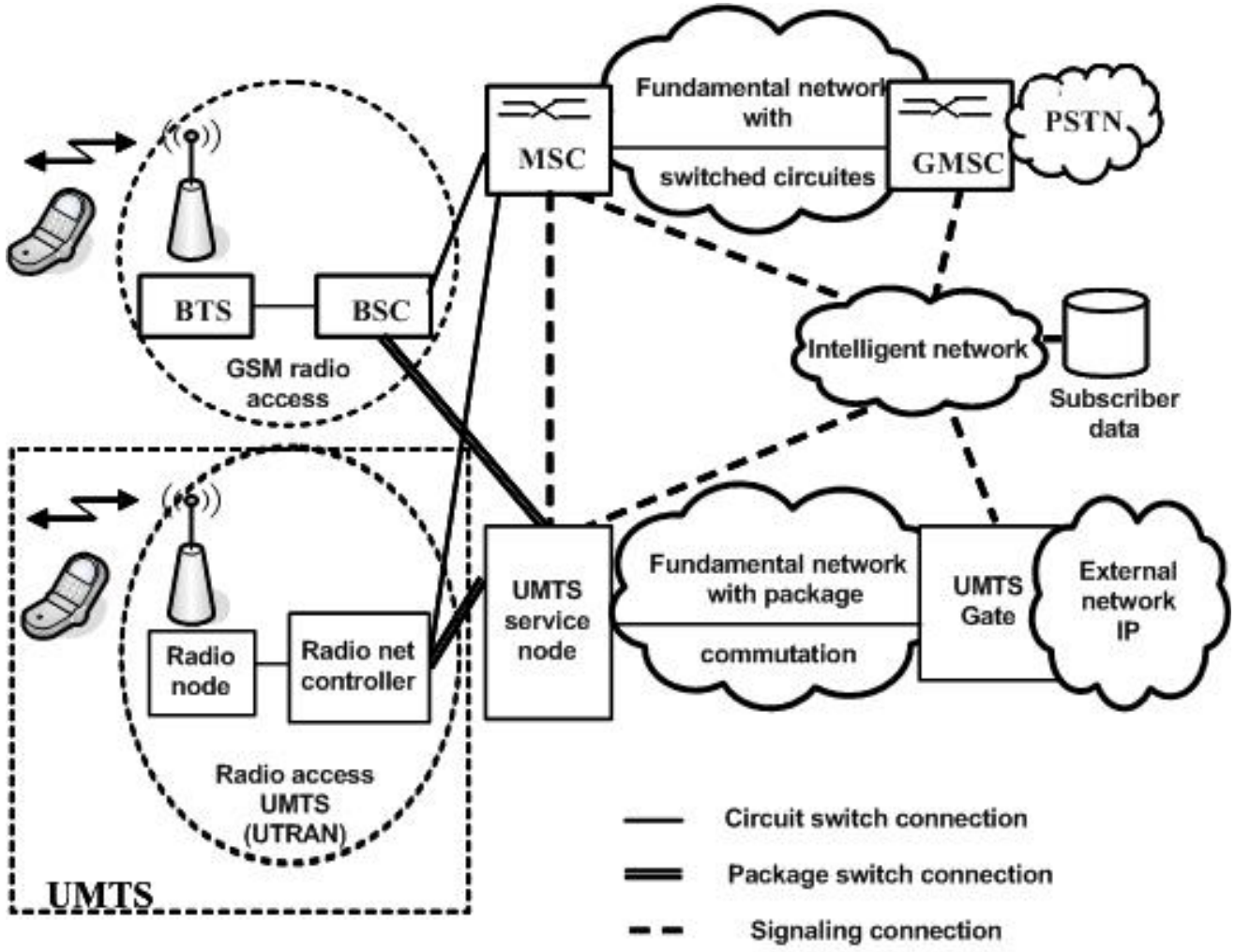
1.1 GSM Networks Overview

Wireless Spectrum and Speed Rates





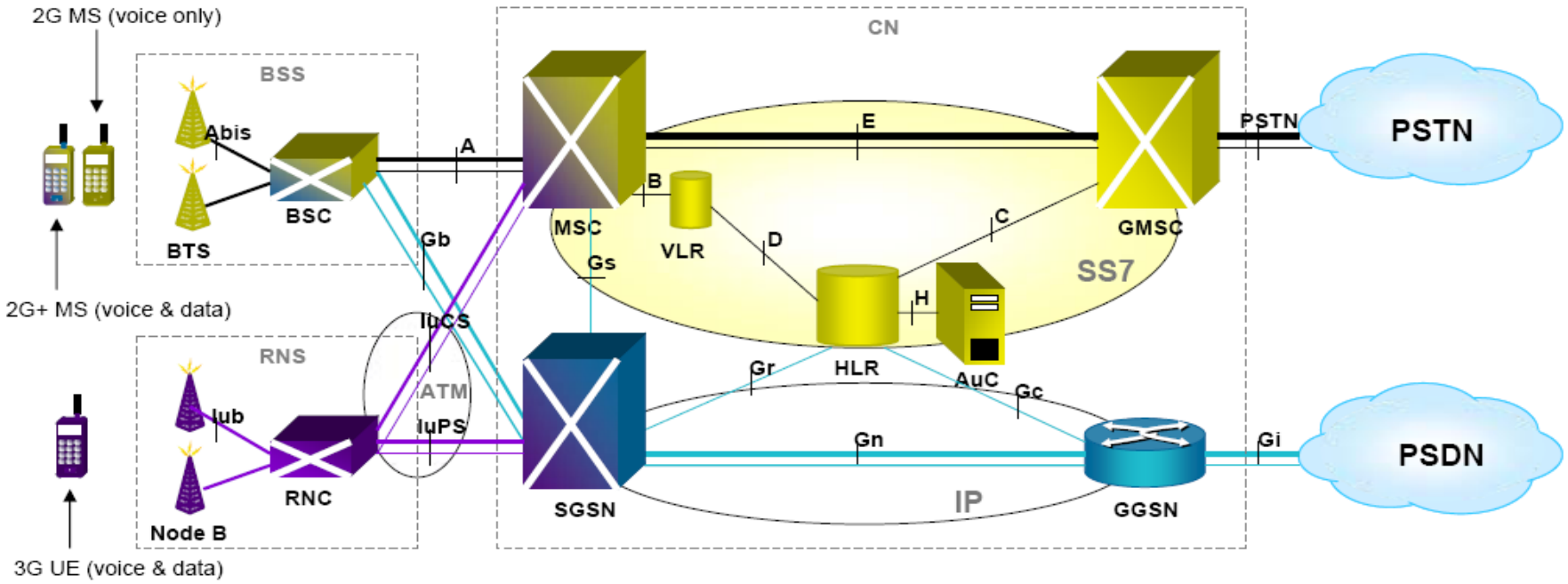
Migration from GPRS 2 UMTS '99 Architecture



3G-UMTS '99 Features

- ITU called 3G Networks as IMT – 2000 (International Mobile Communication System) or in EU as UMTS (Universal Mobile Telecommunication System)
- 3G is based on Wide CDMA technology for radio network resources acces
- On the UMTS 99 networks, it is 2.5G network architecture plus:
  - UTRAN – UMTS Terrestrial Radio Access Network aka RNS – Radio Network System
  - Node B – has special hardware for WCDMA signaling processing
  - RNC – Radio Network Controller
  - UMTS enable devices
  - Quite more headaches … but Government license fee

3G Architecture Overview - UMTS 99



BSS — Base Station System

BTS — Base Transceiver Station

BSC — Base Station Controller

RNS — Radio Network System

RNC — Radio Network Controller

CN — Core Network

MSC — Mobile-service Switching Controller

VLR — Visitor Location Register

HLR — Home Location Register

AuC — Authentication Server

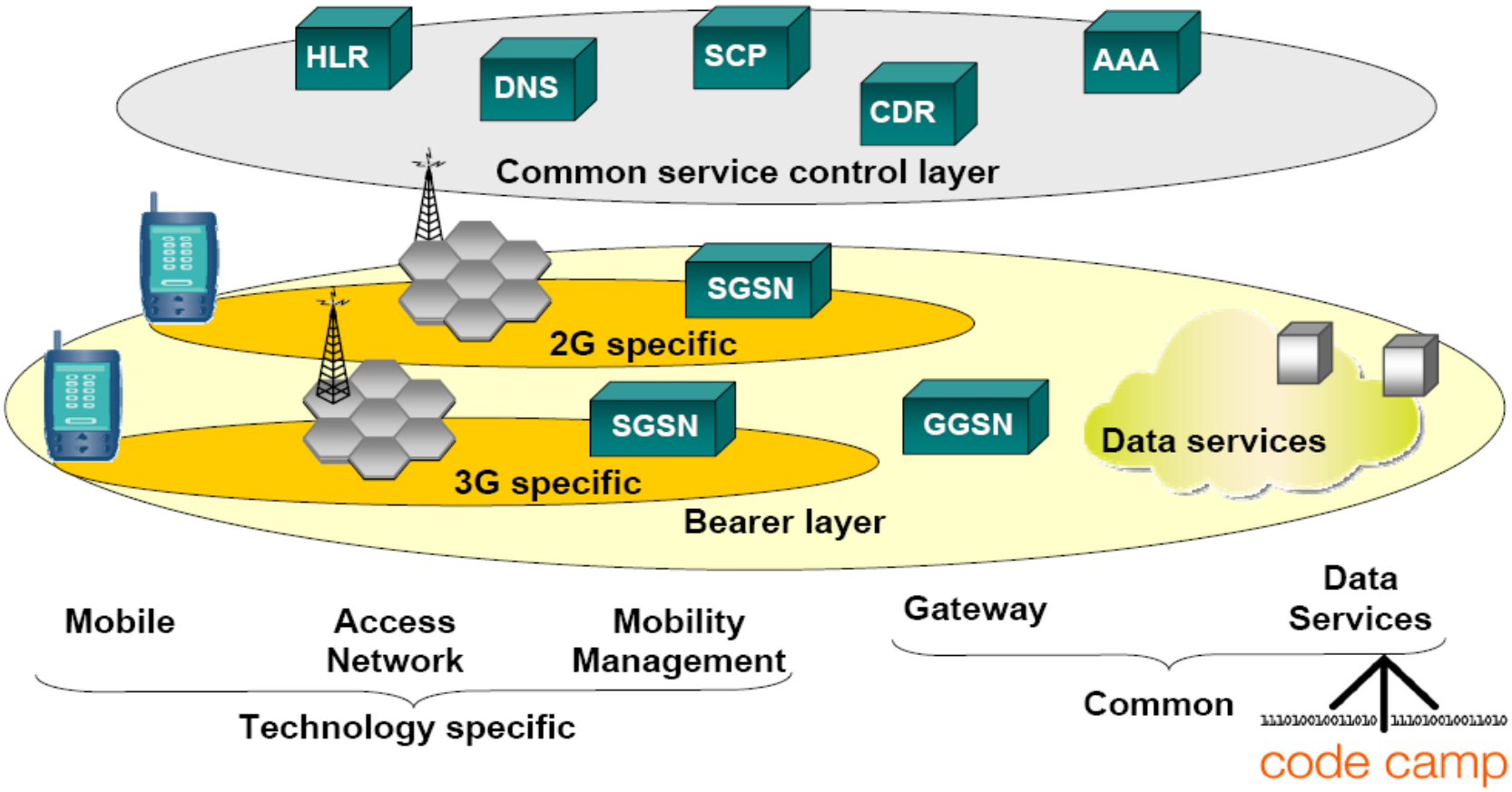
GMSC — Gateway MSC

SGSN — Serving GPRS Support Node

GGSN — Gateway GPRS Support Node

UMTS — Universal Mobile Telecommunication System

3G Bearer Service Elements – UMTS 99



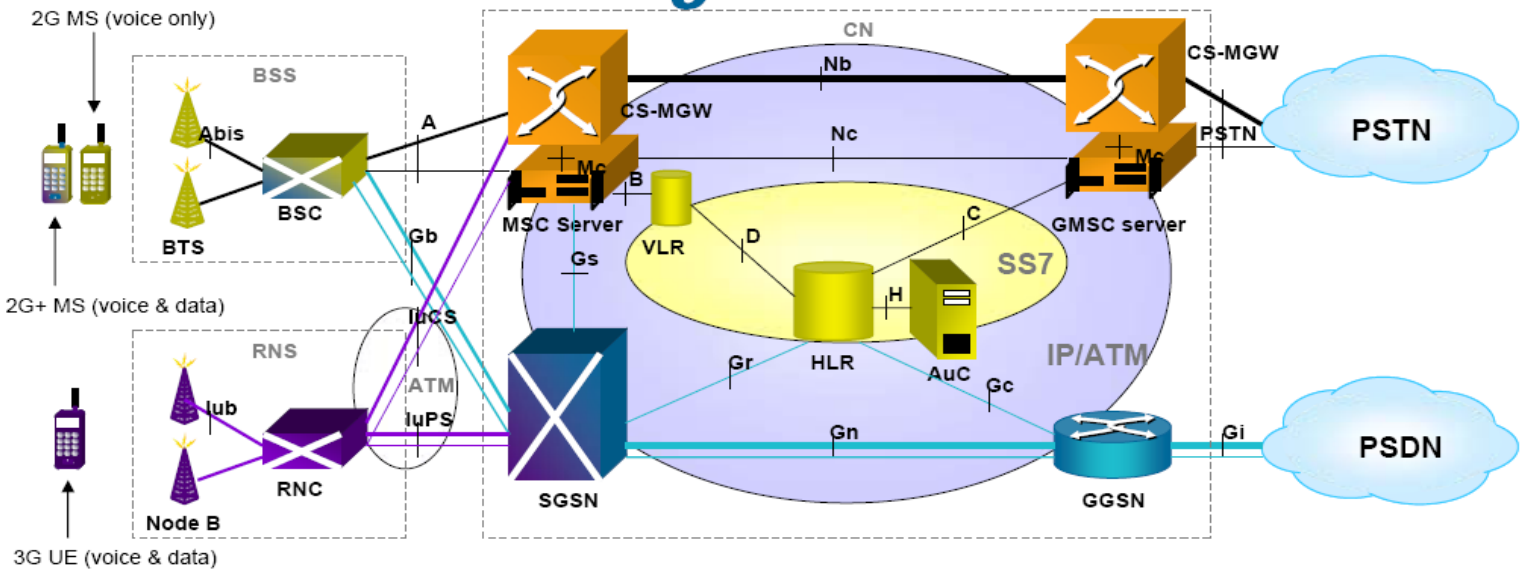
3G Versions

3G Partnership Project (3GPP)

- 3GPP defining migration from GSM to UMTS (W-CDMA)
  - Core network evolves from GSM-only to support GSM, GPRS and new W-CDMA facilities
- 3GPP Release 99
  - Adds 3G radios
- 3GPP Release 4
  - Adds softswitch/ voice gateways and packet core
- 3GPP Release 5
  - First IP Multimedia Services (IMS) w/ SIP & QoS
- 3GPP Release 6
  - "All IP" network; contents of r6 still being defined

Migration from GPRS 2 UMTS 4 Architecture

3G rel4 Architecture (UMTS) — *Soft Switching*



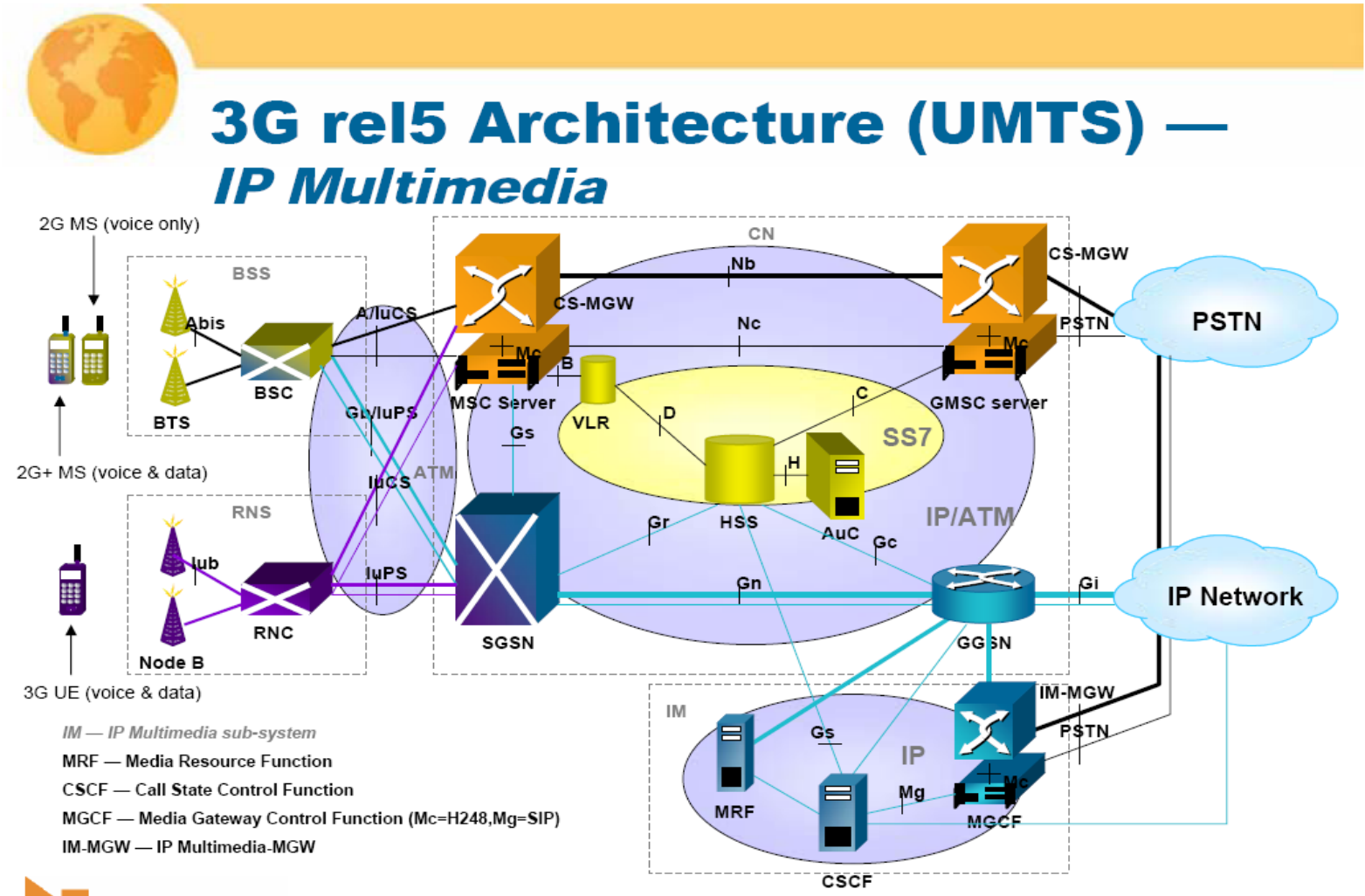
BSS — Base Station System
BTS — Base Transceiver Station
BSC — Base Station Controller

RNS — Radio Network System
RNC — Radio Network Controller

CN — Core Network
MSC — Mobile-service Switching Controller
VLR — Visitor Location Register
HLR — Home Location Register
AuC — Authentication Server
GMSC — Gateway MSC

SGSN — Serving GPRS Support Node
GGSN — Gateway GPRS Support Node

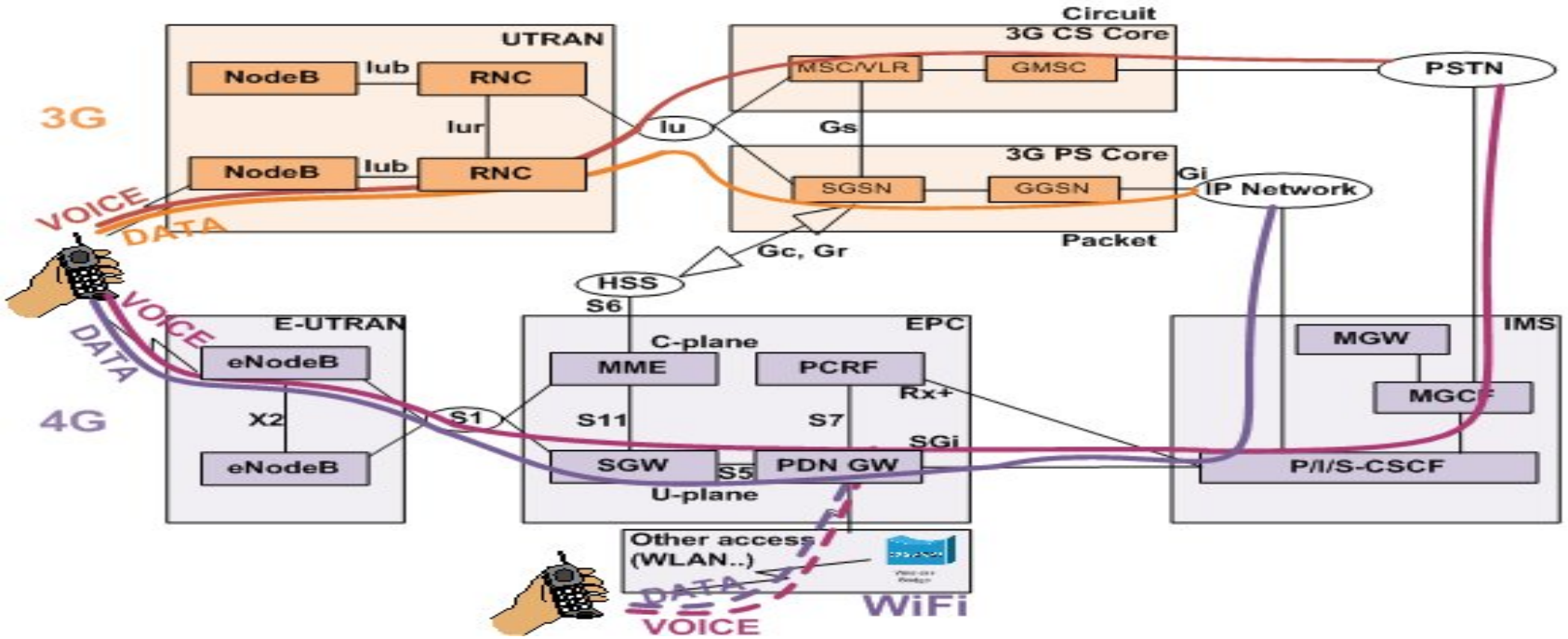
Migration from GPRS 2 UMTS 5 Architecture



1.1 GSM Networks Overview

GSM 4G - Voice and data bearer paths in Universal Mobile Telecommunication System (UMTS) and Long Term Evolution (LTE) network architectures

http://stack.nil.si/ipcorner/VoiceoverLTE/



When comparing the voice and data bearer paths in 3G and 4G network architectures, VoLTE uses a few new nodes:

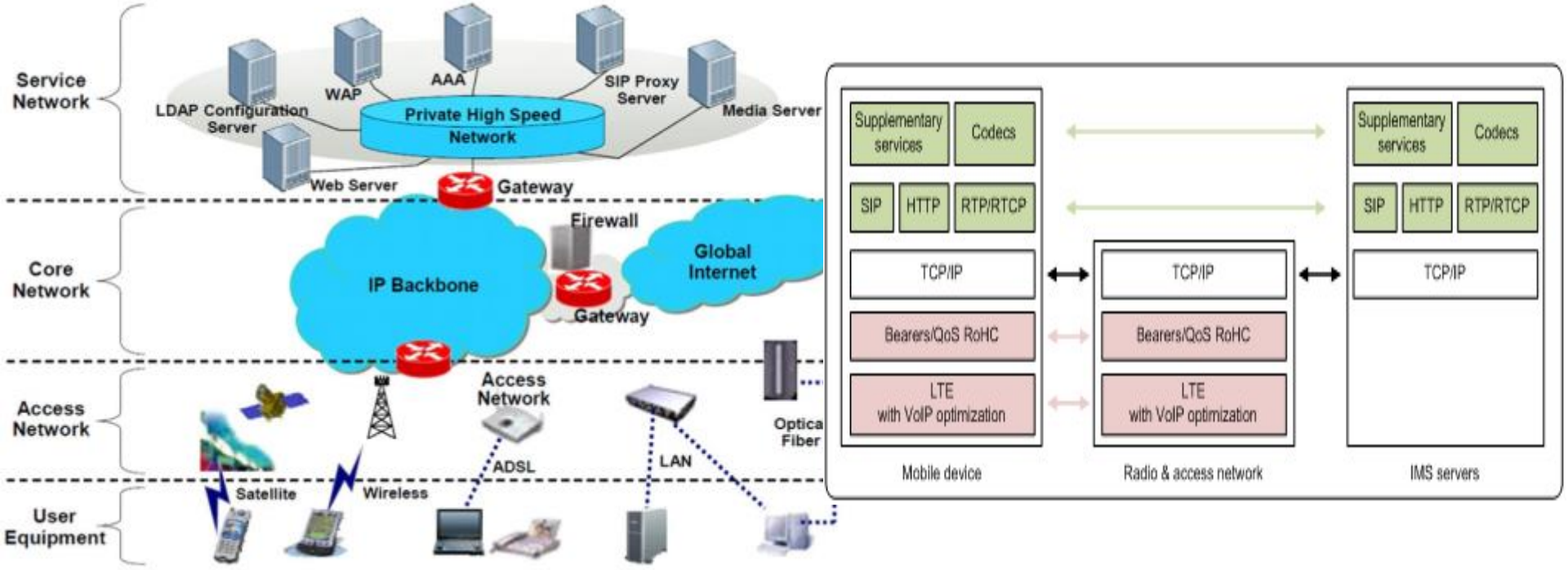
- Home Subscriber Server (**HSS**) performs AAA and subscriber database functionality, eliminating the need for a separate Radius server
- The Policy and Charging Rules Function (**PCRF**) server is an important element, taking care of the direct control of resources (quality of service) according to user profile
- Mobility Management Entity (**MME**) has a role similar to that of the Serving GPRS Support Node (SGSN) in the 3G packet core, except that it only carries the signaling path
- Serving GW (**SGW**) has inherited the SGSN's bearer role in 4G, transporting large amounts of traffic
- Packet Data Network Gateway (**PDN GW**) has replaced the Gateway GPRS Support Node (GGSN) in the 3G packet core, thus having the Network Access Server (NAS) role
- eNodeB serves in 4G instead of NodeB and Radio Network Controller (RNC) in 3G
- Acronyms: Evolved Packet Core (EPC) + IP Multimedia Subsystem (IMS) + E-UTRAN (evolved UMTS terrestrial radio access)..

1.1 GSM Networks Overview

GSM 4G – Protocol stack in LTE and IMS architecture

http://stack.nil.si/ipcorner/VoiceoverLTE/

http://www.telefocal.com/macrosite/home/254-management-of-next-generation-4g-lte-networks



The Role of the All-IP Network (AIPN)

The all-IP network (AIPN, 3GPP TR 22.978 specification) is divided into the Core Network (CN), called the Evolved Packet Core (EPC) in a non-radio-related System Architecture Evolution (SAE), and the radio access network E-UTRAN (evolved UMTS terrestrial radio access). The EPC is an efficient network element, capable of delivering mobile Internet services over a variety of access technologies (2G/3G/4G, WiMAX, WiFi etc.), and performing the inter-RAT (radio access technology) handover to provide service continuity. A bearer is an IP packet flow with the designated quality of service (QoS) between the gateway and the User Equipment (UE), performing optimized Robust Header Compression (RoHC), of IP, UDP, RTP and TCP headers, considered for the wireless links (having a high packet loss rate). The UE and the network are required to support both IPv4 and IPv6. This concept has changed since evolved NodeB (eNodeB), which is directly connected to the IP cloud. Similarly to 3G is the GPRS tunneling protocol (GTP) tunnel between eNodeB, SGW and Packet Data Network Gateway PGW.

1.1 GSM Networks Overview

GSM – Evolution

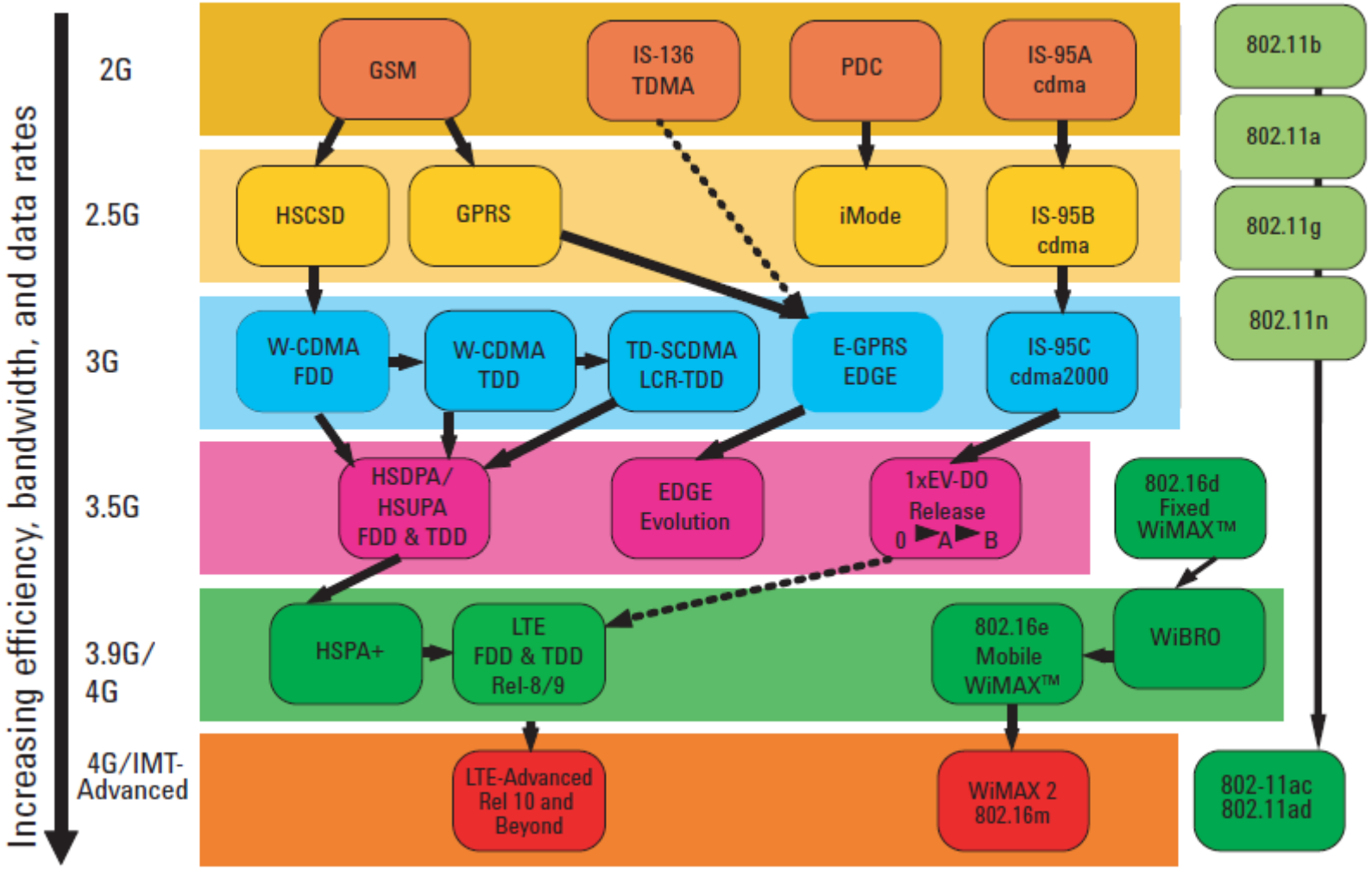
http://www.radio-electronics.com/info/cellulartelecomms/lte-long-term-evolution/3gpp-4g-imt-lte-advanced-tutorial.php

	WCDMA (UMTS)	HSPA HSDPA / HSUPA	HSPA+	LTE	LTE Advanced (IMT Advanced)
Max downlink speed bps	384 k	14 M	28 M	100M	1G
Max uplink speed bps	128 k	5.7 M	11 M	50 M	500 M
Latency round trip time approx	150 ms	100 ms	50ms (max)	~10 ms	less than 5 ms
3GPP releases	Rel 99/4	Rel 5 / 6	Rel 7	Rel 8	Rel 10
Approx years of initial roll out	2003 / 4	2005 / 6 HSDPA 2007 / 8 HSUPA	2008 / 9	2009 / 10	
Access methodology	CDMA	CDMA	CDMA	OFDMA / SC-FDMA	OFDMA / SC-FDMA

1.1 GSM Networks Overview

GSM – Evolution

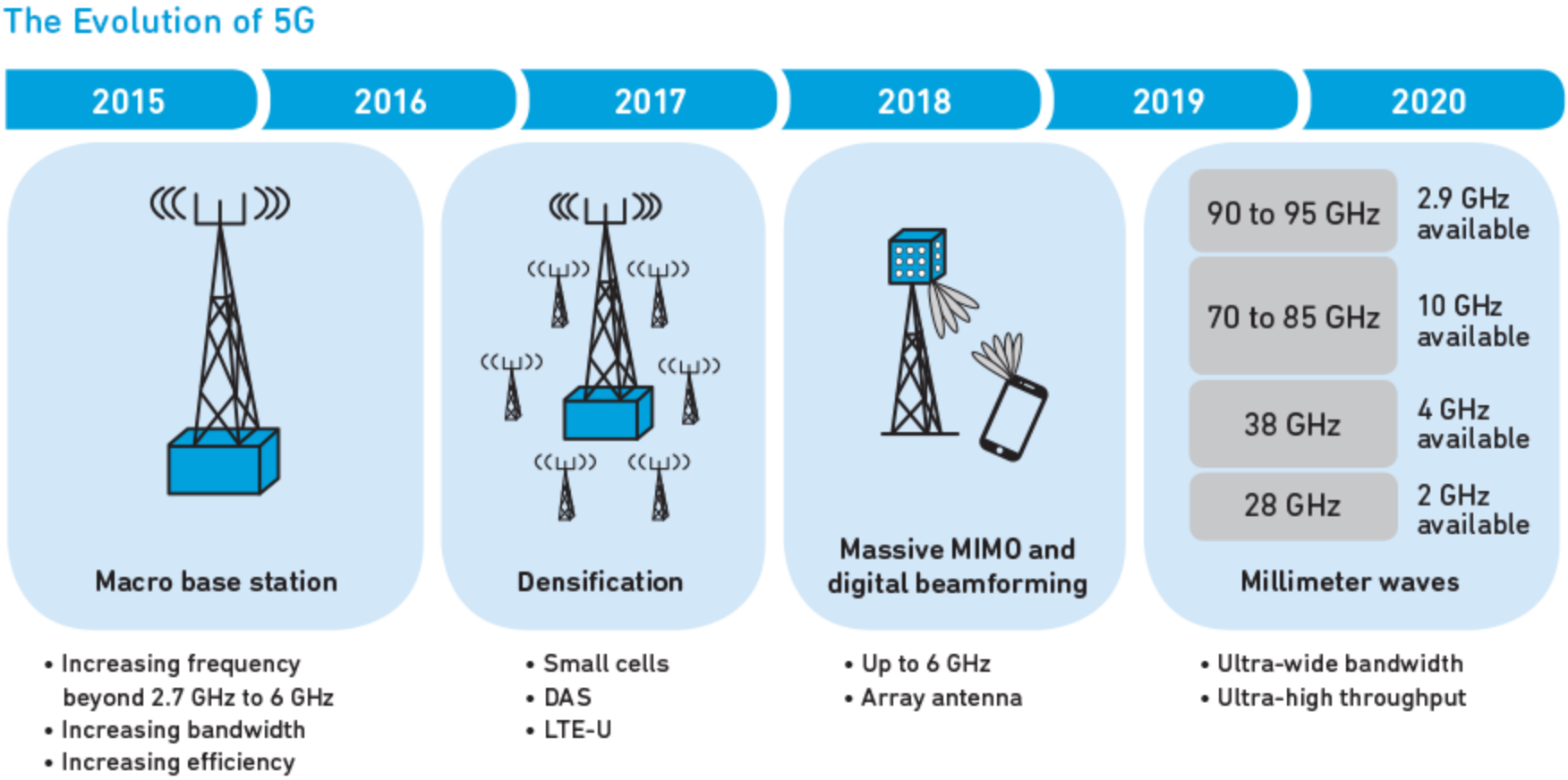
http://cp.literature.agilent.com/litweb/pdf/5990-6706EN.pdf



1.1 GSM Networks Overview

GSM – 5G

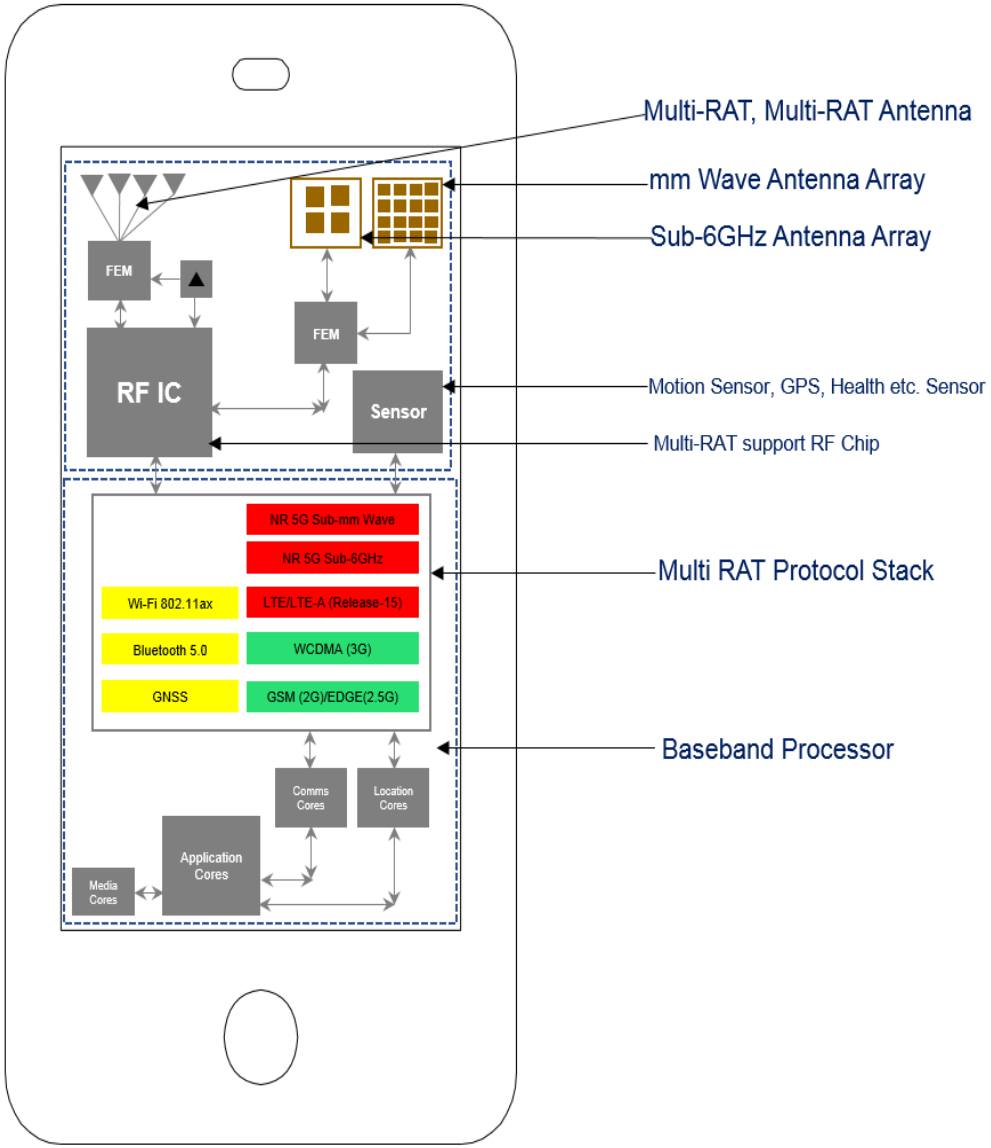
https://www.qorvo.com/design-hub/blog/small-cell-networks-and-the-evolution-of-5g



1.1 GSM Networks Overview

GSM – 5G

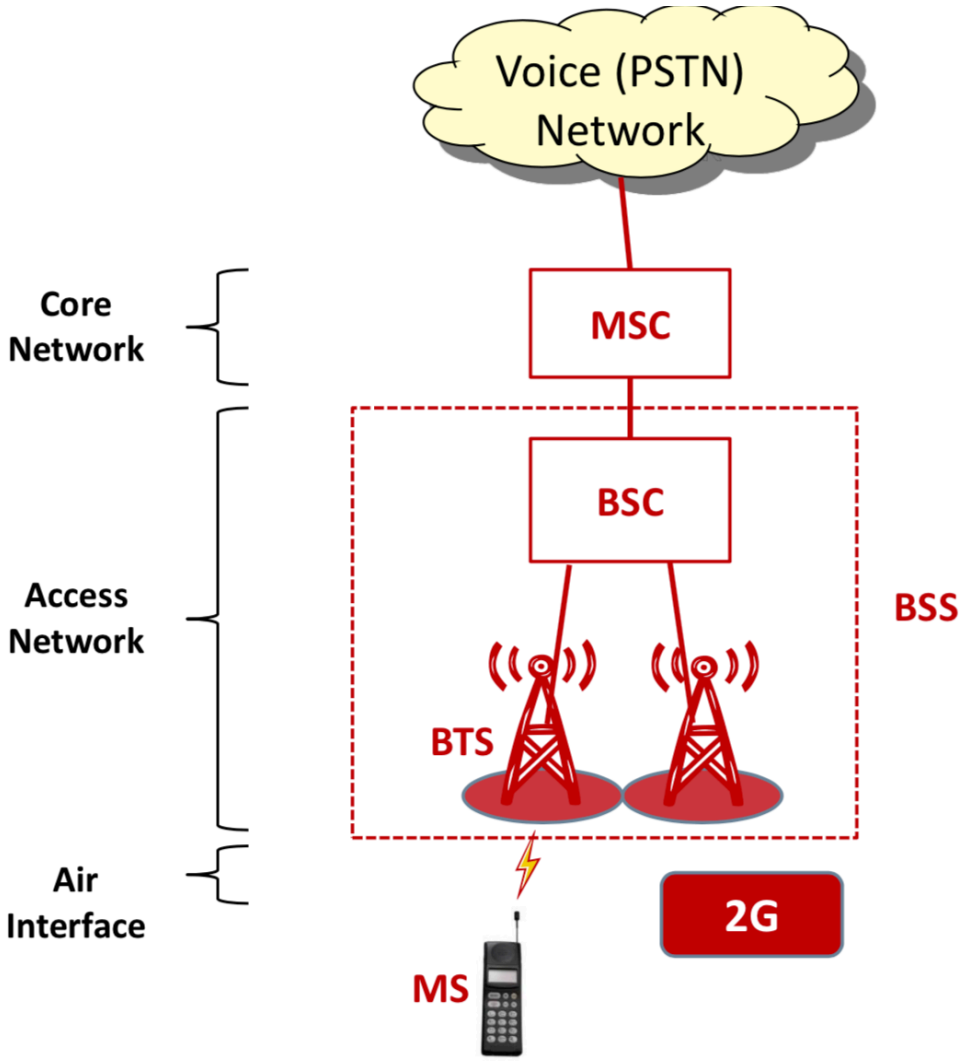
http://www.techplayon.com/5g-cell-phone-architecture/



	Peripheral connectivity	Local (home) networking	Wide area networking
Typical range	<30 ft.	<300 ft.	Outdoor (miles)
Content distribution Focus on high data rates ergy consumption secondary	Bluetooth	Wi Fi IEEE 802. llax .11ah: HaLow	GSM Lte 5G GPRS LTE Cat-M NB-IoT
Sense and control Low energy/long battery life Data rate is secondary	Bluetooth SMART	ZigBee	
Proprietary solutions	ANT	Sub-GHz enocean	LoRa SIGFOX INGENU
Typical applications	Personal appliances (wristband, smartwatch, step counter, keyboard, mouse, pointer, etc.)	Indoor networks (internet, email, phone, security, energy management, smart home monitoring, etc.)	Outdoor networks (smartphone, internet, city, industry 4.0, agriculture, smart logistics, etc.)

1.1 GSM Networks Overview

GSM – Infrastructure Recap

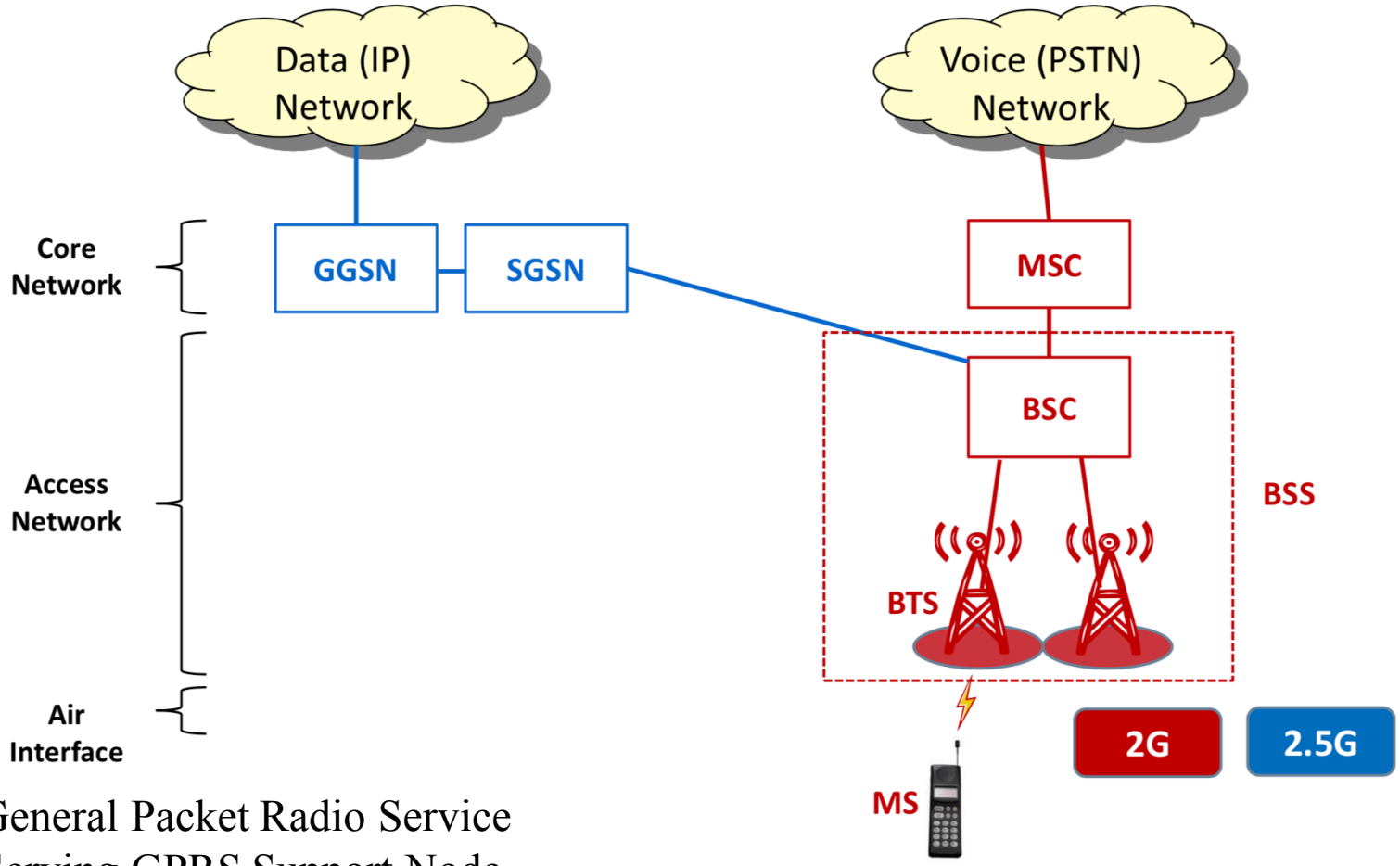


MSC = Mobile Switching Centre
BSS = Base Station Subsystem
BSC = Base Station Controller
BTS = Base Transceiver Station
MS = Mobile Station

https://www.slideshare.net/3G4GLtd/highlevel-architecture-of-mobile-cellular-networks-from-2g-to-5g

1.1 GSM Networks Overview

GSM – Infrastructure Recap



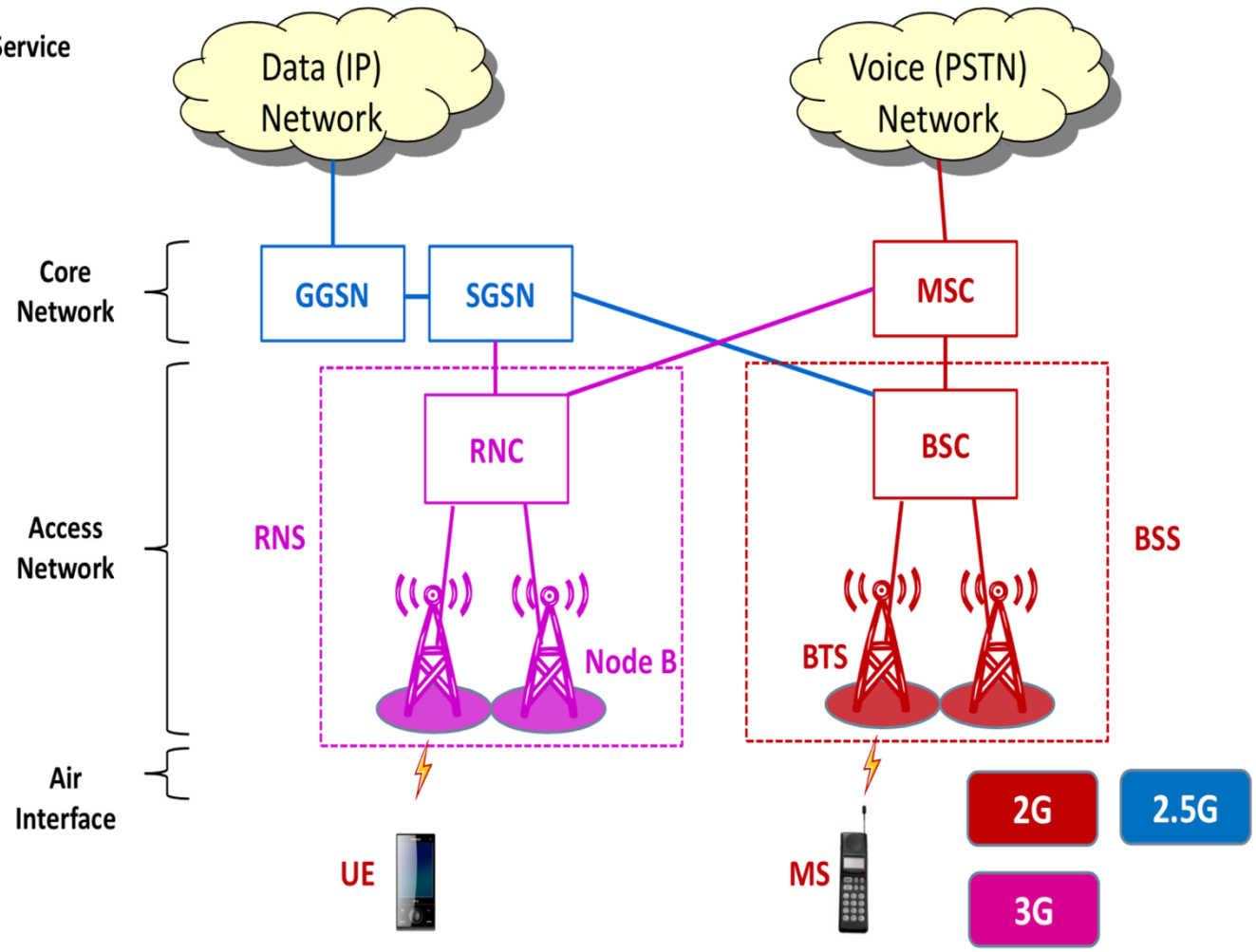
GPRS = General Packet Radio Service
SGSN = Serving GPRS Support Node
GGSN = Gateway GPRS Support Node

https://www.slideshare.net/3G4GLtd/highlevel-architecture-of-mobile-cellular-networks-from-2g-to-5g

1.1 GSM Networks Overview

GSM – Infrastructure Recap

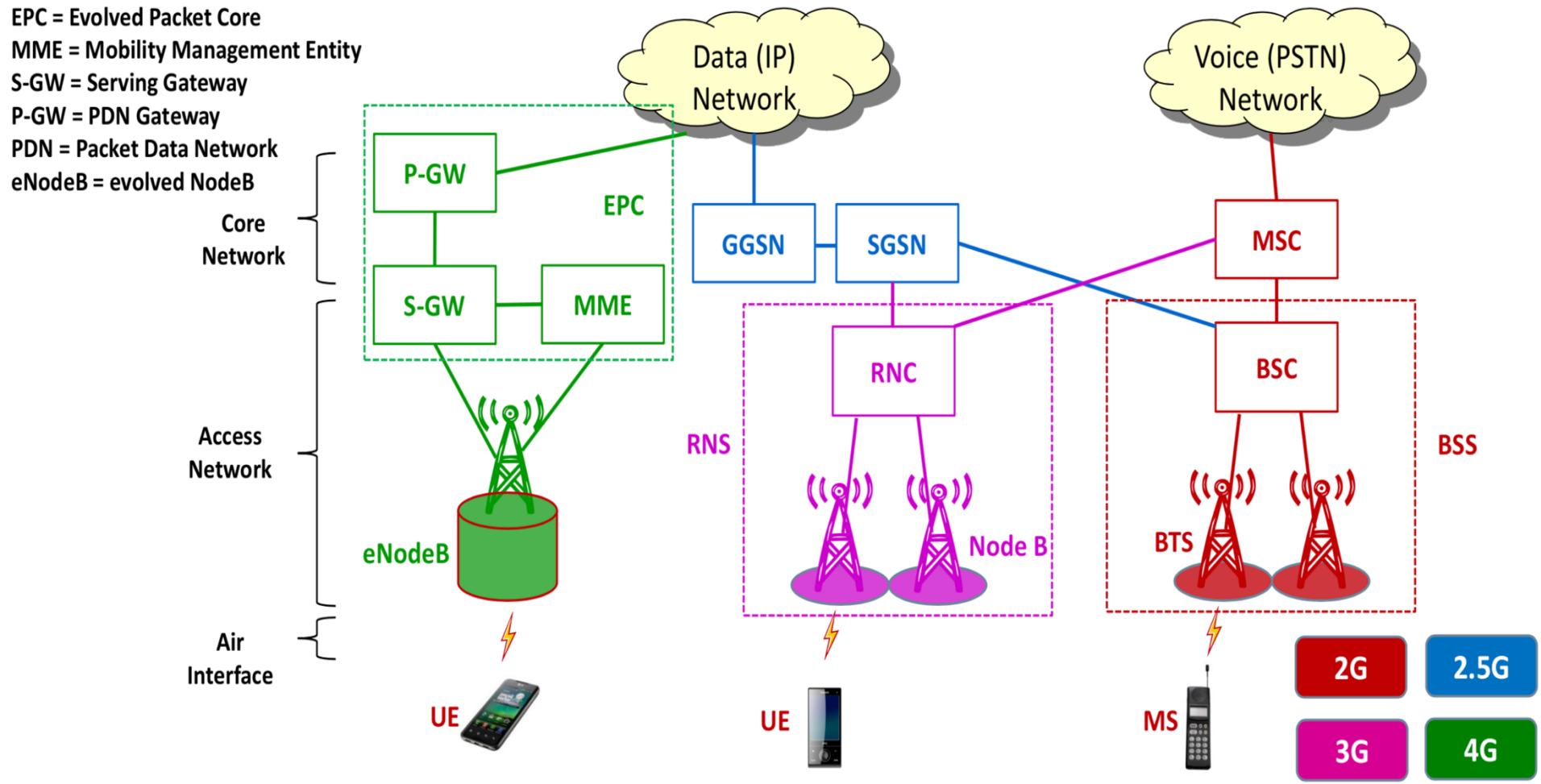
UMTS = Universal Mobile Telecommunications Service
RNC = Radio Network Controller
RNS = Radio Network Subsystem
UE = User Equipment



https://www.slideshare.net/3G4GLtd/highlevel-architecture-of-mobile-cellular-networks-from-2g-to-5g

1.1 GSM Networks Overview

GSM – Infrastructure Recap

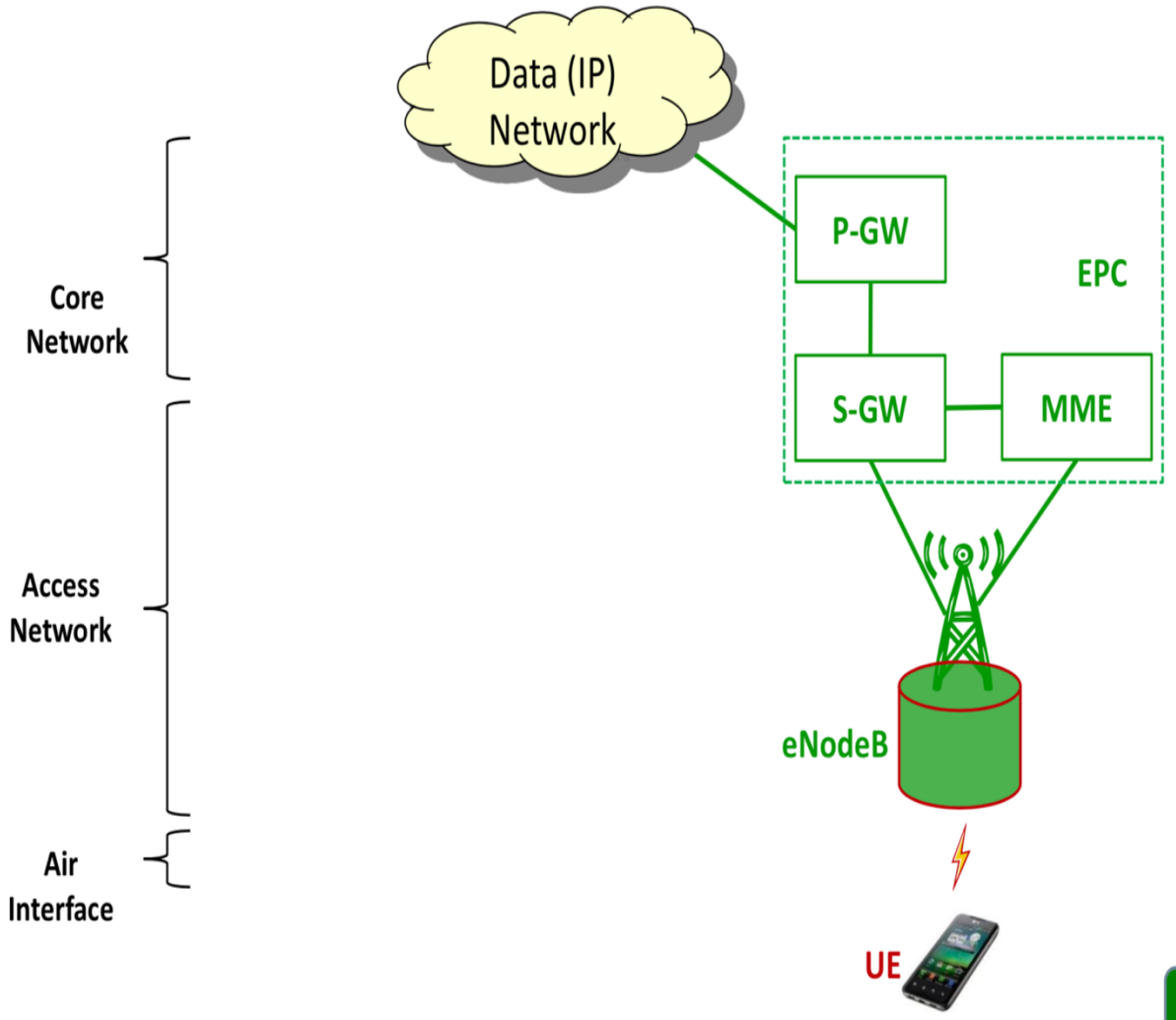


https://www.slideshare.net/3G4GLtd/highlevel-architecture-of-mobile-cellular-networks-from-2g-to-5g

1.1 GSM Networks Overview

GSM – Infrastructure Recap

EPC = Evolved Packet Core
MME = Mobility Management Entity
S-GW = Serving Gateway
P-GW = PDN Gateway
PDN = Packet Data Network
eNodeB = evolved NodeB



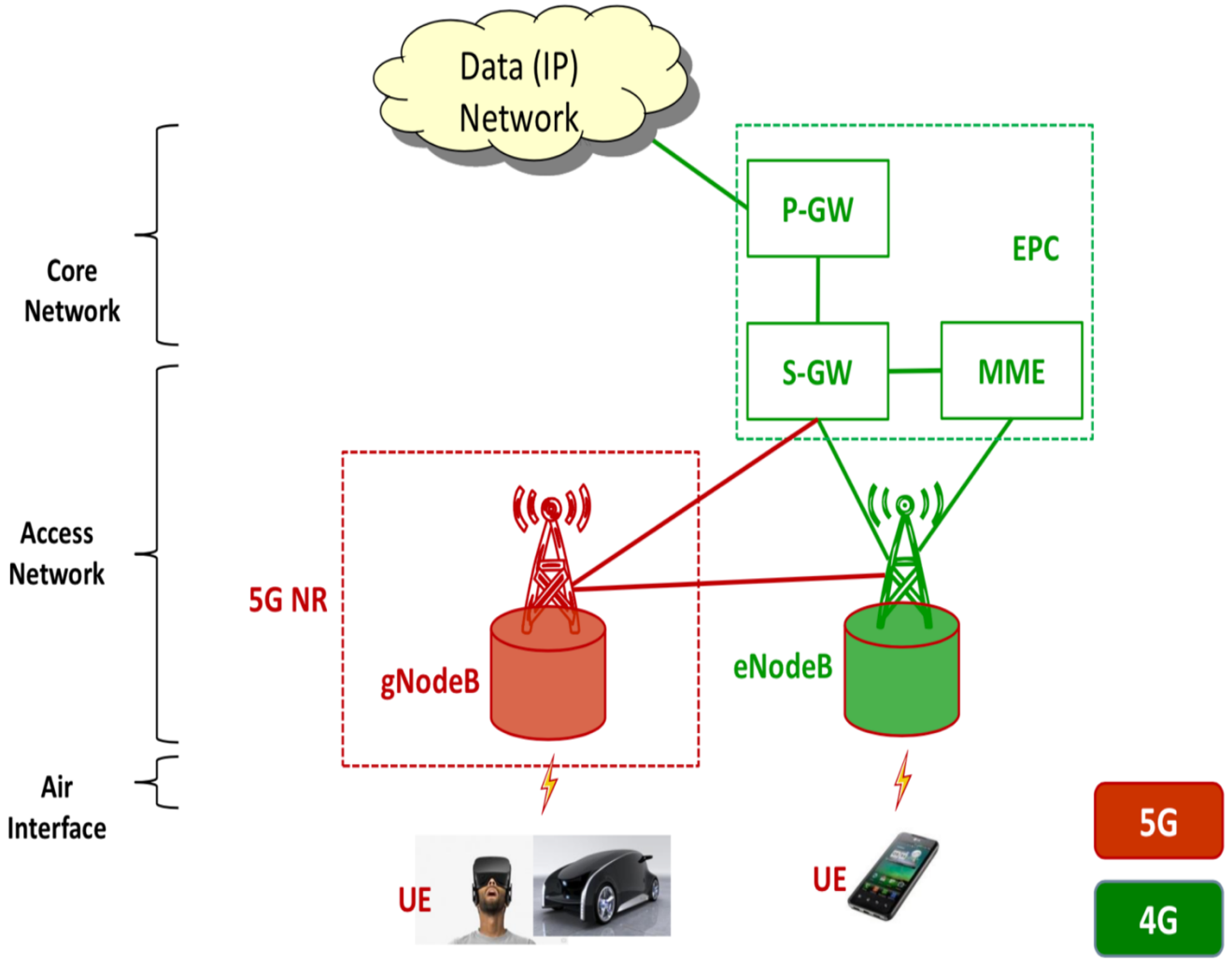
4G

https://www.slideshare.net/3G4GLtd/highlevel-architecture-of-mobile-cellular-networks-from-2g-to-5g

1.1 GSM Networks Overview

GSM – Infrastructure Recap – 5G Phase 1

gNodeB = next generation NodeB
NR = New Radio

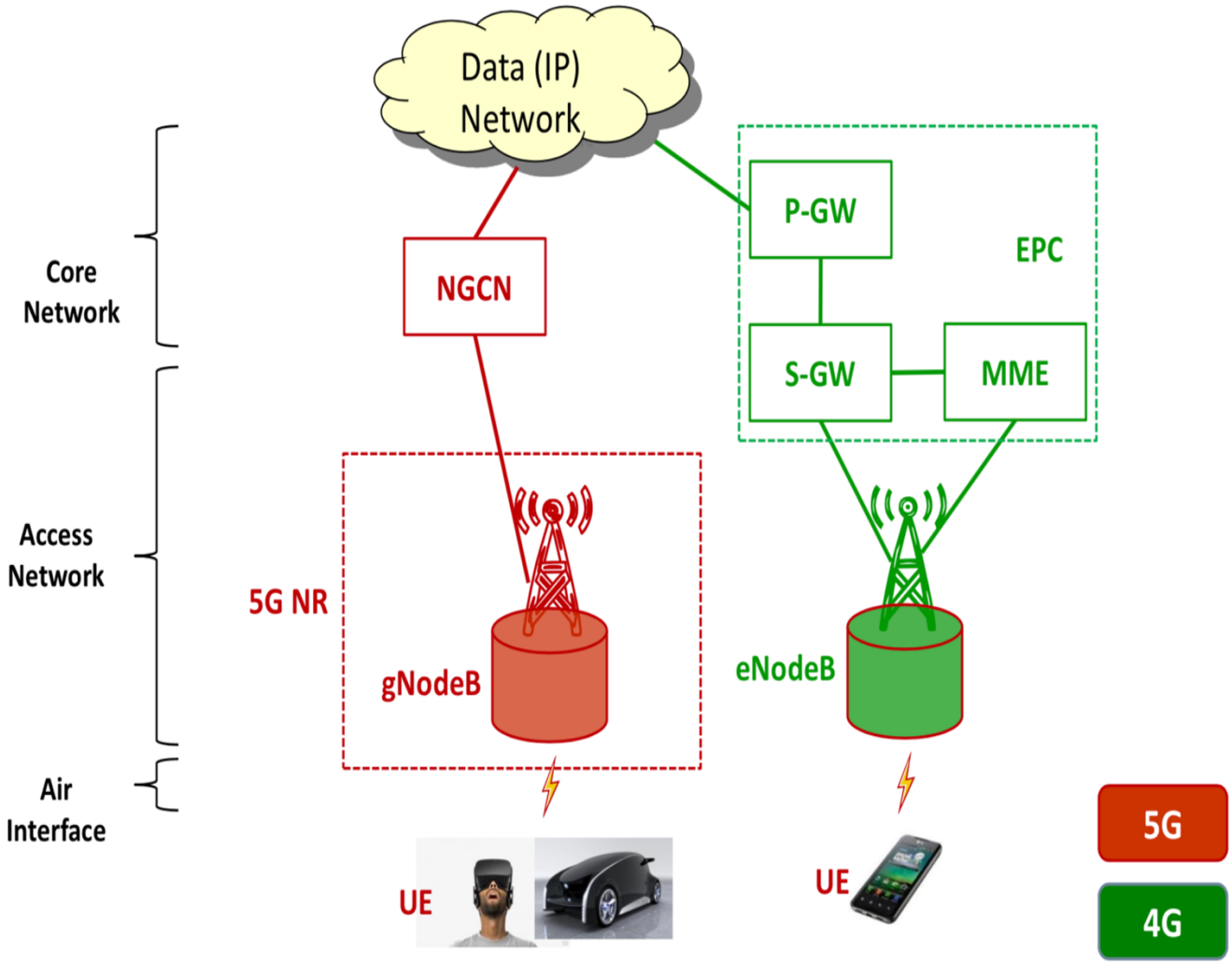


https://www.slideshare.net/3G4GLtd/highlevel-architecture-of-mobile-cellular-networks-from-2g-to-5g

1.1 GSM Networks Overview

GSM – Infrastructure Recap – 5G Phase 2

NGCN = Next Generation Core Network
gNodeB = next generation NodeB
NR = New Radio

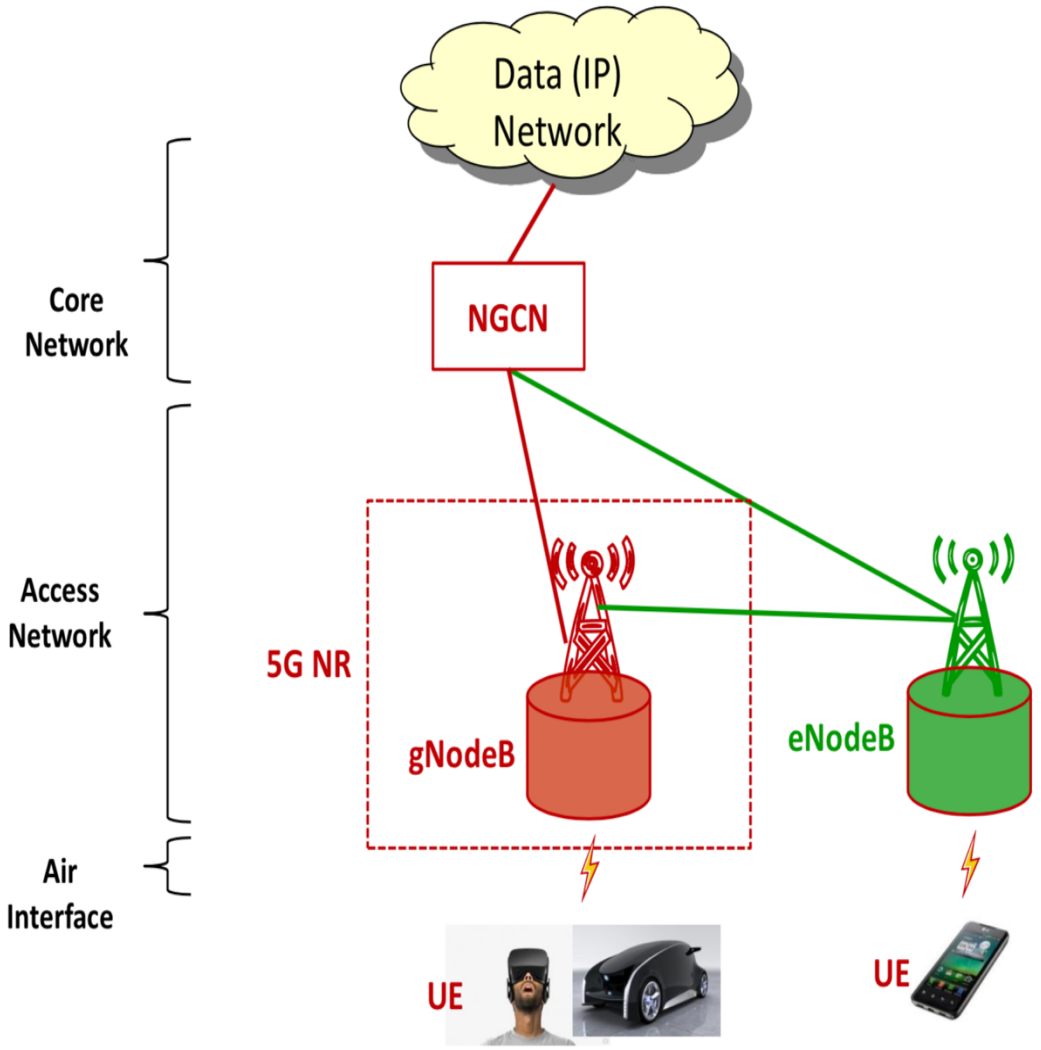


https://www.slideshare.net/3G4GLtd/highlevel-architecture-of-mobile-cellular-networks-from-2g-to-5g

1.1 GSM Networks Overview

GSM – Infrastructure Recap – 5G Phase 3

NGCN = Next Generation Core Network
gNodeB = next generation NodeB
NR = New Radio



5G

4G

1.1 GSM Networks Overview

GSM – Infrastructure Security Recap

AN – Access Network
AS – Access Stratum
RRC – Radio Resource Control
NAS – Non-Access Stratum
CP – Control Plane
UP – User Plane

MS / UE

BTS / NodeB

BSC / RNC / eNodeB

MSC/SGSN/EPC

GSM
- Handset Authentication
- Ciphering (AN CP, UP)

GPRS
- Handset Authentication + Ciphering (AN CP, UP)

UMTS
- Mutual Authentication
- Ciphering (RRC / AN CP, UP) + Signalling Integrity (RRC)
- IPSec (Optional)

LTE
- Mutual Authentication
- Ciphering (RRC / AN CP, UP) + Signalling Integrity (RRC)
- IPSec (Optional)
- Ciphering (NAS) + Signalling Integrity (NAS)

1.1 GSM Networks Overview

GSM – Infrastructure Security Recap

Summary of Algorithms for 2G, 3G & 4G

	GSM	GPRS	UMTS	LTE
Authentication Algorithms	GSM Milenage	GSM Milenage	Milenage TUAK	Milenage TUAK
Integrity Algorithms			UIA0 – NULL UIA1 – Kasumi UIA2 – Snow3G	EIA0 – NULL EIA1 – Snow3G EIA2 – AES EIA3 – ZUC
Ciphering Algorithms	A5/1 A5/2 A5/3 A5/4	GEA3 GEA4	UEA0 - NULL UEA1 – Kasumi UEA2 – Snow3G	EEA0 – NULL EEA1 – Snow3G EEA2 – AES EEA3 – ZUC

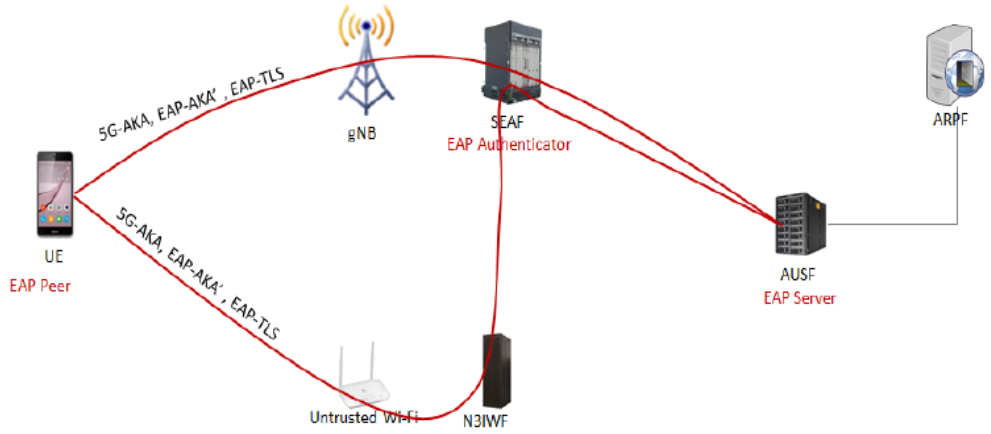
GSM Milenage - 3GPP TS 55.205, Milenage - 3GPP TS 35.206, TUAK - 3GPP TS 35.231, A5/3 & GEA3 - 3GPP TS 55.216, A5/4 & GE4 - 3GPP TS 55.226

For other specifications see GSMA Security Algorithms: (https://www.gsma.com/aboutus/workinggroups/working-groups/fraud-security-group/security-algorithms)

GSM – 5G Security

Unified Authentication Framework

- Build up an unified authentication framework for different access technology, enable security context sharing among different access technology:
  - ARPF: credential repository
  - AUSF: authentication server
  - SEAF: security anchor
  - EAP framework are supported, a critical step for 5G to become an open network platform
  - EAP extended type, EAP-5G, is used to carry the NAS signaling over untrusted N3GPP link
    - EAP-5G is an vendor specific message format to carry NAS signaling between UE and N3IWF.

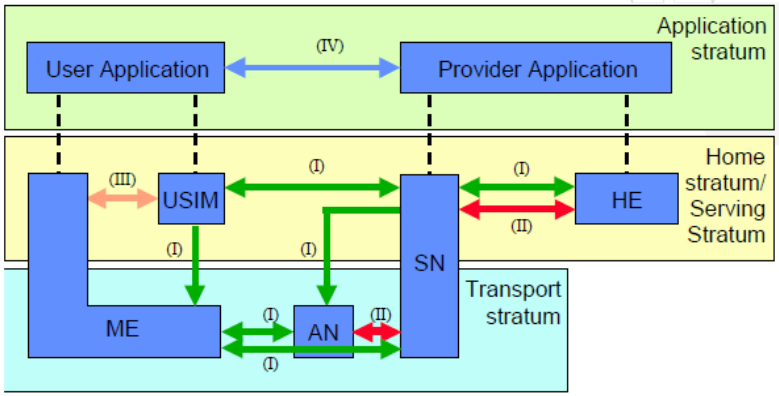


1.1 GSM Networks Overview

GSM – 5G Security

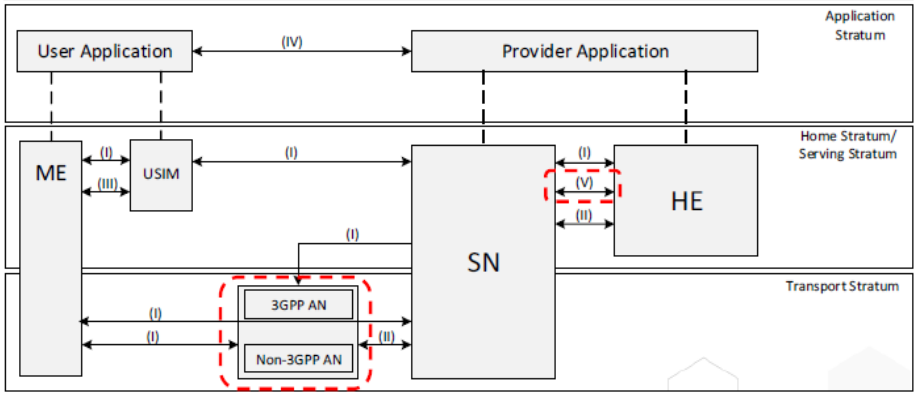
Security Architecture

4G Security Architecture (TS 33.401)



I. Network access security (I)
II. Network domain security (II)
III. User Domain Security (III)
IV. Application domain security (V)
V. Visibility and configurability of security (VI)

5G Security Architecture (TS 33.501)



I. Network access security (I)
II. Network domain security (II)
III. User Domain Security(III)
IV. Application domain security (V)
V. SBA domain security (V)
VI. Visibility and configurability of security (VI)

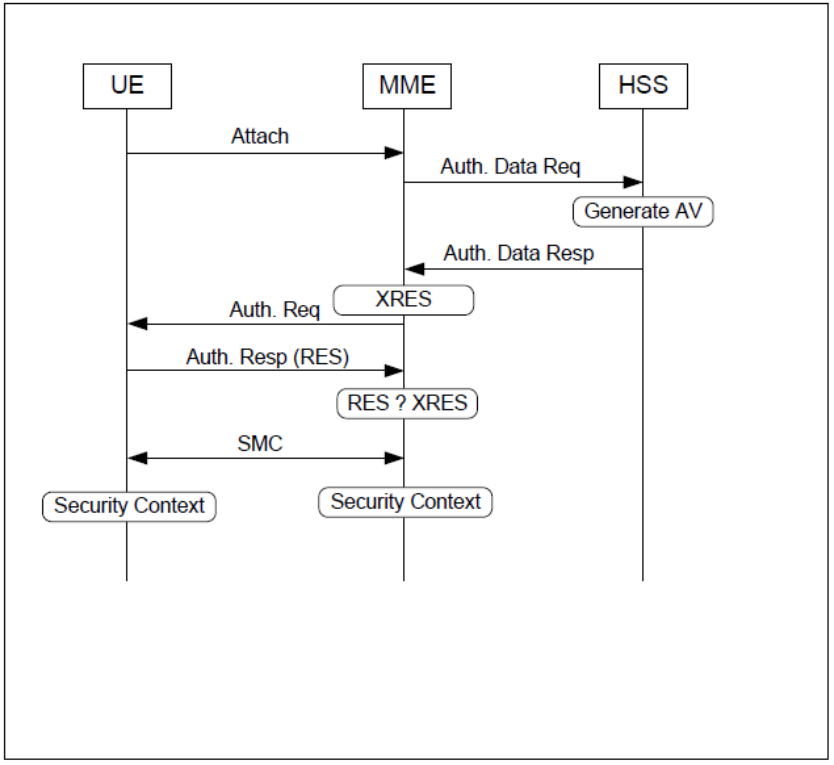
Enhancement

1. AN : 3GPP and non-3GPP access network treated more equally in access network.
2. SN←→ HE(V) : interface for Service-based Architecture

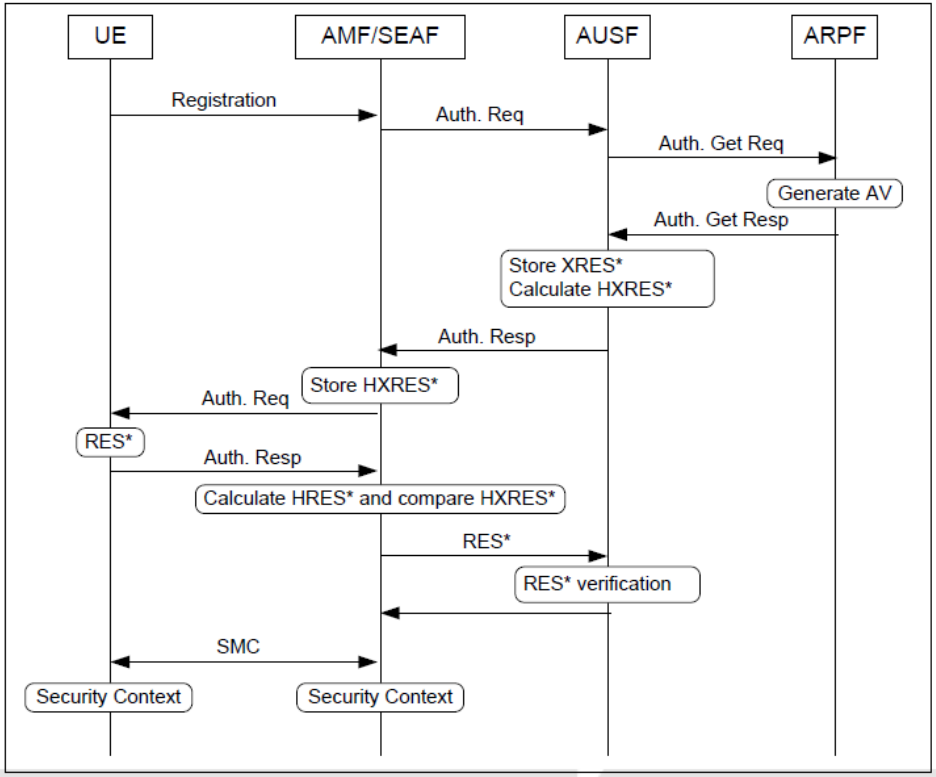
GSM – 5G Security

Authentication Protocols : EPS-AKA vs. 5G-AKA

4G: EPS-AKA



5G: EPS-AKA

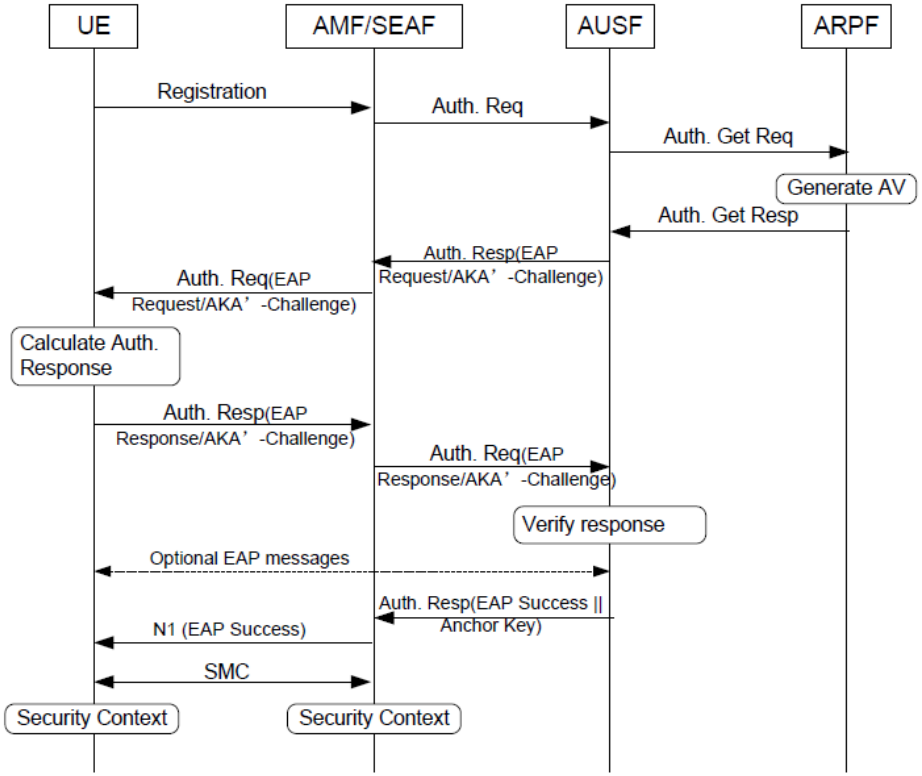


GSM – 5G Security

EAP-AKA' in 5G

Basic procedure:

1. UE send registration request to AUSF and ARPF
2. ARPF decide authentication method
3. AUSF start EAP-AKA'
4. UE and AUSF perform mutual authentication
5. AUSF send anchor key to SEAF/AMF for further key derivation
6. UE derive keys for communication.



1.1 GSM Networks Overview

GSM – 5G Security

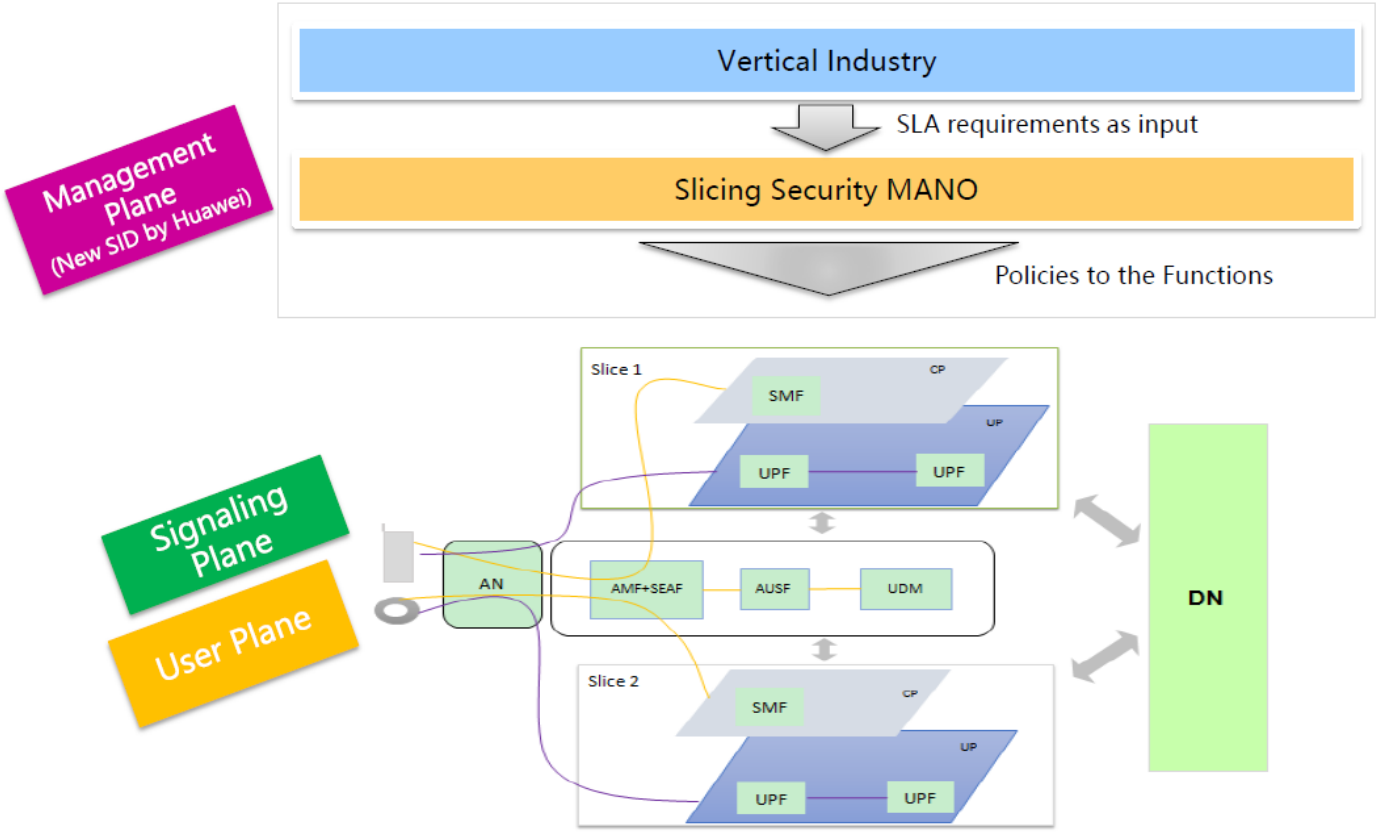
Slice security

Management Plane : Configure and manage the slicing security policies for MANO.

A new Study Item proposed by Huawei was approved in August 2017. The study results will be captured in a separate Technical Report.

Signaling Plane: Execute the slicing accessing security procedure and release the security enablers

User Plane: Execute the protection solution by analyzing the flowing map of data.



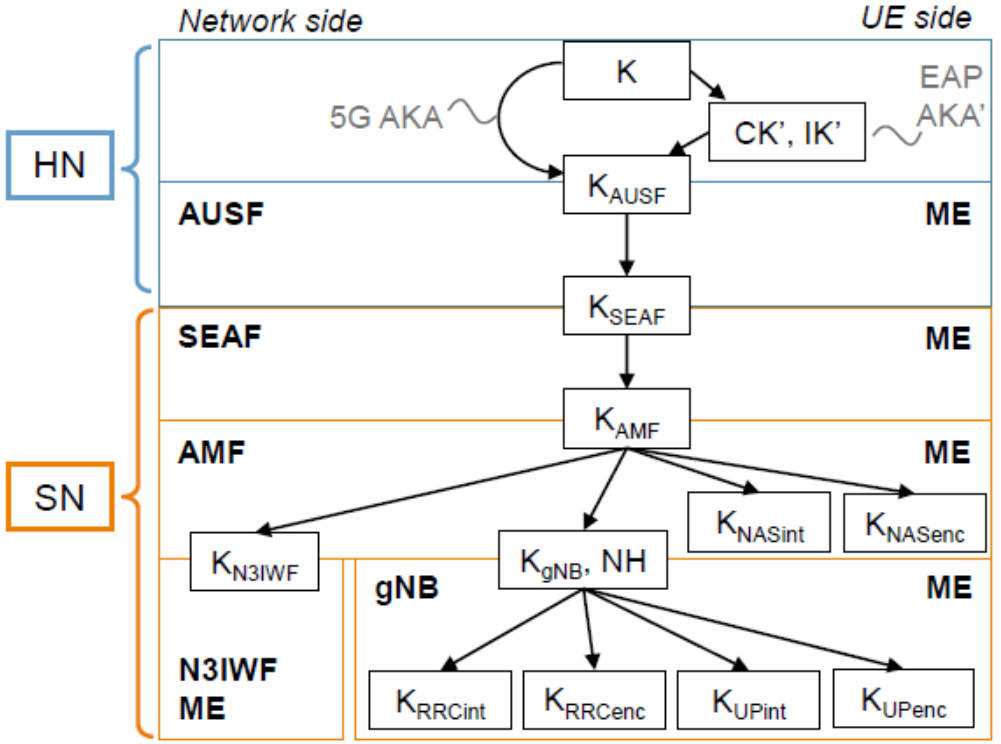
1.1 GSM Networks Overview

GSM – 5G Security

TNO innovation for life

MAJOR CHANGES IN 5G – KEY HIERARCHY

- Key hierarchy extended to also include:
  - K_{AUSF} at home network
  - K_{SEAF} at visited network
- Reasons for K_{AUSF}
  - Quick reauthentication
  - Protecting home to UE traffic, e.g. steering of roaming under discussion
- Reasons for K_{SEAF}:
  - Separate security anchor from mobility anchor
  - Pre-empts AMF at insecure locations

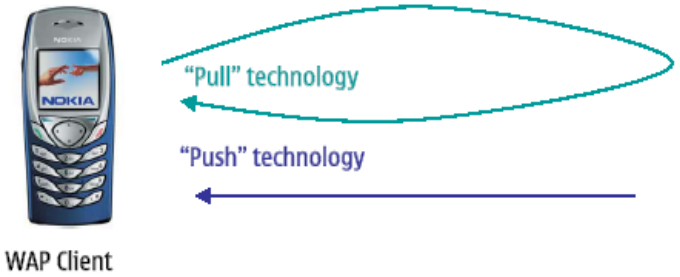


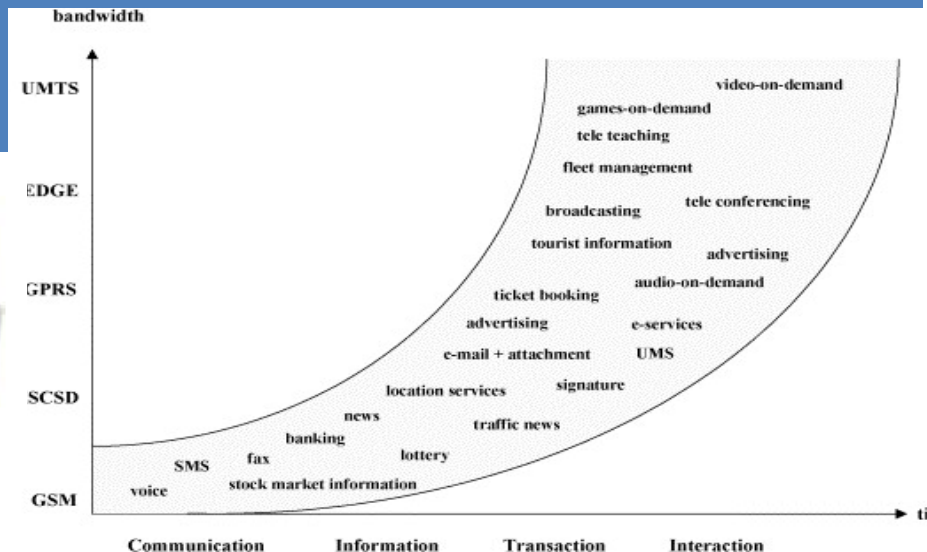
3GPP SA3 - 5G Security

1.2 Mobile Applications Overview

Mobile Applications Types & Business Services

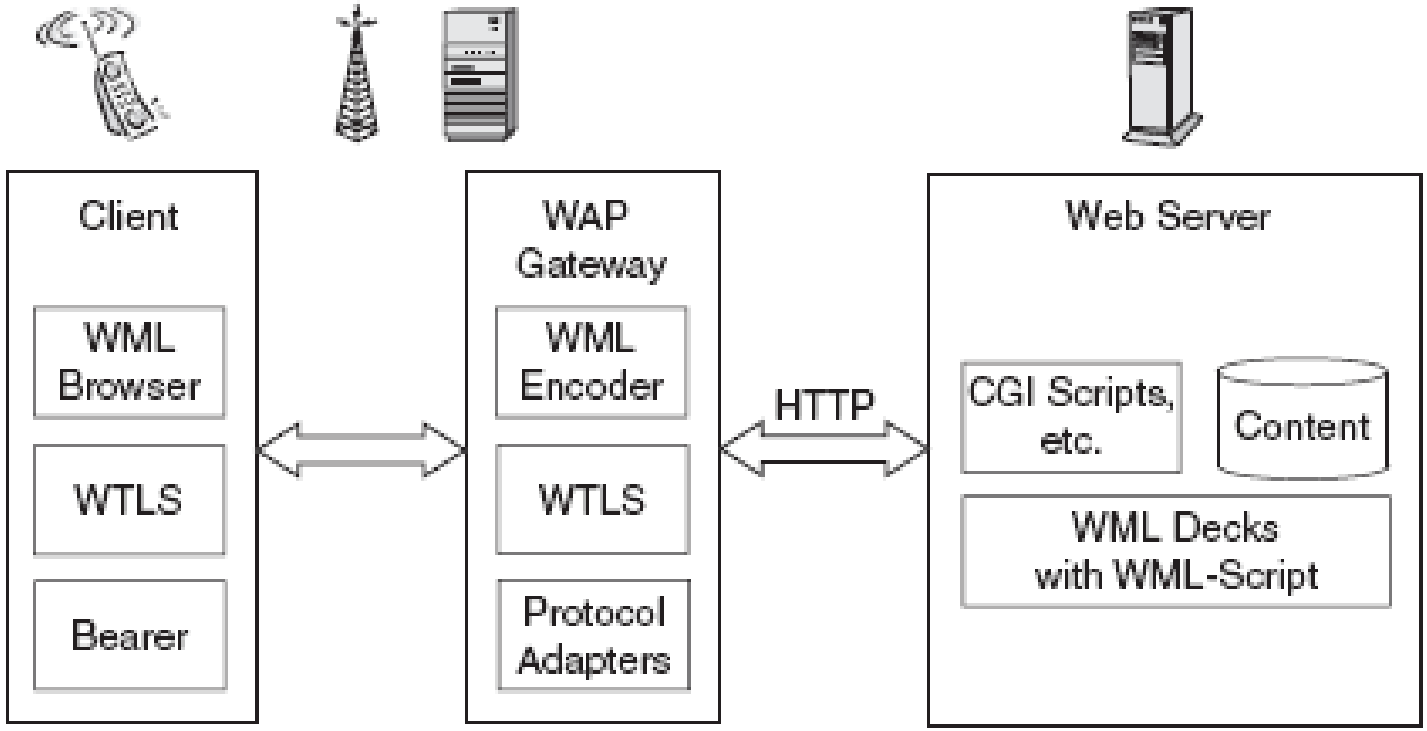
- Client-Server Model
  - Pull Model – Web/WAP Client-Server Model
    - Weather Forecast, Financial Quotations, Digital Rights Management Content 4 Download Provisioning, Video Streaming
  - Push Model – SMS, MMS, Push Messages (SL, SI, CO)
    - Advertising, Digital Rights Management, and asynchronous delivery (Weather Forecast, Financial News – but different business model etc.)
- Cross-Platforms Mobile App – JME / HTML5 / CSS 3 / JS
- SIM Toolkit Apps
- Mobile OS Platforms Native App
- Hybrid Apps





Servlet & Nokia WML Obsolete approach

- Client-Server
- WTLS Nokia Security

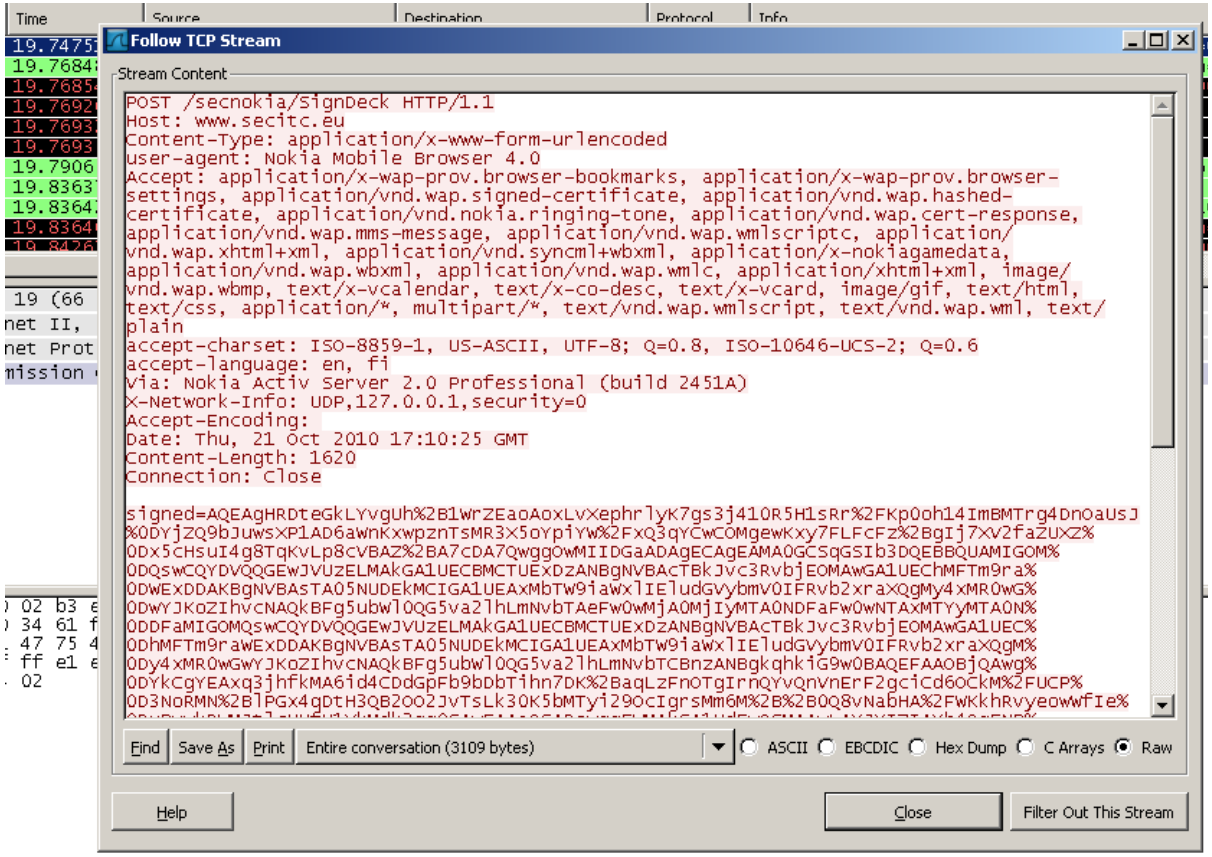


Nokia WAP Development – Deprecated / Obsolete

- Installation and Implementation of:
  - JDK 1.5.0
  - Java Server Class
  - Tomcat Apache
  - Java Simple Servlet Development
  - Nokia Internet Mobile Toolkit
    - NMIT 4.1
    - Service Pack 4 JDK 1.5.0
    - Install Nokia WAP Gateway
    - Install Nokia Browser 4.0
- We were communicating about:
  - Servlets
  - XHTML, WML and WMLS
  - WTLS and & Signing Procedure

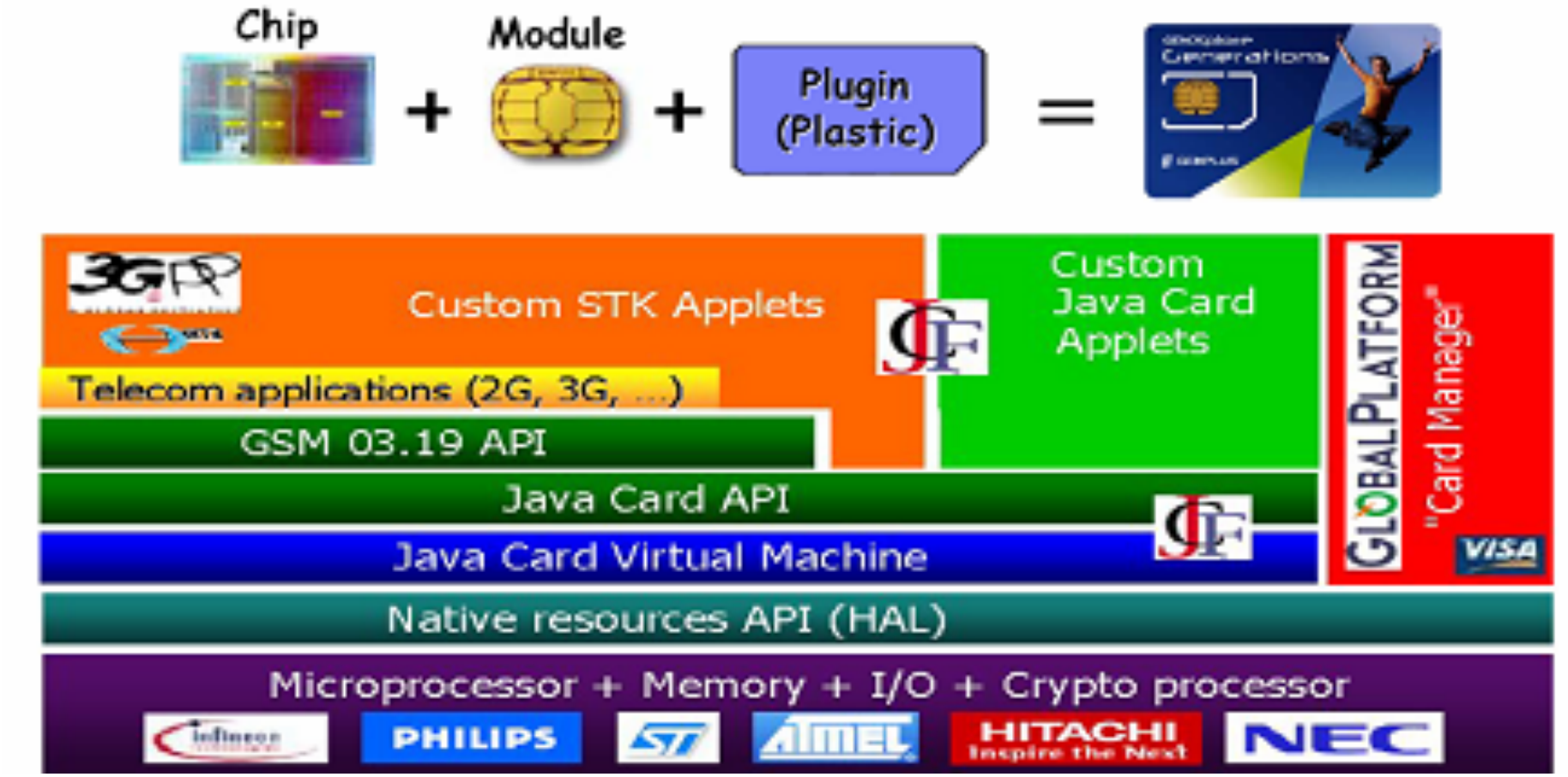
Servlet&Nokia Samples Obsolete

- Client-Server
- WTLS Nokia Security



1.2.1 (U)SIM Overview

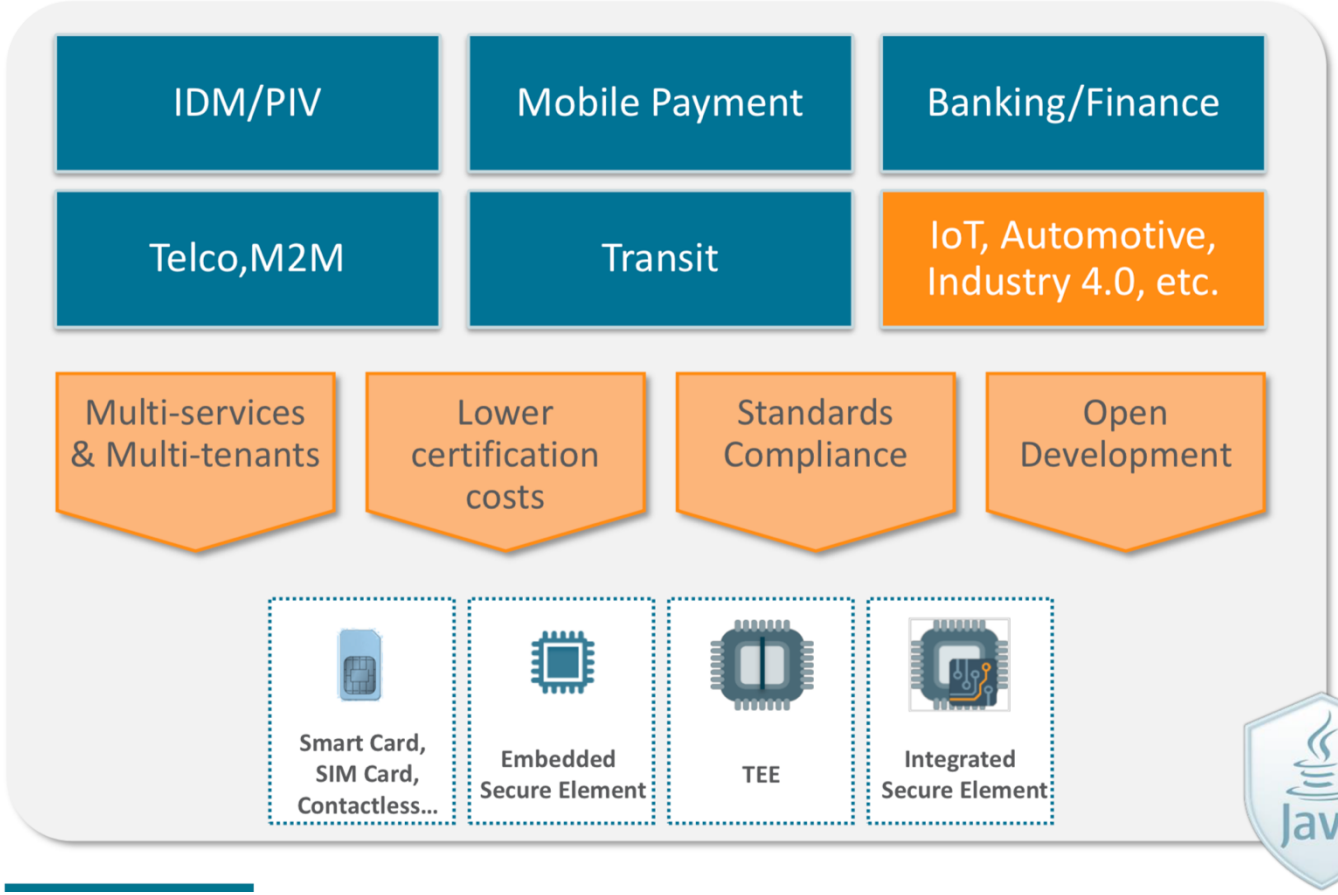
(U)SIM Components



1.2.1 (U)SIM Overview

(U)SIM JavaCard

Enabling Security with the Java Card Platform



In Vertical Markets

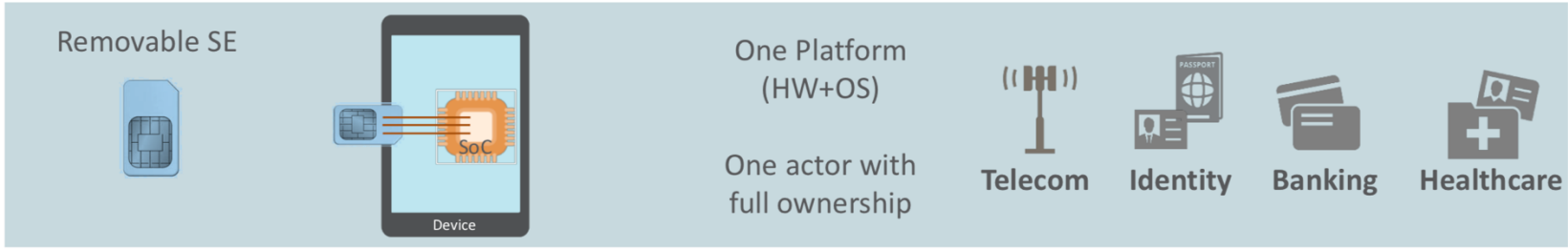
Via Secure Open Application Platform

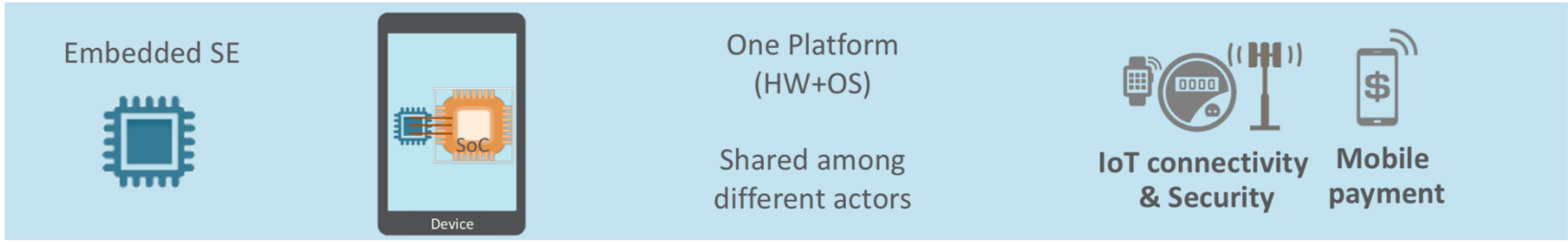
With Choice of Security Form Factors

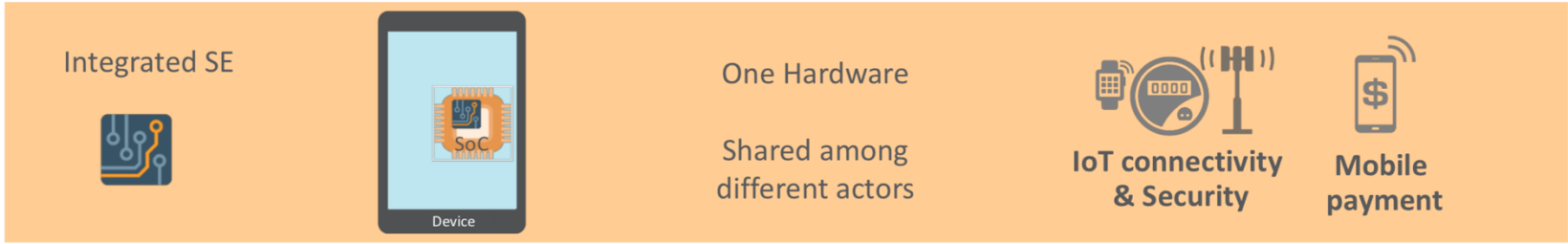
1.2.1 (U)SIM Overview

(U)SIM Secure Element

Secure Element: Form Factor evolutions



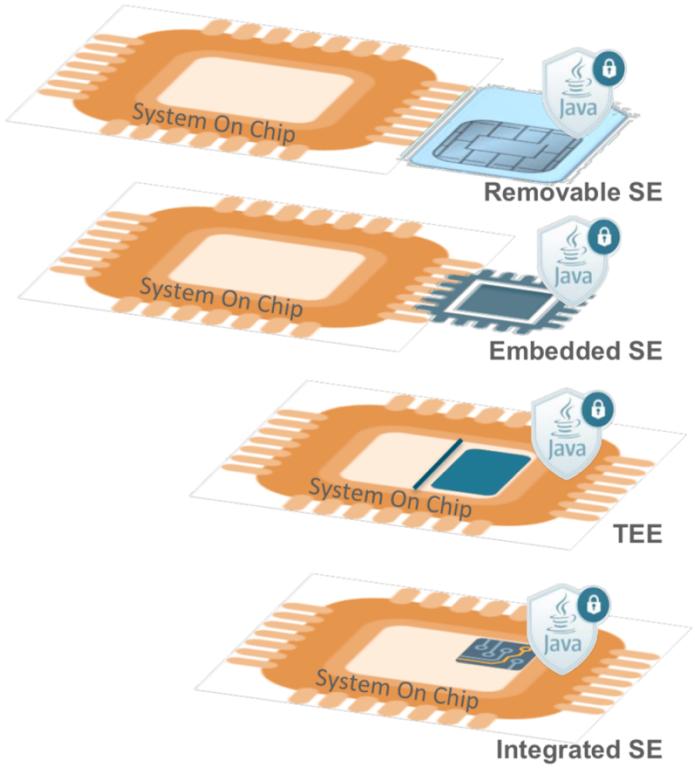




1.2.1 (U)SIM Overview

(U)SIM JavaCard

Java Card as Unified Security Framework



Java Card

- **Scalable Architecture for embedded security**
  - Certifiable platform can be applied across secure segments
  - Small footprint enables any form factor
- **Standards based implementation and certification**
  - Public specifications (Oracle, GP, ETSI) and verifiable compatibility
  - Certified protection profile as standard input for product security targets
- **Proven, extensible and Manageable platform**
  - Wider range of available solutions, tools and expertise
  - Ability to deploy and manage applications from different providers in the value chain (chip maker, OEM, MNO, SSP, user)
- **Content portability across hardware form Factors**
  - Service development / deployment is abstracted from the target HW
  - Easy Migration path for existing applications : eUICC and Payment
  - Hardware choice becomes a factor of commercial and security requirements

1.2.1 (U)SIM Overview

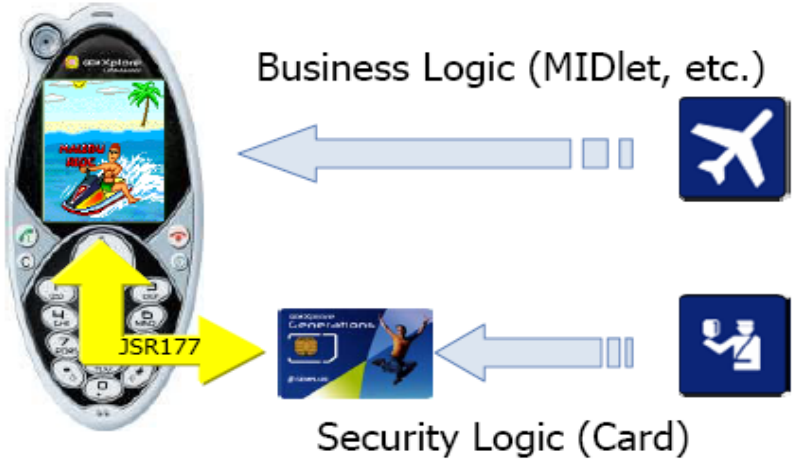
(U)SIM Components

Edge Security technologies for Mobile & IoT - Comparative

	Certification Level	Applications Deployment	cost	Flash Memory Processing	Platform Resources Access	Typical/Potential Use Cases
SW	+	Pre & Post issuance	+	SoC	Everything	Application isolation
TEE	++	Pre & Post issuance Depends on framework	+	~256Kb-2+Mb ~128Kb - 2+Mb ~200Mhz 1+Ghz	Everything	fingerprint authentication / IoT security/ …
TPM	+++	One unique pre-issued application	++	~64Kb ~4Kb ~20Mhz	Confined to Microcontroller	Device attestation / IoT security
SE	++++	Pre & Post issuance Depends on framework	+++	~64-512Kb ~3-12Kb ~20Mhz	Confined to Microcontroller	SIM / Payment / ID / Transportation
iSE	+++	Pre & Post issuance Depends on framework	+	~64-512+Kb ~3-12+Kb ~200Mhz	Dedicated but the Flash shared with SoC. Could evolve.	SIM / Payment / ID / Transportation / IoT security

1.2.1 (U)SIM Overview

(U)SIM Applications



(U)SIM Features

- User Identity
- Secure Storage
- Remote Management & Personalization
- Cryptographic features

Applications

- Content Protection, DRM (aka *SIM Sentry*)
- Corporate Services (VPN, User identification)
- Secured transactions (Digital Signature, mobile banking)
- Store user settings
- Operator Services (Enhance User Interface)
- Link a web application to the card (Proxy, OTA high speed)

1.2.1 (U)SIM Overview

(U)SIM JSR177 – JME MIDlet-SIM Java Card App

MIDlet MIDlet MIDlet

JSR177 API

APDU / SAT JCRMI PKI CRYPTO

Access Control Policy (Filter)

J2ME VM

Native and low level Smart Card driver

Crypto software

Up to any Logical Channels

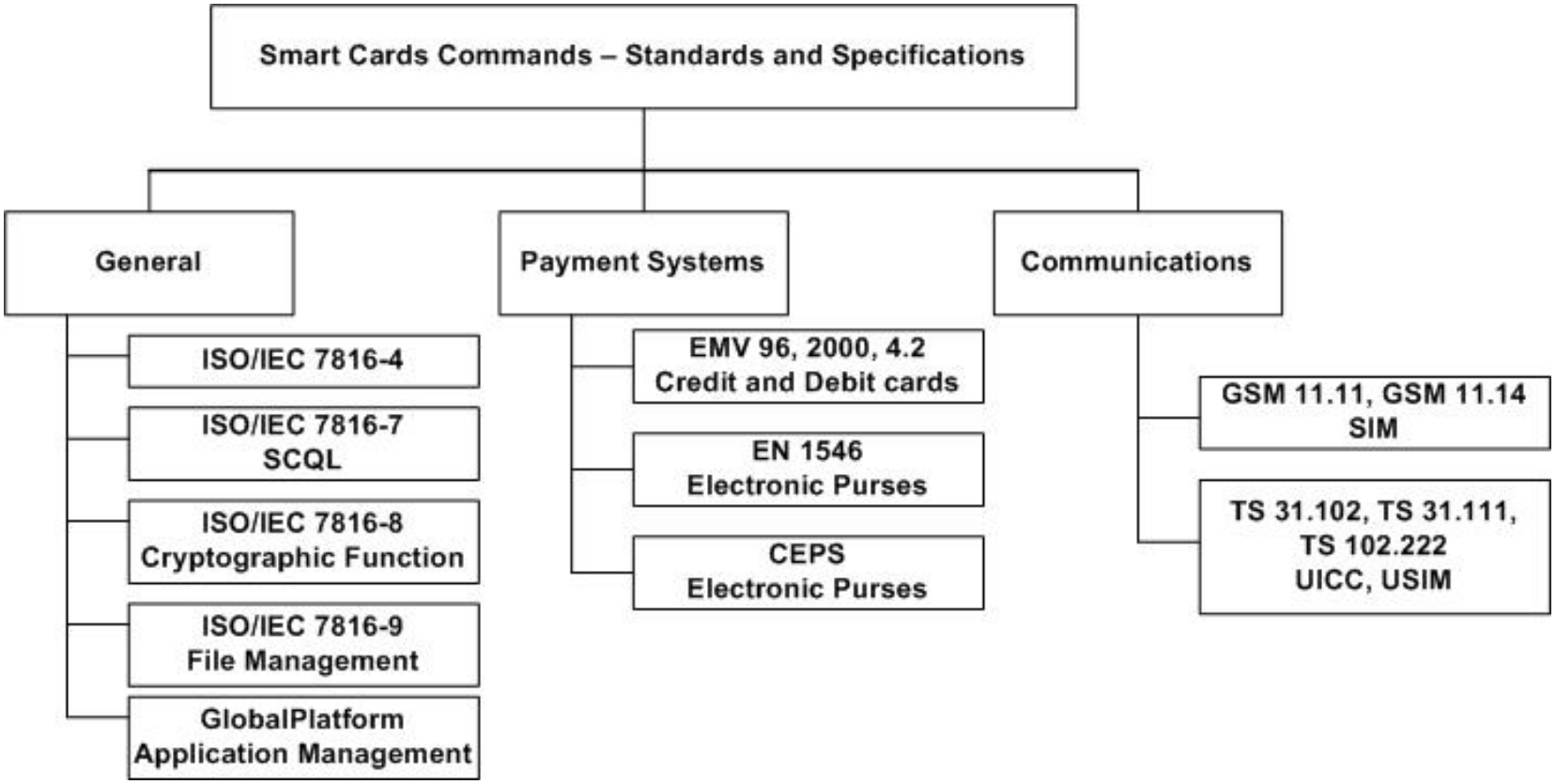
APDU & JCRMI : 1, 2 or 3 USAT : 0 only

Card App. Card App. Card App.

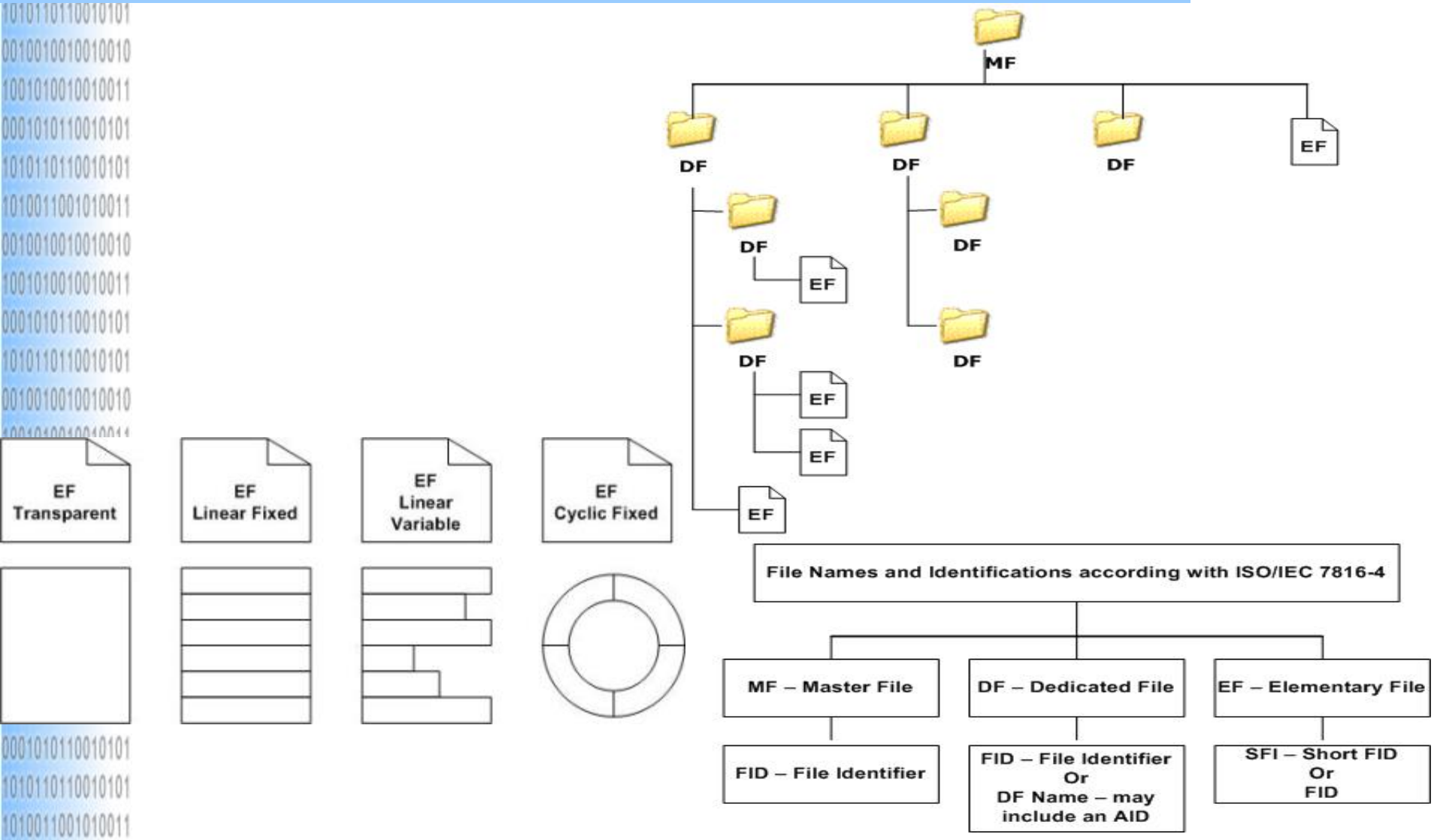
GEMPLUS

code camp

SIM Technologies – Standards



SIM Technologies – Smartcards Card File System



SIM Technologies – Remember APDUs

APDU Command						
CLA	INS	P1	P2	Lc	Data Field	Le
Header-mandatory				Body-optional		

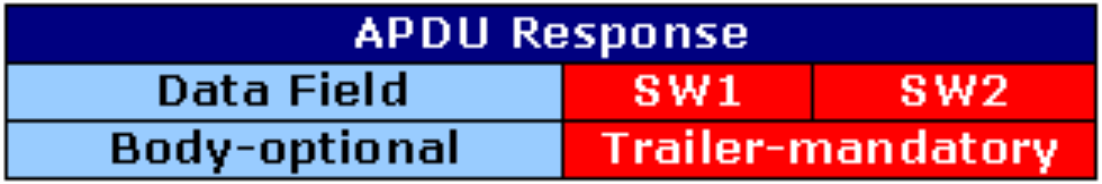
- **CLA** – is one byte – 2 hexadecimal digits, and has different predefined values conform standard ISO/IEC 7816 for selecting the class aplication.
- **INS** – is one byte, and the standard defines a specific instruction in the field CLA;

- **P1** – this represents the first parameter for a instruction and has one byte;
- **P2** – this is the second parameter for an instruction and has one byte. Is used for the same scope like P1;

- **Lc** – has one byte, is optional and represents the bytes length for the field Data Field;
- **Data Field** – is not fixed and has a bytes' length equal with the value from the field's value Lc. In this field are stored data and parameters which are send from host application to applet;

- **Le** – stores the maxim number of bytes that should have Data Field from APDU Response (the number of bytes from response could be any value from the range 0 and the value from this field).

SIM Technologies – Remember APDUs



- **Data Field** – has variable length which is determined by the value of the byte field Le from the APDU Command;

- **SW1** – has one byte and represent the status word 1;
- **SW2** – has one byte and represent the status word 2.

The fields SW1 and SW2 are parsed and interpreted together, but a communication process is called *complete* if there were no problems (SW1=0x61 & SW2=0x90 OR SW1=0x90 & SW2=0x00

SIM Technologies – File Access – Orange SIM DEMO

The commands specified for the SIM in GSM 11.11

Command	Brief description
Security commands	
CHANGE CHV	Change the PIN
DISABLE CHV	Disable PIN queries
ENABLE CHV	Enable PIN queries
RUN GSM ALGORITHM	Execute the GSM-specific cryptographic algorithm
UNBLOCK CHV	Reset the PIN retry counter from its terminal count
VERIFY CHV	Verify the PIN

SIM Technologies – File Access – Orange SIM DEMO

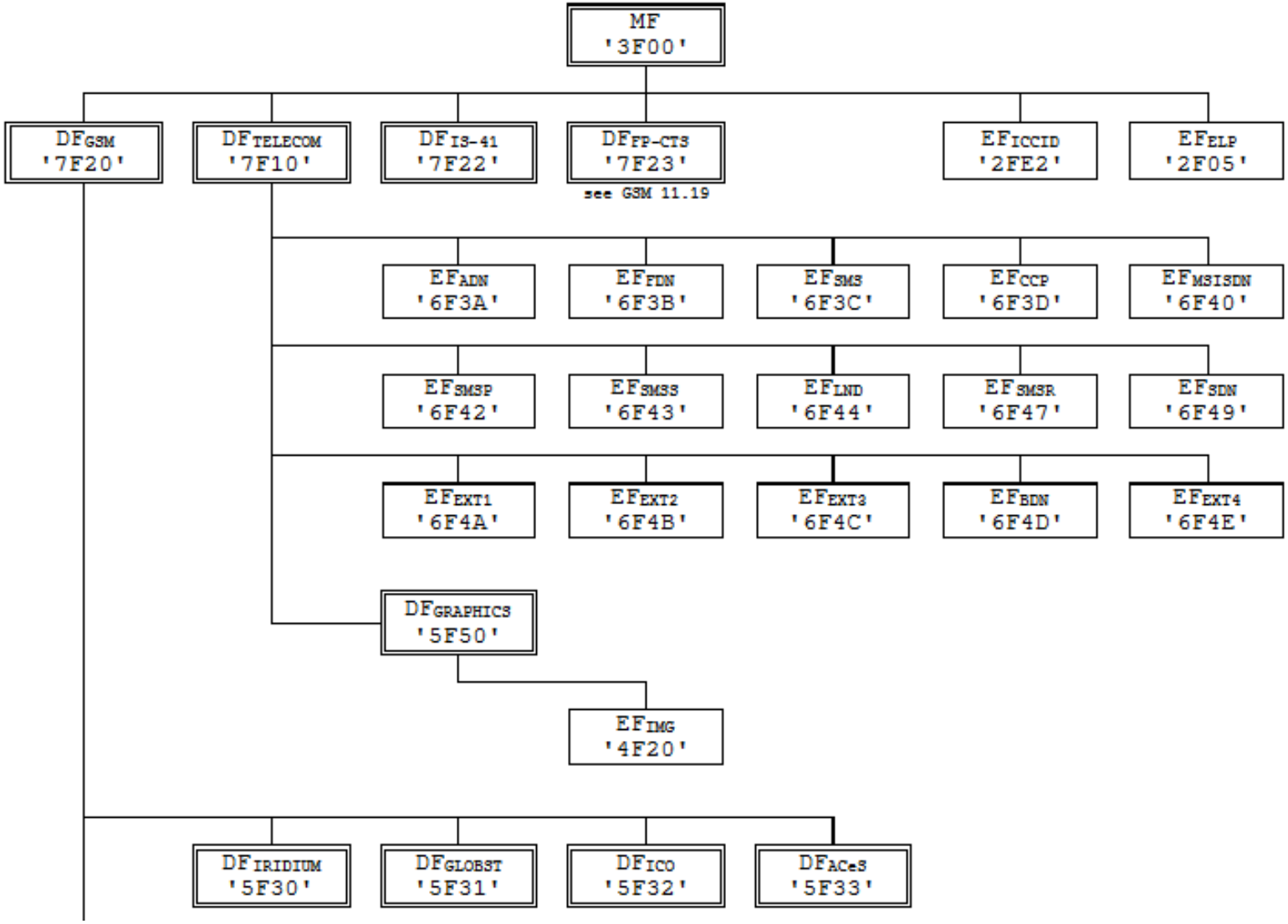
The commands specified for the SIM in GSM 11.11

Commands for operations on files	
INCREASE	Increase the value of a counter in a file
INVALIDATE	Reversibly block a file
READ BINARY	Read from a file with a transparent structure
READ RECORD	Read from a file with a record-oriented structure
REHABILITATE	Unblock a file
SEEK	Seek a text string in a file with a record-oriented structure
SELECT	Select a file
STATUS	Read various data from the currently selected file
UPDATE BINARY	Write to a file with a transparent structure
UPDATE RECORD	Write to a file with a record-oriented structure

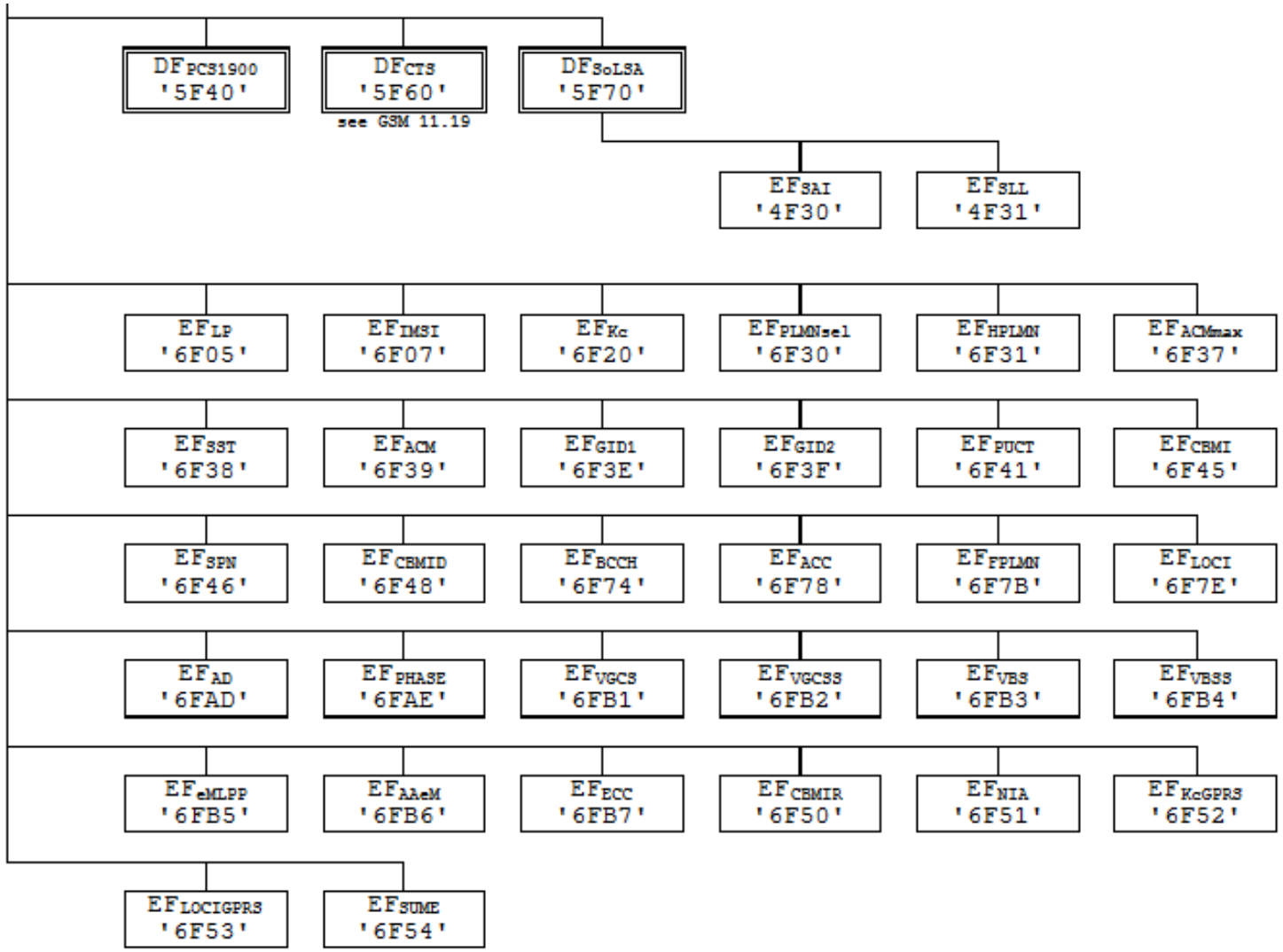
SIM Technologies – File Access – Vodafone/Orange SIM DEMO

SIM Application Toolkit commands	
ENVELOPE	Pass data to a value-added service of the SIM forming part of the SIM Application Toolkit
FETCH	Retrieve a SIM Application Toolkit command from the SIM in the mobile equipment
TERMINAL PROFILE	List all functions of the mobile equipment with respect to the SIM Application Toolkit
TERMINAL RESPONSE	Convey the response of the mobile equipment to a previous SIM Application Toolkit command of the SIM
Miscellaneous commands	
GET RESPONSE	Command specific to T = 0 for requesting data from the smart card
SLEEP	Obsolete command for putting the smart card into a low-power state

SIM Technologies – File Access – Orange SIM DEMO



SIM Technologies – File Access – Orange SIM DEMO



SIM Technologies – File Access – Orange SIM DEMO

```
//JCOP APDUs on a Orange/Vodafone (U)SIM
// https://www.primianotucci.com/os/smartcard-explorer
/terminal "winscard:4|Gemplus USB Smart Card Reader 0"

//Answer 2 Reset
/atr

//First auth for IMSI with PIN 1234
/send A02000010831323334FFFFFFFF
=> A0 20 00 01 08 31 32 33 34 FF FF FF FF          . ...1234....
 (65986 usec)
 <= 90 00                                          ..
Status: No Error

//select DF GSM - select EF 7F20 file
/send A0A40000027F20
 => A0 A4 00 00 02 7F 20                           ......
 (51210 usec)
 <= 9F 1A
```

SIM Technologies – File Access – Orange SIM DEMO

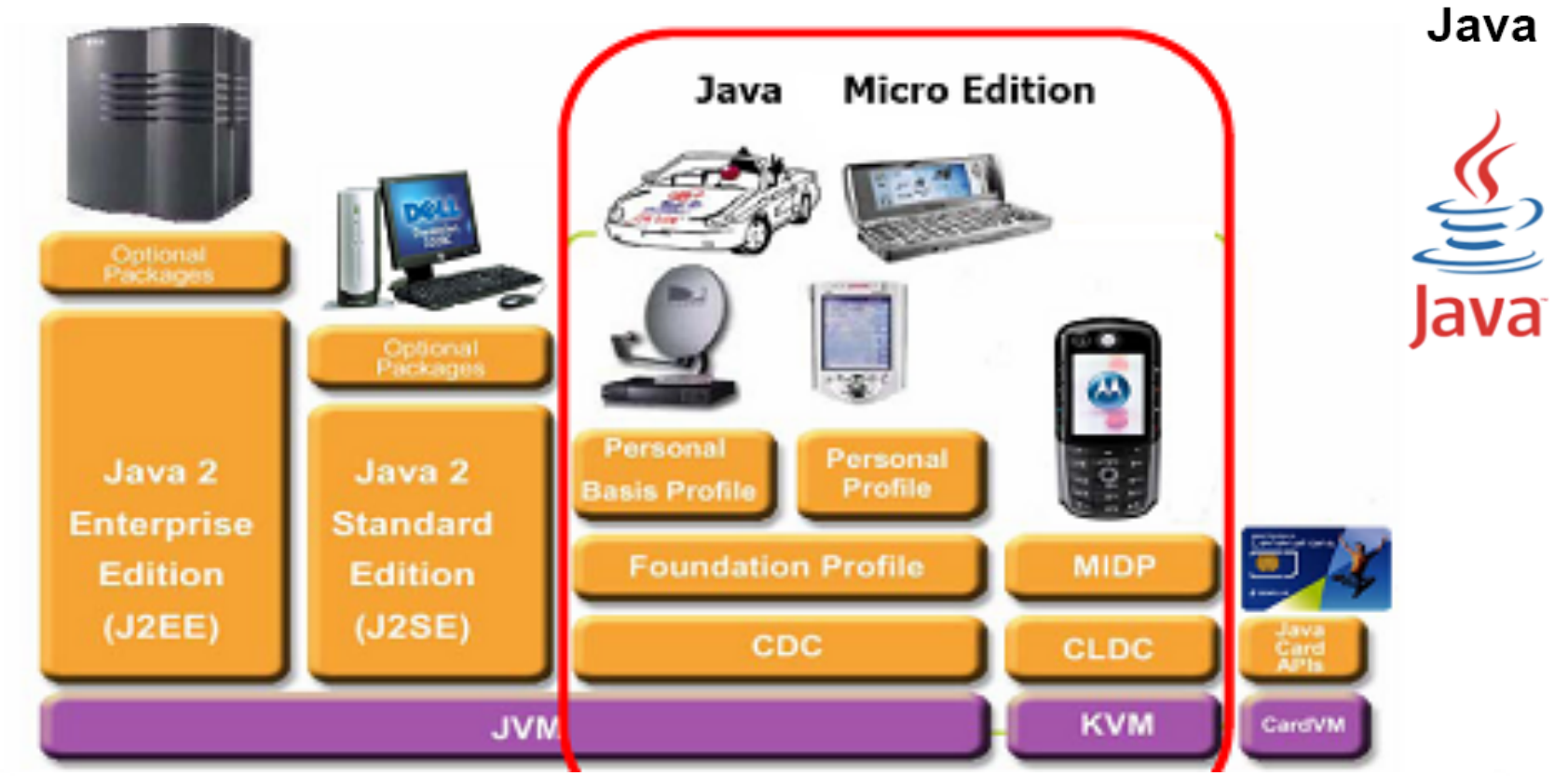
```
//JCOP APDUs on a Orange/Vodafone (U)SIM

//GET RESPONSE  for EF 7F20
/send A0C000001A
 => A0 C0 00 00 1A                                  .....
 (13067 usec)
 <= 00 00 00 00 7F 20 02 00 00 00 00 00 0D 13 00 19    ..... .....
    04 00 83 8A 83 8A 00 03 00 00 90 00                ............
Status: No Error

//select EF IMSI - 6F07
/send A0A40000026F07
=> A0 A4 00 00 02 6F 07                               .....o.
 (29702 usec)
 <= 9F 0F                                          ..
Status: 0x9F0F
```

SIM Technologies – File Access – Orange SIM DEMO

```
//JCOP APDUs on a Orange/Vodafone (U)SIM

//GET RESPONSE for EF 6F07 file
/send A0C000000F
 => A0 C0 00 00 0F                                    .....
 (10837 usec)
<= 00 00 00 09 6F 07 04 00 14 00 14 01 01 00 00 90    ....o...........
    00                                                .
Status: No Error

//READ BINARY from EF 6F07 - IMSI
//IMSI = 296210042558267 7
/send A0B0000009
 => A0 B0 00 00 09                                    .....
 (21796 usec)
 <= 08 29 62 10 04 25 58 26 77 90 00                  .)b..%X&w..
Status: No Error

//08 is IMSI length, the IMSI value: 296210042558267 7
```

1.2.2 Cross-platforms App Overview

JME - Java Micro-Edition



1.2.2 Cross-platforms App Overview

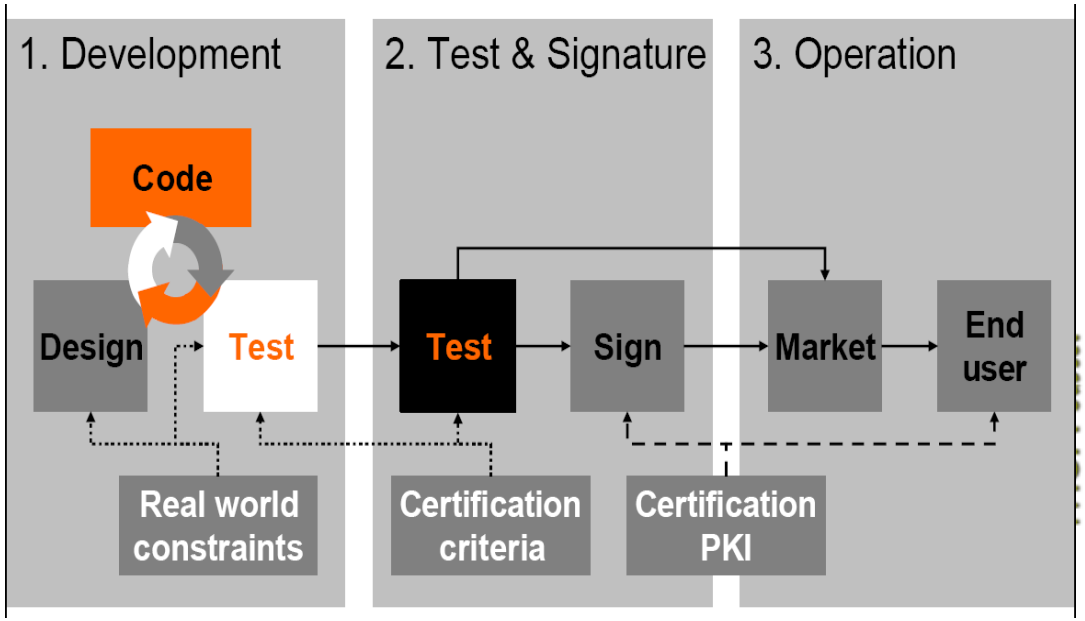
JME - Java Micro-Edition

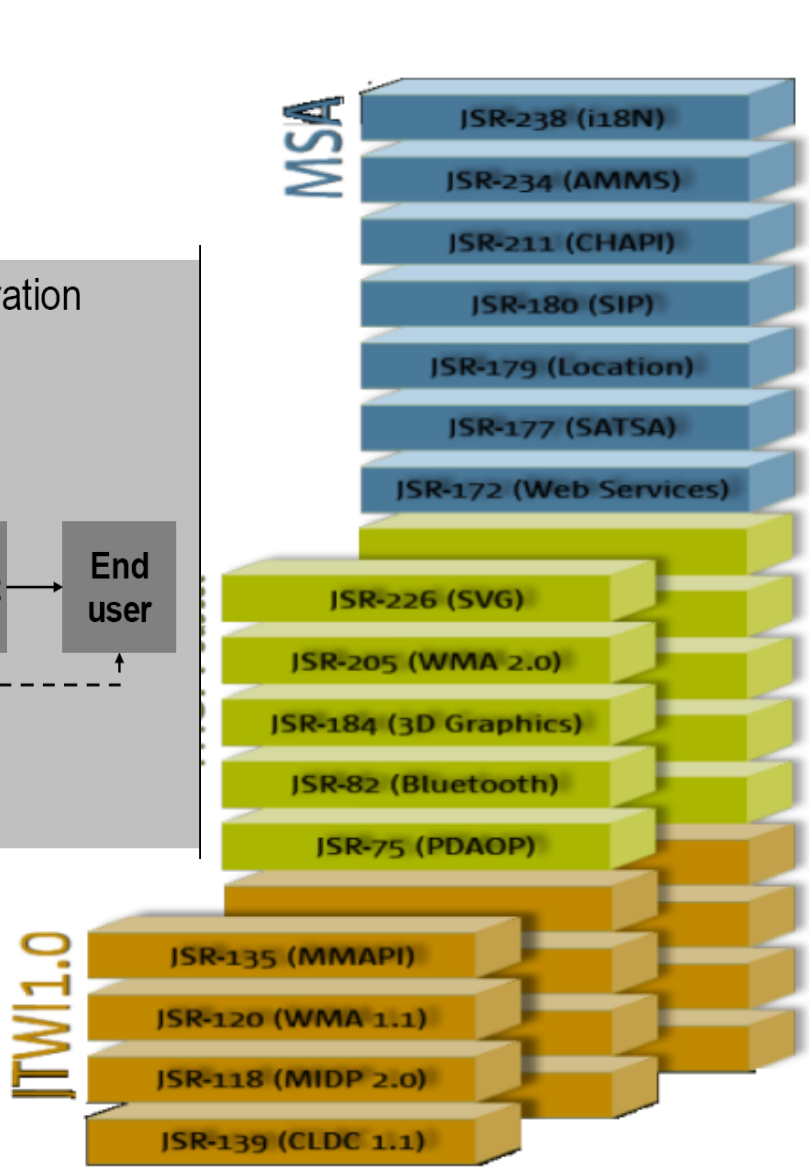
- CLDC (JSR 139)
- MIDP (JSR 118)
- Packages of JTWI (JSR 185)
- PDA (JSR 75)
  - File API
  - PIM API
- Bluetooth & OBEX (JSR 82)
- Mobile 3D graphics (JSR 184)
- SVG (JSR 226)
- Mobile I18N (JSR 238)
- Content Handler (JSR 211)
- Security & Trust (JSR 177)
  - Crypto
  - APDU
  - PKI
- Web Services (JSR 172)
  - XML Parsing
  - JAX RPC
- Location (JSR 179)
- SIP (JSR 180)
- Payment (JSR 229)
- Advanced Multimedia Supplements (JSR 234)

Java

1.2.2 Cross-platforms App Overview

JME - Java Micro-Edition





1.2.2 Cross-platforms App Overview

JME - Java Micro-Edition

- Sun's Wireless Toolkit (WTK)
  - A PC emulator and a neat suite of tools & libraries
- IDE
  - Eclipse with ME plugin
  - Sun Net Beans Mobility Pack
  - Borland JBuilder Mobile Studio
  - I use Emacs and a Makefile
- Proprietary tools & IDE plugins
  - Very useful
  - Rarely faithful (for emulation)

> Developments tools for Java ME

- CLDC Wireless Toolkit 2.5
  - MSA
- NetBeans Mobility Pack 5.5
  - SVG support
- Next release of NetBeans mobility pack
  - CDC Toolkit

- Web
  - Sun's portal on Java ME
  - Device manufacturers developers portal: tools, technical specifications and documentation

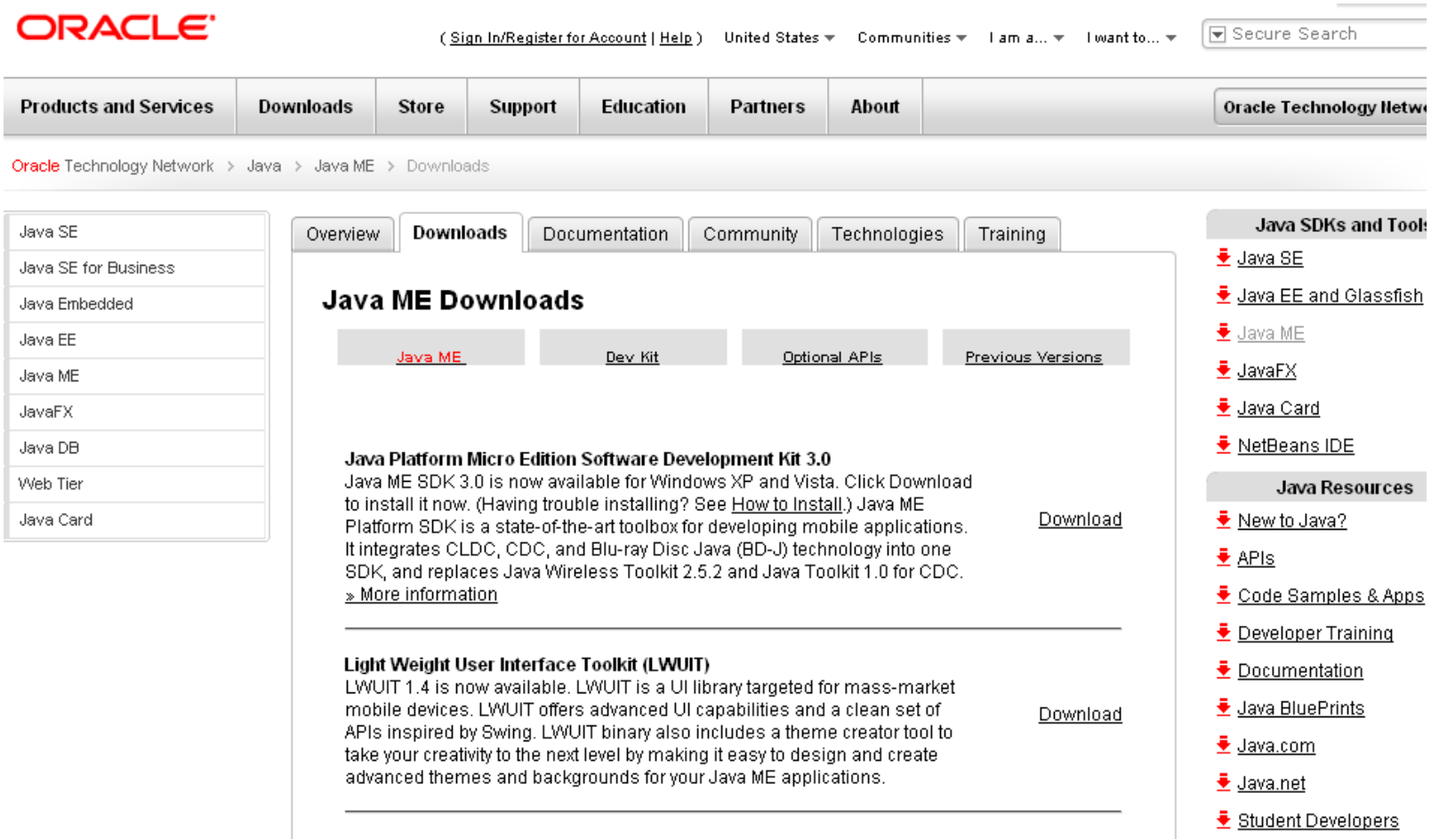
Java

ORACLE

1.2.2 Cross-platforms App Overview

JME - Java Micro-Edition

http://www.oracle.com/technetwork/java/javame/downloads/index.html



1.2.3 Mobile OS "Native" Applications Overview

Mobile OS in the market?:

1. Google Android OS / Fuchsia OS?
2. Apple iOS
3. RIM BlackBerry OS
4. Microsoft Windows Mobile
5. *Sailfish OS – MeeGo, Intel Tizen (Intel Wearable-Yocto OS) & Ubuntu Mobile – quite new (open source)*

- *…, no more new mobile devices with:*
  - *Symbian, Garnet / ACCESS ALP / Palm OS, HP WebOS*

  Because of ARM processors, there are Nokia or HTC devices that run Symbian OS or Windows Mobile + on top, JME virtual machine

  The trend is for mobile rich clients app with HTML5 + CSS3 + JS

1.2.3 Mobile OS "Native" Applications Overview

New Mobile OS	Kernel, Supported processors platforms, Source Models, License	Tools + Developers Web Page
Google Android 	Kernel: Monolithic (modified Linux kernel) OS Family: Unix-like Processors Family: ARM, MIPS (RISC Design) and x86 (CISC Design) Source Model: Open Source Package manager: Google Play, APK License: Apache, GPL v2	Programming Languages: ▪ Java/Kotlin or C/C++ Tools & SDK: ▪ Android SDK – Java/Kotlin: ○ ADT – Android Developer Tools Bundle or Eclipse Java IDE + ADT Plug-in ○ Android Studio 3+ ▪ Android NDK – C/C++ ▪ Google Flutter – Dart ▪ Apache Ionic / Cordova Web: http://developer.android.com
Apple iOS 	Kernel: Hybrid (XNU) OS Family: X is NOT Unix Processors Family: Apple A6 - ARM (RISC Design) Source Model: Closed-Source	Programming Languages: ▪ Objective-C / Swift Tools & SDK: ▪ iOS software development kit (SDK) & Xcode (IDE) Web: https://developer.apple.com/devcenter/ios/i

1.2.3 Mobile OS "Native" Applications Overview

New Mobile OS	Kernel, Supported processors platforms, Source Models, License	Tools + Developers Web Page
BlackBerry 10 – QNX  	Kernel: Real-time microkernel (RTOS – Real-Time Operating System) OS Family: QNX - (Unix-like) Processors Family: ARM (RISC Design) Source Model: Closed-Source Package manager: - License: Proprietary BlackBerry got QNX	Programming Languages: ▪ Java or C/C++ or HTML/CSS/JavaScript or ActionScript Tools & SDK: ▪ Blackberry Native SDK: ○ C/ C++ App Framework, C++/Qt and QML - Cascades SDK ▪ Webworks SDK: ○ HTML5/Javascript/CSS, ▪ Adobe AIR: ○ ActionScript ▪ Java Android Runtime: ○ Repackaging Tool / Eclipse IDE Plug-in ▪ BlackBerry Java – "Always" alternative for BB: ○ BlackBerry Plug-in for Eclipse Java IDE Web: http://developer.blackberry.com/

1.2.3 Mobile OS "Native" Applications Overview

New Mobile OS	Kernel, Supported processors platforms, Source Models, License	Tools + Developers Web Page
Windows / RT 	Kernel: Hybrid (Microsoft NT Kernel) OS Family: Microsoft Windows Processors Family: ARM (RISC Design) and x86 (CISC Design) Source Model: Closed-Source Package manager: - License: Proprietary	Programming Languages: ▪ C#, VB .NET Tools & SDK: • MC VS 2017+ - Xamarin ▪ Windows Phone SDK 8: ○ Microsoft Visual Studio .NET 2012 / 2013 Web: http://developer.windowsphone.com
Tizen OS 	Kernel: Monolithic (Linux) OS Family: Linux Processors Family: ARM (RISC Design) and x86 (CISC Design) Source Model: Mixed - Open Source and Proprietary Package manager: RPM (Red hat Package Manager) License: GPL – GNU Public License v2, LGPL, Apache, BSD, Proprietary Intel leaves MeeGo project and joins the development of SLP – Samsung Linux Platform	Programming Languages: ▪ HTML5/CSS/JavaScript or C/C++ Tools & SDK: ▪ Tizen Web App Programming – HTML: ○ Tizen Integrated Development Environment (IDE) is based on the JSDT (JavaScript Development Tools) - Web IDE ▪ Tizen Native App Programming – C/C++: ○ Tizen Integrated Development Environment (IDE) is based on the JSDT (JavaScript Development Tools) - Web IDE and Eclipse CDT (C/C++ Development Tools) Web: https://developer.tizen.org/

1.2.3 Mobile OS "Native" Applications Overview

New Mobile OS	Kernel, Supported processors platforms, Source Models, License	Tools + Developers Web Page
Ubuntu Touch 	Kernel: Monolithic (Linux) OS family: Unix-like Processors Family: ARM (RISC Design) and x86 (CISC Design) Source Model: Open Source Package manager: dpkg (Debian) License: Mainly GPL	Programming Languages: ▪ QML (JavaScript based) or HTML5/CSS/JavaScript Tools & SDK: ▪ Qt Creator for C/C++ and QML Web: http://developer.ubuntu.com/ http://developer.ubuntu.com/apps/sdk/
Firefox OS 	Kernel: Monolithic (Linux) OS family: Unix-like Processors Family: ARM (RISC Design) Source Model: Open Source Package manager: ZIP Manager License: Mainly GPL and Apache Mobile OS developed by Mozilla's Boot to Gecko (B2G) project.	Programming Languages: ▪ HTML5/CSS/JavaScript Tools & SDK: ▪ Mozilla Debugger Web: https://developer.mozilla.org/en/docs/Mozilla/Firefox_OS

1.2.3 Mobile OS "Native" Applications Overview

New Mobile OS	Kernel, Supported processors platforms, Source Models, License	Tools + Developers Web Page
Sailfish OS – by Jolla Sailfish *Nokia former team from MeeGo OS development | based on Mer & Nemo Mobile runs on Jolla Mobile device and Nokia N9 | runs also on Samsung i9300, Google Nexus S*	**Kernel: Monolithic (Linux)** **OS Family: UNIX-like (Linux)** **Processors Family: ARM (RISC Design) - Qualcomm Snapdragon 400** **Source Model: Open-Source for OS** **Package manager: RPM (Red Hat)** **License: Open-Source**	**Programming Languages:** ▪ **Qt 5 C/C++ and QML** **Tools & SDK:** ▪ **Qt Sailfish OS IDE on Linux (Ubuntu)** ▪ **C++ Qt & Silica Library:** ○ **C/ C++ Qt 5 and QML @ QtQuick2** ▪ **Mer Build Engine:** ▪ **For cross-platform compiling** ▪ **Java Android Runtime:** ○ **Repackaging Tool** **Web: http://sailfishos.org**

1.2.3 Mobile OS "Native" Applications Overview

Mobile Convergence for M-App Development

HTML 5/CSS 3/JavaScript – Apache Cordova / Oracle Mobile JET | MS Xamarin (C#)

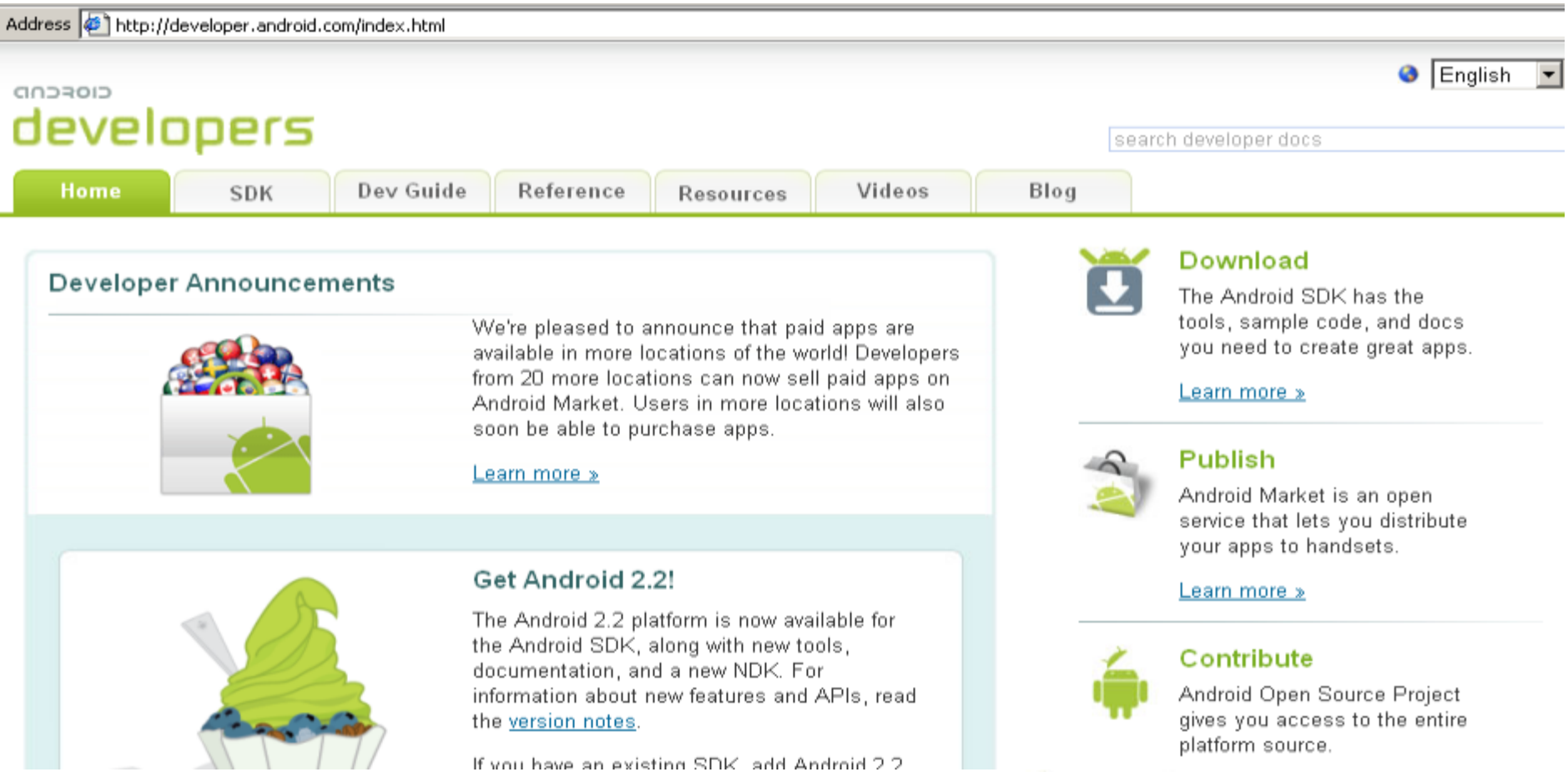
WebKIT Engine / Similar Engine

Mobile IoT: Android/Java, iOS, C-Posix, JavaScript

ANDROID		BlackBerry	Windows 8/10 IoT Core	Intel Tizen	SailfishOS & Ubuntu Touch	Firefox OS
Java/ Kotlin	C/C++	Objective-C / Swift	Java & Native C/C++	Web: HTML 5	C#.NET	C/C++	Web: HTML 5	Native: C++ Qt	Web: QML or HTML 5	Web: HML 5

1.2.3 Mobile OS "Native" Applications Overview

1. Google Android Main Developer Site



Google ANDROID OS

http://developer.android.com/

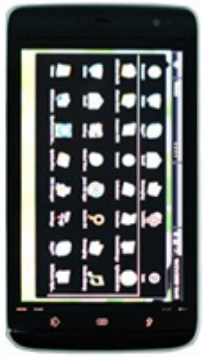
1.2.3 Mobile OS "Native" Applications Overview

1. Google Android Devices

Wildfire
Available
HTC
June, 2010

EVO
Available
HTC
June, 2010

Aero / Mini 3iX
Available
Dell
July, 2010



Streak
Available
Dell
June 5th, 2010 (UK)

Galaxy S / Fascinate
Available
Samsung
June 3rd, 2010

Optimus Q (LG C710 Aloha / LG LU2300)
Available
LG
May, 2010

Xperia X10 Mini
Available
Sony Ericsson
2010, May

Galaxy A / Apollo / M100S
Available
Samsung
2010, May (Koreea)

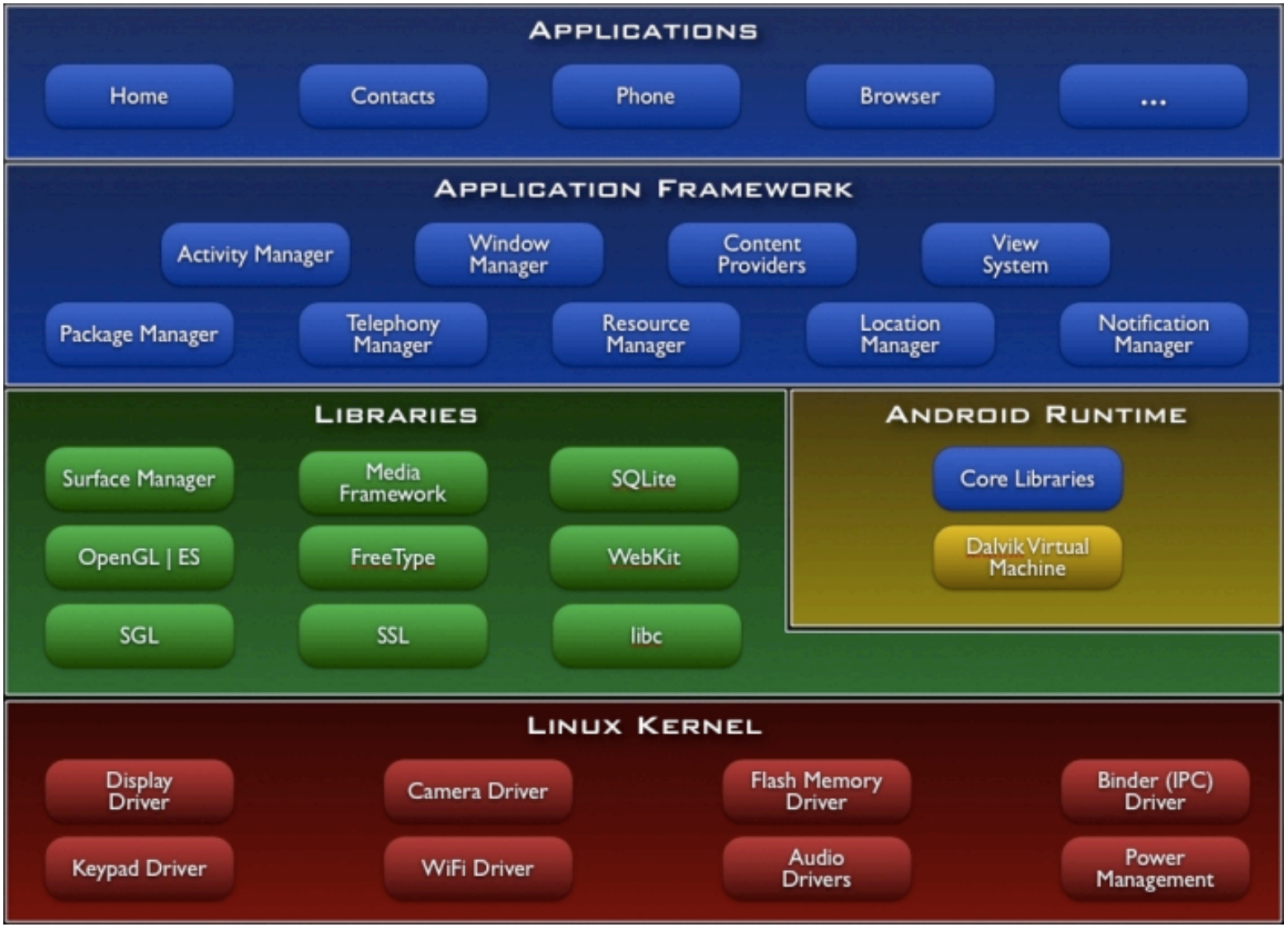
Droid Incredible
Available
HTC
April 29th, 2010

GT540 Optimus
Available
LG
1st May, 2010

Pulse Mini (T-Mobile)
Available
Huawei
April, 2010

1.2.3 Mobile OS "Native" Applications Overview

1. Google Android OS Architectures



1.2.3 Mobile OS "Native" Applications Overview

1. Google Android Features

- **Application framework** enabling reuse and replacement of components
- **Dalvik/Java virtual machine** optimized for mobile devices
- **Integrated browser** based on the open source WebKit engine
- **Optimized graphics** powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- **SQLite** for structured data storage
- **Media support** for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- **GSM Telephony** (hardware dependent)
- **Bluetooth, EDGE, 3G, and WiFi** (hardware dependent)
- **Camera, GPS, compass, and accelerometer** (hardware dependent)
- **Rich development environment** including a device emulator, tools for debugging, memory and performance profiling, and a plugin for the Eclipse IDE

- JAVA based DEVELOPMENT

1.2.3 Mobile OS "Native" Applications Overview

Google Android Demo

https://sites.google.com/site/mobilesecuritylabware/3-data-location-privacy/lab-activity/cryptography/cryptography-mobile-labs/encryption-decryption/2-lab-activity/lab-activity

http://www.tutorialspoint.com/android/

1.2.3 Mobile OS "Native" Applications Overview

2. Apple iOS Features

Models

Legend

Discontinued | Current

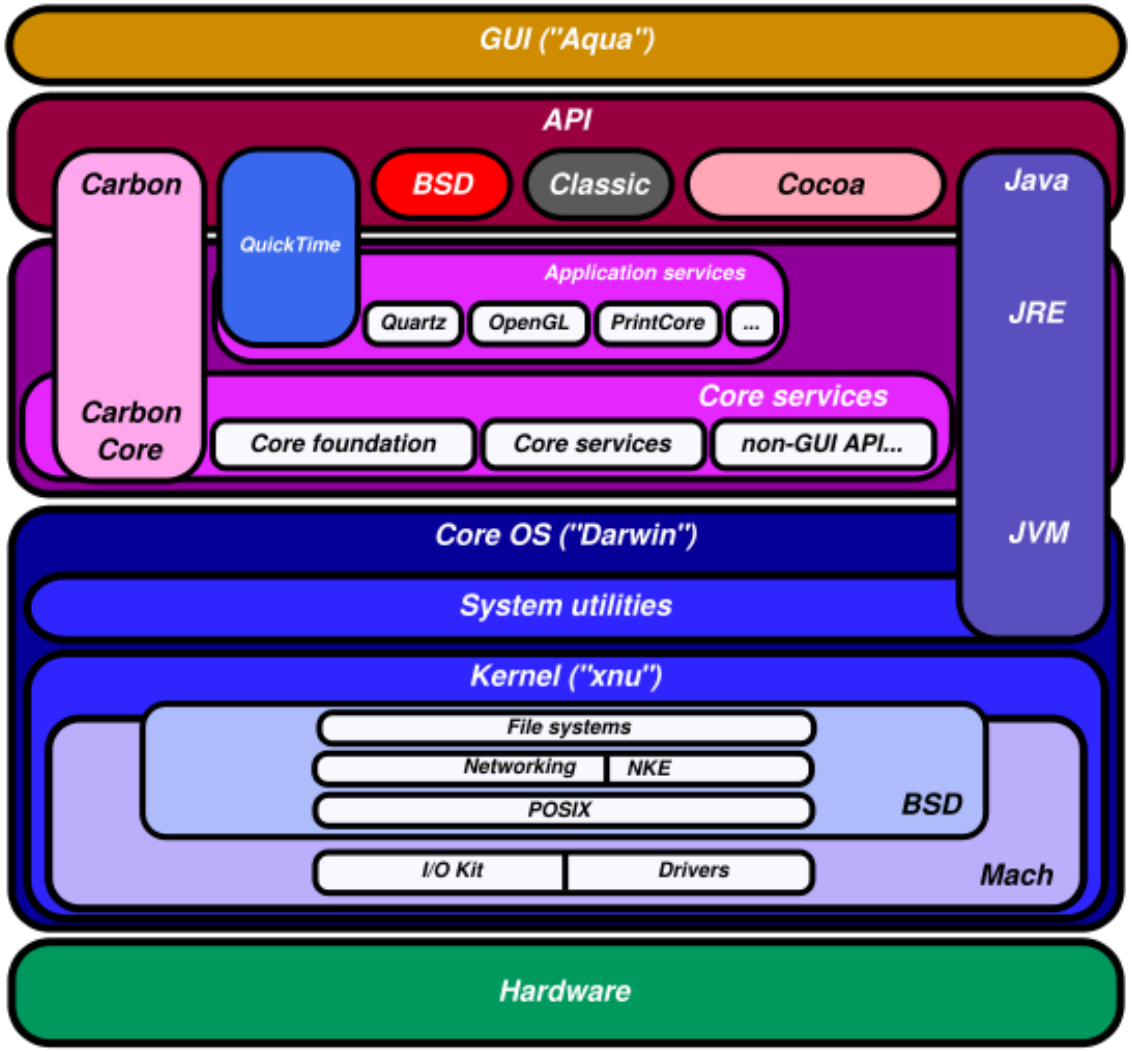
http://developer.apple.com/devcenter/ios/index.action

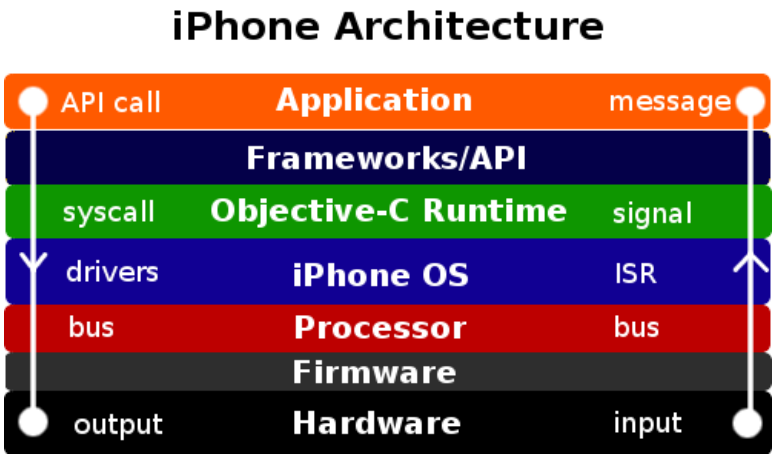
iPhone

Model	iPhone	iPhone 3G	iPhone 3GS	iPhone 4
Initial operating system	iPhone OS 1.0	iPhone OS 2.0	iPhone OS 3.0	iOS 4.0
Highest Supported operating system	iPhone OS 3.1.3	iOS 4.2 Beta 3		
Display	3.5 in (89 mm), 3:2 aspect ratio, scratch-resistant[1] glossy glass covered screen, 262,144-color LCD, 480 × 320 px (HVGA) at 163 ppi		In addition to previous, features a fingerprint-resistant oleophobic coating [2]	3.5 in (89 mm), 3:2 aspect ratio, aluminosilicate glass covered IPS LCD screen, 960 × 640 px at 326 ppi, 800:1 contrast ratio
Storage	4, 8 and 16 GB	8 and 16 GB	8, 16 and 32 GB	16 and 32 GB
Processor	620 MHz (underclocked to 412 MHz) Samsung 32-bit RISC ARM 1176JZ(F)-S v1.0[3][4]		833 MHz (underclocked to 600 MHz) ARM Cortex-A8[5][6] Samsung S5PC100[5][7]	ARM Cortex-A8 Apple A4[8]
Graphics	PowerVR MBX Lite 3D GPU[9]		PowerVR SGX535 GPU[5]	PowerVR SGX535 GPU[10]
Memory	128 MB DRAM[11]		256 MB DRAM[5][6]	512 MB DRAM[12]

1.2.3 Mobile OS "Native" Applications Overview

2. Apple iOS Features





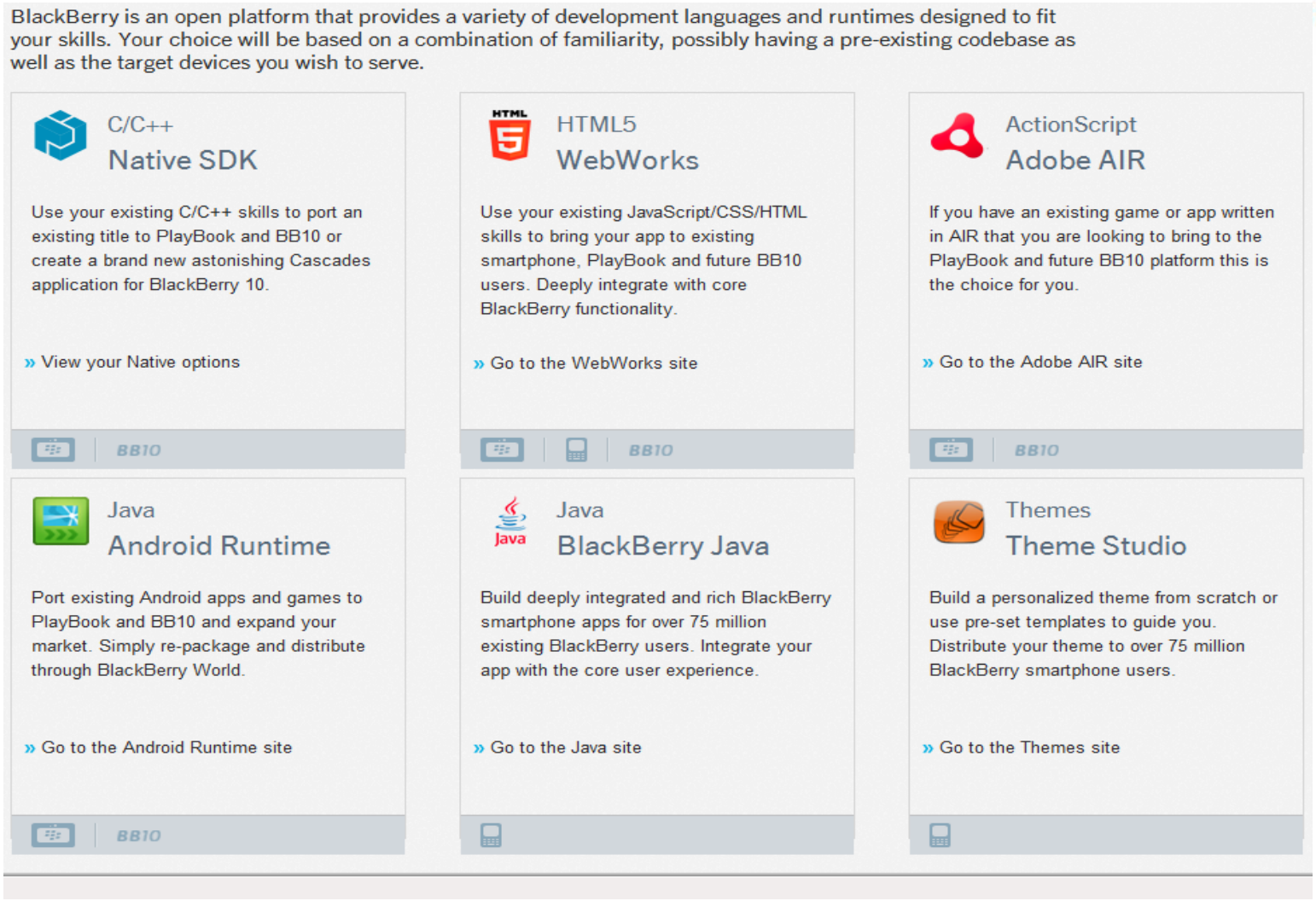
1.2.3 Mobile OS "Native" Applications Overview

3. RIM BlackBerry OS Features

http://developer.blackberry.com

1.2.3 Mobile OS "Native" Applications Overview

3. RIM BlackBerry OS Developing Platform Choice



http://developer.blackberry.com/develop/platform_choice/index.html

Blackberry Mobile OS Technologies

- Blackberry OS or BB10 – QNX are proprietary
- No significant information publicly
- Platform Development choice: C/C++ Native NDK – QNX (BB10), HTML 5 WebWorks (BB10), ActionScript Adobe AIR (BB10), Java Android Runtime (BB10), Java-ME Blackberry (All BB devices)
- For Java-ME, hardware access through RIM developed JVM via
  - Standard Java-ME applications
  - MDS applications
- RIM's attention to building cross-platform functionality
  - Android OS, Windows Mobile, Desktop Connect
- For Java, suggests nothing specific about the underlying OS
  - Focus mainly on the RIM API that wraps MIDP

Blackberry Mobile OS Technologies

JAVA ME STANDARD APPLICATIONS

- JavaME applications have access to several APIs
- Full support for CLDC API
  - java.io, java.lang, java.lang.ref, java.util, javax.microedition.io
- MIDP API extends javax.microedition to include
  - lcdui, lcdui.game, media, media.control, midlet, pki, rms
- RIM API extends MIDP and with BB look-and-feel
- CLDC is the most portable application type

Blackberry Mobile OS Technologies

RIM extended JME API – jargon “RIMlets”

- Tighter device integration and access to user interface, networking, and other capabilities

- All 3 can be used together except for net.rim.device.api.ui
  - Better functionality but non-standard
  - CLDC can use all 3 and be mostly portable
- Unlike MIDlets, RIM API is similar to Swing
  - UI operations on the event thread thus not thread safe
  - Apps must grab event thread locks

- net.rim.system.Application (Background)
- net.rim.system.UiApplication (UI)

- Starting point is the main() method
- Display the UI by pushScreen() method
  - Stack based model
  - BB back-button pops these stacks

Blackberry Mobile OS Technologies

RIM extended JME API – jargon “RIMlets” – RIM API

- Some API packages (net.rim.blackberry.api):
  - browser: browse html and create pages
  - invoke: invoke BB applications (mail, task, memo, etc)
  - mail: Mailbox operations (read, write, send)
    - mail.event: event listeners
  - phone: phone events and operations
  - pdap: PDA apps (Tasks, Calendar, Address Book)
    - javax.microedition.pim also good

- RIM device API
  - bluetooth, battery, compression, LDAP, ui, http, math

- Crypto API
  - RIMKeyStore: Classes for storing and using keys
    - TrustedStore: For wireless operations (trust the issuer)
  - Rich set of ciphers and modes
    - Many pub key functions require Certicom key

Blackberry Mobile OS Technologies

Security Model & Sensitive API

- Class files verified for interface compliance
- Limited API set (CLDC)
- Downloading and management within the JVM
- No user-defined class loaders
- No Java Native Interface or user extensions
- System classes cannot be overridden.

- Use of the BlackBerry and RIM API is restricted
  - Tracked for security and export reasons
  - Not in simulator
- The JVM checks for valid code signatures
  - Developer just sends code hashes to a webservice
  - Gets a RIM signed signature
  - Linktime verification
  - Runtime verification

- LBS (GPS) Package
- MultiMedia
- RMS
  - Persistent Storage API
  - Local long term storage
  - Protected API and needs key for encoding

1.2.3 Mobile OS "Native" Applications Overview

4. Microsoft Windows Mobile OS Features

Main App Engine Runner –
Microsoft .NET Compact Edition

- Common Language Runtime for devices
  - Managed Code Execution for applications
  - Memory, thread and security management
  - Code compilation and verification
  - More robust applications
- Just-In-Time compilation
  - Applications are in Microsoft Intermediate Language
  - No speed penalties associated with interpreted code
  - All programming languages are equal

1.2.3 Mobile OS "Native" Applications Overview

4. Microsoft Windows Mobile OS Features

Main App Engine Runner –
Microsoft .NET Compact Edition

- Framework optimized for limited resources
  - Small footprint of 2MB
  - Native components written from ground up
  - Managed bits ported and modified from full Framework
  - Designed for portability
- Base Class Library
  - Subset of desktop .NET Framework library
  - Has additional classes for device-specific functions
  - Same programming model as Desktop

1.2.3 Mobile OS "Native" Applications Overview

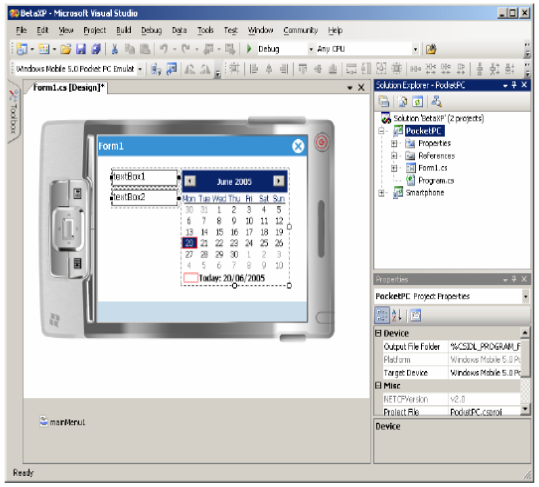
4. Microsoft Windows Mobile OS

Minimum Backward Compatibility Start

- What do I need to get started?
- What will I have?
  - Visual Studio 2005
  - Compact Framework 2.0
- How do I maximize the Compact Framework?

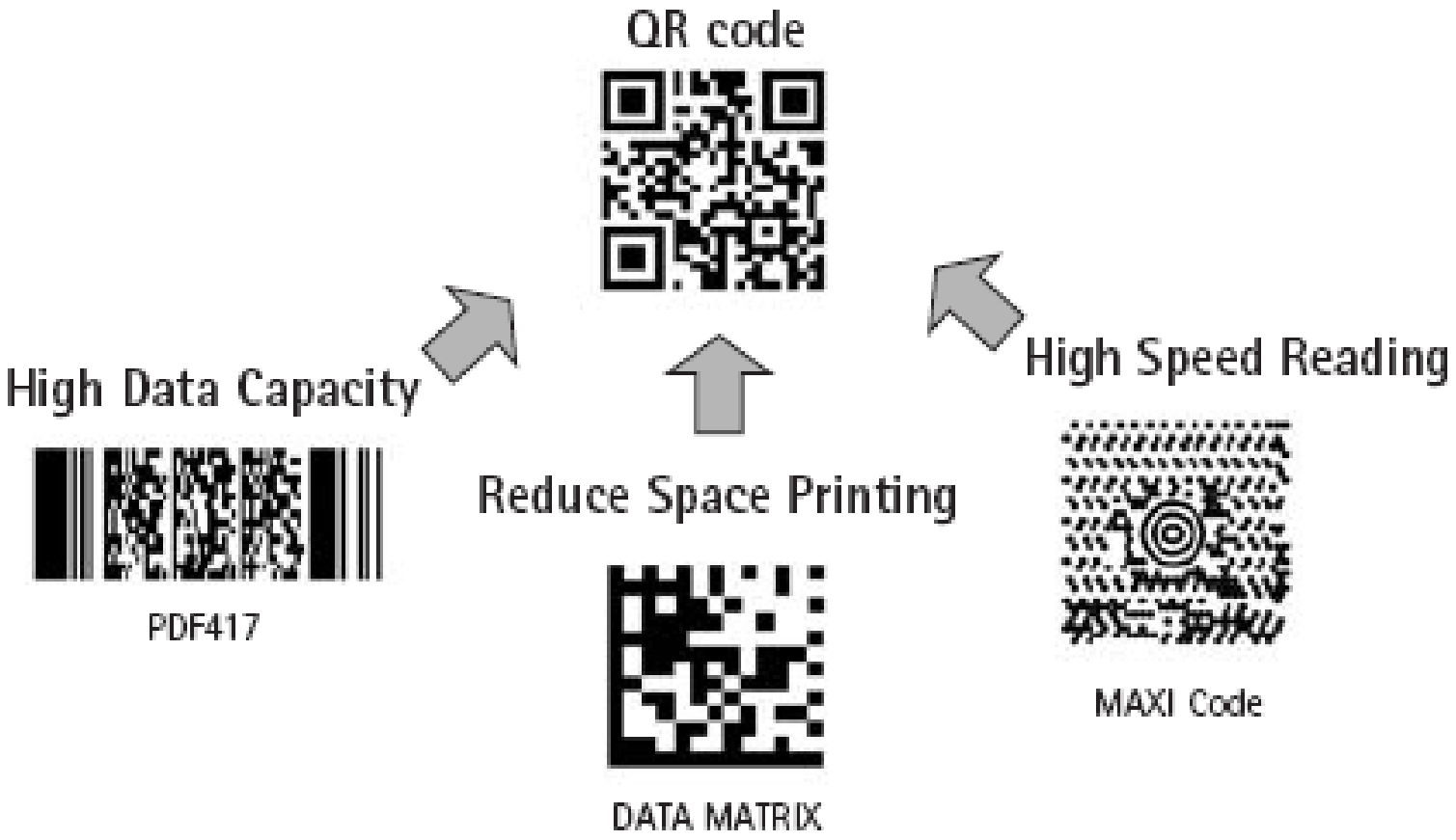
- Visual Studio 2005 Release Candidate
  - http://lab.msdn.microsoft.com/vs2005/get
- ActiveSync 4.0 Developer Preview
  - http://msdn.microsoft.com/mobility/downloads/tools
- Windows Mobile 5.0 SDK Refresh
  - http://msdn.microsoft.com/mobility/downloads/sdks

- Form Designers (Orientation & Resolution)



1.2.4 2D Barcodes Overview

A. **Data Matrix** – ISO/IEC 16022
B. **QR** – ISO 18001



1.2.4 – Data Matrix – ISO/IEC 16022

Step 1: Encoding

The ASCII representation is:

Data character: ‘A’ ‘n’ ‘a’

Decimal: 65+1 110+1 97+1

Hex: 0x42 0x6F 0x62

Table 2 — ASCII encodation values

Codeword	Data or function
1 - 128	ASCII data (ASCII value + 1)
129	Pad
130 - 229	2-digit data 00 - 99 (Numeric Value + 130)

Table 7 — ECC 200 symbol attributes

Symbol size[a]		Data region		Mapping matrix size	Total codewords		Reed-Solomon block		Inter-leaved blocks	Maximum data capacity		
Row	Col	Size	No.		Data	Error	Data	Error		Num.	Alphanum.[d]	Byte
10	10	8 x 8	1	8 x 8	3	5	3	5	1	6	3	1
12	12	10 x 10	1	10 x 10	5	7	5	7	1	10	6	3
14	14	12 x 12	1	12 x 12	8	10	8	10	1	16	10	6

1.2.4 – Data Matrix – ISO/IEC 16022

Step 2: Error Checking & Correction

The Reed Solomon algorithm for Error Checking:

***Annex E in ISO 16022** describes the error correction process for ECC 200 and Annex E.3 in gives an example of a routine to perform the calculation of the error correction code-words.*

Data: Ana

CW No: 1 2 3 4 5 6 7 8

Decimal: 66 111 98 20 66 57 160 115

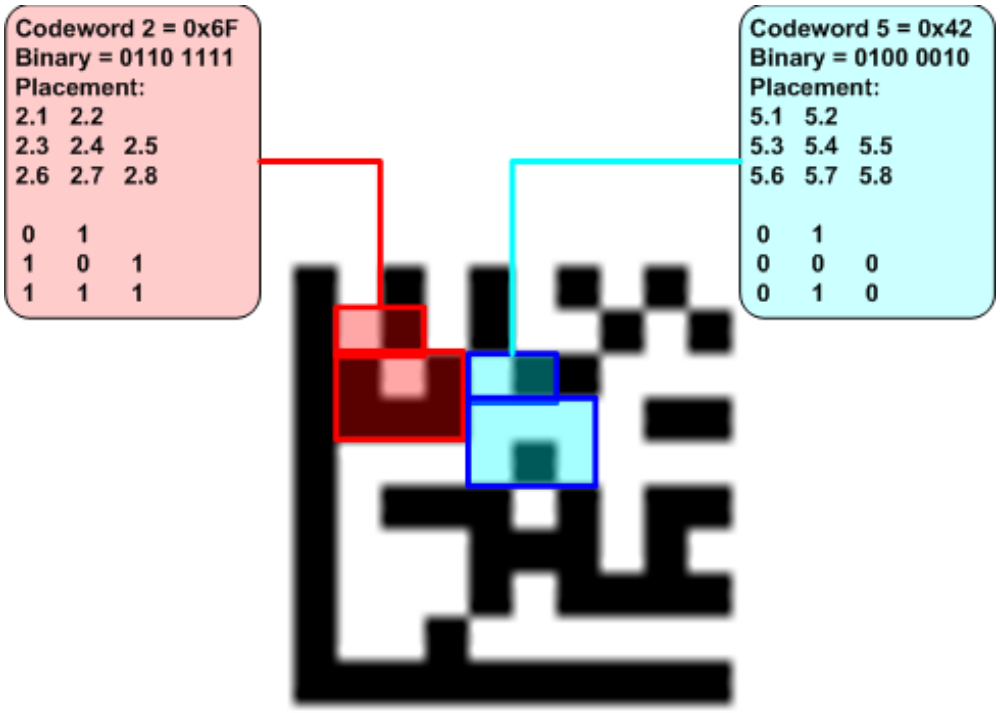
Hex: 42 6F 62 14 42 39 A0 73

____data____/ _____check_____/

1.2.4 – Data Matrix – ISO/IEC 16022

Step 3: Module Place in Matrix
Step 4: Actual Symbol

2.1	2.2	3.6	3.7	3.8	4.3	4.4	4.5
2.3	2.4	2.5	5.1	5.2	4.6	4.7	4.8
2.6	2.7	2.8	5.3	5.4	5.5	1.1	1.2
1.5	6.1	6.2	5.6	5.7	5.8	1.3	1.4
1.8	6.3	6.4	6.5	8.1	8.2	1.6	1.7
7.2	6.6	6.7	6.8	8.3	8.4	8.5	7.1
7.4	7.5	3.1	3.2	8.6	8.7	8.8	7.3
7.7	7.8	3.3	3.4	3.5	4.1	4.2	7.6



Hex: 42 6F 62 14 42 39 A0 73

____data____/ _____check_____/

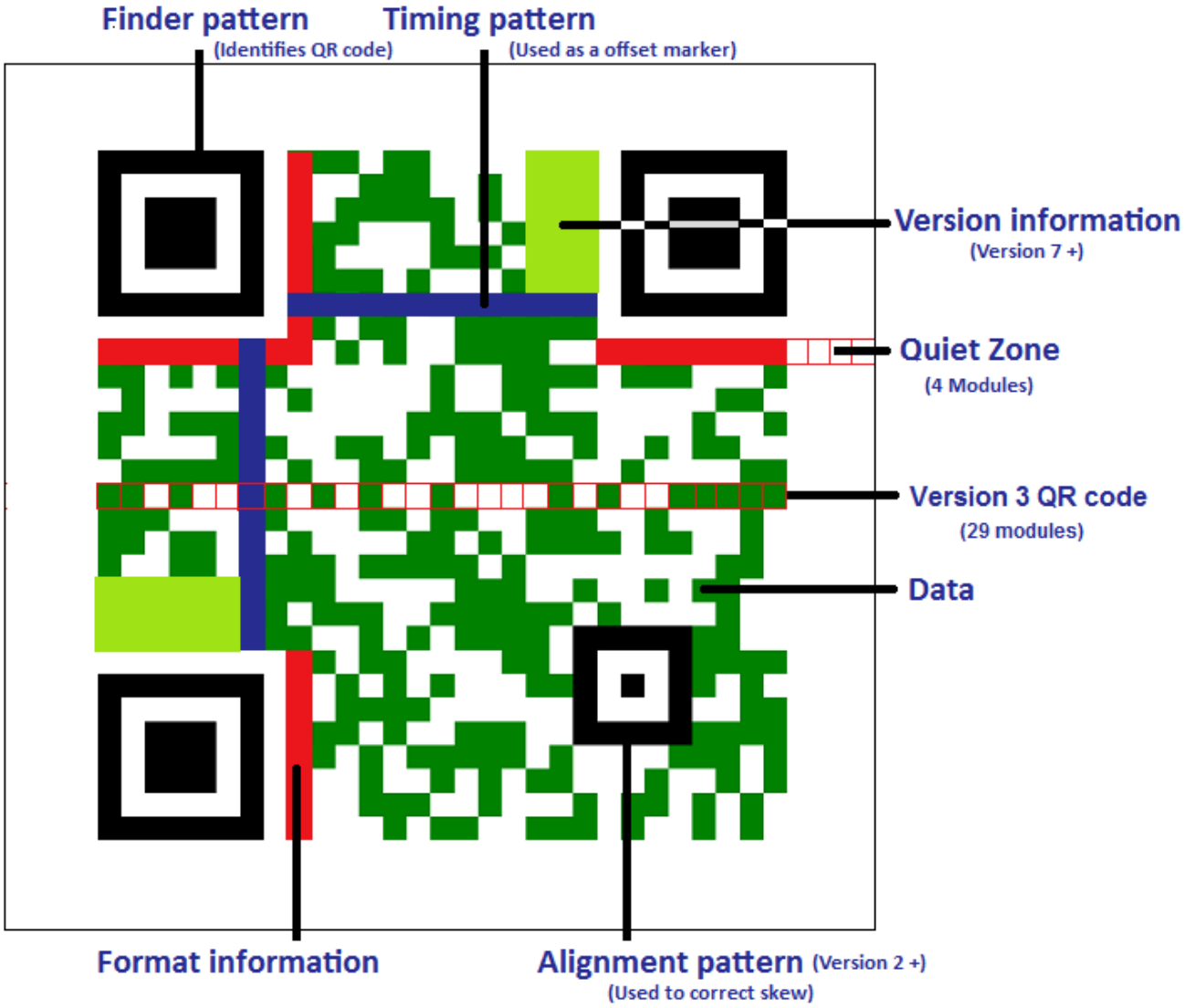
1.2.4 – QR Code – ISO 18004

QR Code Features

Symbol size	Min. 21x21 cell - Max. 177x177 cell (with 4-cells interval)	
Information type and volume	Numerical characters	7,089 characters at maximum
	Alphabets, signs	4,296 characters at maximum
	Binary (8 bit)	2,953 characters at maximum
	Kanji characters	1,817 characters at maximum
Conversion efficiency	Numerical characters mode	3.3 cells/character
	Alphanumerical/signs mode	5.5 cells/character
	Binary (8 bit) mode	8 cells/character
	Kanji character mode (13 bit)	13 cells/character
Error correction functionality	Level L	Approx. 7% of the symbol area restored at maximum
	Level M	Approx. 15% of the symbol area restored at maximum
	Level Q	Approx. 25% of the symbol area restored at maximum
	Level H	Approx. 30% of the symbol area restored at maximum
Linking functionality	Possible to be divided into 16 symbols at maximum	

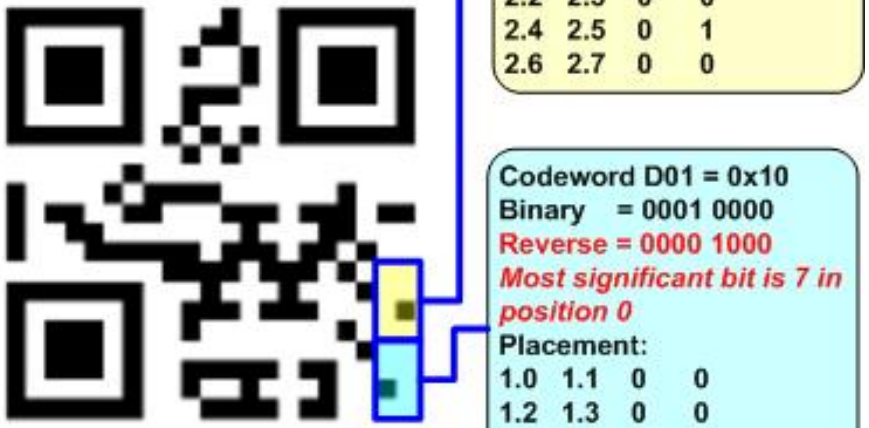
1.2.4 – QR Code – ISO 18004

QR Code Structure



1.2.4 – QR Code – ISO 18004

QR Code Creation



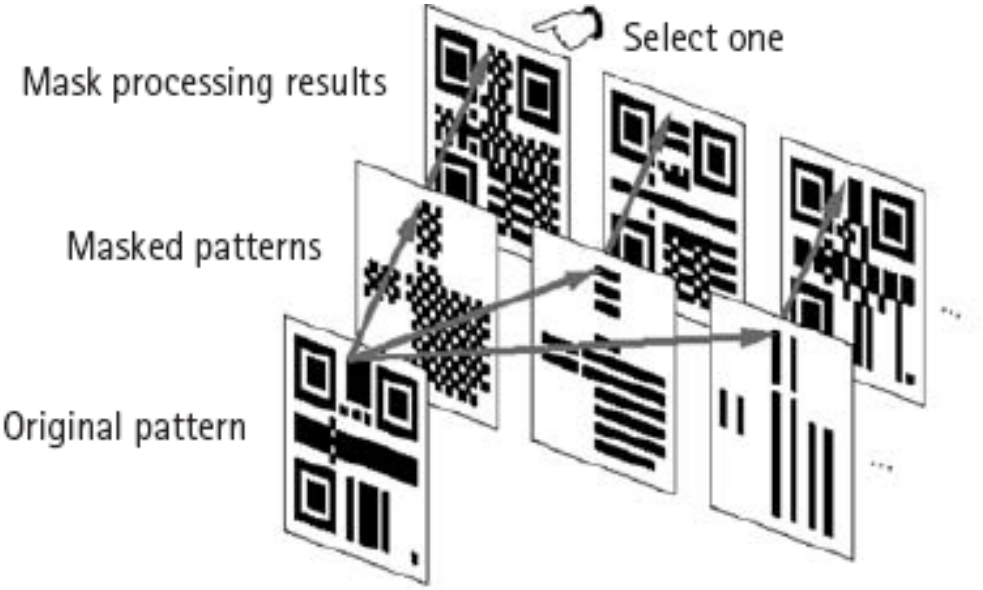
Codeword D02 = 0x20
Binary = 0010 0000
Reverse = 0000 0100
Most significant bit is 7 in position 0
Placement:

2.0	2.1	0	0
2.2	2.3	0	0
2.4	2.5	0	1
2.6	2.7	0	0

Codeword D01 = 0x10
Binary = 0001 0000
Reverse = 0000 1000
Most significant bit is 7 in position 0
Placement:

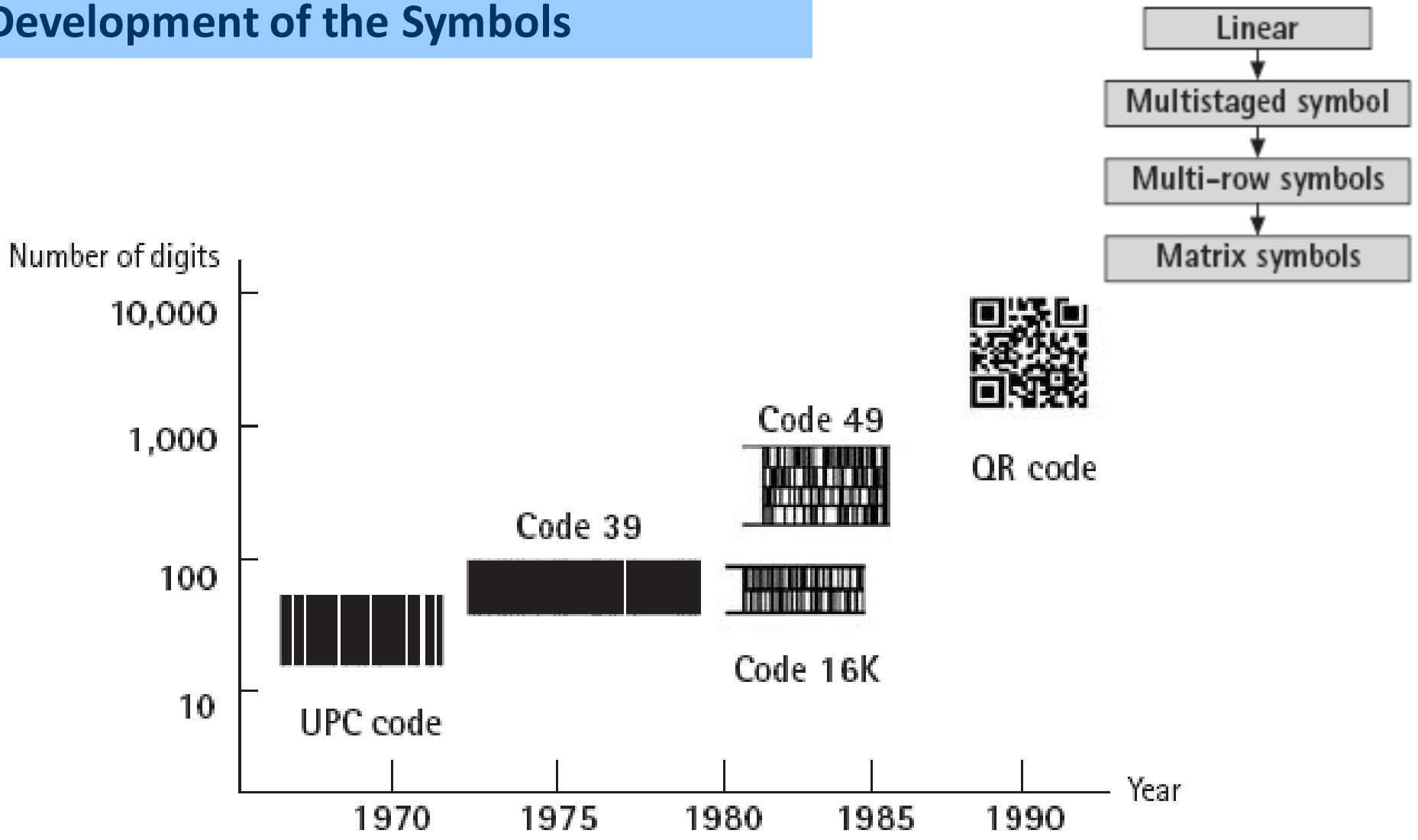
1.0	1.1	0	0
1.2	1.3	0	0
1.4	1.5	1	0
1.6	1.7	0	0

The most significant bit (shown as bit 7) of each codeword shall be placed in the first available module position.



1.2.4 – QR vs. Others 2D Barcodes

Development of the Symbols



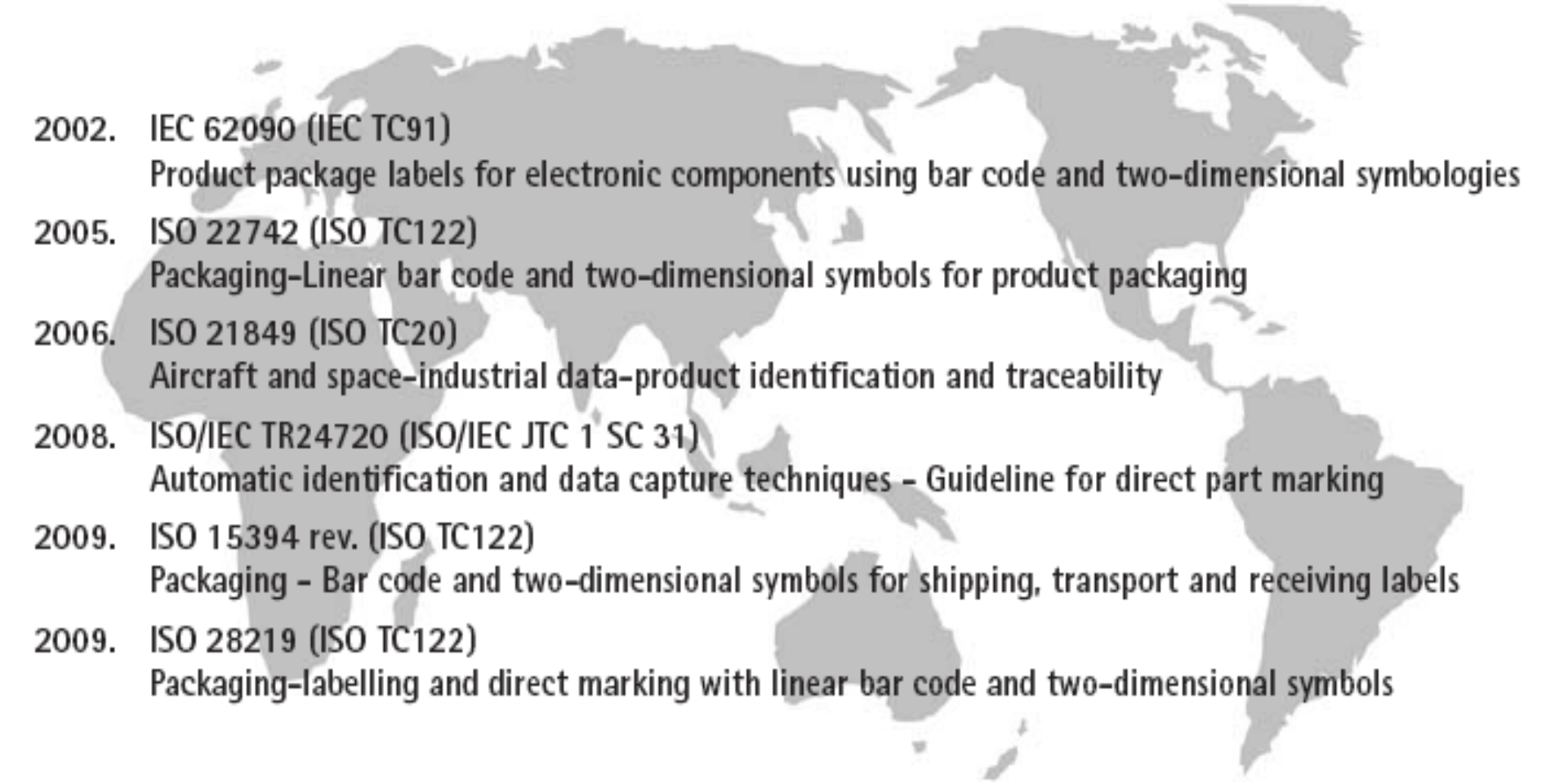
1.2.4 – QR Code – ISO/IEC 18004

QR Standardization

1997/10.	AIM International Automatic Identification Manufacturers	AIM-ITS 97/001
1999/01.	Japanese Industrial Standard	JIS-X0510
1999/09.	JAMA Japan Automobile Manufacturers Association	JAMA-EIE001
2000/06.	ISO International Organization for Standardization	ISO/IEC 18004
2000/12.	Chinese National Standard	GB/T 18284
2002/12.	Korea National Standard	KS-X ISO/IEC18004
2003/12.	Vietnam National Standard	TCVN7322

1.2.4 – QR Code – ISO/IEC 18004

Application Standards using QR Code



2002. IEC 62090 (IEC TC91)
Product package labels for electronic components using bar code and two-dimensional symbologies

2005. ISO 22742 (ISO TC122)
Packaging-Linear bar code and two-dimensional symbols for product packaging

2006. ISO 21849 (ISO TC20)
Aircraft and space-industrial data-product identification and traceability

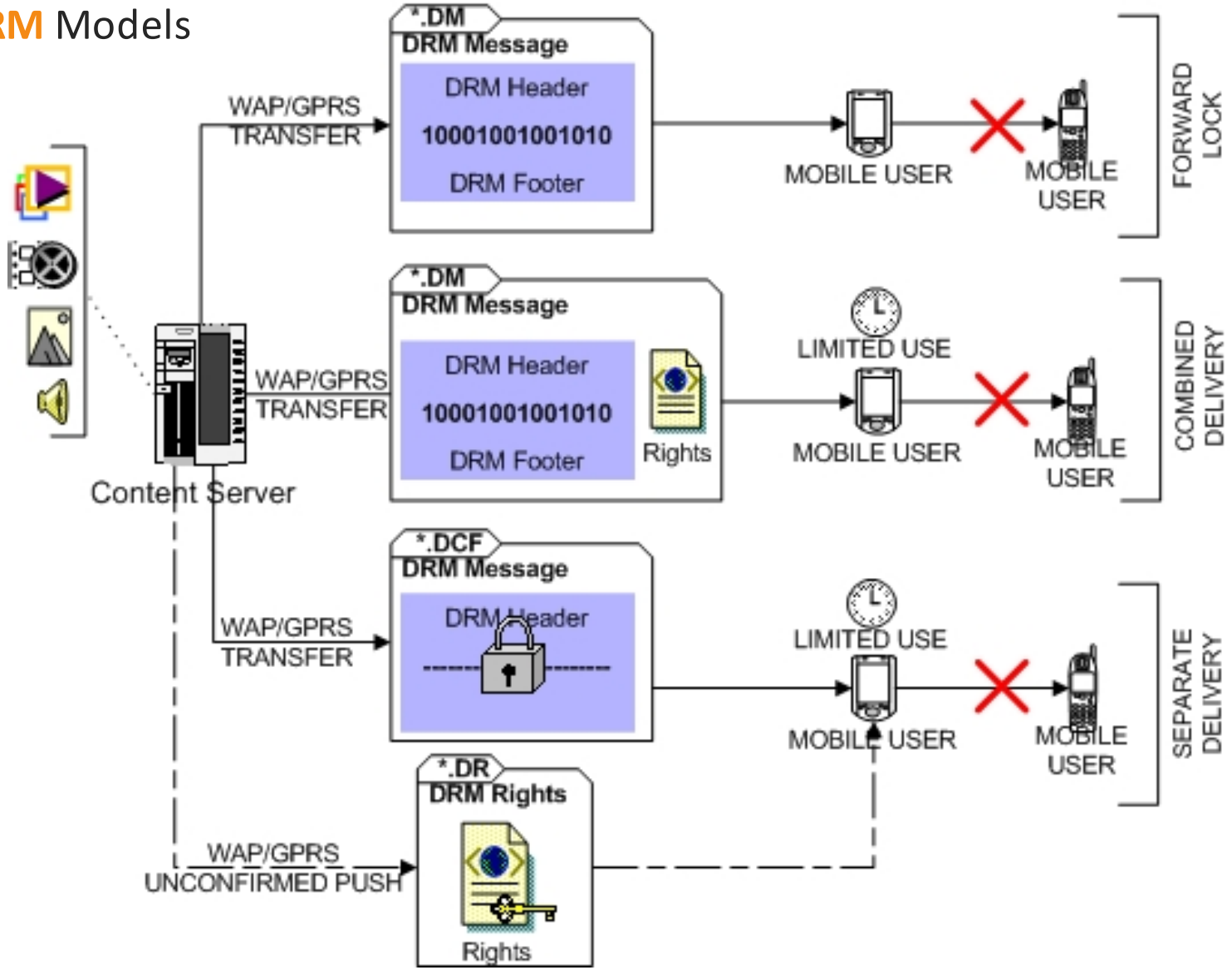
2008. ISO/IEC TR24720 (ISO/IEC JTC 1 SC 31)
Automatic identification and data capture techniques - Guideline for direct part marking

2009. ISO 15394 rev. (ISO TC122)
Packaging - Bar code and two-dimensional symbols for shipping, transport and receiving labels

2009. ISO 28219 (ISO TC122)
Packaging-labelling and direct marking with linear bar code and two-dimensional symbols

1.2.5 Mobile OMA DRM Overview

OMA m-DRM Models



1.2.5 Mobile OMA DRM Overview

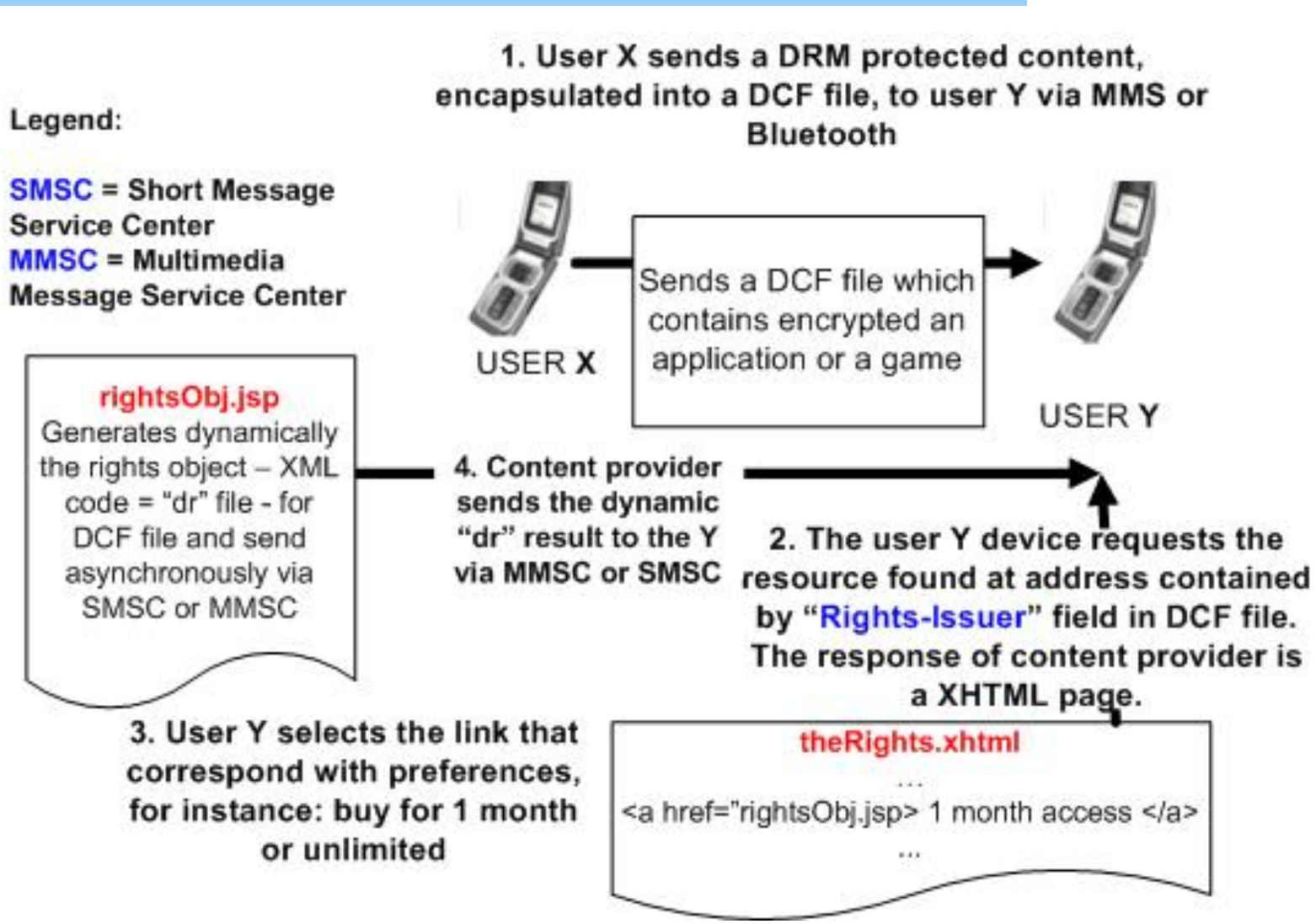
OMA m-DRM Sample Message in HTTP Response

Combined Delivery Model

```
--boundary-1
Content-type:
      application/vnd.oma.drm.rights+xml
Content-Transfer-Encoding: binary
<o-ex:rights
 xmlns:o-ex="http://odrl.net/1.1/ODRL-EX"
 xmlns:o-dd="http://odrl.net/1.1/ODRL-DD"
 xmlns:ds="http://www.w3.org/2000/09/xmldsig#/">
 <o-ex:context>
  <o-dd:version>1.0</o-dd:version>
 </o-ex:context>
 <o-ex:agreement>
  <o-ex:asset> <o-ex:context>
     <o-dd:uid>cid:http://content-id-here</o-
     dd:uid>
   </o-ex:context></o-ex:asset>
```

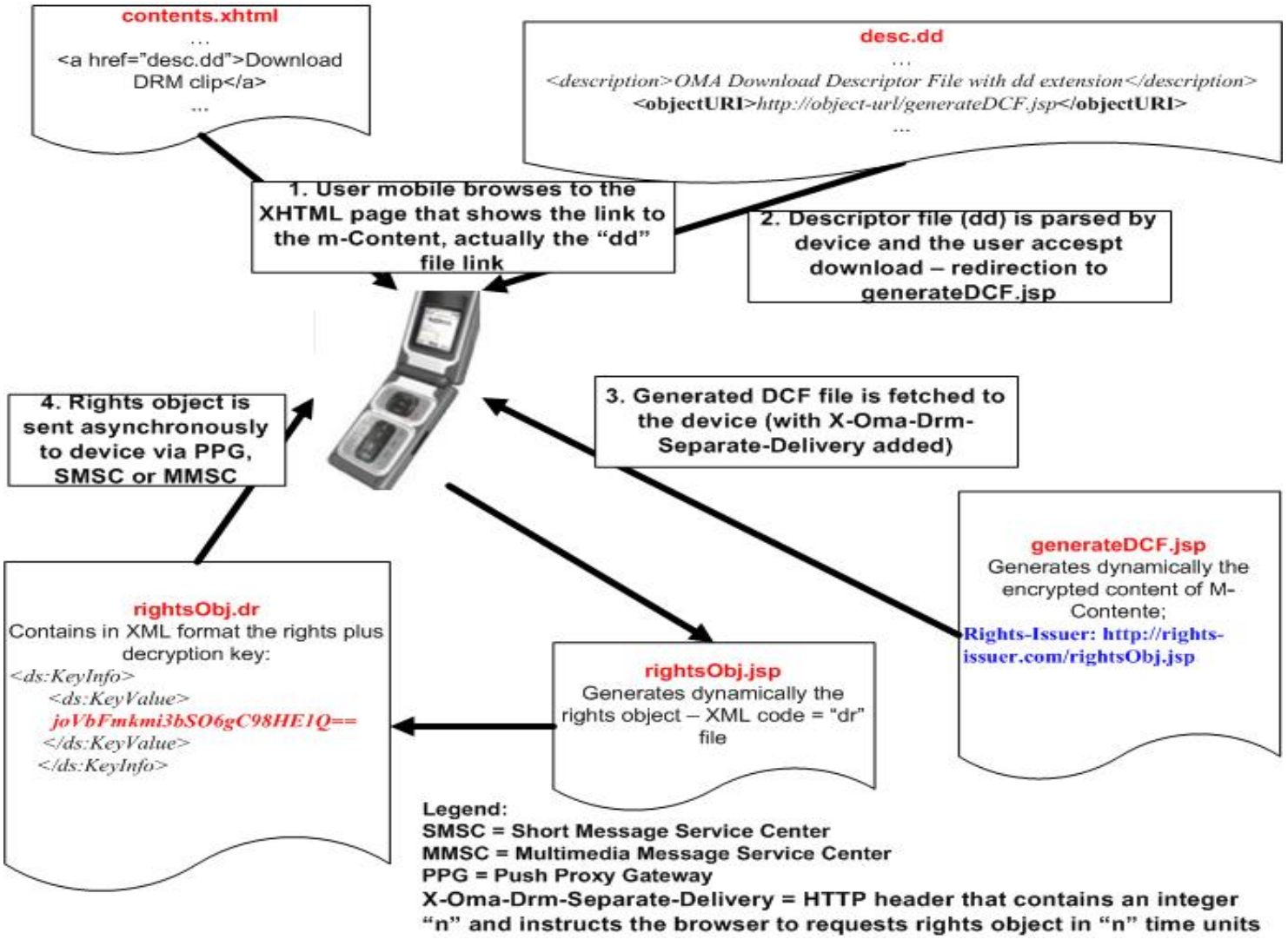
```
<o-ex:permission>
    <o-dd:play>
      <o-ex:constraint>
        <o-dd:count>3</o-dd:count>
        <o-dd:datetime>
         <o-dd:start>2012-08-01T20:59:10</o-
      dd:start>
         <o-dd:end>2012-11-01T20:59:10</o-
      dd:end>
        </o-dd:datetime>
      </o-ex:constraint>
    </o-dd:play>
  </o-ex:permission>
 </o-ex:agreement>
</o-ex:rights>
--boundary-1
Content-type: image/jpeg
Content-ID: <http://content-id-here>
Content-Transfer-Encoding: binary
ÿØÿà…Binary representation of the M-CONTENT
--boundary-1--
```

Separated Delivery Model – Business Approach



Section 1.2.5 – Mobile Digital Rights Management

Separated Delivery Model – Technical Approach



Separated Delivery Model – 1/3 Download Descriptor File

```
<media xmlns="http://www.openmobilealliance.org/xmlns/dd">
  <DDVersion>1.0</DDVersion>
  <name>Name Of Product</name>
  <size>1234</size>
  <type>image/jpg</type>
  <vendor>Media Vendor Company</vendor>
  <description>Description</description>
  <objectURI>http://object-url</objectURI>
  <iconURI>http://icon-url</iconURI>
  <infoURL>http://info-url</infoURL>
  <nextURL>http://next-url</nextURL>
  <installNotifyURI>
    http://install-notify-url
  </installNotifyURI>
  <installParam>-param1 -param2</installParam>
</media>
```

Separated Delivery Model – 2/3 Download DCF File

image/jpegcid:http://content-id-here□gŽŒEncryption-Method: AES128CBC
Content-Name: "NameOfContent"
Rights-Issuer: http://rights-issuer.com/content
Content-Description: "DescriptionOfContent"
Content-Vendor: "VendorName"
Icon-Uri: http://vendor.com/content-icon.gif
”¶{…Binary encrypt representation of the M-CONTENT using AES-Rijndael symmetric key algorithm in CBC mode

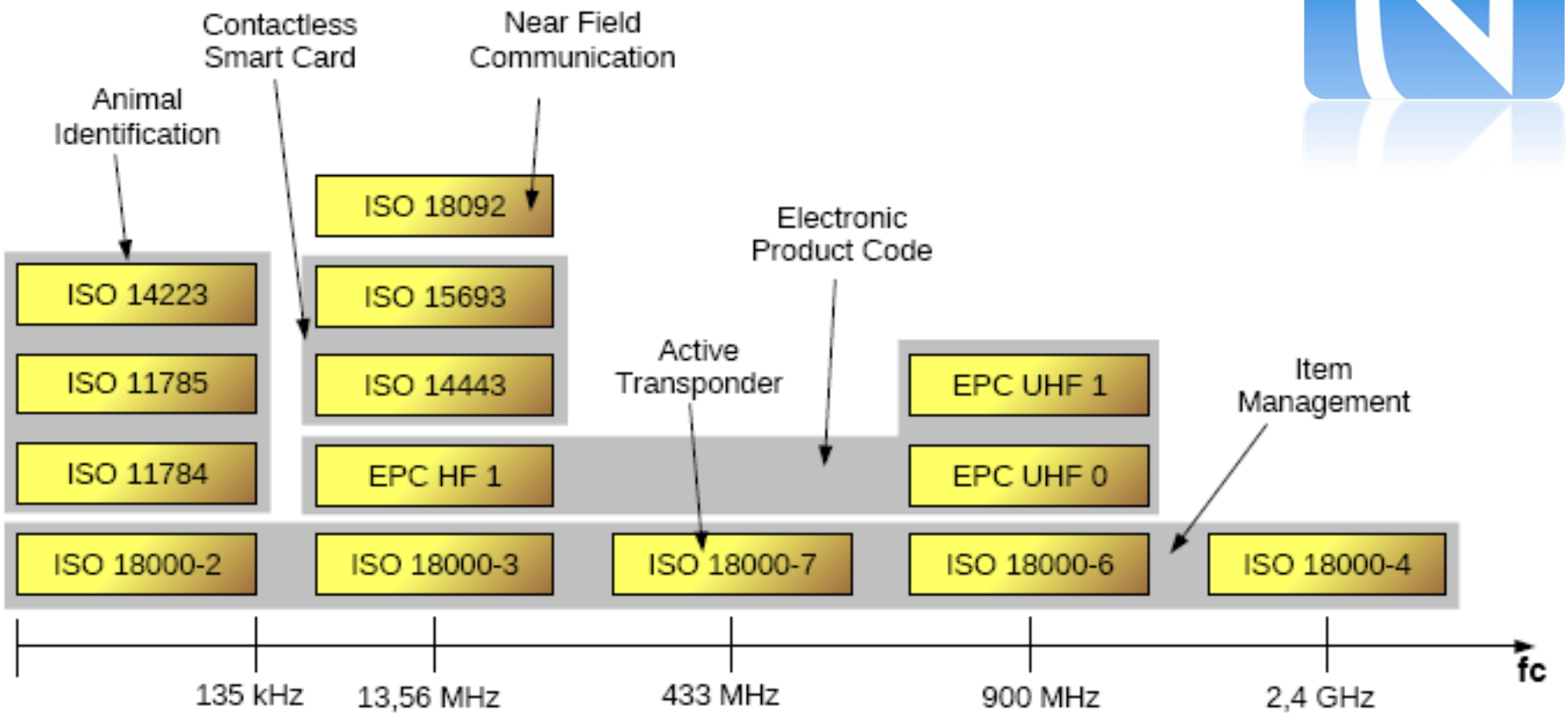
Separated Delivery Model – 3/3 Download Digital Rights Object

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE o-ex:rights PUBLIC "-//OMA//DTD DRMREL
      1.0//EN"
      "http://www.oma.org/dtd/dr">
<o-ex:rights
  xmlns:o-ex="http://odrl.net/1.1/ODRL-EX"
  xmlns:o-dd="http://odrl.net/1.1/ODRL-DD"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#/">
  <o-ex:context>
    <o-dd:version>1.0</o-dd:version>
  </o-ex:context>
  <o-ex:agreement>
    <o-ex:asset>
      <o-ex:context>
        <o-dd:uid>cid:http://content-id-here</o-dd:uid>
      </o-ex:context>
      <ds:KeyInfo>
       <ds:KeyValue>
        joVbFmkmi3bSO6gC98HE1Q==
      </ds:KeyValue>
     </ds:KeyInfo>
    </o-ex:asset>
```

```
  <o-ex:permission>
    <o-dd:play>
      <o-ex:constraint>
        <o-dd:count>2</o-dd:count>
        <o-dd:datetime>
          <o-dd:start>2006-09-27T20:59:10</o-
        dd:start>
          <o-dd:end>2007-09-27T20:59:10</o-
        dd:end>
        </o-dd:datetime>
      </o-ex:constraint>
    </o-dd:play>
  </o-ex:permission>
  </o-ex:agreement>
</o-ex:rights>
```

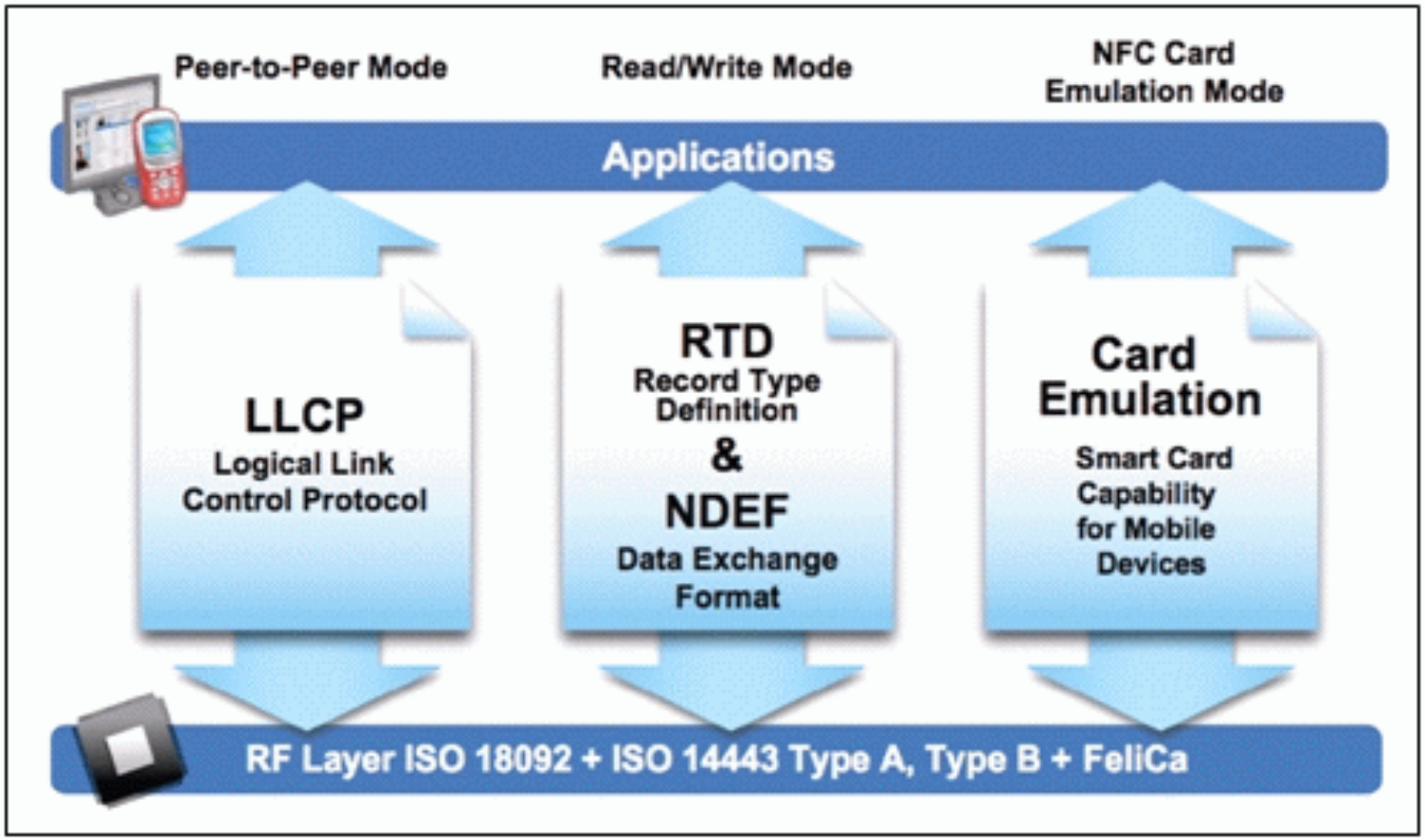
1.2.6 NFC Overview

NFC/RFID Standards



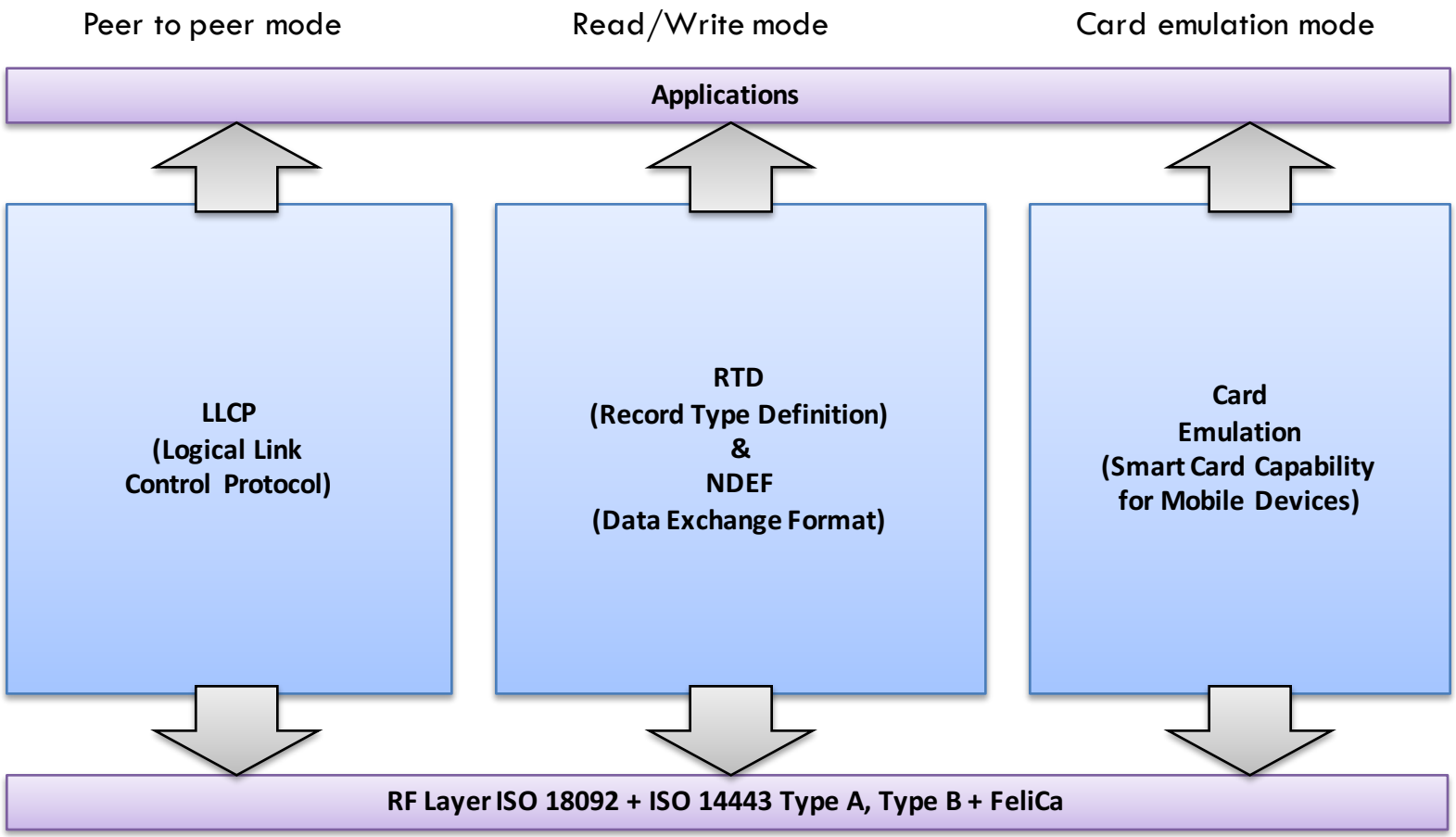
1.2.6 NFC Overview

NFC Modes

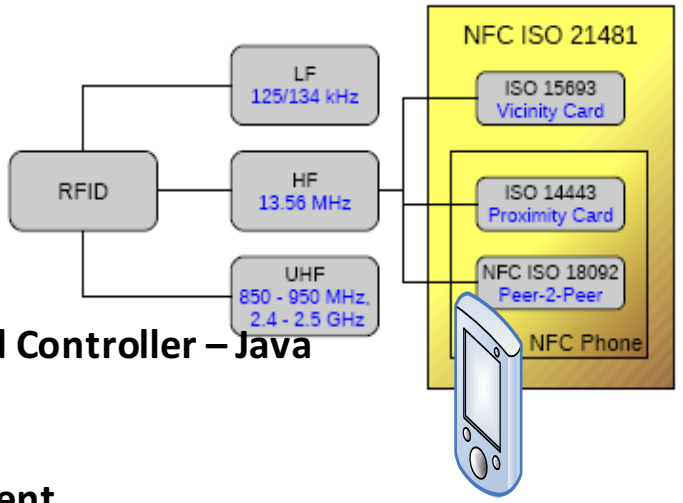


1.2.6 NFC Overview

NFC Forum Specs



1.2.6 NFC Overview

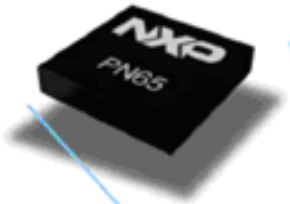


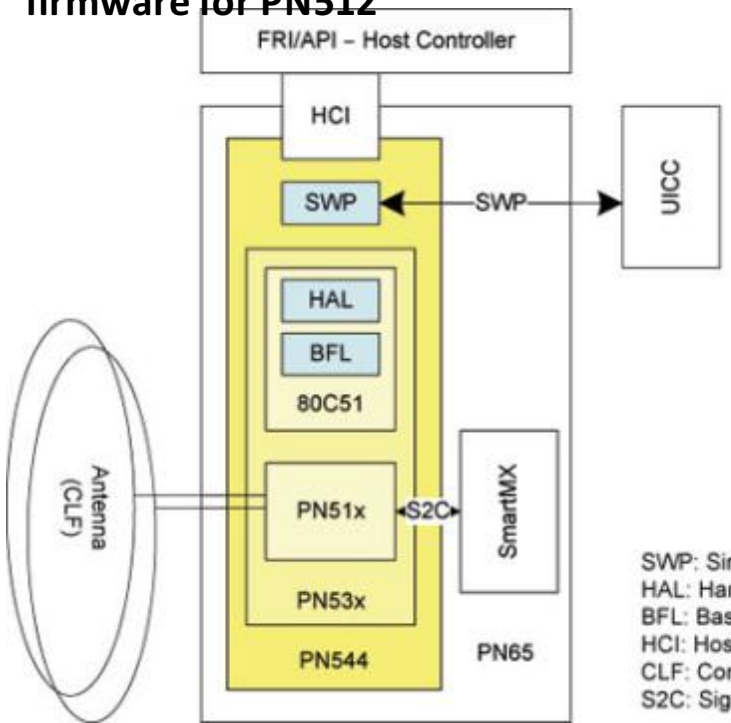
NFC/RFID Mobile Implementation Case

NXP PN65:

- **SmartMX - P5CN072 Secure Dual Interface PKI Smart Card Controller – Java Card**
- **PN 544**
  - **SWP – Single Wired Protocol for SIM as secure element**
  - **PN531**
    - **PN 512 - NFC transmission module @ 13,56MHz**
    - **Microcontroller 80C51 – 32KB ROM and 1KB RAM running firmware for PN512**

***Samsung Nexus S (Android OS) + ISO 15693 + Proprietary P2P
Nokia C7 (Symbian OS)

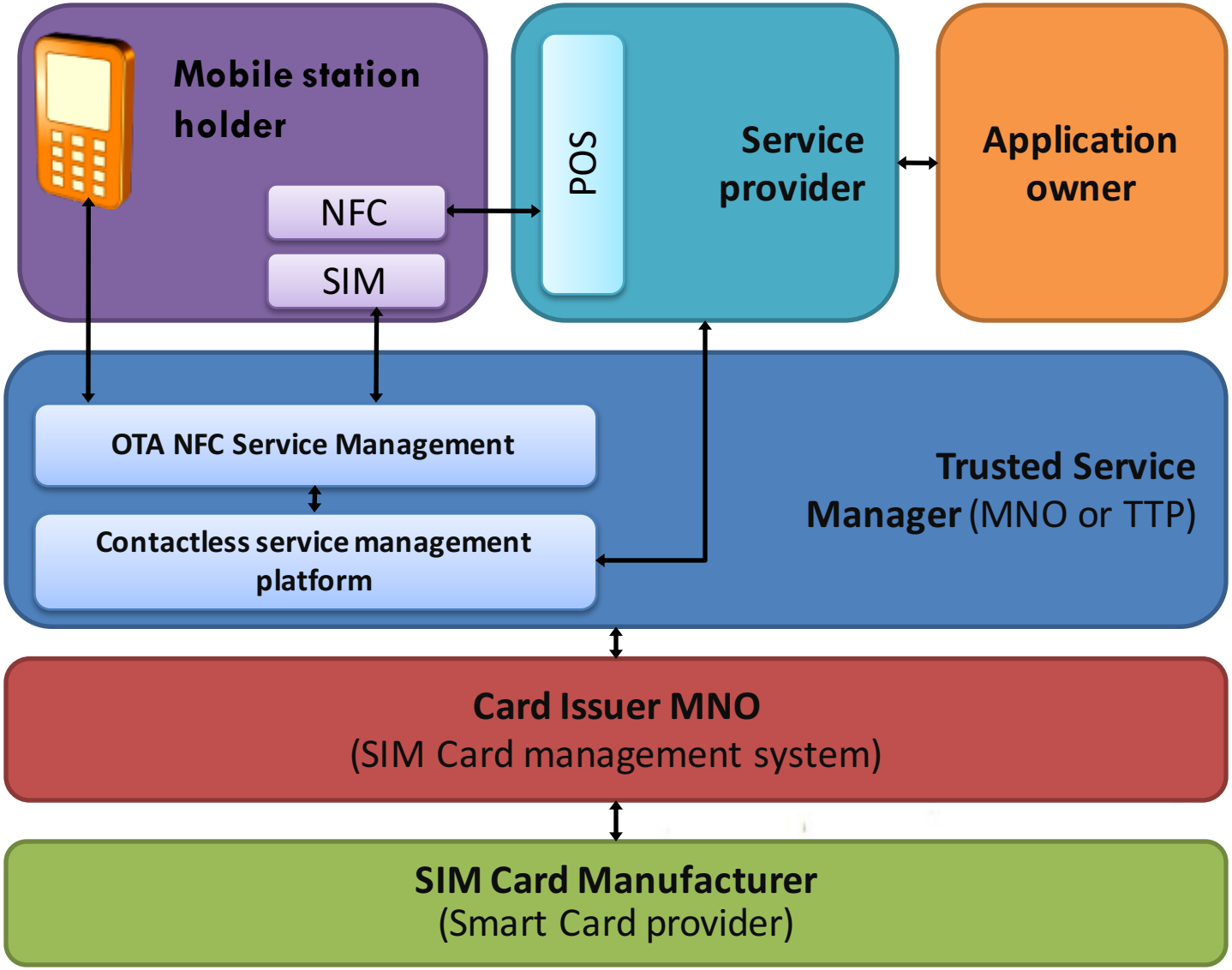




SWP: Single Wire Protocol
HAL: Hardware Abstraction Layer
BFL: Basic Function Library
HCI: Host Controller Interface
CLF: Contactless Frontend
S2C: SigIn-SigOut-Connection

1.2.6 NFC Overview

NFC Roles and Actors for M-Payment Services



Too much information?

This was **Section 1 – Technical Issues**

- ASN.1 and DER
- Hash Functions
  - MD2, MD5, SHA-1, SHA-2, SHA-3, RIPEMD-128, RIPEMD-160 and MAC
- Symmetric Encryptions Schemes
  - DES, 3DES, IDEA, AES-Rijndael in ECB, CBC, PCBC Modes
  - RC2, RC4, RC5 and RC6
- Asymmetric Encryptions Schemes
  - RSA, DSA, El-Gammal, and Elliptic Curves
  - Digital Signature and Enveloping
  - X509 v3 Certificates and PKIs
  - Dual Signature & Nonce
  - Blind Signatures
- TCP/IP Networking Security
  - IPSec
  - SSL / TSL
  - HTTPS, S-MIME, ESMTP (AUTH CRAM-MD5)

There are a lot of MOBILE devices, technologies, concepts and APIs/SDKs.

+ CRYPTO SECURITY

Section Conclusion

For M-Payment Secure and Reliable Services, it is a MUST to combine MOBILE technologies with CRYPTO SECURITY.

It's about multi-disciplinary work, and IT **development & integration** tasks.

It's about where is going to be STORED sensitive data – in SIM, NFC chipset, mobile device or data services back-end.

Technologies may be used for the mobile payment service solutions:

- Short Message Service (SMS) / Multimedia Message Service (MMS) / Unstructured Supplementary Services Delivery (USSD)
- Web Applications over WAP/GPRS/3G (UMTS) & 4G (LTE) Data Connections
- Mobile Device Application (OS based: Google Android, Apple iOS, RIM Blackberry OS, Microsoft Windows Mobile, MeeGo – Intel Tizen
- Cross-platform frameworks: JME – Java Micro Edition versus HTML5/CSS3/JS
- (U)SIM-based Application
- Near Field Communication (NFC) / NFC 2.0 – including Data over Voice.

M-Payments Services Features, models, case-studies, and success stories

M-Payments Features and Case Scenarios

It’s not just about the ideas, but technologies, architectures & security

M-Payment Services

What about the **M-PAYMENT SERVICES**

Requirements & Features?

They are not new.

The products and goods that can be achieved with the m-payments services methods:

Electronic content: **Applications, e-Books, games, VOD – Video on Demand, music, ringtones, wallpapers, etc.**

Hard goods: **Concerts tickets, Books, journals, magazines, etc.**

Services access: **transportation fare – bus, subway or train, parking meters, cinema access and other services.**

2. M-Payment Services Requirements

According with (Karnouskos & Fokus, 2004), the mobile payment service requirements:

- *Simplicity & Usability:* The m-payment application should have an ergonomic GUI – Graphical User Interface with small learning curve to the customer. The application's personalization is a criterion for the end-user satisfaction.

- *Universality:* M-payments service should be possible for low value micro-payments and high value macro-payments and it should include domestic, regional and global environments.

- *Interoperability:* The m-payments service should be able to interact with other systems and it should be based on open standards and technologies.

2. M-Payment Services Requirements

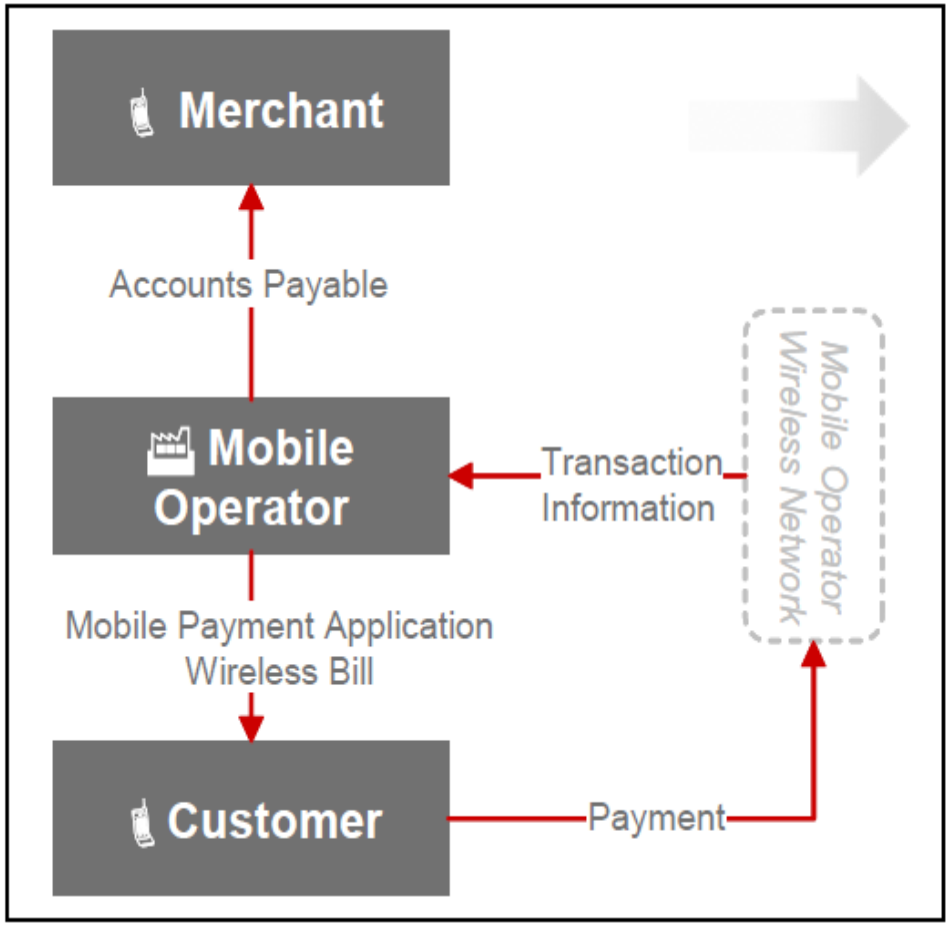
- *Security, Privacy & Trust:*
  - As an objective the mobile payments have to be as anonymous as cash transactions.
  - If the mobile payments system is not anonymous, then a customer must be able to understand how his or her private information is protected.
  - Also, when mobile payments transactions are recorded, the customer privacy should not be openly available for the public access – the credit histories and spending patterns of the customer.
  - The system should be "bullet-proof", resistant to inside or outside attacks.

- *Cost:* From macro and micro systems point of view the costs of the usability and deployment for the m-payments systems should be lower than the existing payment mechanisms.

- *Speed:* The speed of the mobile payments execution should be acceptable to customers and merchants.

- *Cross border payments:* The m-payment application and transactions should be available globally, in order to be widely accepted – regional or world-wide.

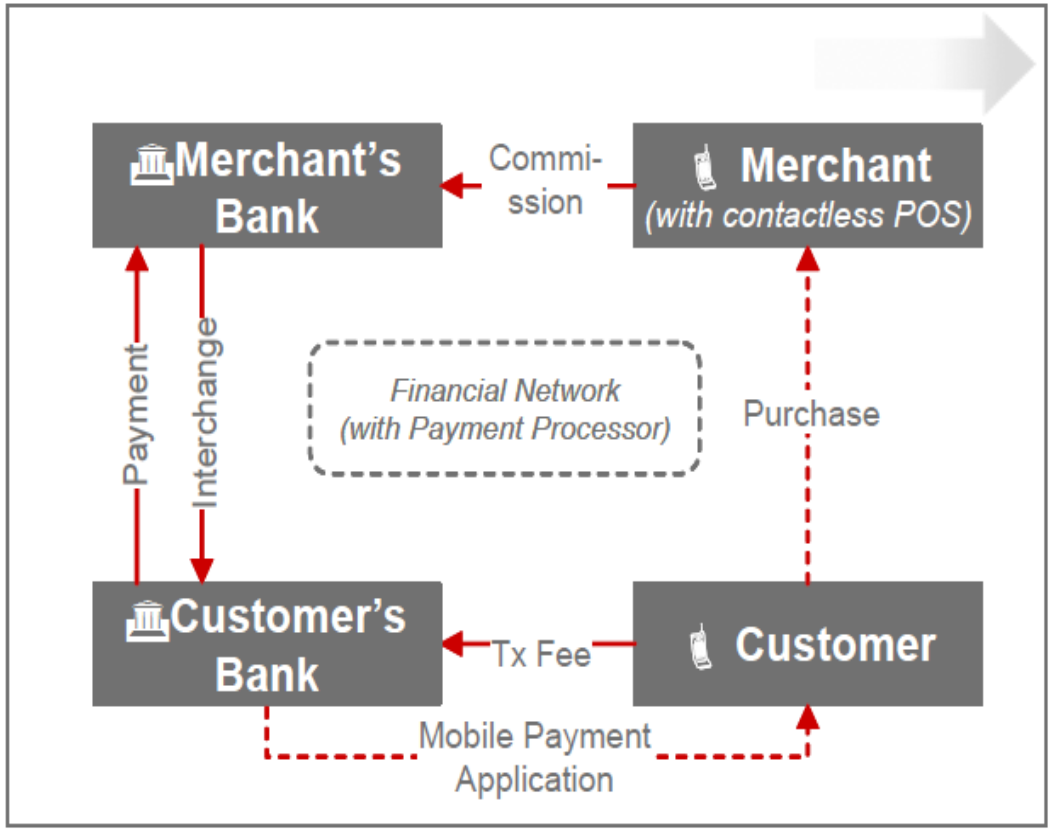
2. M-Payment Services Models



A. MNO-Mobile Operator Centric Model

The mobile payment service is deployed independently by a MNO – mobile network operator. An independent mobile wallet with electronic cash or money (stored in the SIM, internal/external crypto-chip to the mobile device or software application) may be provided by the mobile operator. The charging of the electronic wallet may be done through the user mobile account (telephony company bill) and the money withdraw may be done using specialized offices with mobile operator agreement.

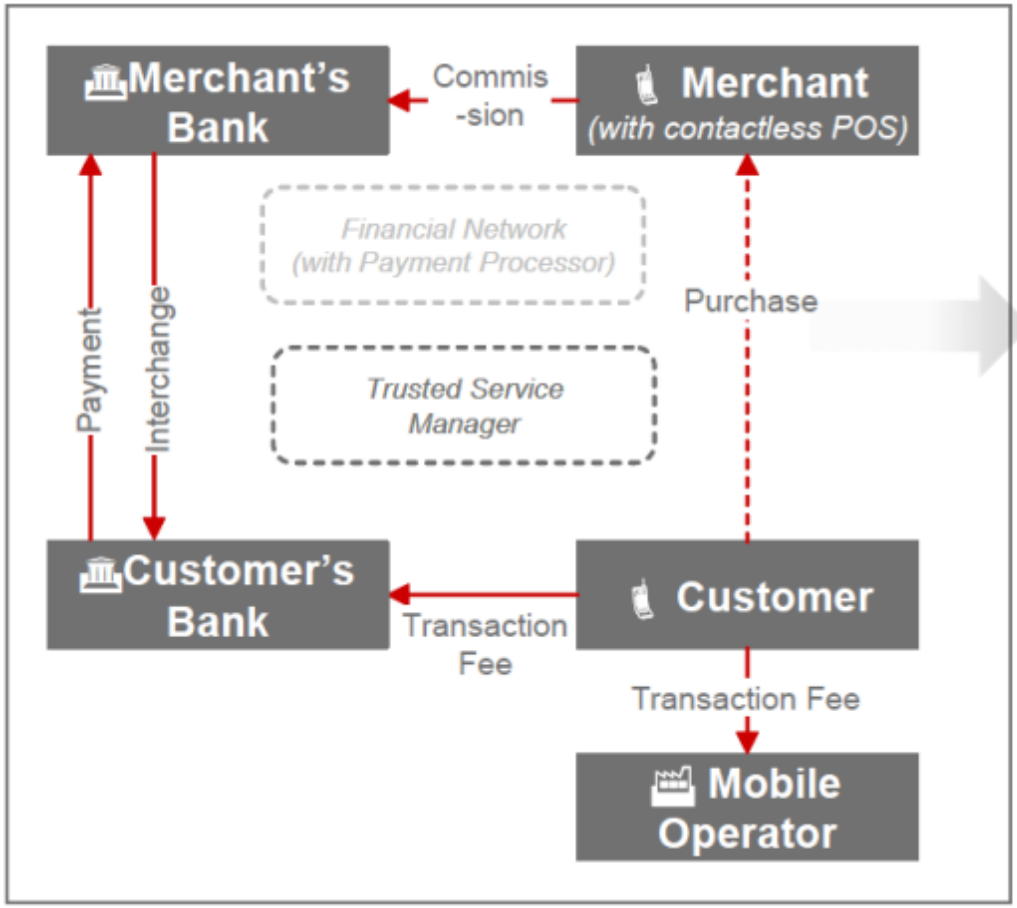
2. M-Payment Services Models



B. Bank Centric Model

The mobile applications or devices are provided by a bank to the customers for the mobile payment transaction achievement and the bank provides to the merchants the compliant point-of-sale (POS). Mobile network operators are used as simple carriers or device providers.

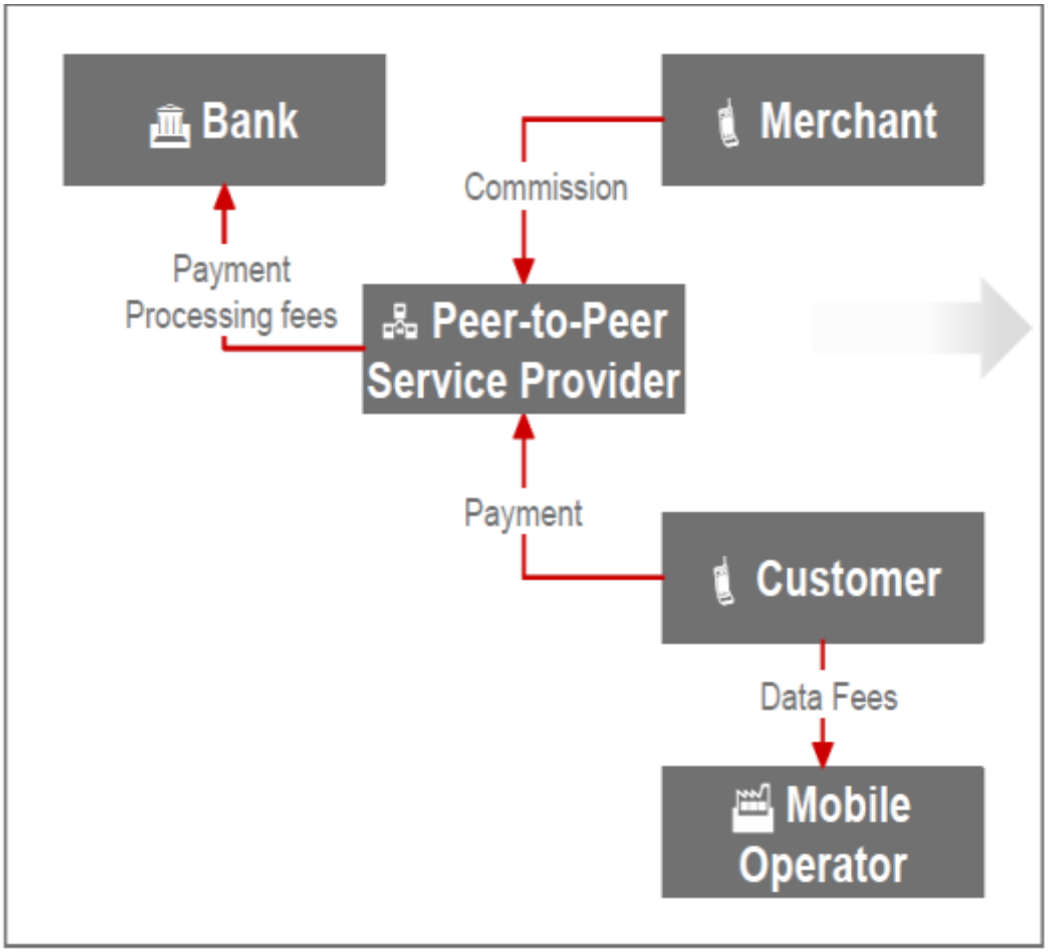
2. M-Payment Services Models



C. Collaboration Model

The banks, mobile operators and a trusted third party are collaborating for providing the mobile payment service, including the issuing of co-branded devices that ensures the customer loyalty.

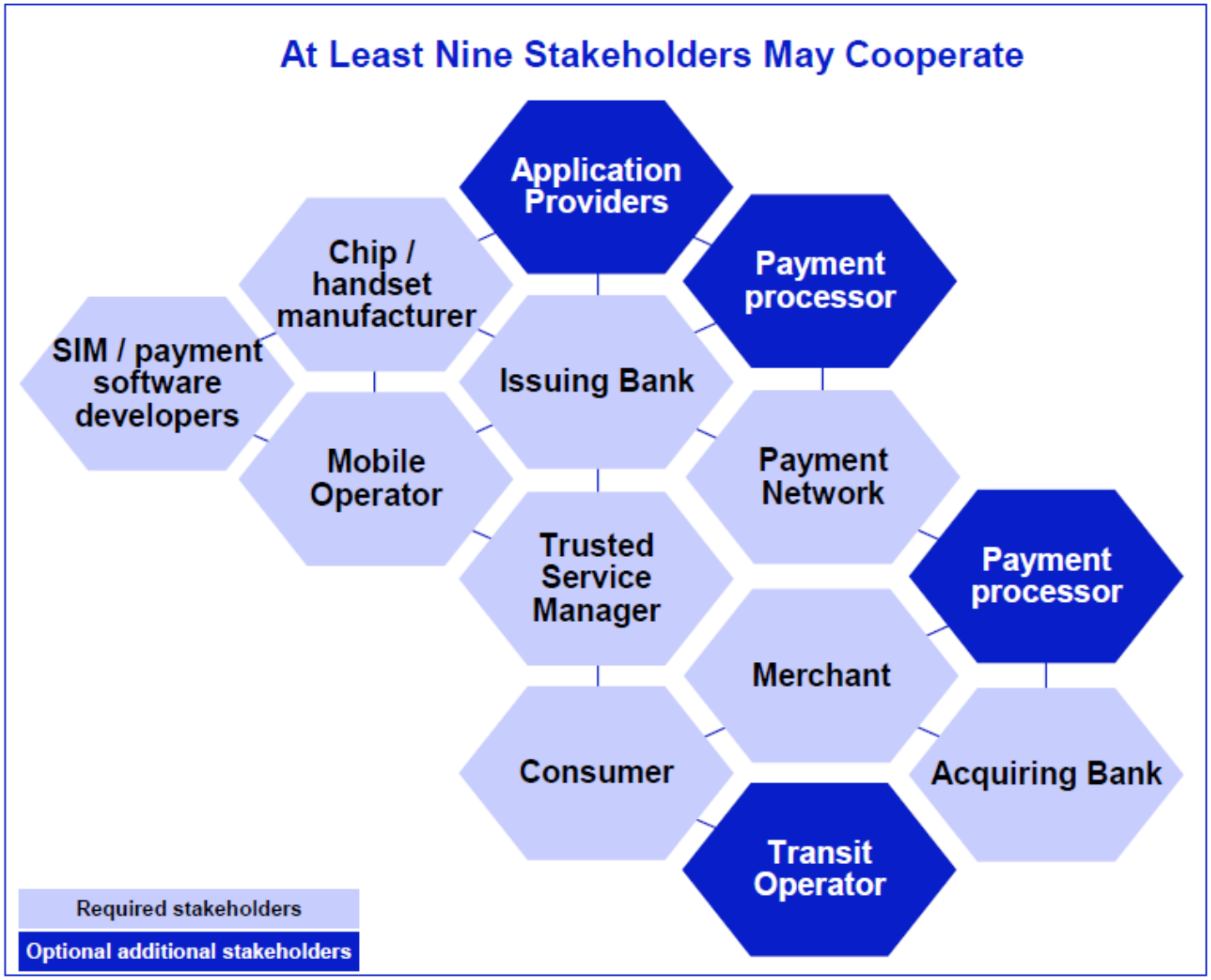
2. M-Payment Services Models



D. Peer to Peer Model

A private/public institution or company, independently from financial institutions and mobile network operators, is the mobile payment service provider.

2. M-Payment Service Stakeholders



2. M-Payment System Samples

One advantage of a mobile payment system administered by a network operator is that payments can be charged directly to the subscriber's phone bill. With third-party account schemes, the payment must be obtained from the user by other means.

- P2P Model: Mobile Bitcoin (Blockchain.info + Mycellium)
- Bank centric (?) Model: Deutschland Mobile EC/Geldkarte / ING SMS | ING/BT Tokenization Mobile Payment – Bank+VISA
- Collaboration Model: Google/Apple Pay (VISA Tokenization) vs. Mobile EMV | Google Wallet (eSE - Java Card)
- FinTech Collaboration Model: Seqr.com + MobilePay + Revolut
- Mobile Network Operator Model: M-Pesa + Orange Money – MNO Model
- Other/Old "Mobile Micro-Payments Systems":
  - 1. Sonera MobilePay – "bank account or credit card based" – SMS
  - 2. Paybox – PayBox - Sybase mPayment 365 – Internet Connection
  - 3. GiSMo – confirmation SMS secure code for Internet PCs
  - 4. Mobile SET – Internet Connection + Credit Card
  - 5. Mobile Evolution of GeldKarte? – "mobile SIM wallet"
  - 6. 2D BATS – case study
  - 7. Sony FeliCa / NFC

Section 2.1 – Mobile Payment Systems

1. Mobile Bitcoin Wallet – Mycellium & Blockchain.info (P2P Model)

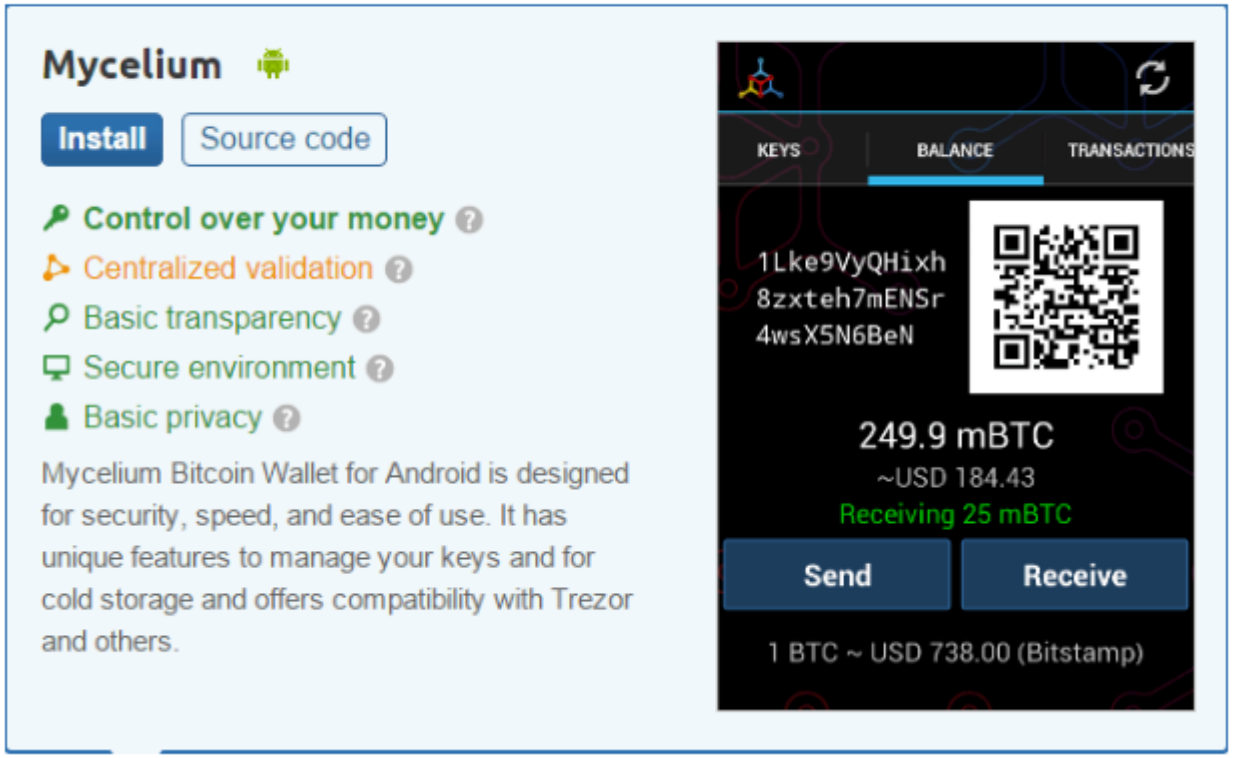
A Bitcoin wallet is a software or application which lets you use the Bitcoin payment network and helps you manage your Bitcoin currency. Just like you need an email application (like "Gmail", "Yahoo" or "Hotmail") to send and receive emails, you need a Bitcoin wallet to send and receive bitcoins.

http://bitcoindaily.org/bitcoin-mobile-wallets

Section 2.1 – Mobile Payment Systems

1. Mobile Bitcoin Wallet – Mycellium & Blockchain.info

Mycelium – Available for Android



https://play.google.com/store/apps/details?id=com.mycelium.wallet
http://satoshicounter.com/2015/07/09/mycelium/

http://bitcoindaily.org/bitcoin-mobile-wallets

Section 2.1 – Mobile Payment Systems

1. Mobile Bitcoin Wallet – Mycellium & Blockchain.info

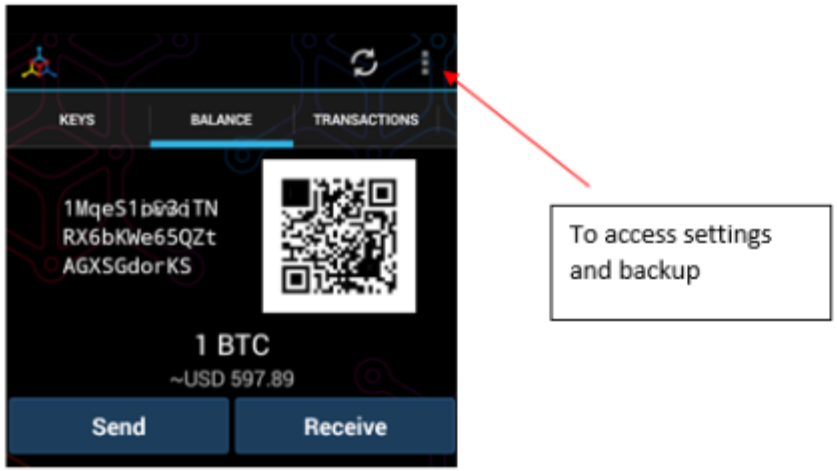
Step 1: create your wallet

https://play.google.com/store/apps/details?id=com.mycelium.wallet&hl=en

Step 2: set-up and secure your wallet

Protect your wallet from theft with a PIN code

- First go to *settings* (found in the top right corner by clicking the vertically stacked three small squares)



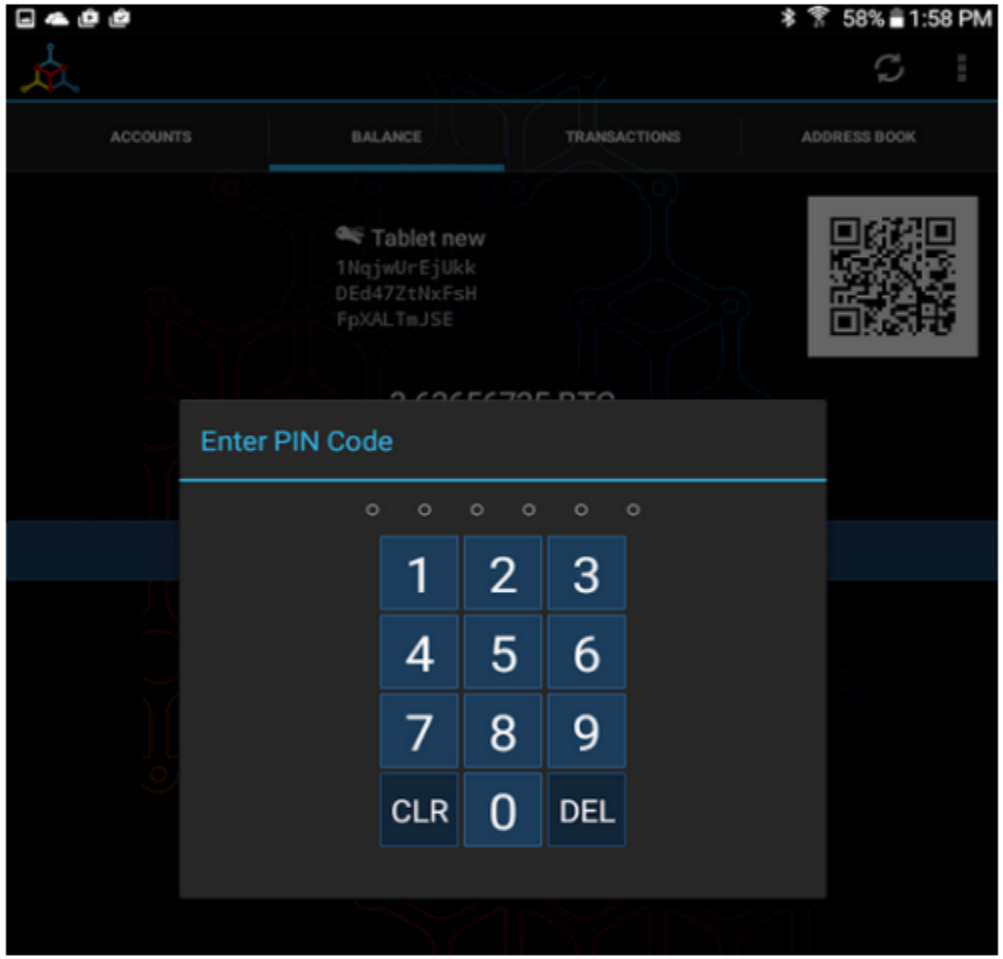
- Click "set pin code"
- Set a pin code and make sure not to forget it, without this code, you cannot spend bitcoin and you cannot access your master recovery seed

http://satoshicounter.com/2015/07/09/mycelium/

Section 2.1 – Mobile Payment Systems

1. Mobile Bitcoin Wallet – Mycellium & Blockchain.info

The next time you want to make a transaction or change important settings, this screen will appear:



http://satoshicounter.com/2015/07/09/mycelium/

Section 2.1 – Mobile Payment Systems

1. Mobile Bitcoin Wallet – Mycellium & Blockchain.info

Step 3: Receive bitcoins - Receiving Payments

Your receiving address QR code

To request a specific amount

To share your bitcoin address via email, SMS, Skype, etc.

1. To receive a payment go to your *Balance* tab and select "Receive"

2. (Optional) Request an amount by clicking on the keyboard icon under "Optional Amount"

3. Share your bitcoin address with the sender by:
   - 3.1 Letting them scan your QR code
   - 3.2 Manually sharing your address with them by using either "Copy to Clipboard" and then pasting it in a message or "Share Bitcoin Address" which will give you various sharing options

4. Wait for the sender to send the funds

5. You can check to make sure you have received the funds by going to the *Transactions* tab

NOTE: you will not be able to send the funds you have just received until you have received one confirmation (approx. 10 minutes)

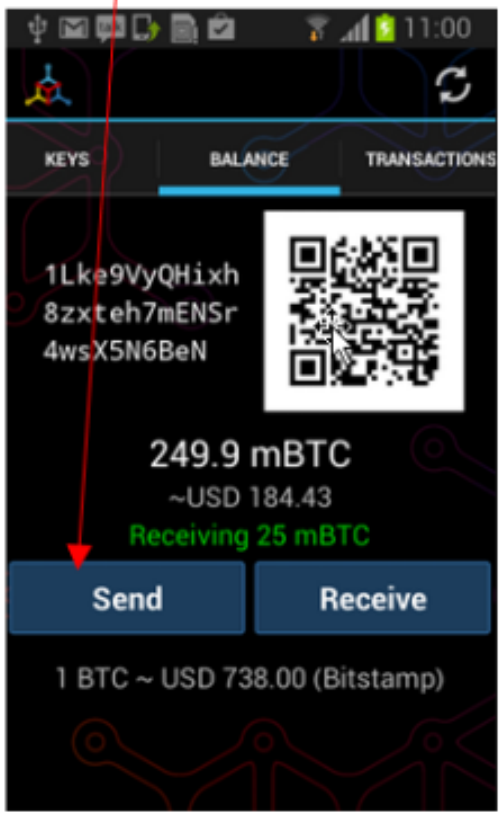
http://satoshicounter.com/2015/07/09/mycelium/

Section 2.1 – Mobile Payment Systems

1. Mobile Bitcoin Wallet – Mycellium & Blockchain.info

Step 4: Sending bitcoins

1. To send a payment go to the *Balance* tab and click "send"



2. Now either
-Scan the recipient's QR code using "Scan QR Code" (this is the simplest and fastest option)
-Paste a copied bitcoin address from your clipboard using "Clipboard"
-Or manually enter their bitcoin address using "Manual Entry"

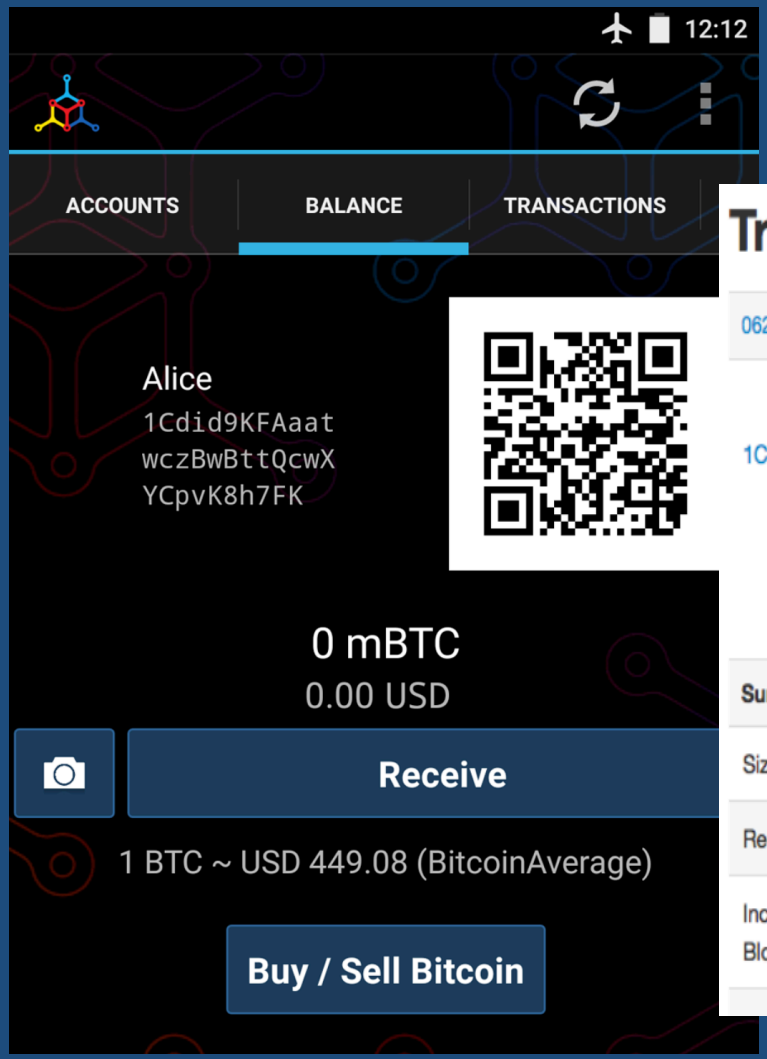
3. Once you have their address, enter the amount you wish to send by clicking on the keyboard icon under "Enter Amount"
-You can choose which currency to enter the amount in (BTC, USD, CAD etc) by clicking on the displayed currency in the top right hand corner

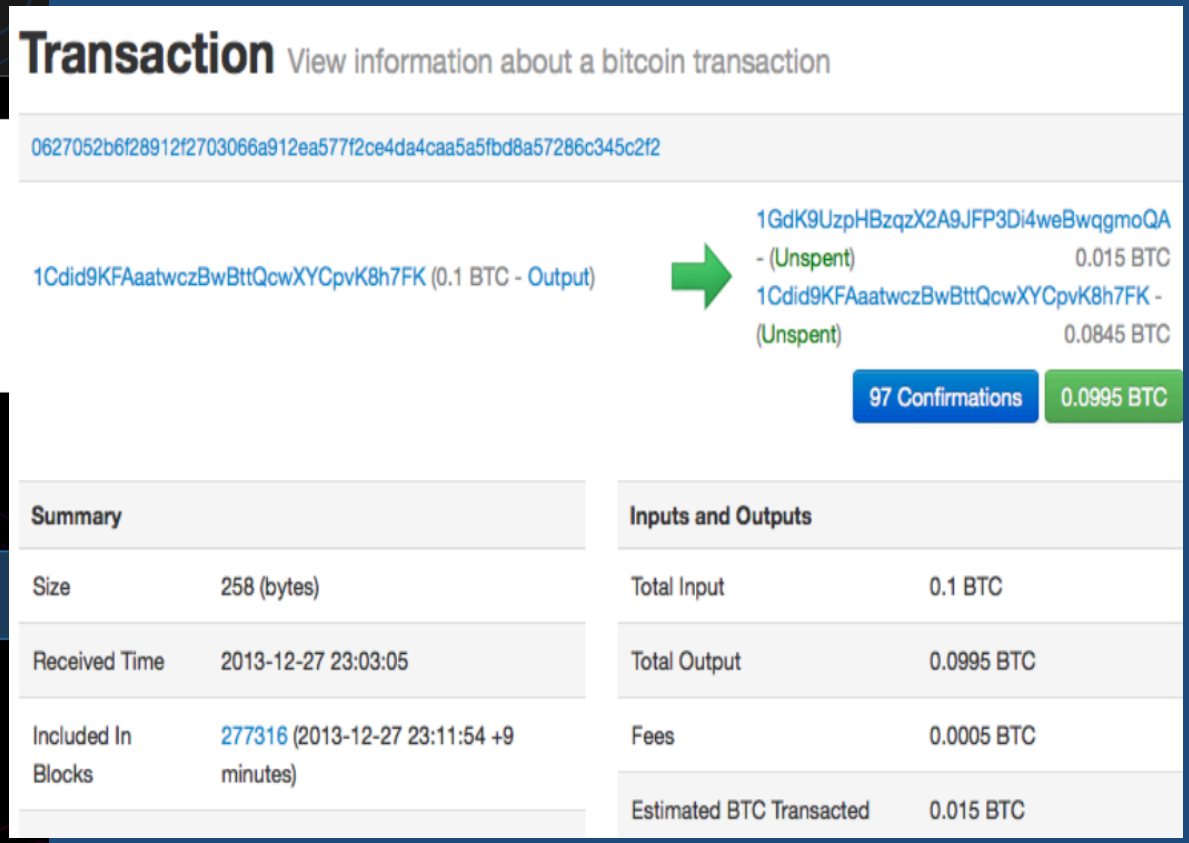
–**Note:** if you are making an online payment, the amount to send will sometimes be automatically entered when scanning the QR code of the merchant. In this case, DO NOT CHANGE the amount. Try to always know from the merchant the amount of Bitcoins he wants to receive, and not the amount of USD (or EUR) he wants to receive.

http://satoshicounter.com/2015/07/09/mycelium/

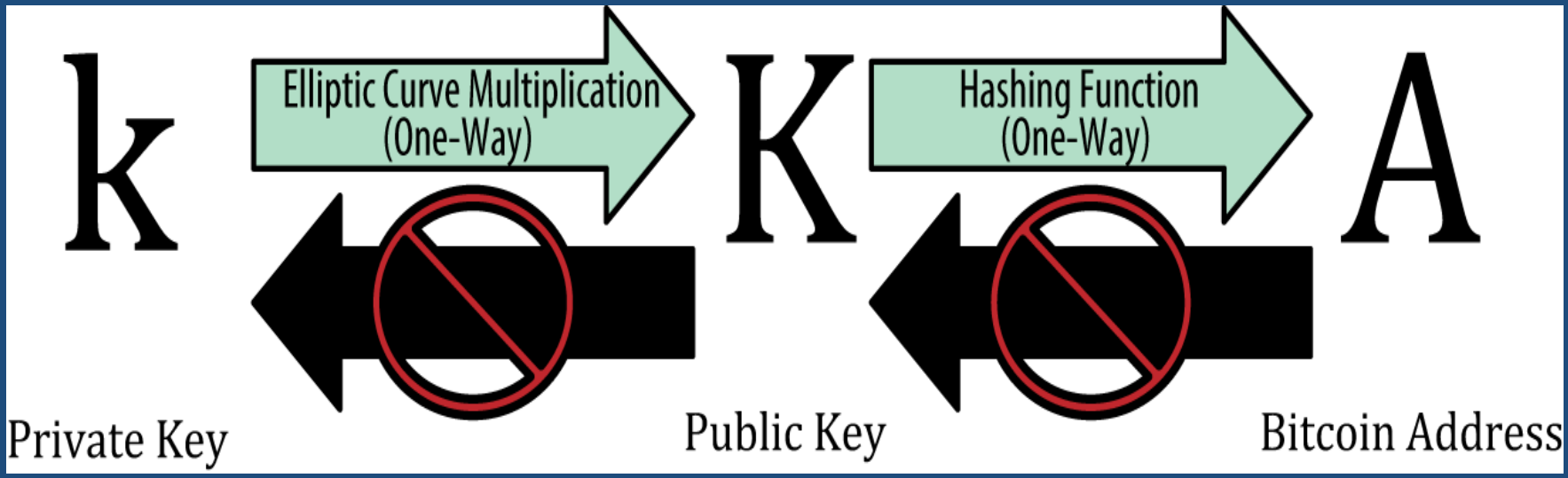
2.1 Bitcoin / Ethereum Blockchain Technology

2.1 Bitcoin / Ethereum Blockchain Technology





2.1 Bitcoin / Ethereum Blockchain Technology - Keys



k = private key is simply a number, picked at random. (e.g. *bitcoin-cli getnewaddress*)

$ bitcoin-cli getnewaddress
1J7mdg5rbQyUHENYdx39WVWK7fsLpEoXZy

$ bitcoin-cli dumpprivkey
1J7mdg5rbQyUHENYdx39WVWK7fsLpEoXZy
KxFC1jmwwCoACiCAWZ3eXa96mBM6tb3TYzGmf6YwgdGWZgawvrtJ

2.1 Bitcoin / Ethereum Blockchain Technology - Keys

K = public key is calculated from the private key using elliptic curve multiplication, which is irreversible: K = k * G, where k is the private key, G is a constant point called the generator point, and K is the resulting public key.

Bitcoin/Ethereum uses a specific elliptic curve and set of mathematical constants, as defined in a standard called ***secp256k1****, established by the National Institute of Standards and Technology (NIST).*

The ***secp256k1*** *curve is defined by the following function, which produces an elliptic curve:*

$$y^2 = (x^3 + 7) \text{ over } (\mathbb{F}_p)$$

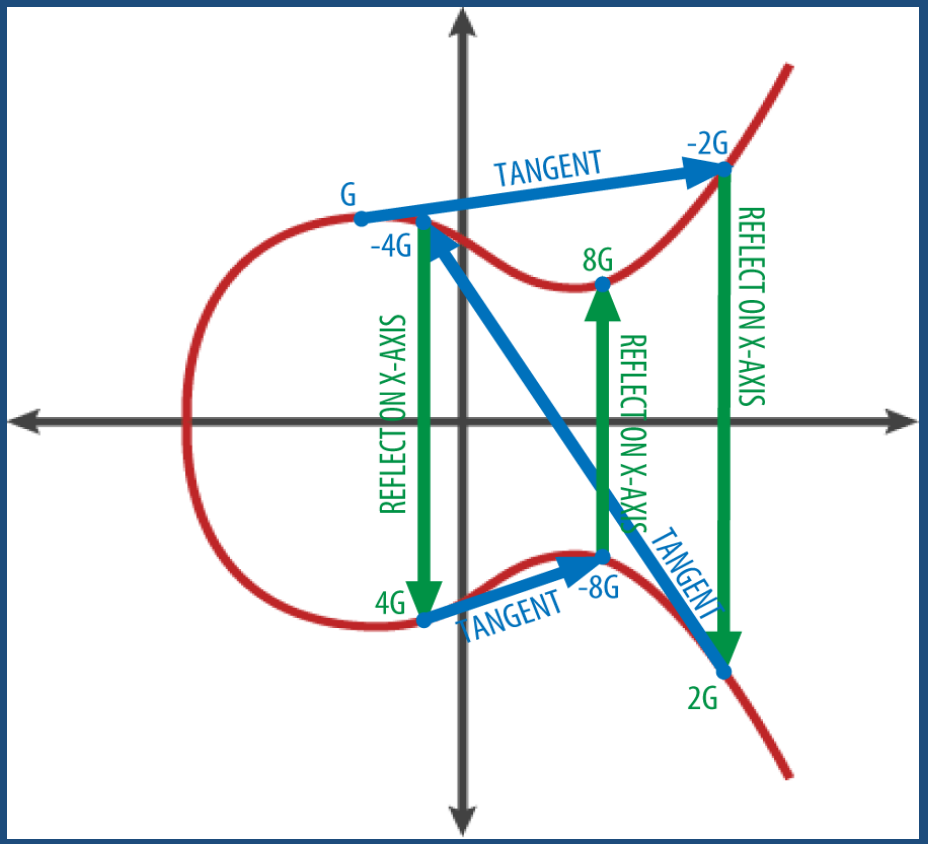
or

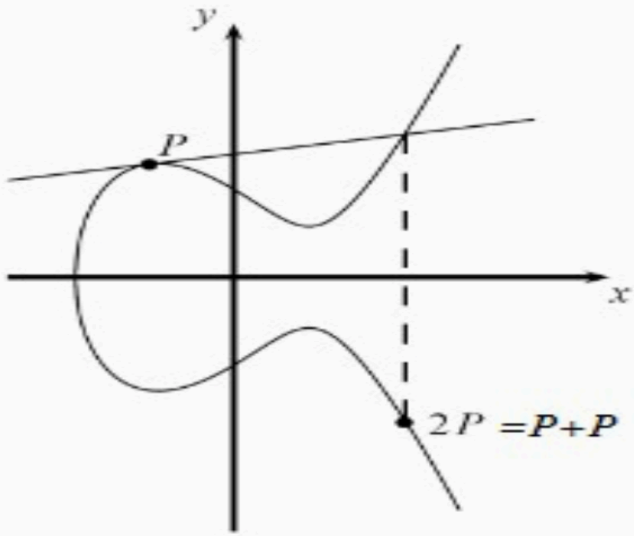
$$y^2 \bmod p = (x^3 + 7) \bmod p$$

The *mod p* (modulo prime number p) indicates that this curve is over a finite field of prime order p, also written as $\mathbb{F}_p$, where $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$, a very large prime number.

2.1 Bitcoin / Ethereum Blockchain Technology – Addresses & Keys

Most bitcoin/ETH implementations use the OpenSSL cryptographic library to do the elliptic curve math. For example, to derive the public key, the function EC_POINT_mul() is used.





Graph representing point doubling over real numbers

In the case of point doubling, the equation becomes:

$$x_3 = s^2 - x_1 - x_2 \ mod \ p$$

$$y_3 = s(x_1 - x_3) - y_1 \ mod \ p$$

$$S = \frac{3x_1^2 + a}{2y_1}$$

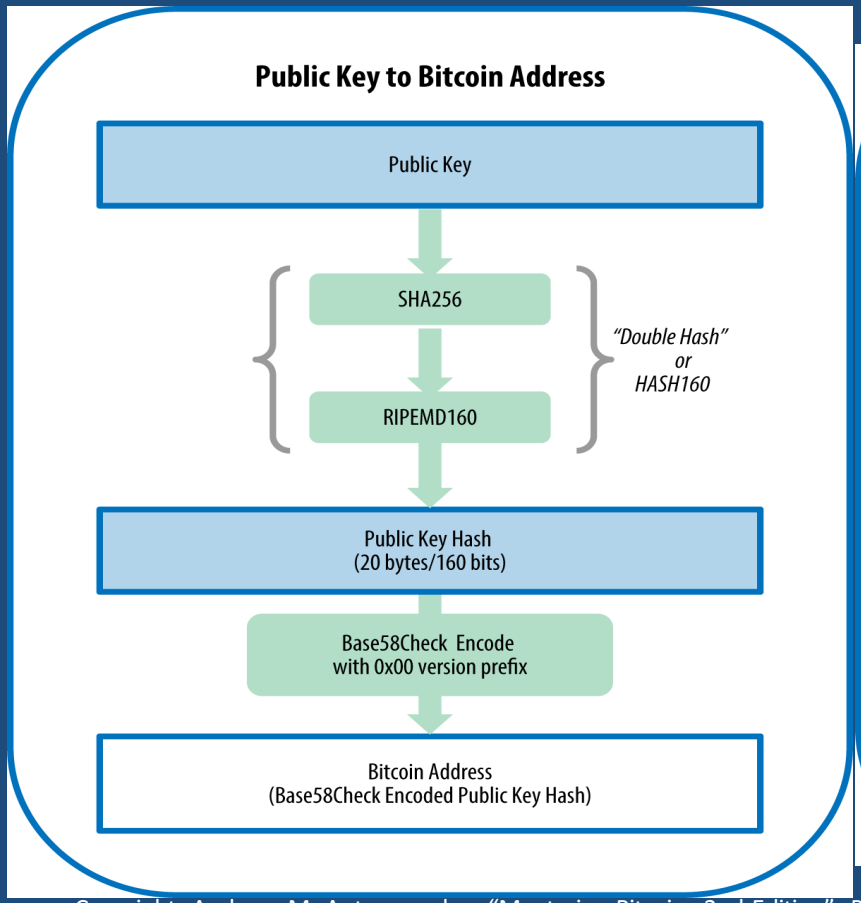
2.1 Bitcoin / Ethereum Blockchain Technology – Addresses & Keys

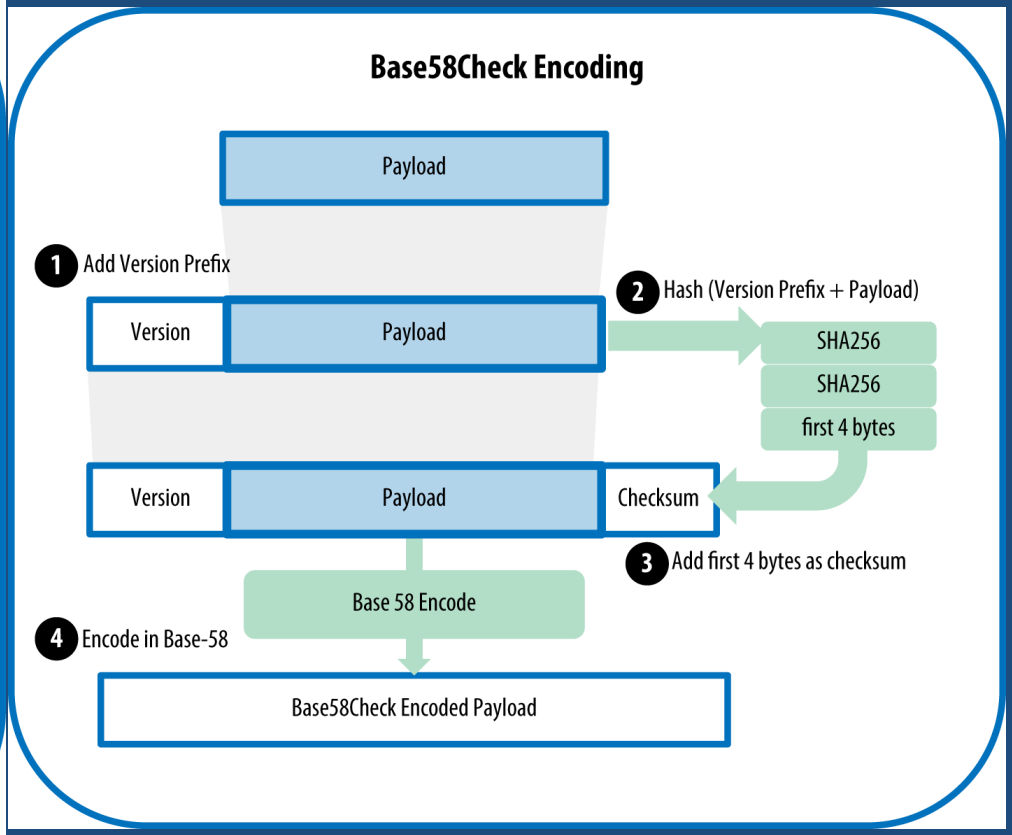
Starting with the public key K, we compute the SHA256 hash and then compute the RIPEMD160 hash of the result, producing a 160-bit (20-byte) number:

$$A = RIPEMD160(SHA256(K))$$

where ***K*** is the public key and ***A*** is the resulting bitcoin address.

Public key to bitcoin address: conversion of a public key into a bitcoin address:

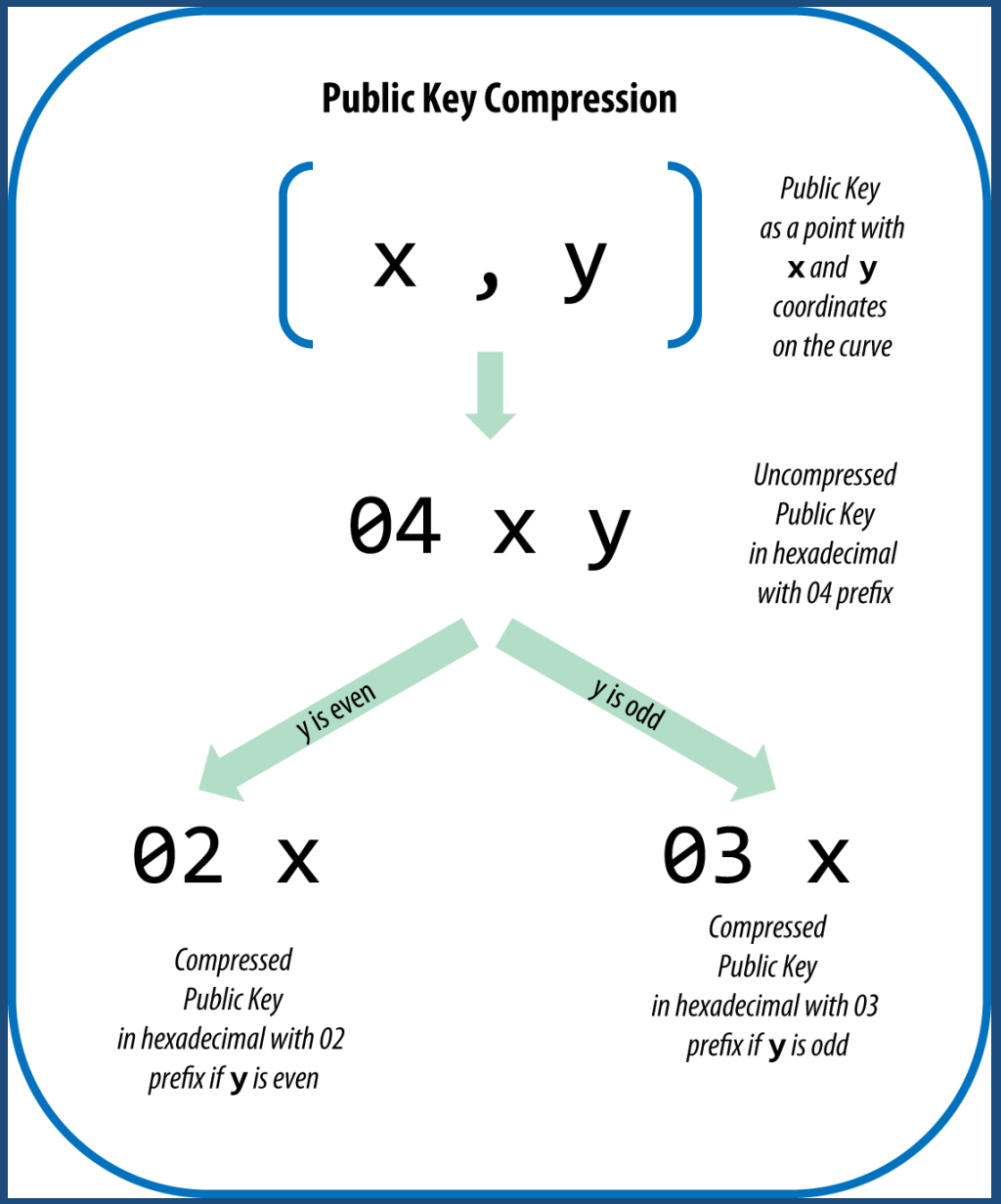




2.1 Bitcoin / Ethereum Blockchain Technology – Addresses & Keys

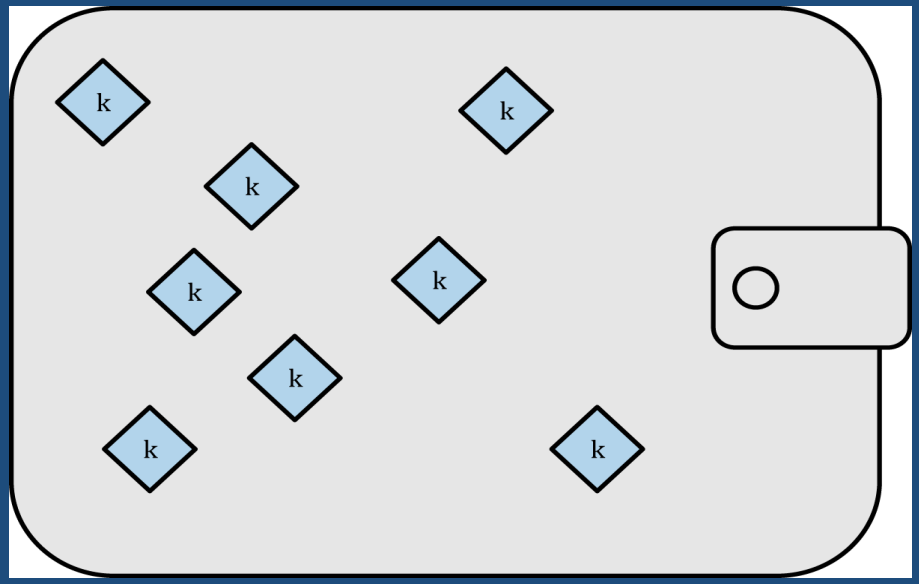
*A public key is a point **(x,y)** on an elliptic curve. Because the curve expresses a mathematical function, a point on the curve represents a solution to the equation and, therefore, if we know the x coordinate we can calculate the y coordinate by solving the equation **y^2 mod p = (x^3 + 7) mod p**.*

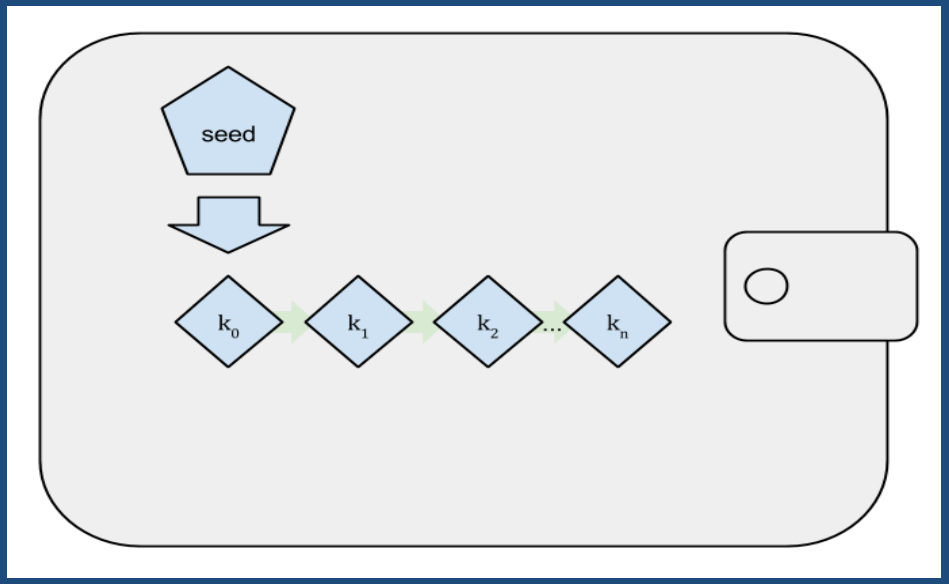
That allows us to store only the x coordinate of the public key point, omitting the y coordinate and reducing the size of the key and the space required to store it by 256 bits. An almost 50% reduction in size in every transaction adds up to a lot of data saved over time!



2.1 Bitcoin / Ethereum Blockchain Technology - Wallets

A common misconception about bitcoin is that bitcoin wallets contain bitcoin. In fact, the wallet contains only keys. The “coins” are recorded in the blockchain on the bitcoin network. Users control the coins on the network by signing transactions with the keys in their wallets. In a sense, a bitcoin wallet is a keychain.





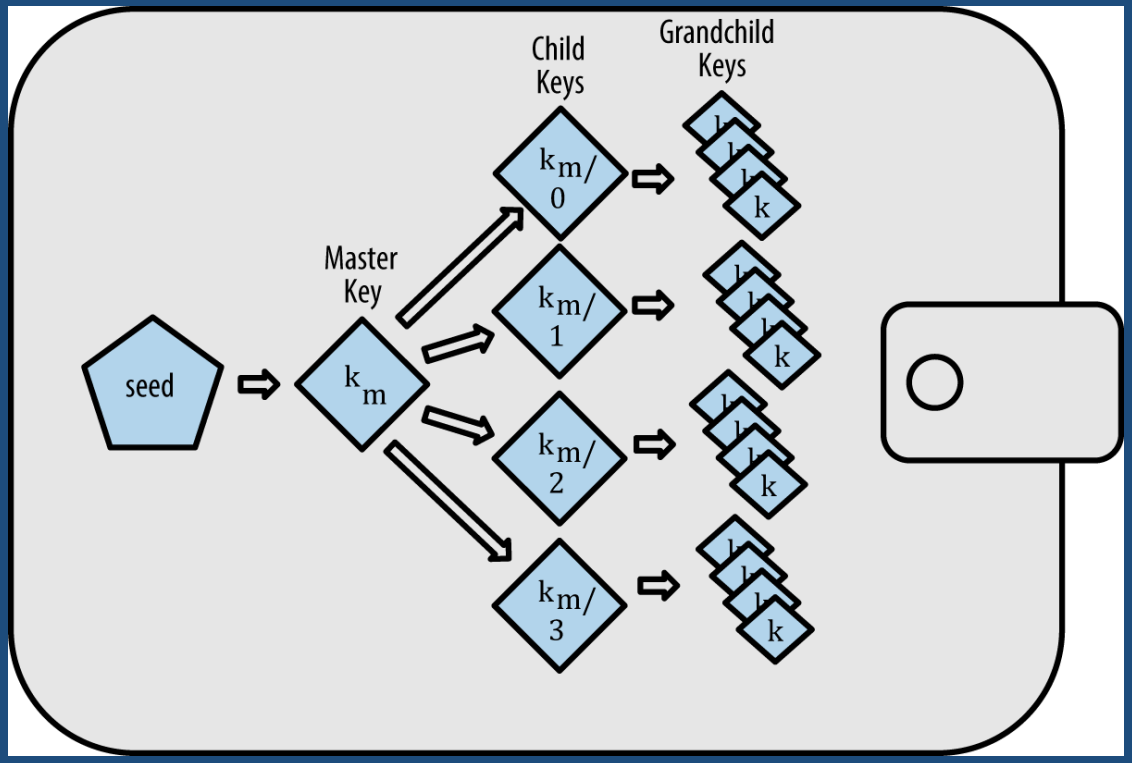
The first type is a nondeterministic wallet, where each key is independently generated from a random number. The keys are not related to each other. This type of wallet is also known as a JBOK wallet from the phrase “Just a Bunch Of Keys.”

Type-0 nondeterministic (random) wallet: a collection of randomly generated keys

The second type of wallet is a deterministic wallet, where all the keys are derived from a single master key, known as the seed. All the keys in this type of wallet are related to each other and can be generated again if one has the original seed.

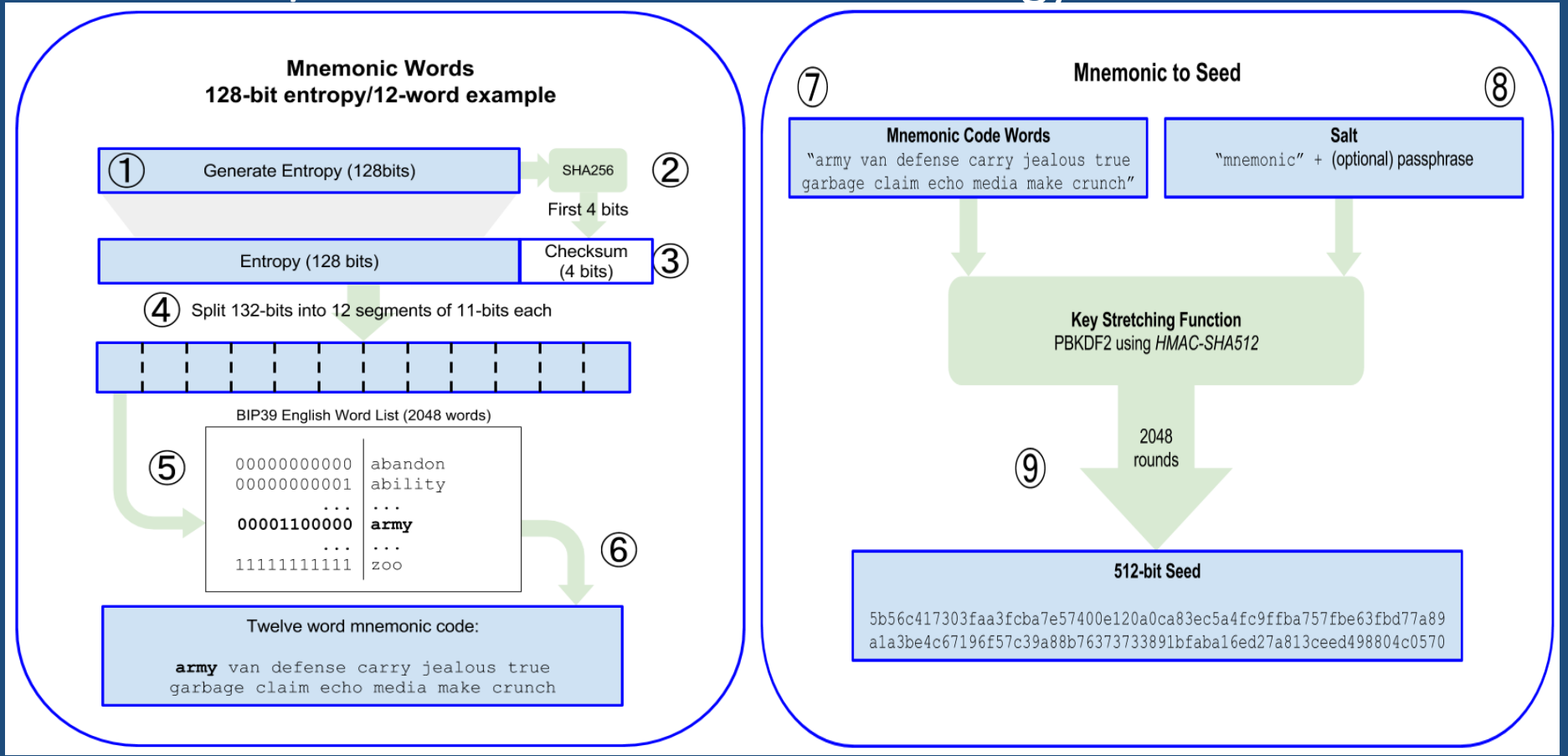
Type-1 deterministic (seeded) wallet: a deterministic sequence of keys derived from a seed

2.1 Bitcoin / Ethereum Blockchain Technology - Wallets



Type-2 HD (*Hierarchical Deterministic*) Wallet: a tree of keys generated from a single seed
HD Wallets (Bitcoin Improvement Proposal: BIP-32/BIP-44)

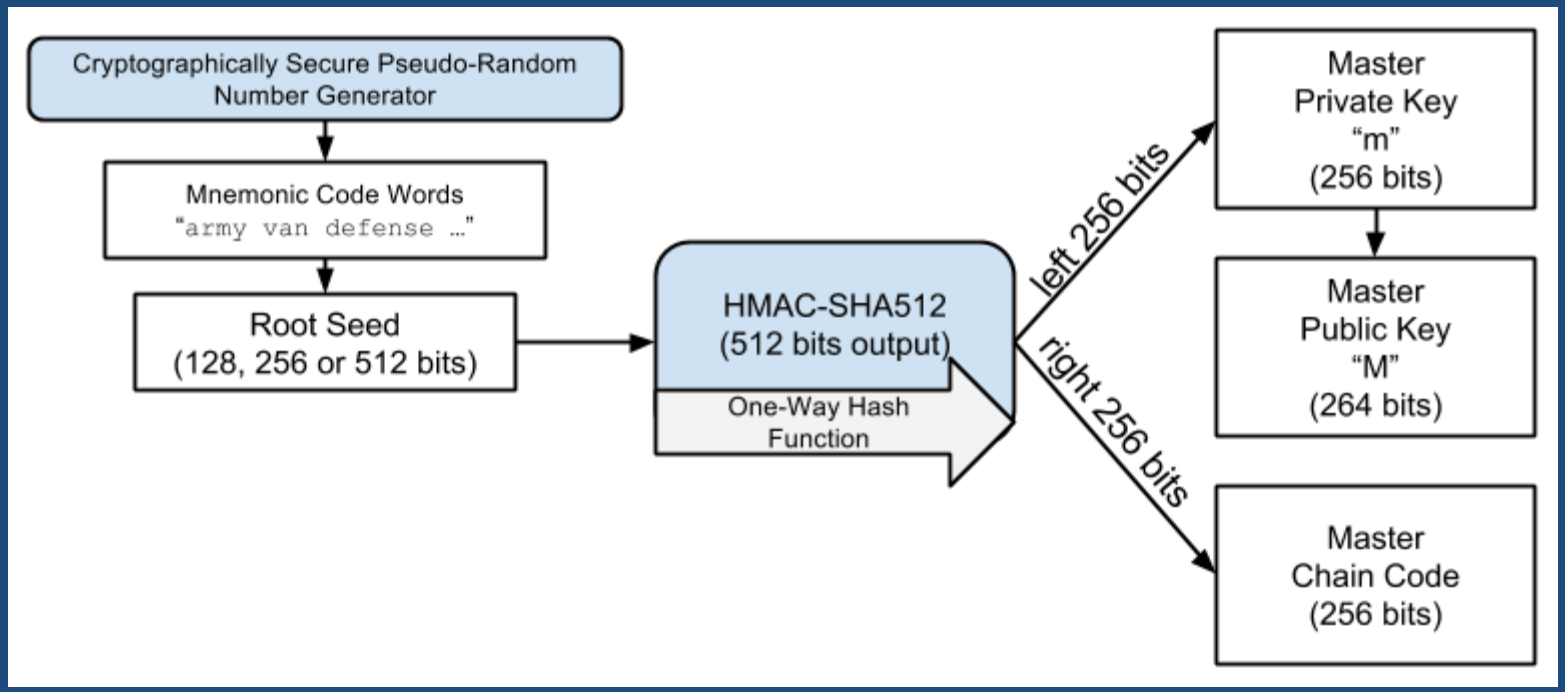
2.1 Bitcoin / Ethereum Blockchain Technology – HD Wallets



Generating entropy and encoding as mnemonic words

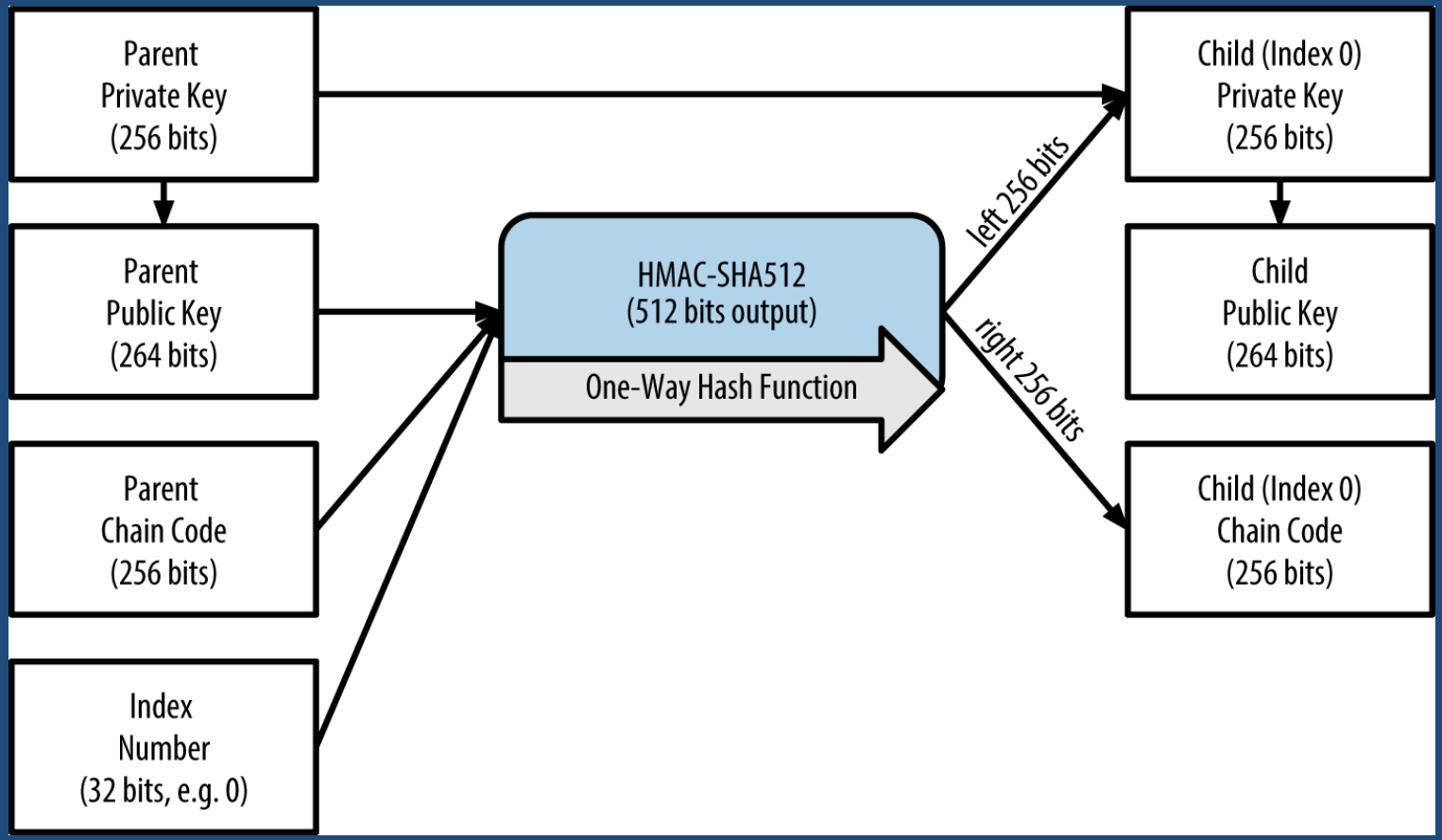
2.1 Bitcoin / Ethereum Blockchain Technology – HD Wallets

Creating an HD Wallet from the Seed HD wallets are created from a single root seed, which is a 128-, 256-, or 512-bit random number. Most commonly, this seed is generated from a mnemonic:



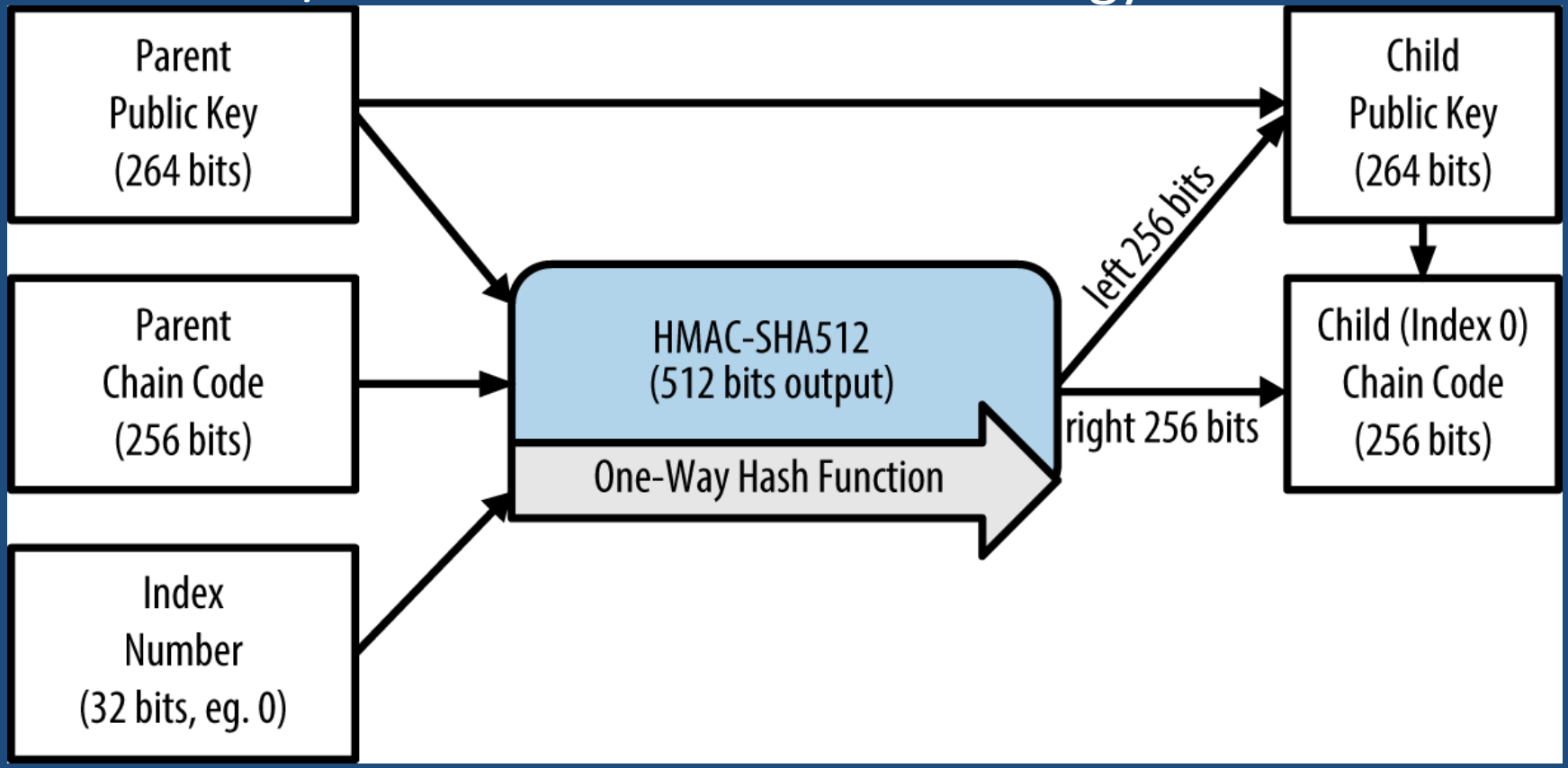
Creating master keys and chain code from a root seed

2.1 Bitcoin / Ethereum Blockchain Technology – HD Wallets



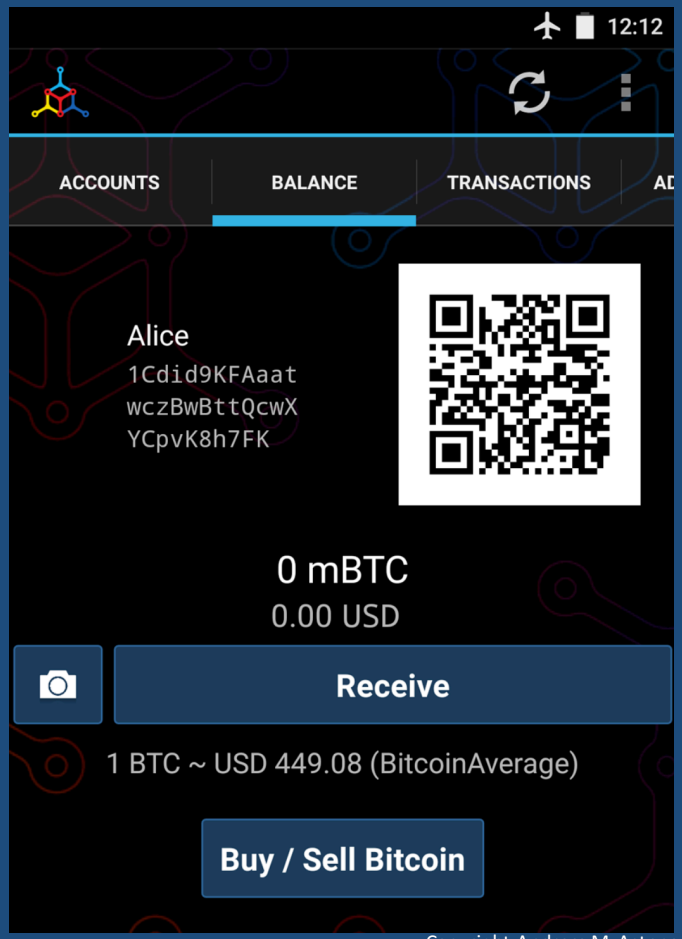
Extending a parent private key to create a child private key

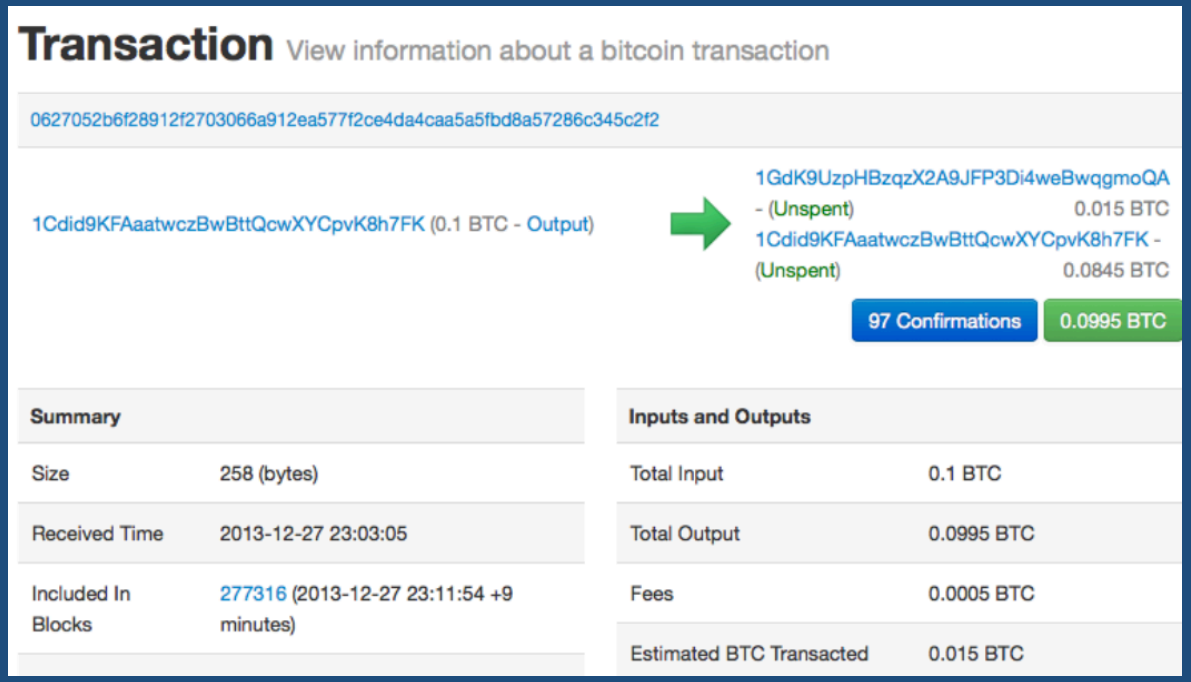
2.1 Bitcoin / Ethereum Blockchain Technology – HD Wallets



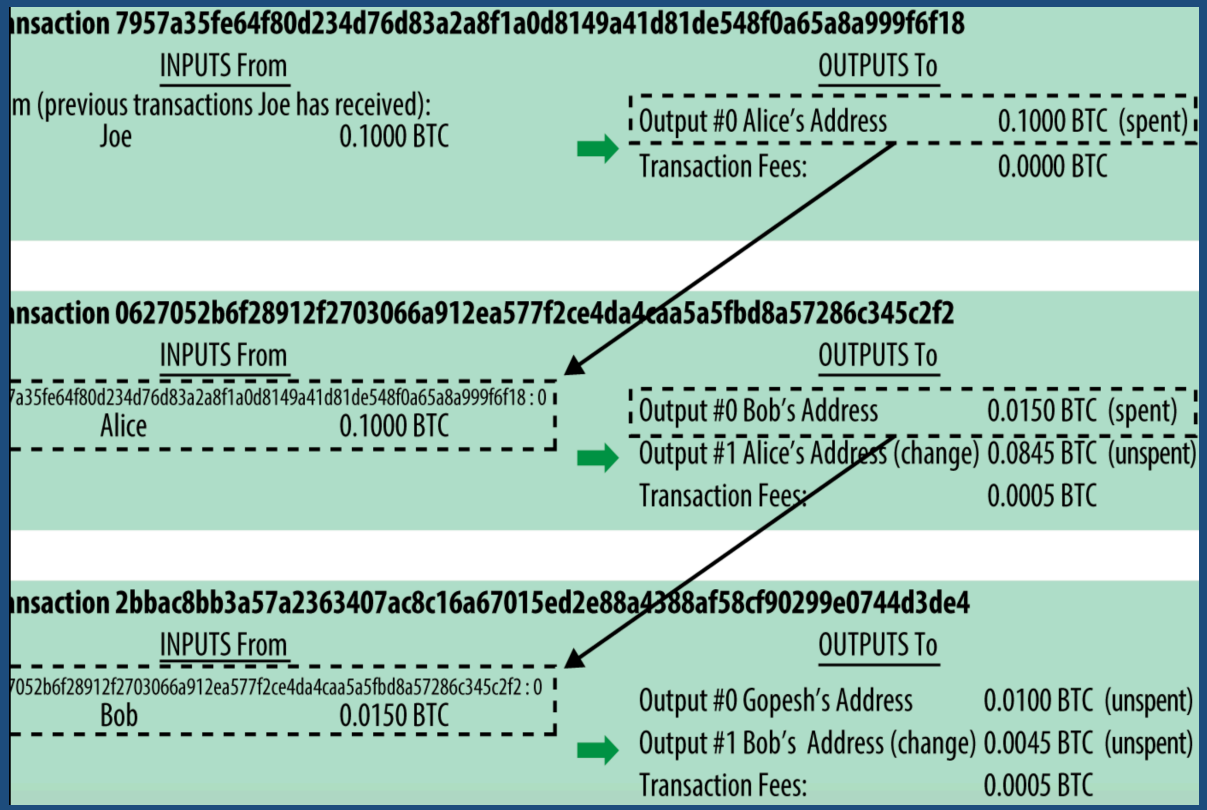
Extending a parent public key to create a child public key

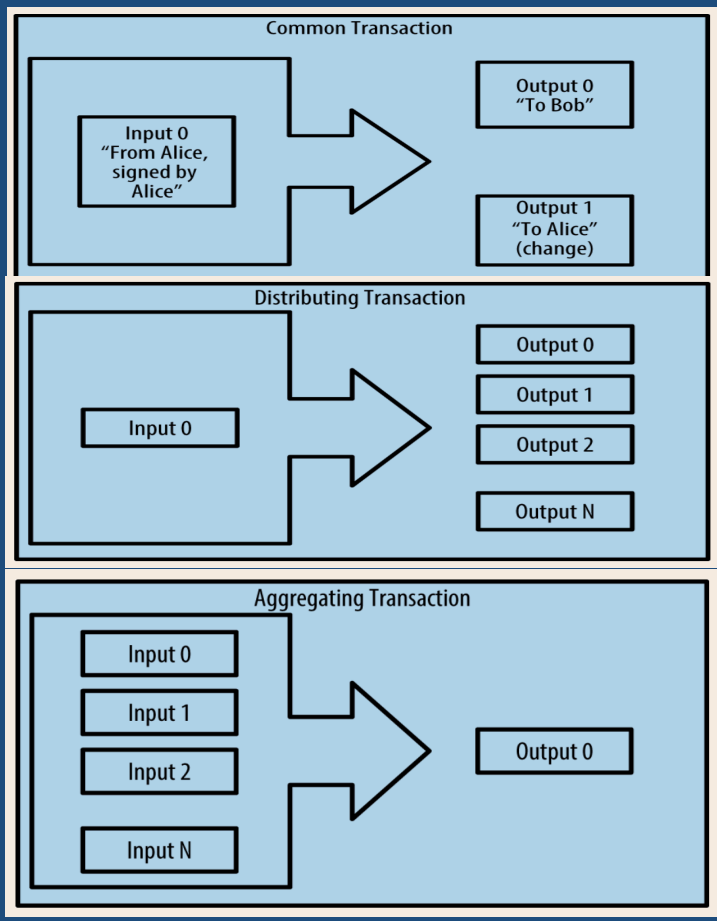
2.1 Bitcoin / Ethereum Blockchain Technology - Transactions



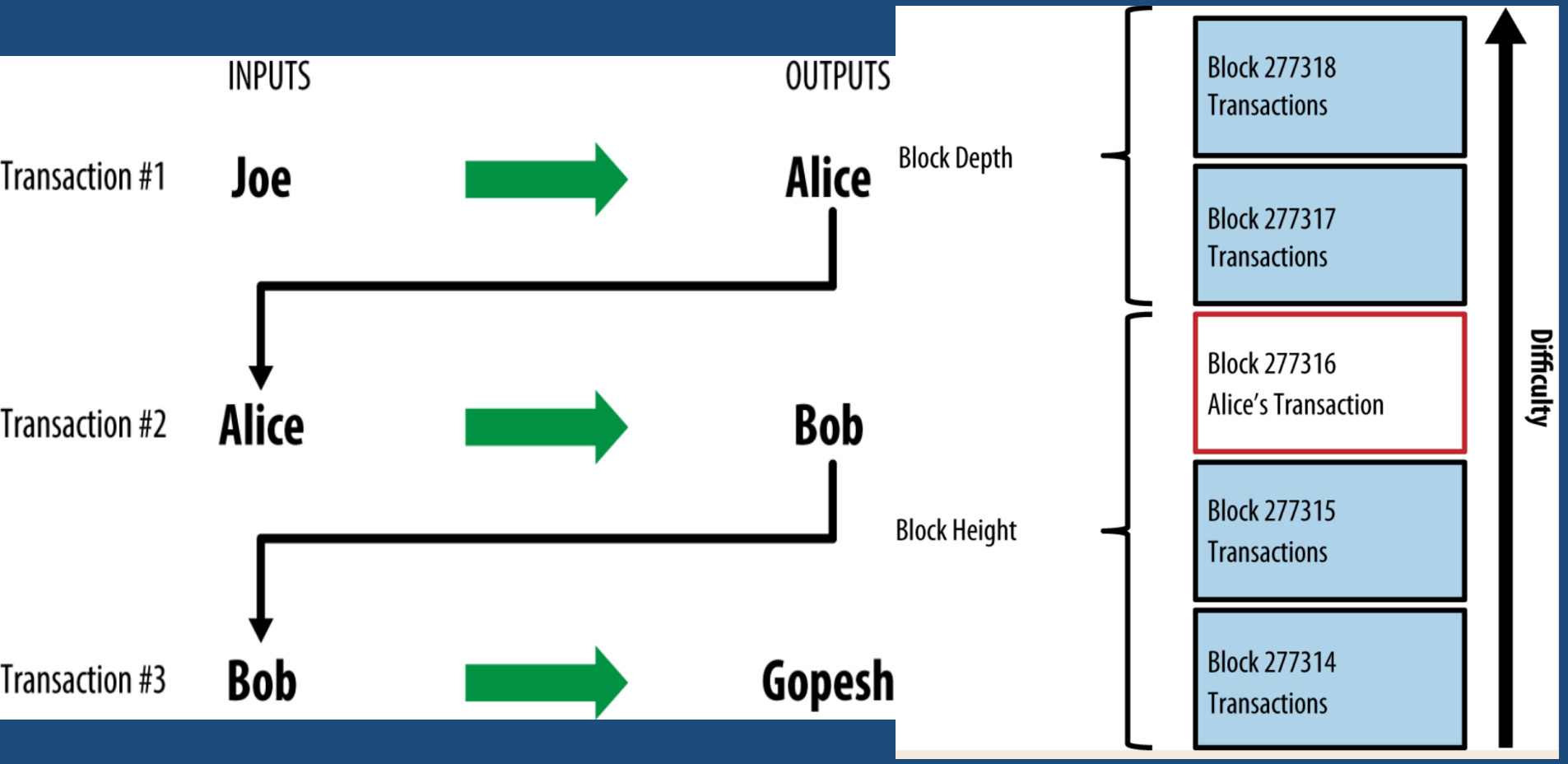


2.1 Bitcoin / Ethereum Blockchain Technology - Transactions



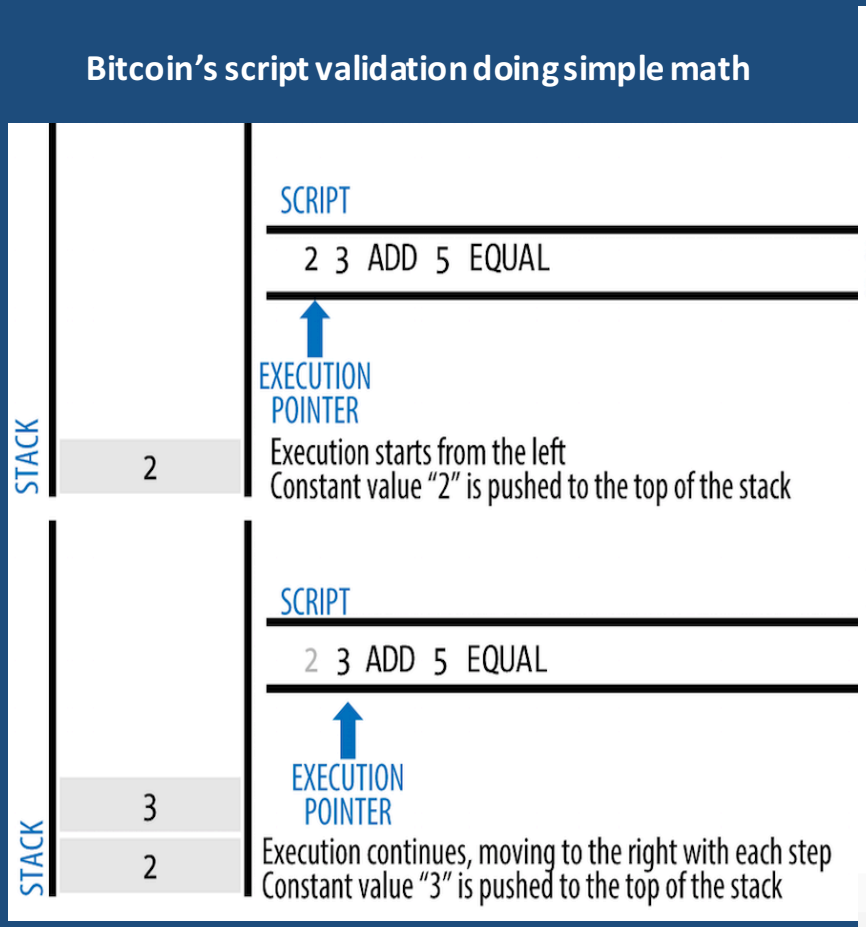


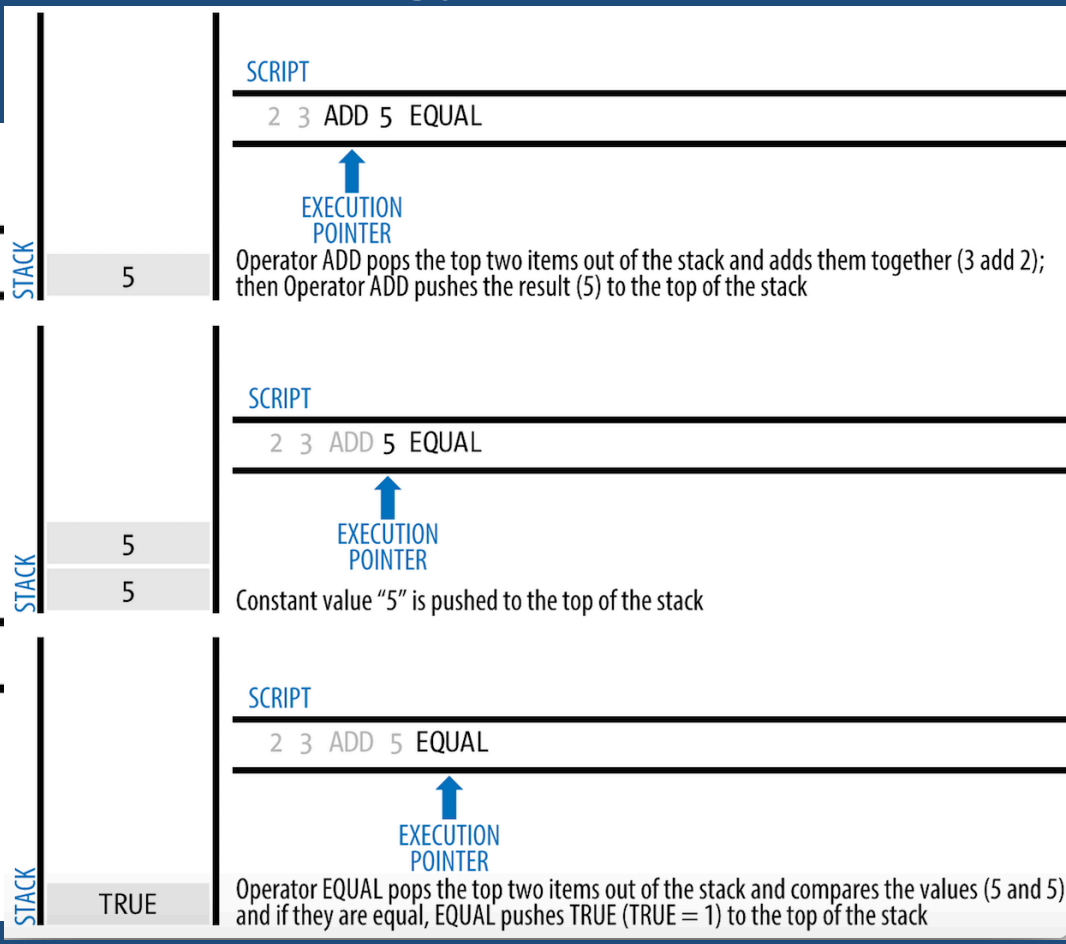
2.1 Bitcoin / Ethereum Blockchain Technology - Transactions



2.1 Bitcoin / Ethereum Blockchain Technology - Transactions

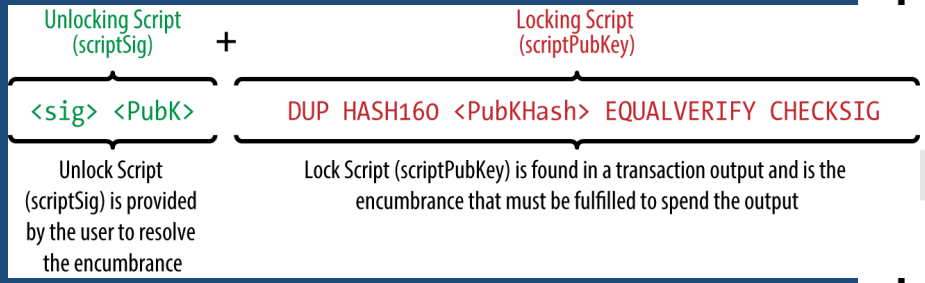
Bitcoin's script validation doing simple math

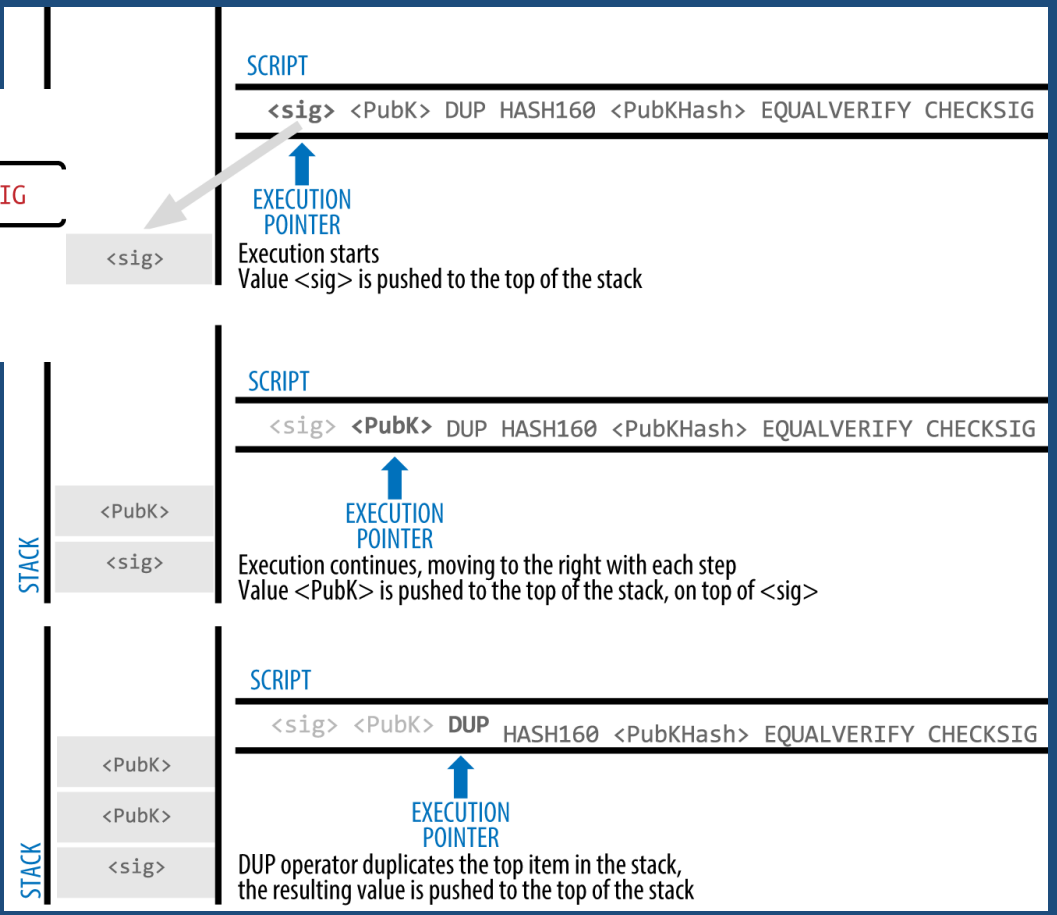




2.1 Bitcoin / Ethereum Blockchain Technology - Transactions

Evaluating a script for a P2PKH (Pay-to-Public-Key-Hash) transaction



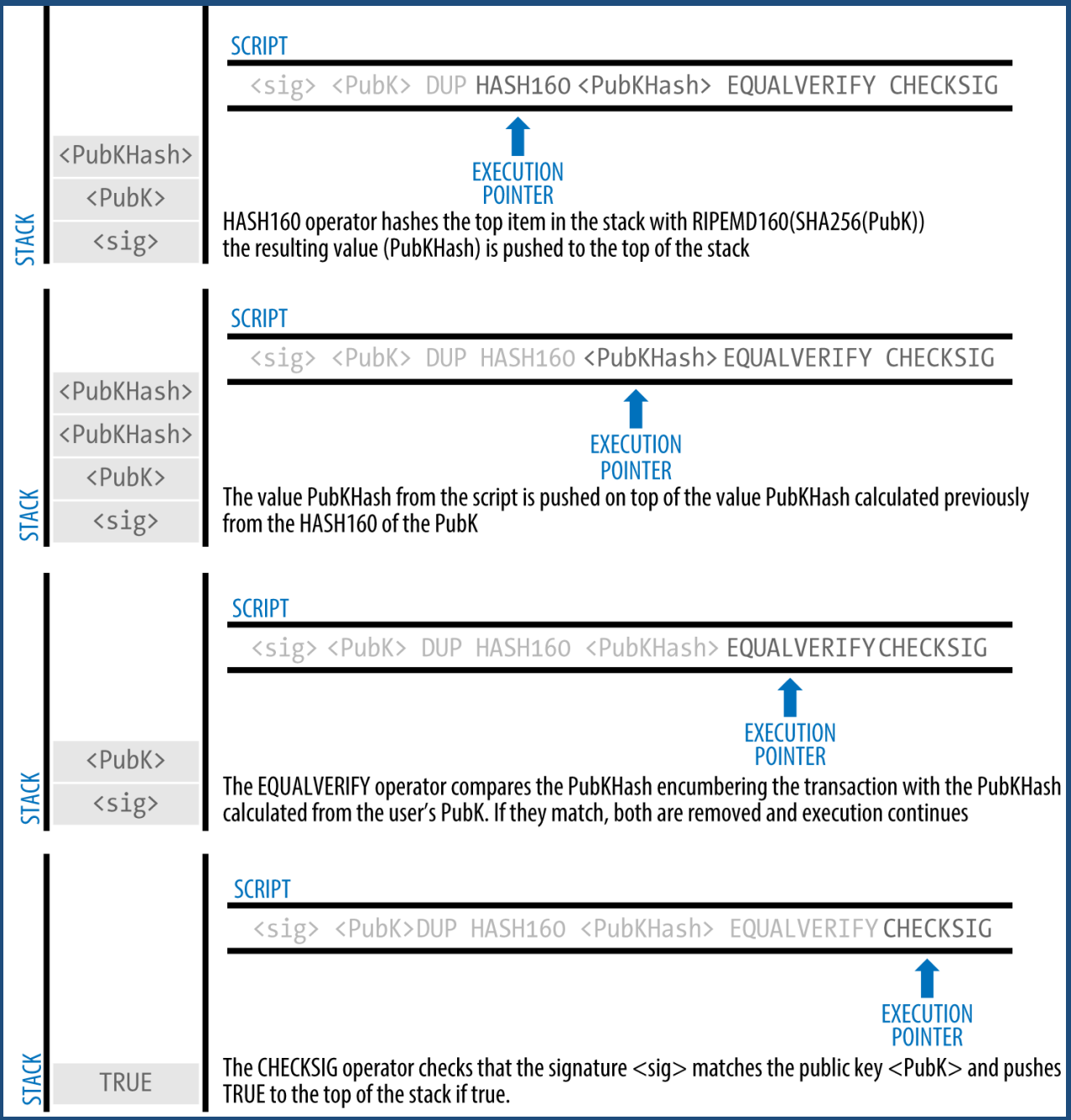


0100000001**186f9f998a5aa6f048e51dd8419a14d8a0f1a8a2836dd734d2804fe65fa35779000000008b483045022100884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb02204b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e381301410484ecc0d46f1918b30928fa0e4ed99f16a0fb4fde0735e7ade8416ab9fe423cc5412336376789d172787ec3457eee41c04f4938de5cc17b4a10fa336a8d752adf**ffffffff0260e31600000000001976a914ab68025513c3dbd2f7b92a94e0581f5d50f654e788acd0ef800000000001976a9147f9b1a7fb68d60c536c2fd8aeaa53a8f3cc025a888ac00000 000

2.1 Bitcoin / Ethereum Blockchain Technology - Transactions

Pay-to-Public-Key-Hash (P2PKH)

The vast majority of transactions processed on the bitcoin network spend outputs locked with a Pay-to-Public-Key-Hash or "P2PKH" script. These outputs contain a locking script that locks the output to a public key hash, more commonly known as a bitcoin address. An output locked by a P2PKH script can be unlocked (spent) by presenting a public key and a digital signature created by the corresponding private key



2.1 Bitcoin / Ethereum Blockchain Technology – Transactions/Signature

Signature = implementation of the ECDSA algorithm; the "message" being signed is the transaction, or more accurately a hash of a specific subset of the data in the transaction. The signing key is the user's private key. The result is the Bitcoin/Ethereum signature:

$$Sig = F_{sig}(F_{hash}(m),\ dA)$$

where:

- dA is the signing private key
- m is the transaction (or parts of it)
- F_{hash} is the hashing function
- F_{sig} is the signing algorithm
- Sig is the resulting signature

Antonopoulos

In Ethereum's implementation of ECDSA, the "message" being signed is the transaction, or more accurately, the Keccak256 hash of the RLP-encoded data from the transaction. The signing key is the EOA's private key. The result is the signature:

$$Sig = F_{sig}(F_{keccak256}(m),\ k)$$

where:

- k is the signing private key
- m is the RLP-encoded transaction
- $F_{keccak256}$ is the Keccak256 hash function
- F_{sig} is the signing algorithm
- Sig is the resulting signature

2.1 Bitcoin / Ethereum Blockchain Technology – Transactions/Signature

The function F_{sig} produces a signature `Sig` that is composed of two values, commonly referred to as `R` and `S`:

```
Sig = (R, S)
```

Now that the two values `R` and `S` have been calculated, they are serialized into a byte-stream using an international standard encoding scheme called the *Distinguished Encoding Rules*, or *DER*.

- `0x30` — indicating the start of a DER sequence
- `0x45` — the length of the sequence (69 bytes)
- `0x02` — an integer value follows
- `0x21` — the length of the integer (33 bytes)
- `R` — `00884d142d86652a3f47ba4746ec719bbfbd040a570b1deccbb6498c75c4ae24cb`
- `0x02` — another integer follows
- `0x20` — the length of the integer (32 bytes)
- `S` — `4b9f039ff08df09cbe9f6addac960298cad530a863ea8f53982c09db8f6e3813`
- A suffix (`0x01`) indicating the type of hash used (`SIGHASH_ALL`)

2.1 Bitcoin / Ethereum Blockchain Technology – Transactions/Block

- The blockchain data structure is an ordered, back-linked list of blocks of transactions.

- The blockchain can be stored as a flat file, or in a simple database.

- A block is a container data structure that aggregates transactions for inclusion in the public ledger, the blockchain.

- The block header is 80 bytes, whereas
- the average transaction is at least 400 bytes and the average block contains more than 1900 transactions.

- A complete block, with all transactions, is therefore 10,000 times larger than the block header.

2.1 Bitcoin / Ethereum Blockchain Technology – Blockchain

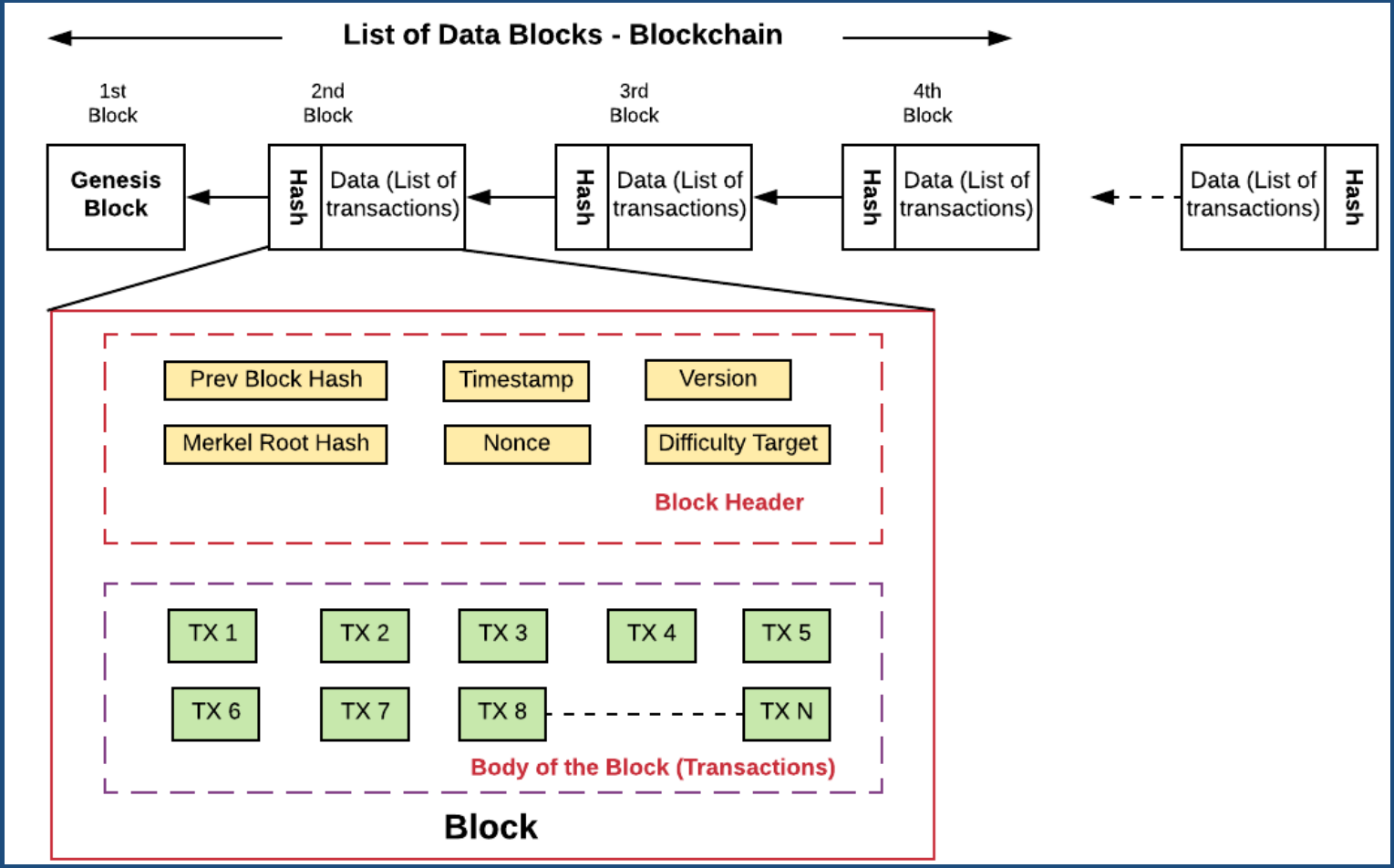
A. The block structure

Size	Field	Description
4 bytes	Block Size	The size of the block, in bytes, following this field
80 bytes	Block Header	Several fields form the block header
1–9 bytes (VarInt)	Transaction Counter	How many transactions follow
Variable	Transactions	The transactions recorded in this block

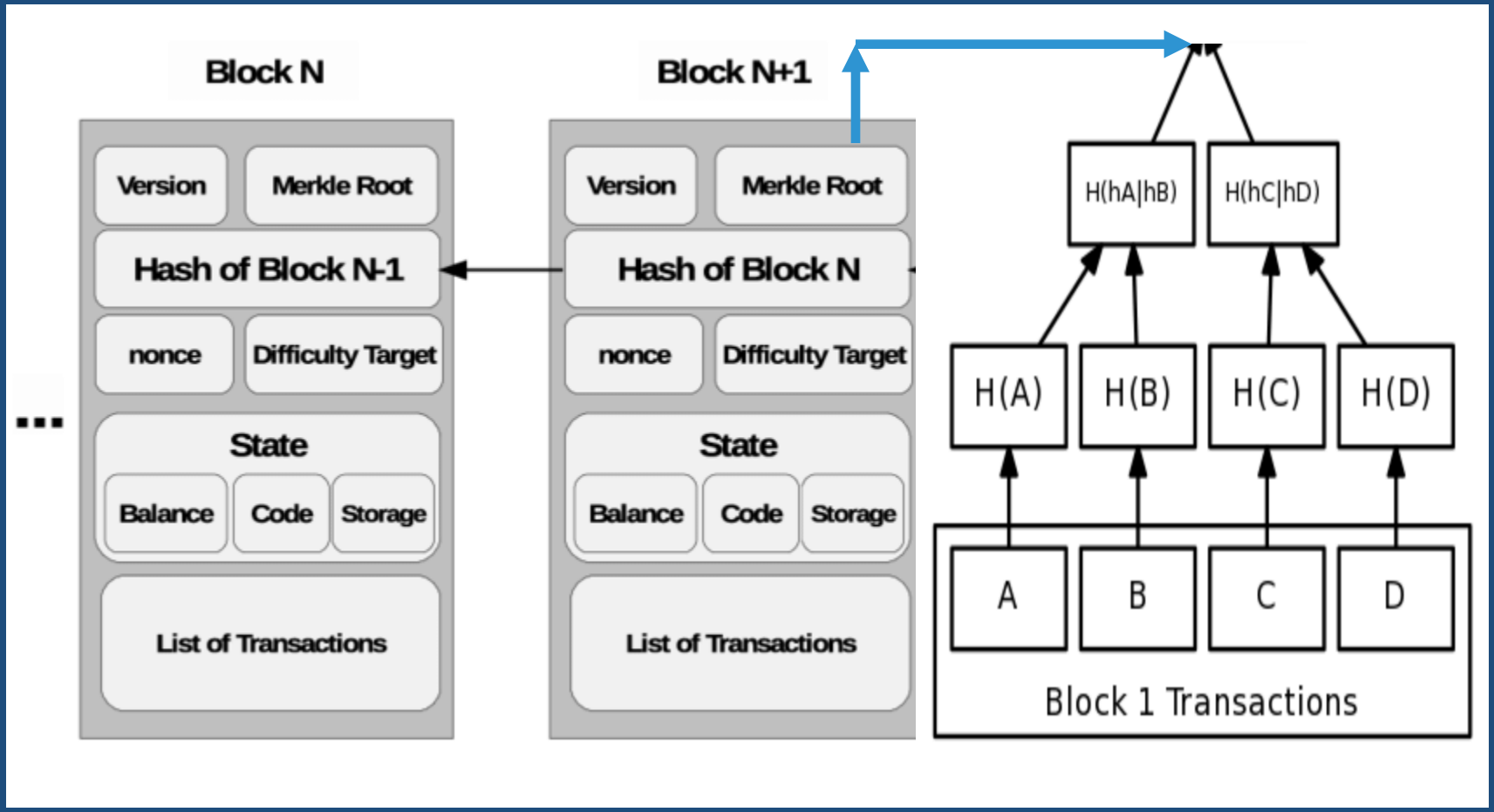
B. The Block Header structure

Size	Field	Description
4 bytes	Version	A version number to track software/protocol upgrades
32 bytes	Previous Block Hash	A reference to the hash of the previous (parent) block in the chain
32 bytes	Merkle Root	A hash of the root of the merkle tree of this block's transactions
4 bytes	Timestamp	The approximate creation time of this block (seconds from Unix Epoch)
4 bytes	Difficulty Target	The Proof-of-Work algorithm difficulty target for this block
4 bytes	Nonce	A counter used for the Proof-of-Work algorithm

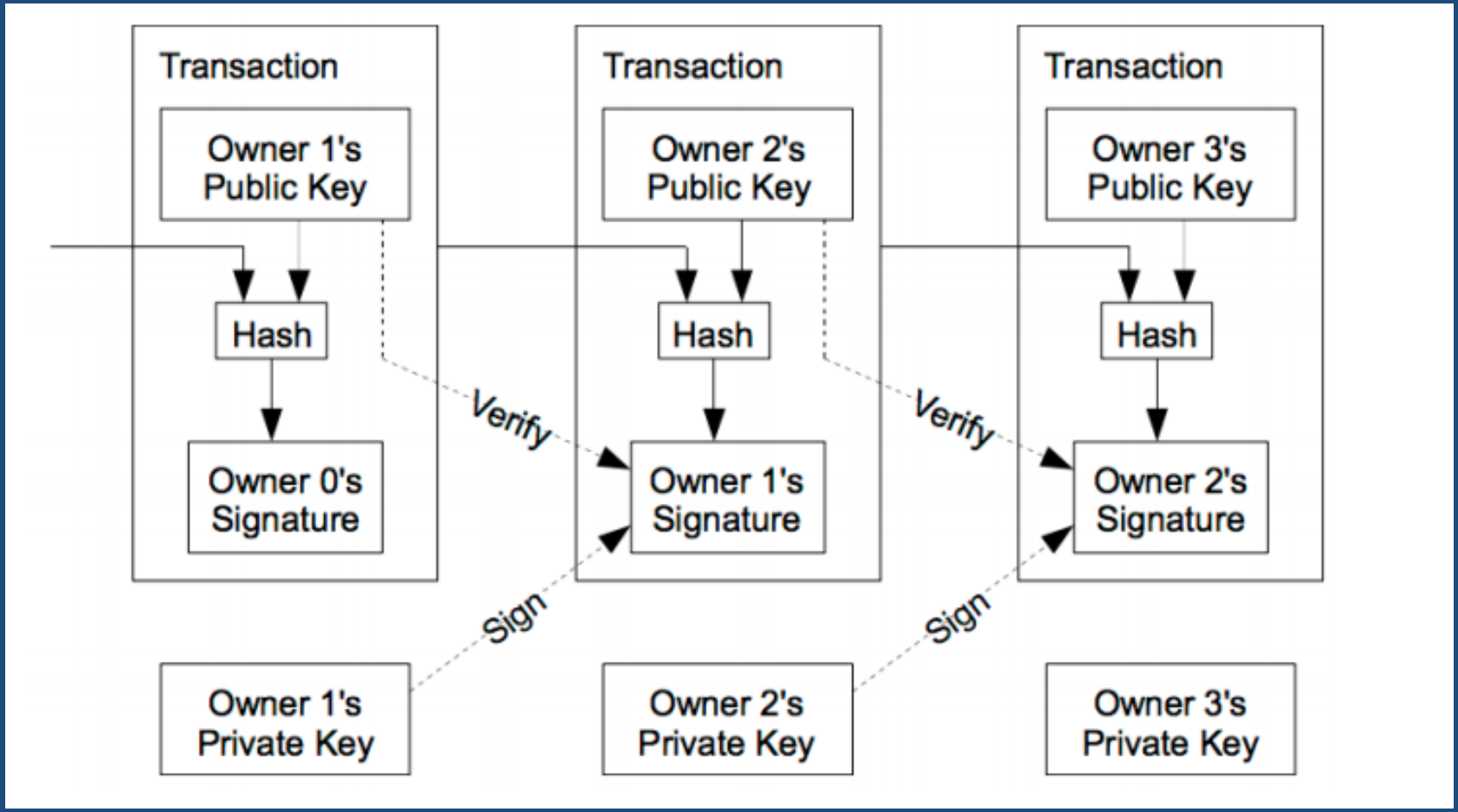
2.1 Bitcoin / Ethereum Blockchain Technology – Blockchain



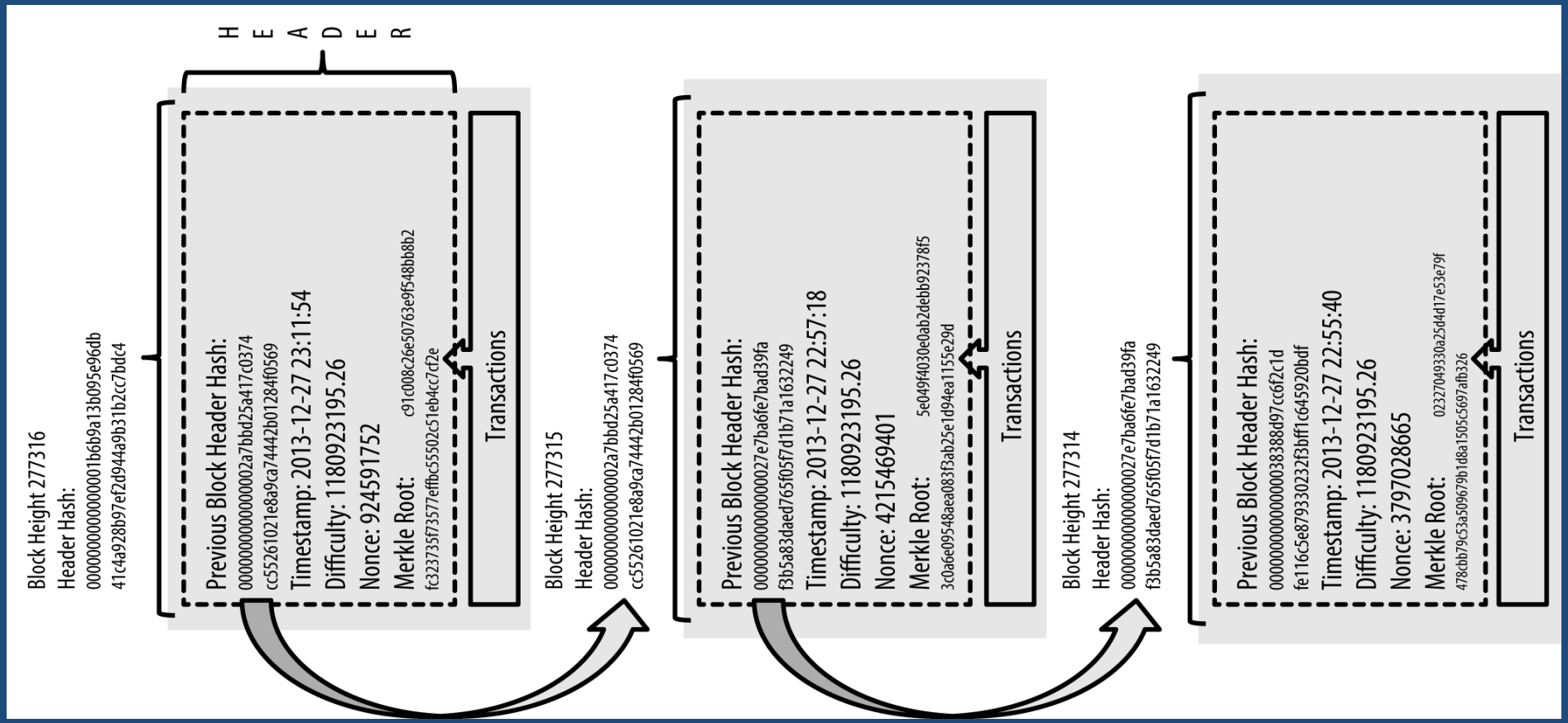
2.1 Bitcoin / Ethereum Blockchain Technology – Blockchain



2.1 Bitcoin / Ethereum Blockchain Technology – Blockchain

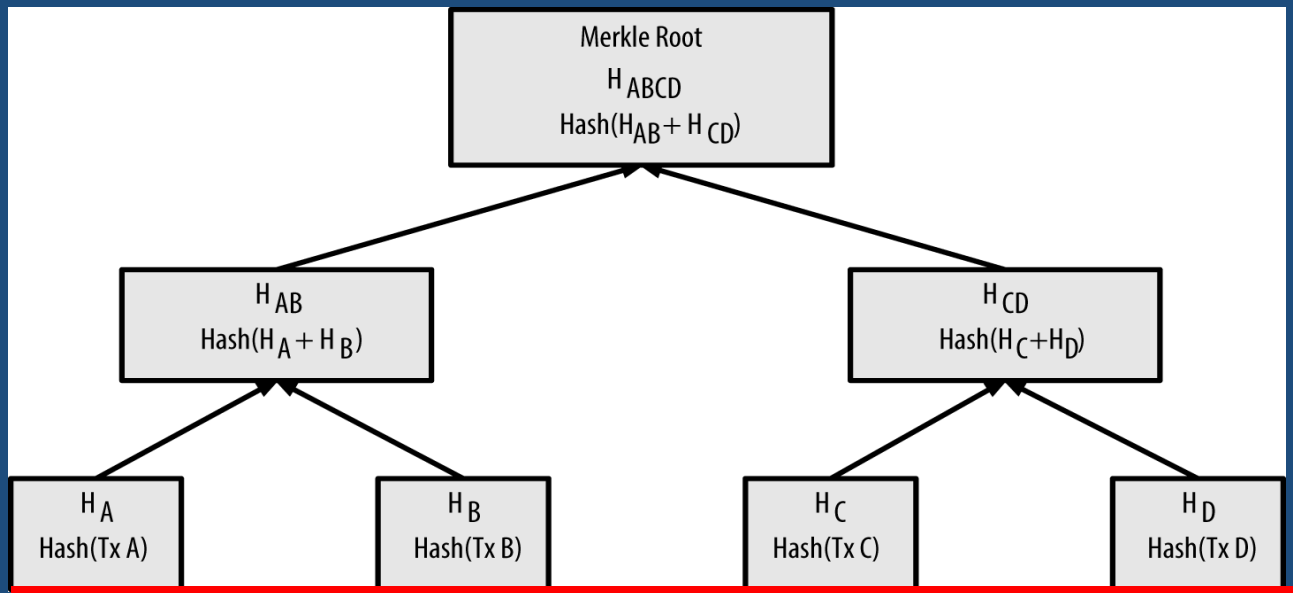


2.1 Bitcoin / Ethereum Blockchain Technology – Blockchain



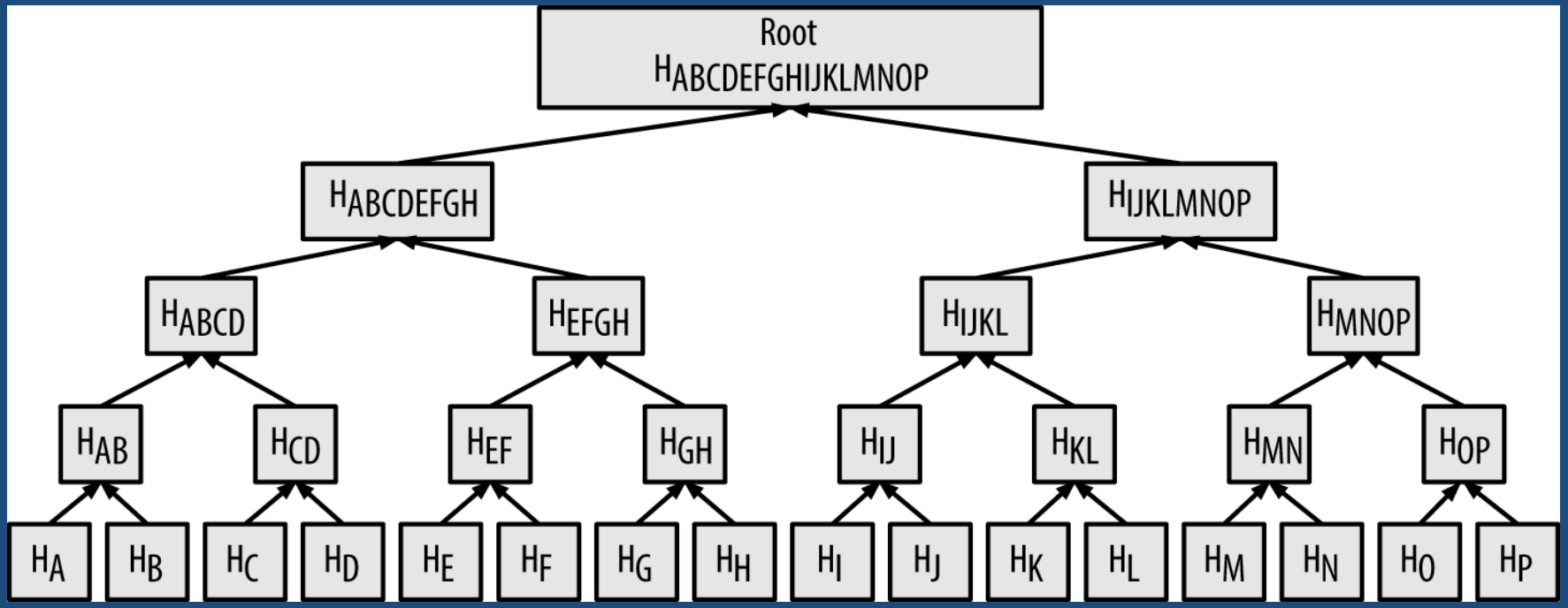
2.1 Bitcoin / Ethereum Blockchain Technology – Blockchain – Transactions Hashes in Merkle Trees

Merkle trees are used in bitcoin to summarize all the transactions in a block, producing an overall digital fingerprint of the entire set of transactions, providing a very efficient process to verify whether a transaction is included in a block. The merkle tree is constructed bottom-up. If we have 4 transactions, A, B, C, and D, which form the leaves of the merkle tree:



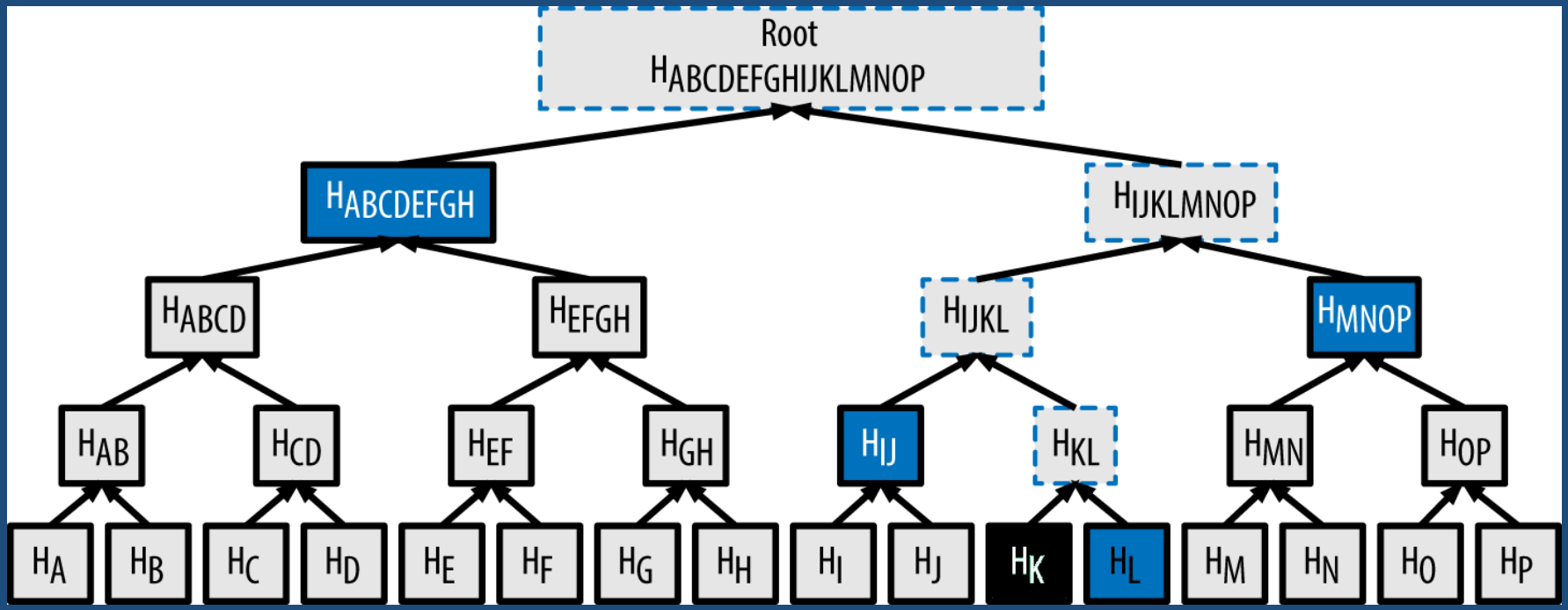
H_A = SHA256(SHA256(Transaction A)); H_{AB} = SHA256(SHA256(H_A + H_B));

2.1 Bitcoin / Ethereum Blockchain Technology – Blockchain – Transactions Hashes in Merkle Trees



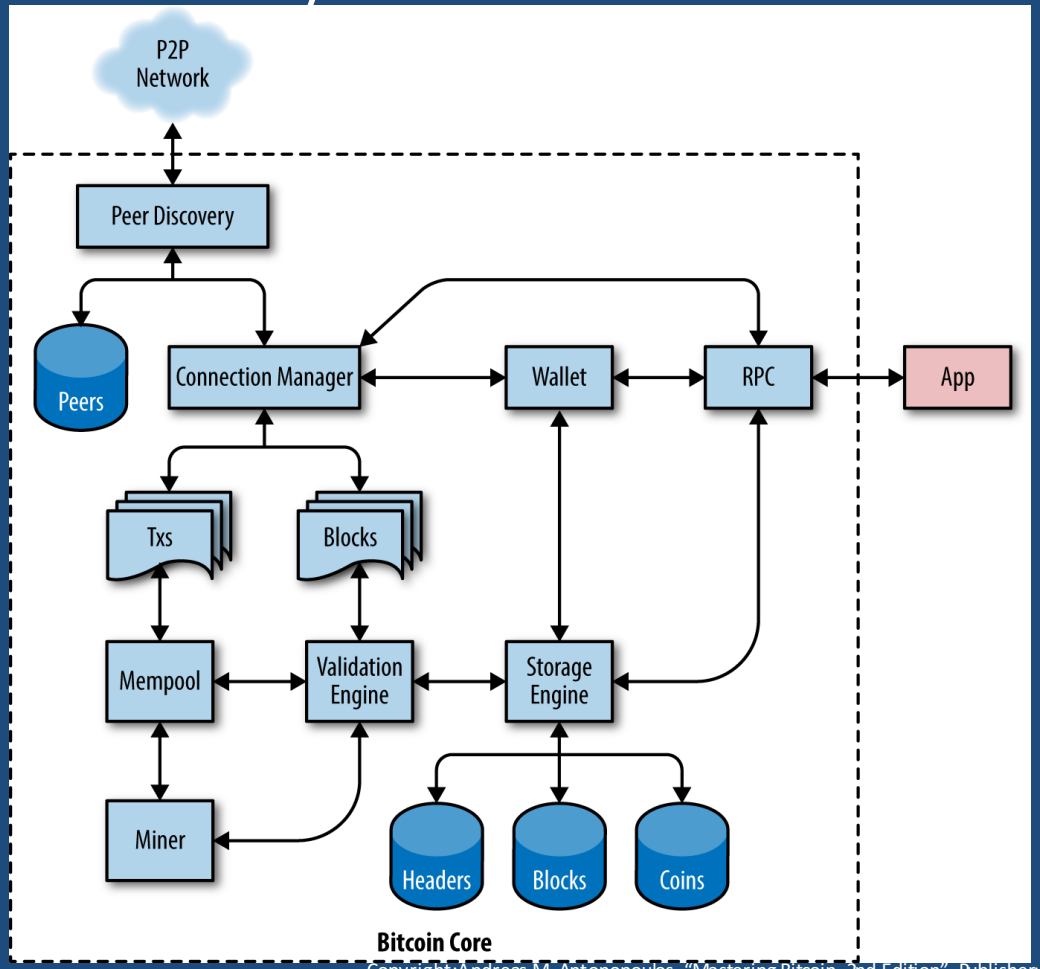
This tree is built from hashes for 16 transactions. Although the root looks bigger than the leaf nodes in the diagram, it is the exact same size, just 32 bytes.

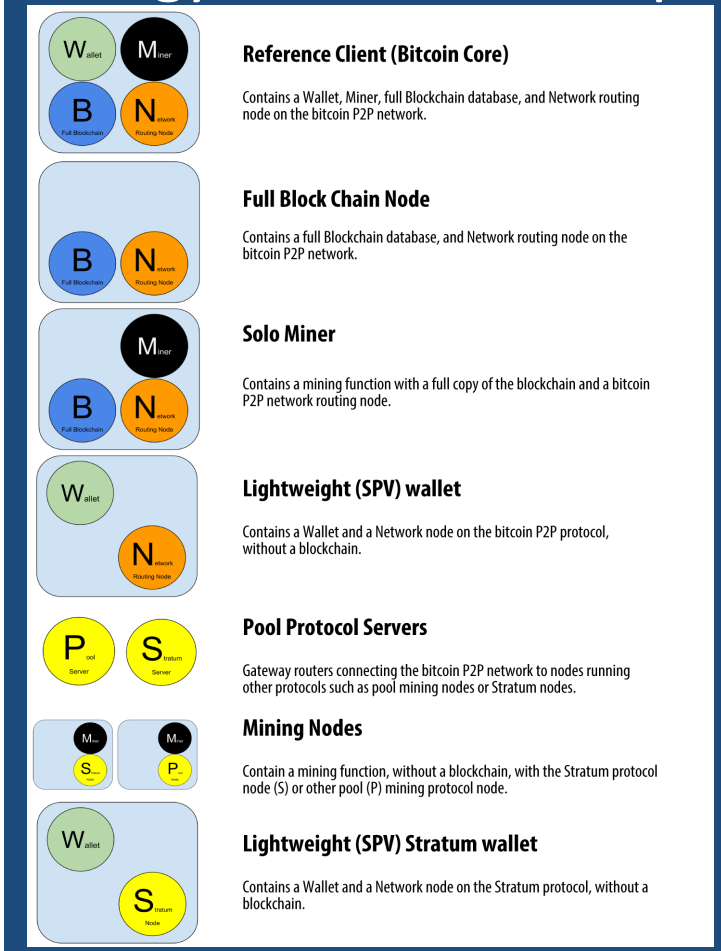
2.1 Bitcoin / Ethereum Blockchain Technology – Blockchain – Transactions Hashes in Merkle Trees



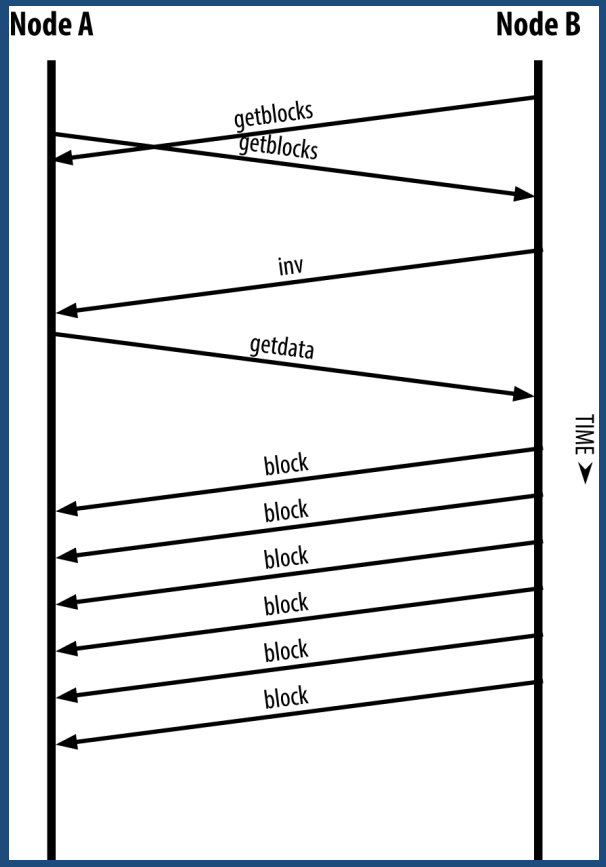
A node can prove that a transaction K is included in the block by producing a Merkle path that is only four 32-byte hashes long (128 bytes total). The path consists of the four hashes H_L, H_{IJ}, H_{MNOP}, and $H_{ABCDEFGH}$. With those 4 hashes provided as an authentication path, any node can prove that H_K (with a black background at the bottom of the diagram) is included in the Merkle root by computing 4 additional pair-wise hashes H_{KL}, H_{IJKL}, $H_{IJKLMNOP}$, and the Merkle tree root (outlined in a dashed line in the diagram).

2.1 Bitcoin / Ethereum Blockchain Technology – Reference Impl.

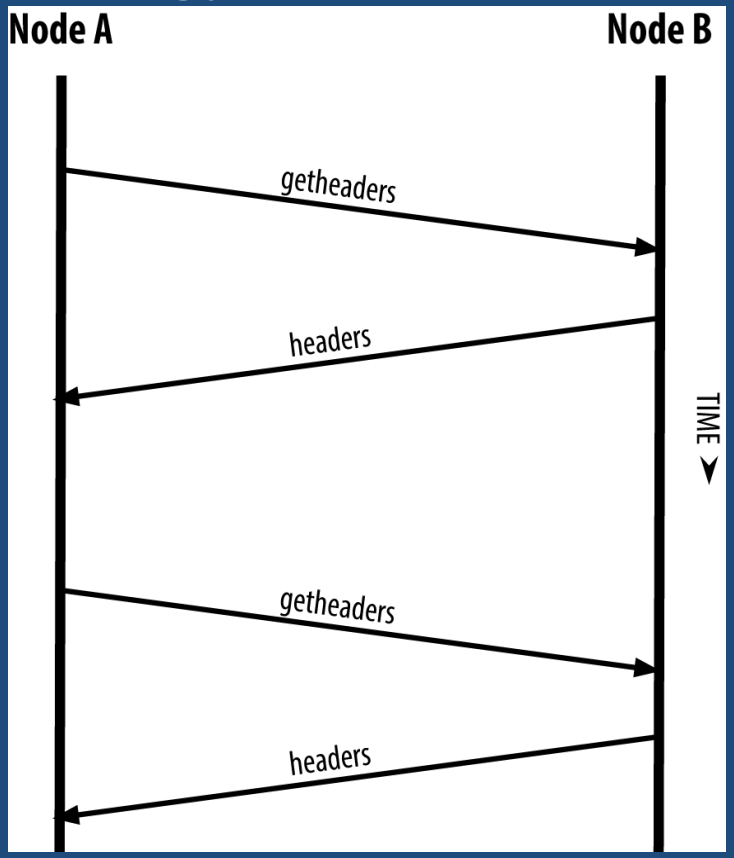




2.1 Bitcoin / Ethereum Blockchain Technology – P2P Network



Node synchronizing the blockchain by retrieving blocks from a peer



SPV node synchronizing the block headers

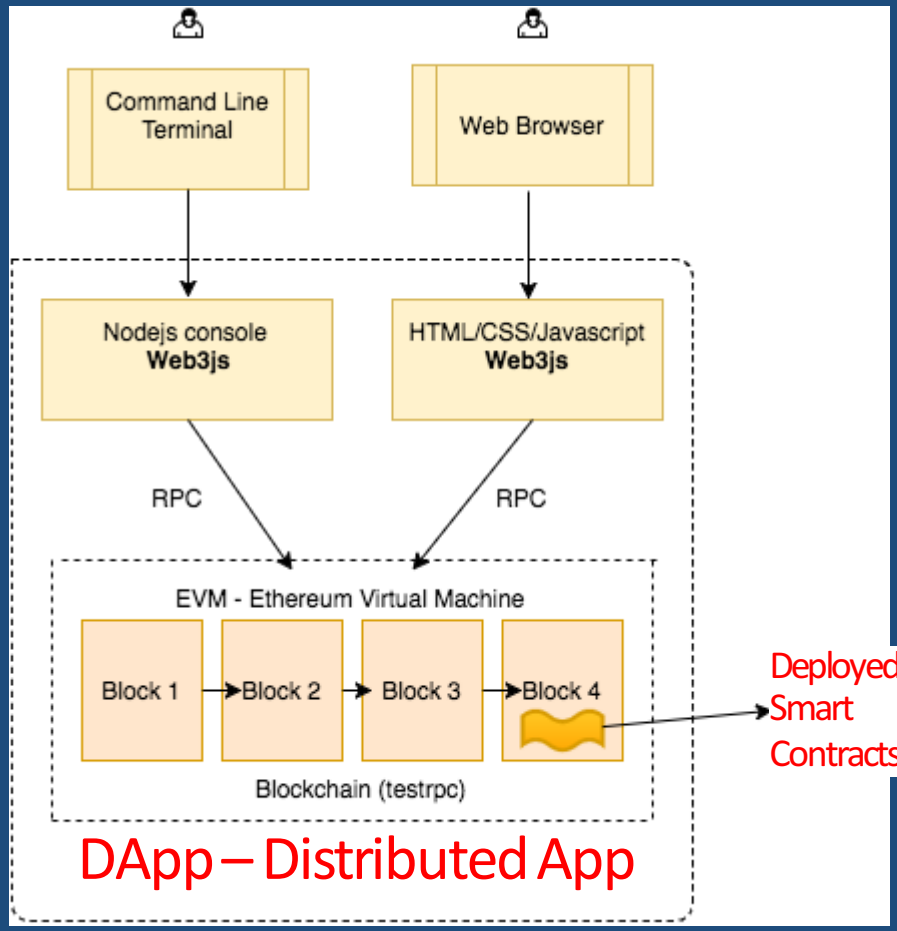
2.1 Consensus Algo - Bitcoin / Ethereum Blockchain Technology – P2P Network

2.1 Bitcoin / Ethereum Blockchain Peer2Peer Network Architecture

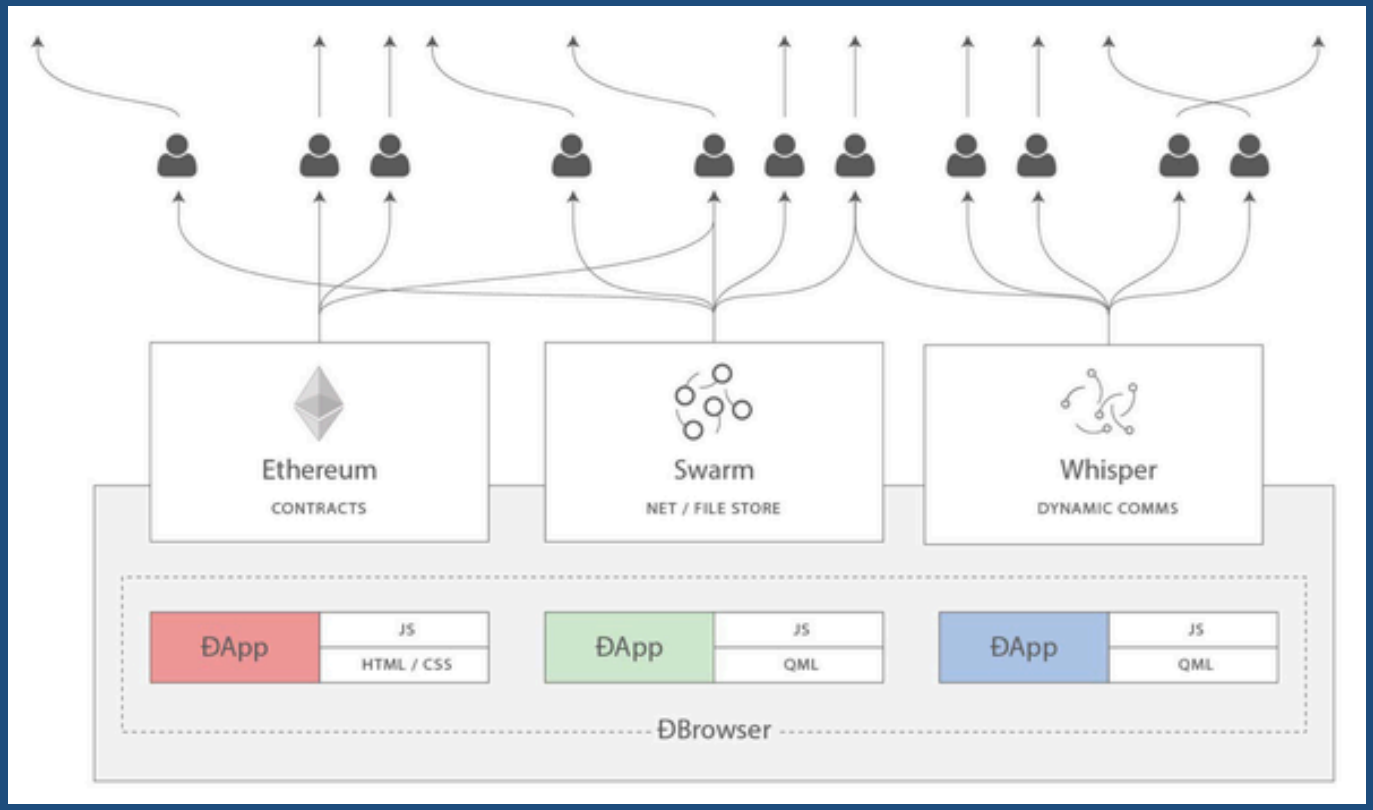
Ethereum shares many common elements with other open blockchains:

- **a peer-to-peer network connecting participants;**
- **a Byzantine fault-tolerant consensus algorithm for synchronization of state updates (a proof-of-work blockchain);**
- **the use of cryptographic primitives such as digital signatures and hashes, and a digital currency (Ether).**

Ethereum's language is *Turing complete*



2.1 Bitcoin / Ethereum Blockchain Peer2Peer Network Architecture



Web3: A suite of decentralized application components for the next evolution of the web

2.1 Bitcoin Standards

luke-jr Merge pull request #740 from skddc/patch-1 ···		Latest commit 954df0d on 14 Dec 2018
bip-0001	Fix formatting	5 years ago
bip-0002	Add obsolete status to process image	2 years ago
bip-0008	Amend BIP8 by height	2 years ago
bip-0009	BIP 9: Misplaced table cells typo	11 months ago
bip-0016	fix bip-0016 link 404	4 months ago
bip-0032	Fix formatting	5 years ago
bip-0039	Fix two errors in the BIP 39 French wordlist	a year ago
bip-0042	Include image for BIP42	5 years ago
bip-0047	BIP-0047: Reusable payment codes	4 years ago
bip-0068	Improve title, add encoding diagram and small fixup	3 years ago
bip-0069	add EOF newlines per @luke-jr	3 years ago
bip-0070	Put BIP 75 in the right place in README, and clean up formatting a bit	3 years ago
bip-0073	Fix formatting	5 years ago
bip-0075	- Update identifier comment in paymentrequest.proto	2 years ago
bip-0098	BIP-0098: Fast Merkle Trees	a year ago
bip-0114	BIP114: MAST proposal v2	3 years ago
bip-0122	Assign BIP 122	3 years ago
bip-0135	Assign BIP 135: Generalized version bits voting	2 years ago
bip-0144	BIP141: commitment clarification. BIP144: new diagram	3 years ago
bip-0152	header message can also get replied by getdata (CMPCT_BLOCK)	3 years ago
bip-0156	renamed files and updated README.mediawiki index	5 months ago
bip-0158	BIP 158: Add more cases to test vectors.	5 months ago
bip-0174	BIP 174 workflow graphics	7 months ago
scripts	scripts/buildtable: Support License-Code header	a year ago

2. EMV Tokenization Pay – Apple Pay and Android-Google Pay ING / BT Pay from 2016+

How Visa Token Service Works

The Visa Token Service enables digital payment service providers and financial institutions to offer their customers a safe way to shop online and with mobile devices. Here's how a token is initiated.

Step 1
Consumer enrolls their Visa account with a digital payment service (such as an online retailer or mobile wallet) by entering their primary account number (PAN), security code and other payment account information.

Step 2
The digital payment service provider requests a payment token from Visa for the enrolled account.

Step 3
Visa shares the token request with the account issuer (such as the consumer's bank).*

Step 5
Visa shares the token with the token requestor for online and mobile (NFC) payment use. A payment token can be limited to a specific mobile device, eCommerce merchant or number of purchases (say, a limit of five) before expiring.

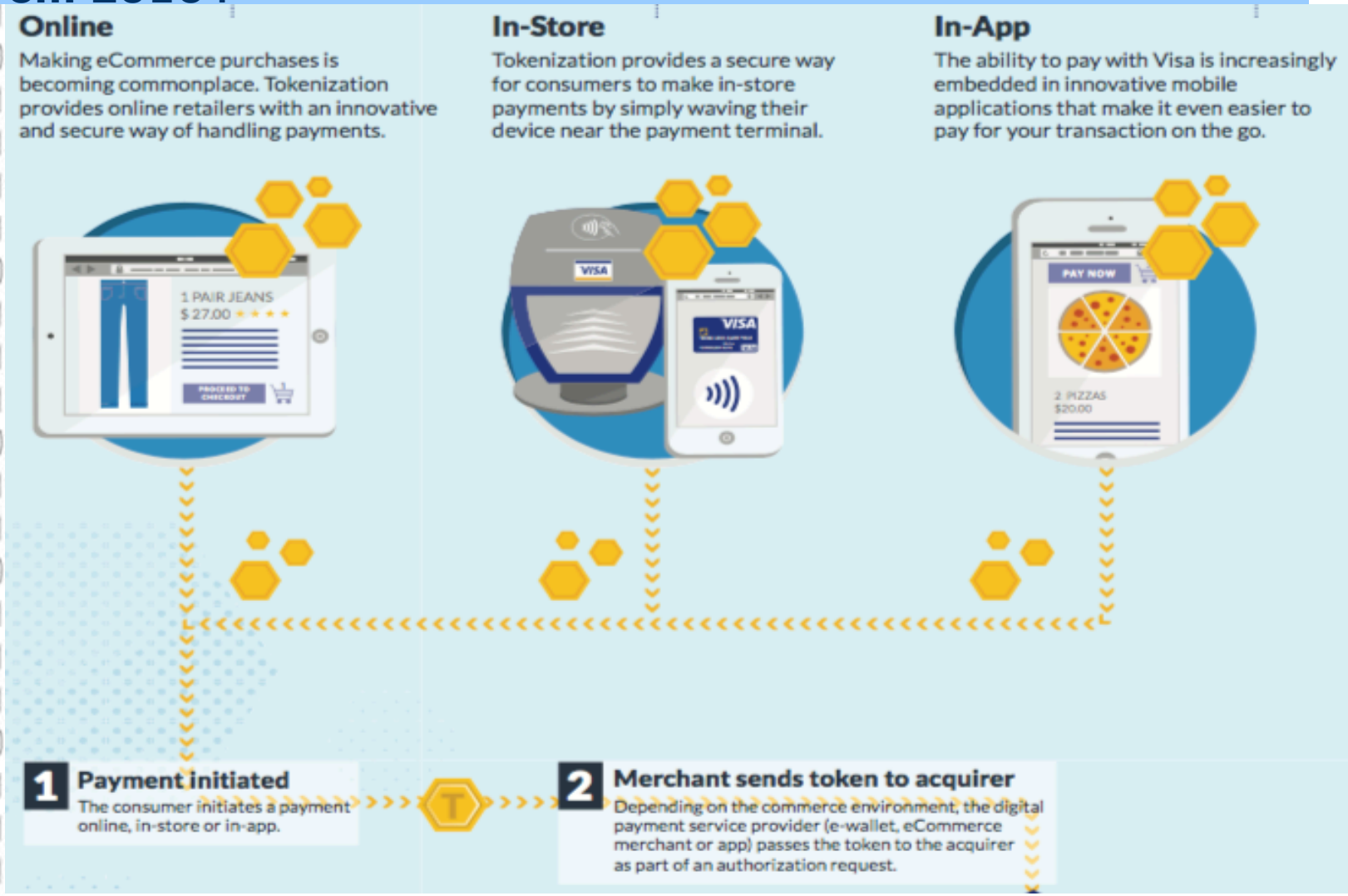
Step 4
With the account issuer's approval, Visa replaces the consumer's PAN with a unique digital identifier (the token).

https://usa.visa.com/dam/VCOM/download/security/documents/visa-security-tokenization-infographic.pdf

*Not applicable to all use cases

Section 2.1 – Mobile Payment Systems

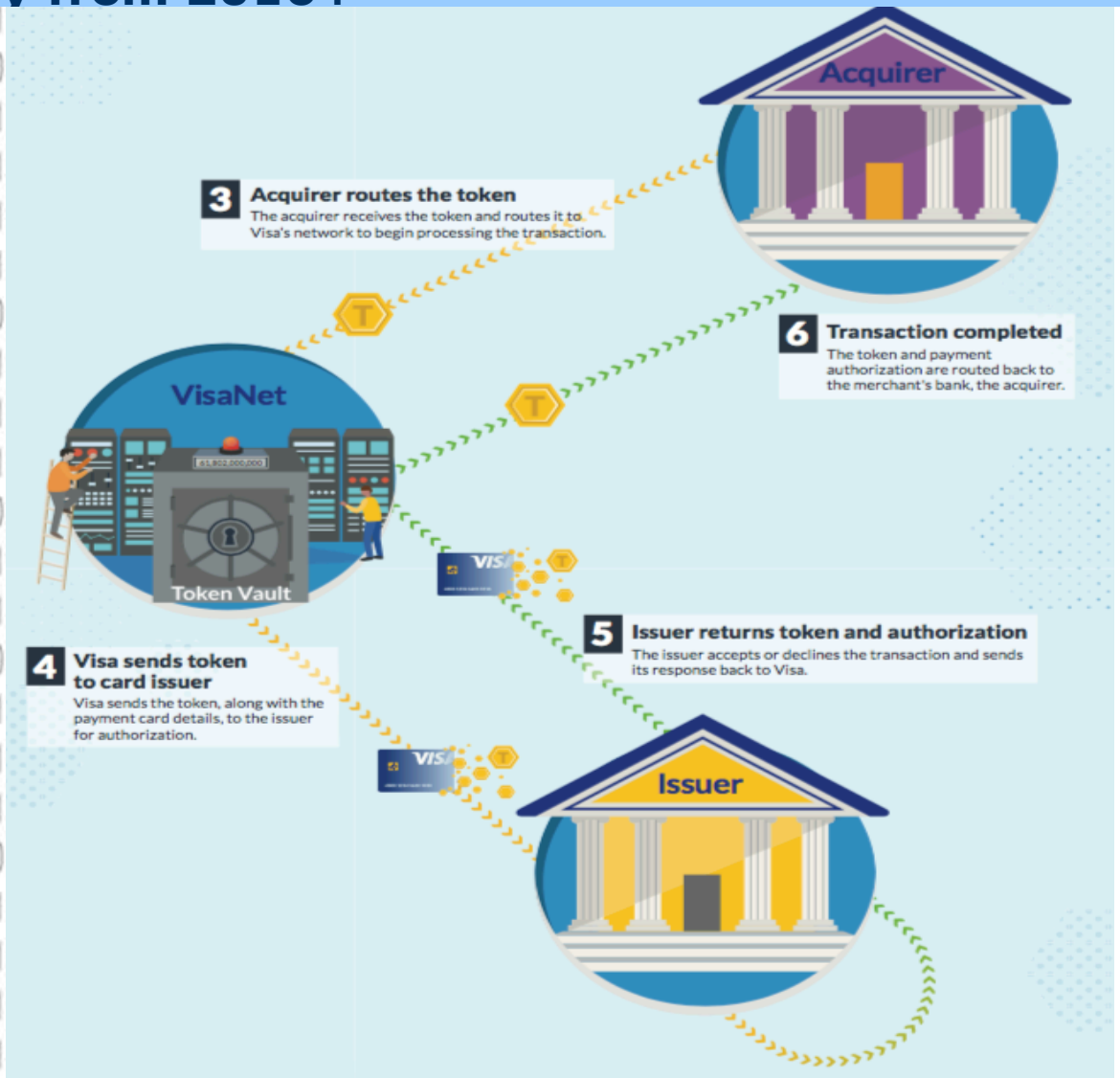
2. EMV Tokenization Pay – Apple Pay and Android Pay ING/ BT Pay from 2016+



https://usa.visa.com/dam/VCOM/download/security/documents/visa-security-tokenization-infographic.pdf

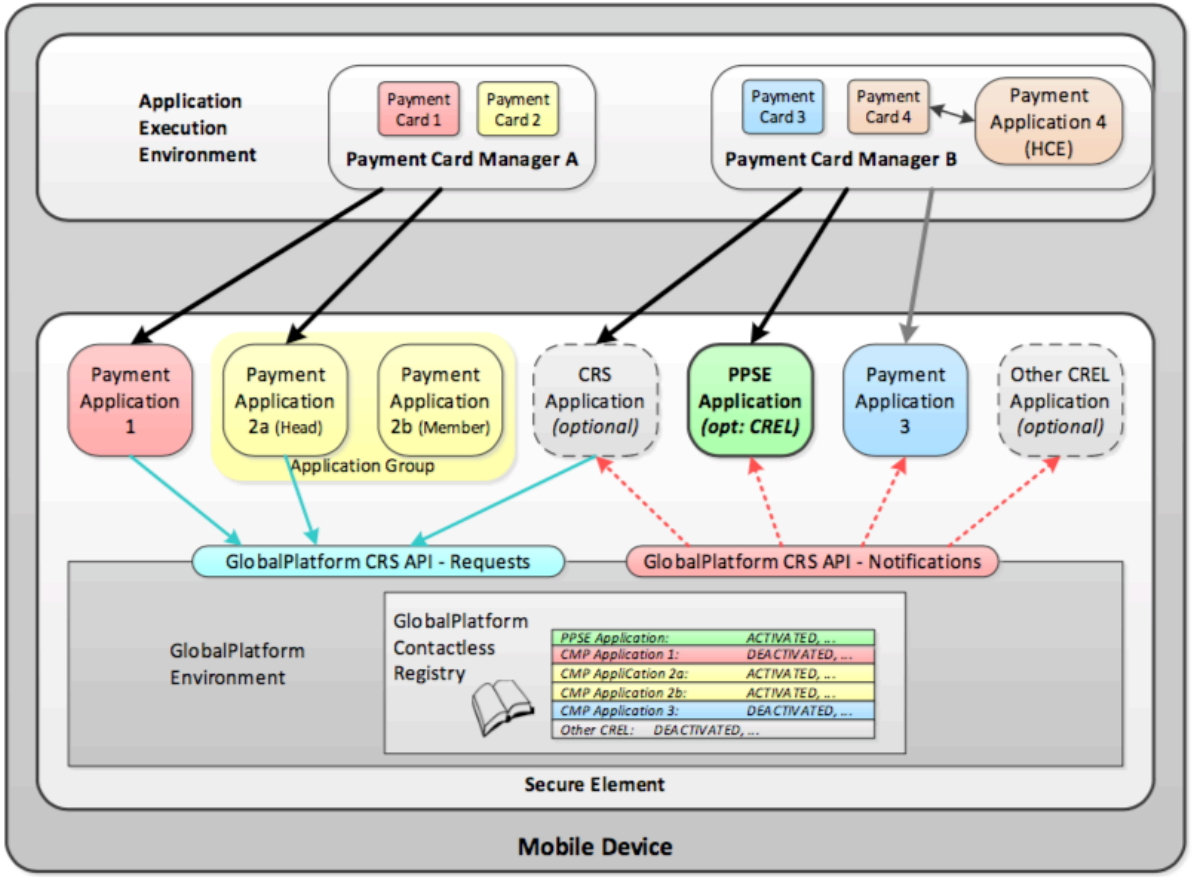
Section 2.1 – Mobile Payment Systems

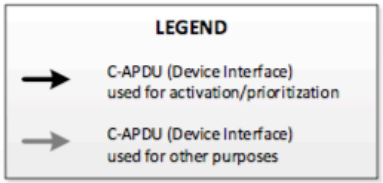
2. EMV Tokenization Pay – Apple Pay and Android Pay ING / BT Pay from 2016+



https://usa.visa.com/dam/VCOM/download/security/documents/visa-security-tokenization-infographic.pdf

3. EMV Mobile Pay – Secure Element Environment and Architecture – Demo Source Code

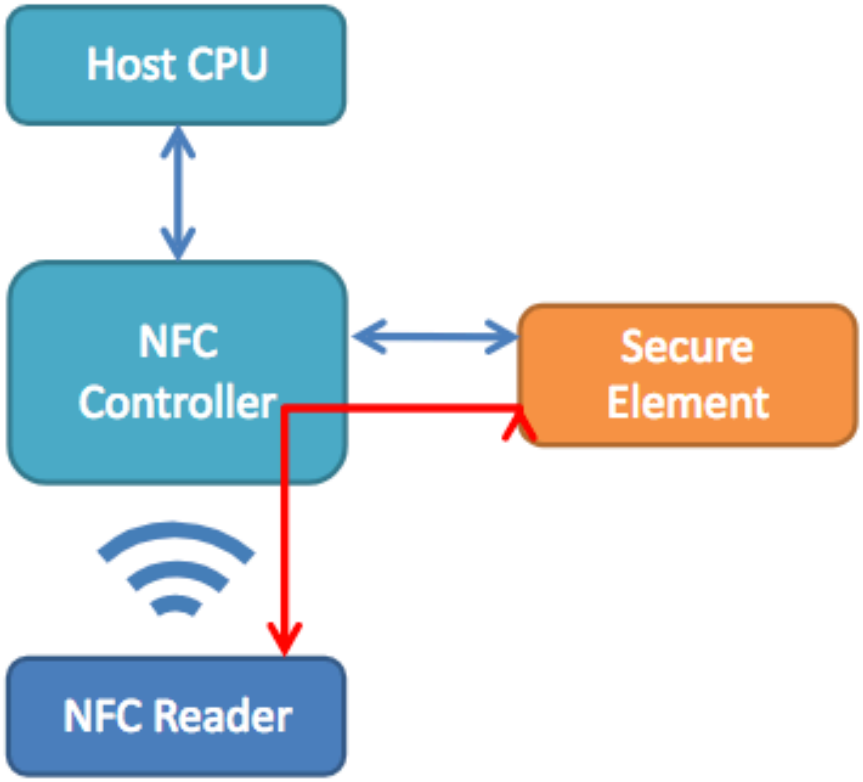




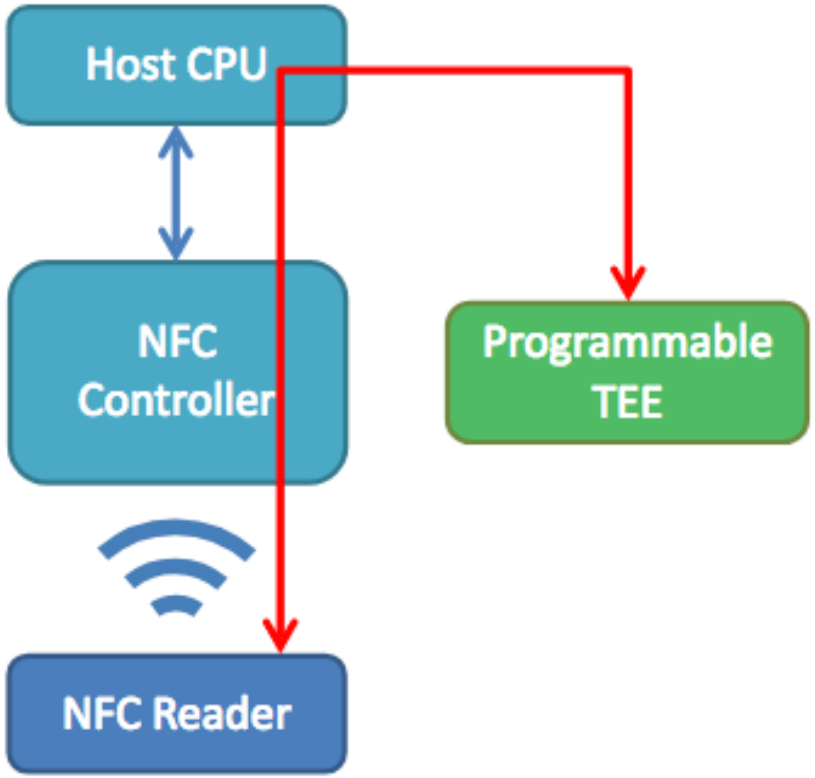
https://www.emvco.com/wp-content/plugins/pmpro-customizations/oy-getfile.php?u=/wp-content/uploads/documents/EMVCo_PPSE_and_Application_Mgmt_for_SE_v1.0-1.pdf

Section 2.1 – Mobile Payment Systems

3. EMV Mobile Pay – Secure Element – Google Wallet up 2014 vs. HCE (Host Card Emulation) – Google Wallet from 2014



NFC card emulation with a secure element



NFC card emulation with a Programmable TEE

http://slideplayer.com/slide/3429113/

Section 2.1 – Mobile Payment Systems

3. EMV Mobile Pay – Secure Element – Google Wallet up 2014 vs. HCE (Host Card Emulation) – Google Wallet from 2014

/Data/Temp/ecomm/Part4/EMVProjectsNetBeans/EMVDemoProj01/src/emvdemo/EMVdemo.java

```
import javacard.framework.*;

/**
 * Contains implementation of EMVdemo Java Card applet class.
 * emvdemo package AID = 9C274369EA 59 (RID + PIX)
 * Pass 4 QDS = 12341234
 *
 *                                    ------- EMVFileSystem – only one
 *         (has) 1           (has) 1 |
 * EMVdemo -------- EMVPurse --------|
 *                                   |
 *                                   ------- CVR (Card Verification Results) – only one
 *
 * File: EMVdemo.java
 * @author critoma
 */
public class EMVdemo extends Applet {
    // EMV applet objects
    private EMVPurse emvPurse;

    public final static byte TRUE = 1;
    public final static byte FALSE = 0;

    /**
     * Installs this applet.
     *
     * @param bArray
     *            the array containing installation parameters
     * @param bOffset
     *            the starting offset in bArray
     * @param bLength
     *            the length in bytes of the parameter data in bArray
     */
    public static void install(byte[] bArray, short bOffset, byte bLength) {
        new EMVdemo();
    }
```

Section 2.1 – Mobile Payment Systems

3. EMV Mobile Pay – Secure Element – Google Wallet up 2014 vs. HCE (Host Card Emulation) – Google Wallet from 2014

/Data/Temp/ecomm/Part4/EMVProjectsNetBeans/EMVDemoProj01/src/emvdemo/EMVPurse.java

```
/*
/send 00A40400069C274369EA107F
/send 80A800000282947F
/send 00B2010C007F
/send 0020008008241234FFFFFFFFFF00
/send 80AE400020000000000010000000000010004000000000003D2141004FFFF42345658ABCD7F
*/

package emvdemo;

import javacard.framework.ISO7816;
import javacard.framework.ISOException;
import javacard.framework.JCSystem;
import javacard.framework.OwnerPIN;
import javacard.framework.Util;
import javacard.security.CryptoException;
import javacard.security.DESKey;
import javacard.security.KeyBuilder;
import javacardx.crypto.Cipher;

/**
 *
 * Contains implementation of the classes: EMVPurse and CVR
 *                    ---------- EMVFileSystem - only one
 *           (has) 1 |
 * EMVPurse ---------|
 *                   |
 *                    ---------- CVR (Card Verification Results) - only one
 *
 * File: EMVPurse.java
 * @author critoma
 */
public class EMVPurse
{
    public final static byte TRUE = 1;
    public final static byte FALSE = 0;

    //Definition of the application-related constants and objects
```

Section 2.1 – Mobile Payment Systems

3. EMV Mobile Pay – Secure Element – Google Wallet up 2014 vs. HCE (Host Card Emulation) – Google Wallet from 2014

~/Data/Temp/ecomm/Part4/EMVProjectsNetBeans/EMVDemoProj01/src/emvdemo/EMVPurse.java — class EMVPurse

```
112     // 4. EMV application specific value constants:
113
114     // AIP  - Application Interchange Profile & AFL - Application File Locator are data objects
115     // stored in ICC (smart card Integrated Circuits Card) and received by the TERMINAL (HOST) in response
116     // to the GPO (GET PROCESSING OPTIONS) command APDU, the first command performed after application selection
117
118     // AIP - Application Interchange Profile, dynamic off-line authentication, for CHV - Cardholder Verification
119     // AIP informs the TERMINAL about the topmost functions supported by the card
120     // TLV - Tag, Lengh, Value = 0x82, 0x02, 2-byte value (which informs the terminal that the ICC supports CHV - cardholder verification and S
121     // but could be modified to support other offline data authentication schemes - e.g. DDA (Dynamic Data Authentication) or fDDA (fast DDA)
122     // 2nd byte is RFU - Reserved for Future Use and 8th bit is reserved for EMV contactless specs
123     // 1st byte of AIP has the value 0x5C which means (according with EMV v4.3 Book 3, Annex C.1) - bits from left to right (b8-b1) / b2 and b8
124     // ------------------------------------------------
125     // | b8      | b7 | b6 | b5 | b4 | b3 |    b2 | b1 |
126     // | 0 (RFU) |  1 |    |    |    |    |       |    |  b7 = 1 => SDA supported (Offline data authentication is supported by the card)
127     // |         |    |  0 |    |    |    |       |    |  NOT used but if b6 = 1 => DDA supported (Offline data authentication is supported by
128     // |         |    |    |  1 |    |    |       |    |  b5 = 1 => CHV - Cardholder verification is supported by the card
129     // |         |    |    |    |  1 |    |       |    |  b4 = 1 => Terminal risk management is to be performed
130     // |         |    |    |    |    |  1 |       |    |  b3 = 1 => Issuer authentication is supported in terms of using the EXTERNAL AUTHENTIC
131     // |         |    |    |    |    |    |0 (RFU)|  0 |  NOT used but if b1 = 1 => CDA (Combined DDA/Application Cryptogram Generation) suppor
132     private final byte[] AIP = { (byte) 0x5C, (byte) 0x00 };
133
134     // AFL - Application File Locator, with SFIs (Short File Identifier) 1 and 2
135     // TLV - Tag, Lengh, Value = 0x94, 4 * n, Data on each from n files (here 2 SFI entries with 4 bytes each =. 8 bytes).
136     // For each aenry according with EMV specs:
137     // Byte 1 (Bits 8-4: SFI - Short File Identifier, Bits 3-1: 0 0 0) | Byte 2 (First record number to read from this file) |
138     // Byte 3 (Last record number to read from this file) | Byte 4 (Number of consecutive records signed in signed application data)
139     // This sample application has 2 EF (Elementary Files):
140     // 1st file has SFI=1 and it has 6 records; (08 01 06 00)
141     // and 2nd file has FCI=2 and it has 2 records (10 01 02 00).
142     private final byte[] AFL = { (byte) 0x08, (byte) 0x01,
143                                  (byte) 0x06, (byte) 0x00,
144                                  (byte) 0x10, (byte) 0x01,
145                                  (byte) 0x02, (byte) 0x00 };
146
147     // PIN - Personal Identification Number (1234) - hard-coded, but might be changed or set at personalization phase
148     // BCD (Binary Code Decimal) - in 1st byte, 1st nibble (4 bits) is control field, and 2nd nibble is the digits number of the PIN
149     private final byte[] DEMO_PIN = { (byte) 0x24, (byte) 0x12, (byte) 0x34,
150                                       (byte) 0xFF, (byte) 0xFF, (byte) 0xFF,  (byte) 0xFF, (byte) 0xFF };
151
```

Section 2.1 – Mobile Payment Systems

3. EMV Mobile Pay – Secure Element – Google Wallet up 2014 vs. HCE (Host Card Emulation) – Google Wallet from 2014

a/Temp/ecomm/Part4/EMVProjectsNetBeans/EMVDemoProj01/src/emvdemo/EMVFileSystem.java

```
package emvdemo;

import javacard.framework.ISO7816;
import javacard.framework.ISOException;
import javacard.framework.JCSystem;
import javacard.framework.Util;

/**
 * Implementation of the EMV card file system
 * The hierarchy has the following "has a"relationship:
 *                                EMVFileSystem
 *                                     | (has) 0..n
 *             ------------------------------------------------
 *             |                                              |
 *          EMVFile                                        EMVFile
 *             | (has) 0..n                                   |
 *    --------------------------                  --------------------------
 *    | (has) 0..n            |                  | (has) 0..n            |
 * EMVFileRecord       EMVFileRecord          EMVFileRecord          EMVFileRecord
 *
 * This file contains the following class: EMVFileSystem and it refers the following classes:
 *
 * File: EMVFileSystem.java
 * @author critoma
 */
public class EMVFileSystem
{
    private byte[] selected_flag;
    private EMVFile[] files;
    private byte max_files_num; //Maximum number of files supported
    private byte next_av = 0; // Next file that can be created
    private byte selected_sfi; // SFI of the currently selected file

    public EMVFileSystem(byte maxFiles) {
        max_files_num = maxFiles;
        files = new EMVFile[maxFiles];
        selected_flag = JCSystem.makeTransientByteArray((short)1, JCSystem.CLEAR_ON_RESET);

        selected_sfi = 0;
    }
```

4. Mobile QR / NFC Payment – SEQR (Pay @ Subway in RO)

1. Enroll to a bank which is providing the interface with the SEQR App (Android, iOS, Windows Phone).
2. Order your product and the merchant is typing the product and the amount of money. The amount is sent via Internet (HTTPs) to the SEQR Server (Seamless Ltd.) + automatically the shop identification info.

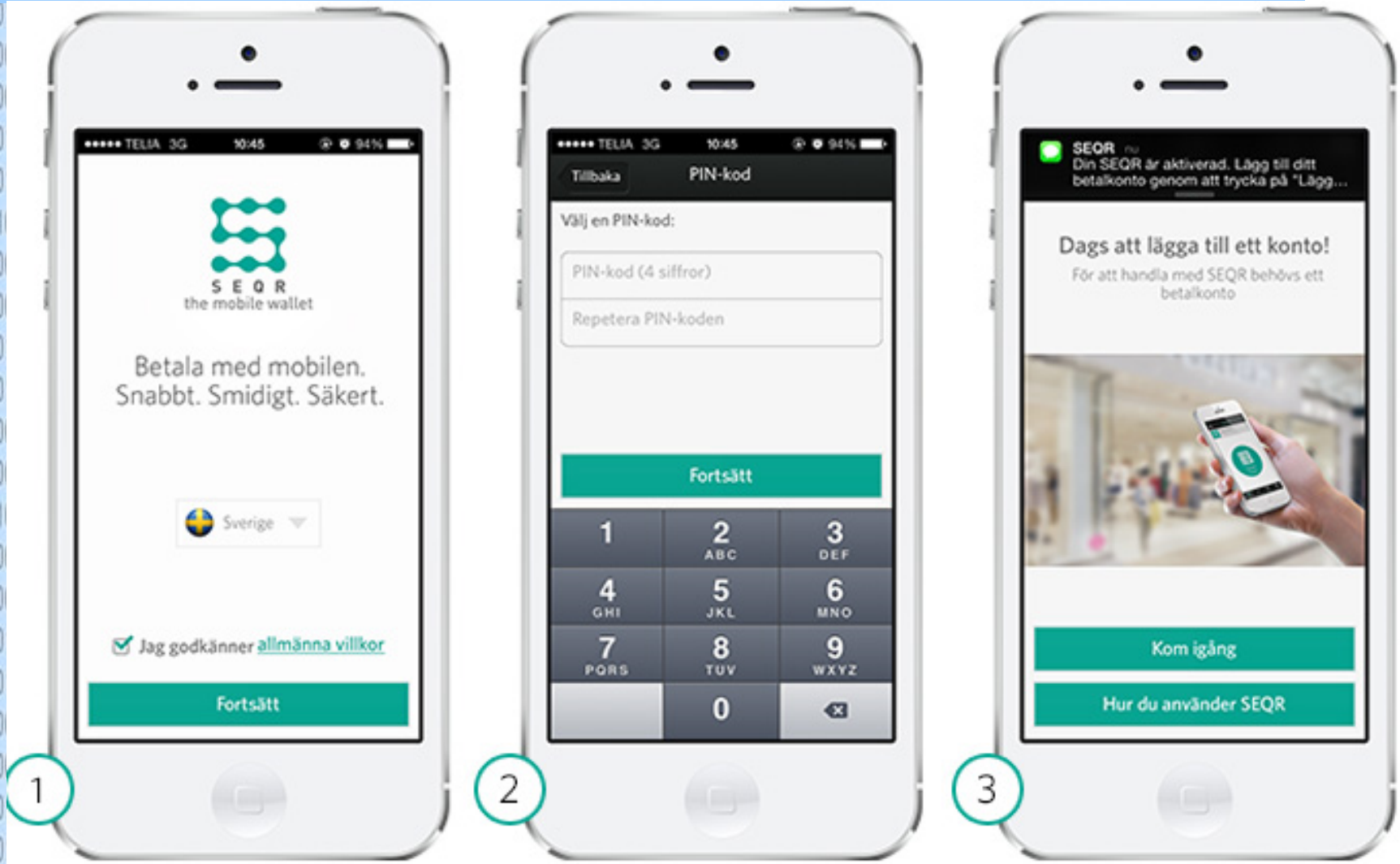
3. After step 2, the buyer scans the Shop QR code with SEQR App. The shop's QR code is not changing in years. The Mobile App is sending the shop QR code info to the SEQR Server. The server is matching the shop QR with your transaction request and it is requiring the buyer to introduce the PIN.

4. If the matching is correct, then the money from the buyer code are transferred to the merchant bank account (like in Debit card transactions).

https://www.seqr.com/se/en/blog/

Section 2.1 – Mobile Payment Systems

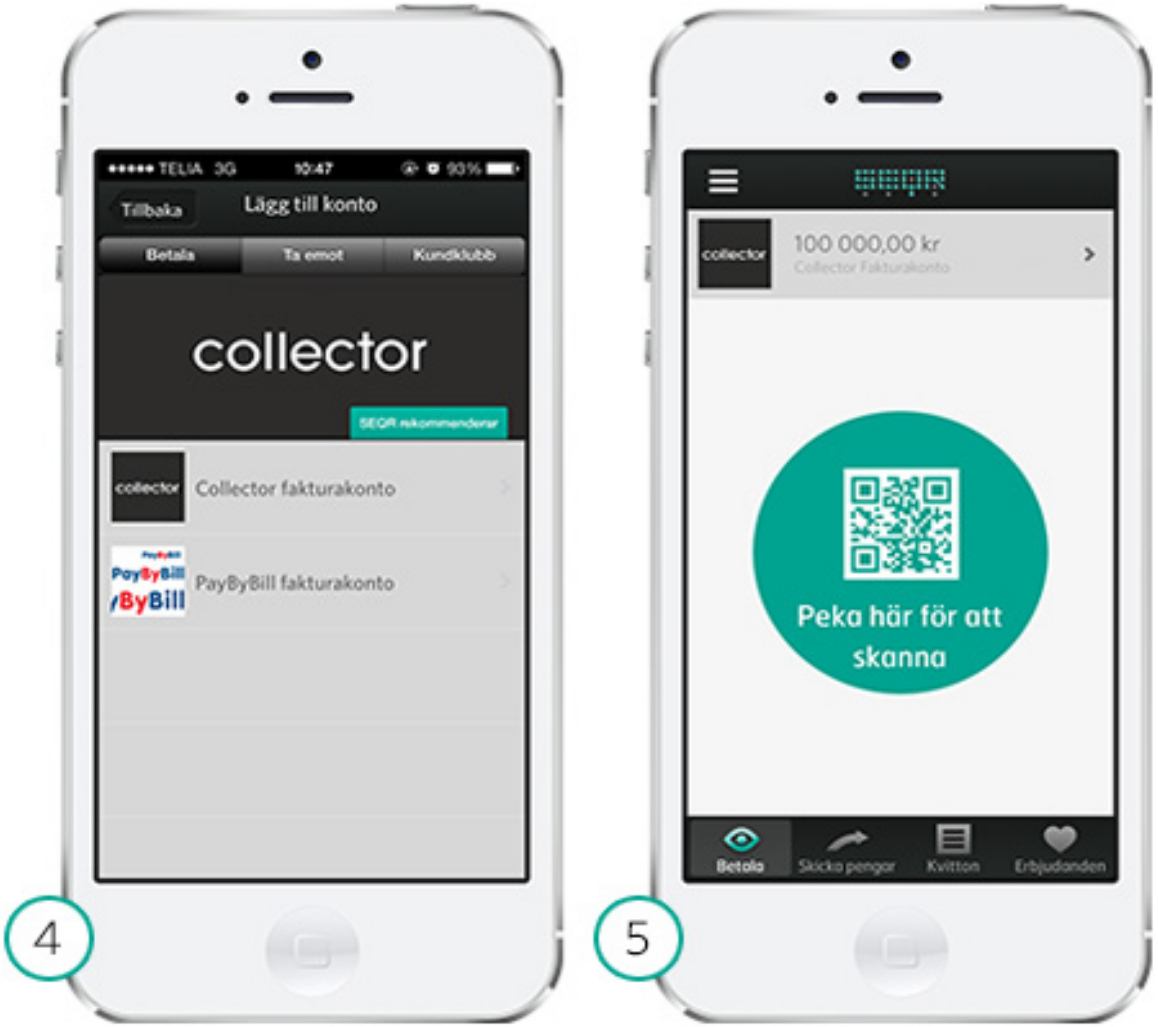
4. Mobile QR / NFC Payment – SEQR (Pay @ Subway in RO)



https://www.seqr.com/se/en/blog/

Section 2.1 – Mobile Payment Systems

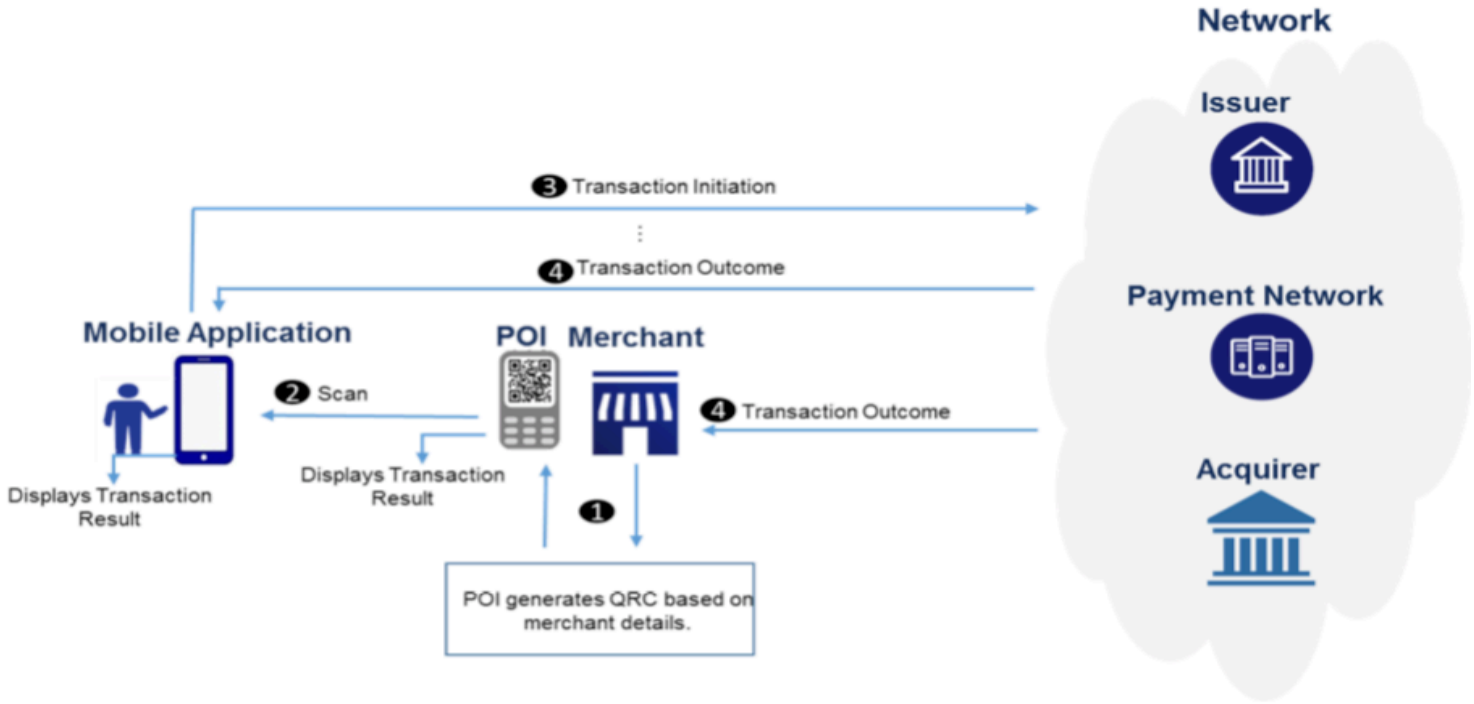
4. Mobile QR / NFC Payment – SEQR (Pay @ Subway in RO)



https://www.seqr.com/se/en/blog/

Section 2.1 – Mobile Payment Systems

4. Mobile QR / EMVCo QR - Merchant-Presented Mode Transaction Flow

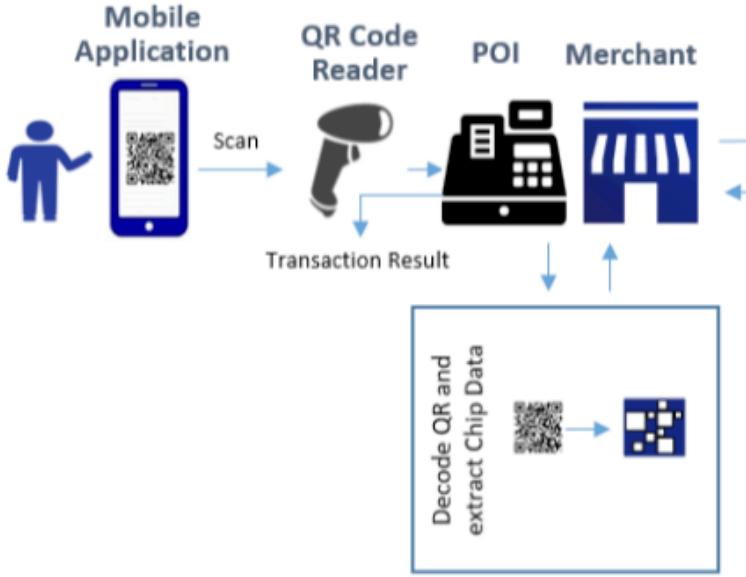


[1] Merchant generates and displays QR Code based on merchant details.
[2] Consumer scans QR Code using a mobile application to initiate the transaction, with CDCVM if required.
[3] Mobile application sends the transaction initiation request to the Network.
[4] The Network processes the transaction and informs the Merchant and the Consumer of the transaction outcome.

https://www.emvco.com/wp-content/plugins/pmpro-customizations/oy-getfile.php?u=/wp-content/uploads/documents/EMVCo-Merchant-Presented-QR-Specification-v1_0.pdf
https://www.emvco.com/wp-content/plugins/pmpro-customizations/oy-getfile.php?u=/wp-content/uploads/documents/EMVCo-Consumer-Presented-QR-Specification-v1.pdf

Section 2.1 – Mobile Payment Systems

4. Mobile QR / EMVCo QR – Consumer Presented Mode Architecture



Main Functionality: (on Level1)

- Scan QR Code
- Convert to string
- Transfer string to POI

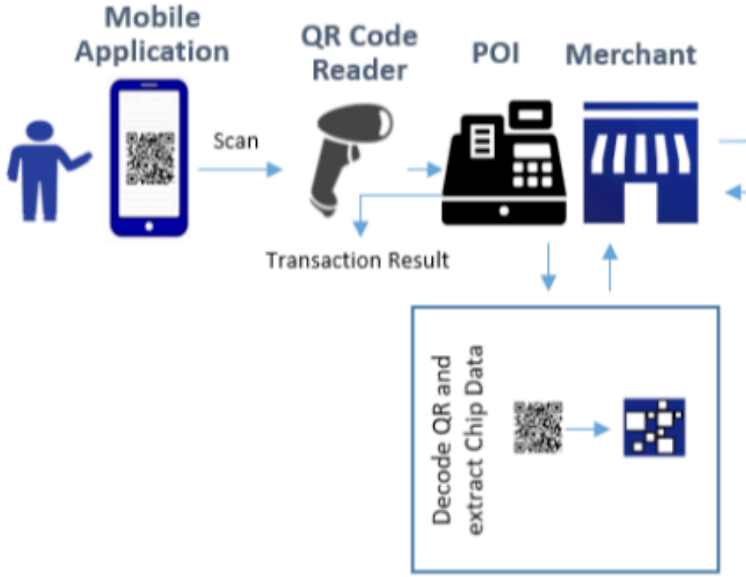
Main Functionality: (on Level2)

- Base64 Decoding
- Parse Data and Check AID
- Transaction Checkout/Processing
- Construct Auth Message

https://www.emvco.com/wp-content/plugins/pmpro-customizations/oy-getfile.php?u=/wp-content/uploads/documents/EMVCo-Merchant-Presented-QR-Specification-v1_0.pdf
https://www.emvco.com/wp-content/plugins/pmpro-customizations/oy-getfile.php?u=/wp-content/uploads/documents/EMVCo-Consumer-Presented-QR-Specification-v1.pdf

Section 2.1 – Mobile Payment Systems

4. Mobile QR / EMVCo QR – Consumer Presented Mode Architecture



Main Functionality: (on Level1)

- Scan QR Code
- Convert to string
- Transfer string to POI

Main Functionality: (on Level2)

- Base64 Decoding
- Parse Data and Check AID
- Transaction Checkout/Processing
- Construct Auth Message

https://www.emvco.com/wp-content/plugins/pmpro-customizations/oy-getfile.php?u=/wp-content/uploads/documents/EMVCo-Merchant-Presented-QR-Specification-v1_0.pdf
https://www.emvco.com/wp-content/plugins/pmpro-customizations/oy-getfile.php?u=/wp-content/uploads/documents/EMVCo-Consumer-Presented-QR-Specification-v1.pdf

Section 2.1 – Mobile Payment Systems

4. Mobile QR / EMVCo QR

Binary Data (shown as hex bytes):

```
85 05 43 50 56 30 31
61 1A
        4F 07 A0 00 00 00 55 55 55
        57 0F 12 34 56 78 90 12 34 58 D1 91 22 01 12 34 5F
```

Base64 Data:

```
hQVDUFYwMWEaTwegAAAAVVVVVw8SNFZ4kBI0WNGRIgESNF8=
```

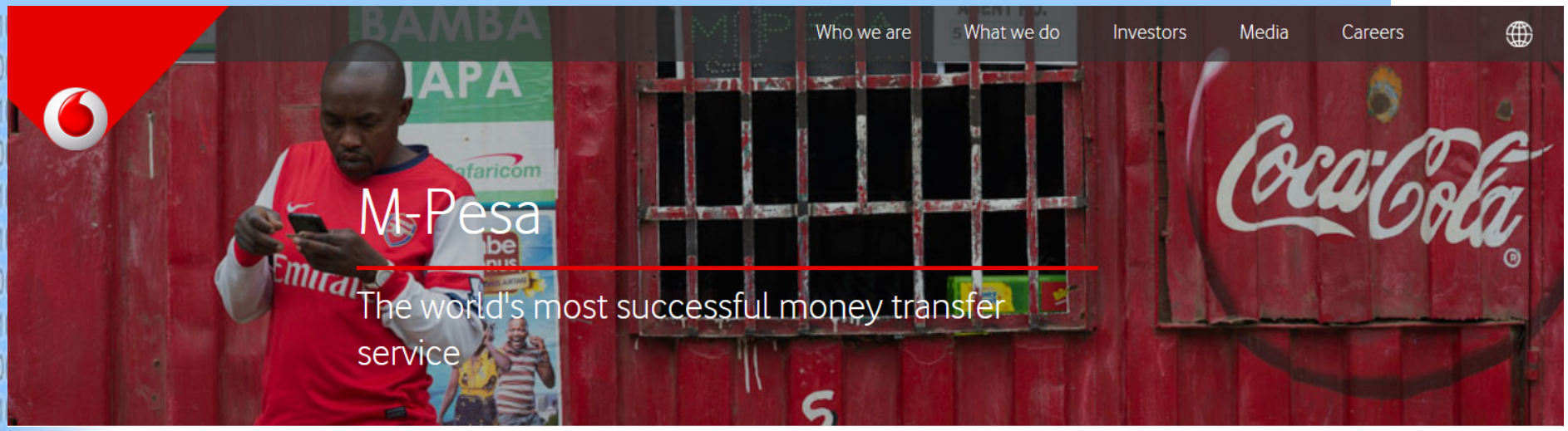
QR Code:

https://www.emvco.com/wp-content/plugins/pmpro-customizations/oy-getfile.php?u=/wp-content/uploads/documents/EMVCo-Merchant-Presented-QR-Specification-v1_0.pdf

https://www.emvco.com/wp-content/plugins/pmpro-customizations/oy-getfile.php?u=/wp-content/uploads/documents/EMVCo-Consumer-Presented-QR-Specification-v1.pdf

Section 2.1 – Mobile Payment Systems

5. M-Pesa – Vodafone (Pay Cab/Taxi Trip in RO) - Cancelled



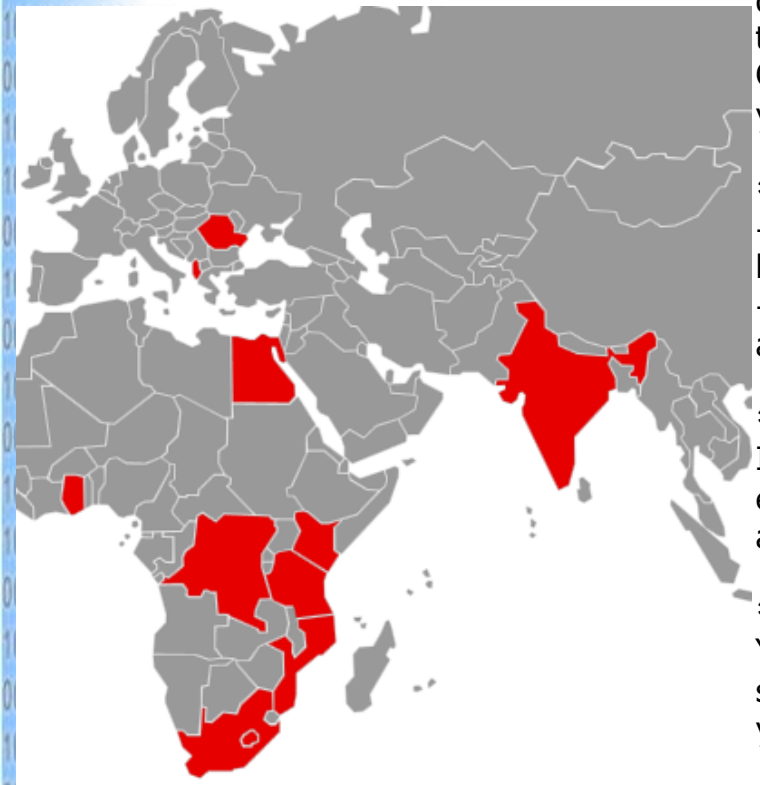
M-Pesa is the world's most successful money transfer service. It enables millions of people who have access to a mobile phone, but do not have or have only limited access to a bank account, to send and receive money, top-up airtime and make bill payments.

Customers register for the service at an authorized agent, often this is a small mobile phone store or retailer, and then deposit cash in exchange for electronic money which they can send to their family or friends. Once they have registered all transactions are completed securely by entering a PIN number and both parties receive an SMS confirming the amount that has been transferred. The recipient, who does not have to use the same network, receives the electronic money in real-time and then redeems it for cash by visiting another agent.

http://www.vodafone.com/content/index/what/m-pesa.html

Section 2.1 – Mobile Payment Systems

5. M-Pesa



It works like this:

****register****
you register to join Mpesa at any Mpesa agent for free. This is quick to do and easy - because Mpesa agents ate everywhere in cities and they seem to be in every village (I'm tryng to get some stats on this). Once registration is done the network sends you an updated menu on your phone - so you're ready go.

****load some money on****
- once registered you can load money on to your phone at any agent by just handing over your cash and they load it onto your phone.
- or a kind relative, friend, or employer can put money onto your account for you.

****send money****
It's easy to send money too. You just go to the menu on your phone, enter the phone number of the recipient and amount and the money arrives near instantly - in the time it takes for an sms to arrive.

****pay bills****
You can also pay all sorts of bills, over 600, by mpesa. Through a very similar process to sending money to friends. You can buy a flight, pay your water bill and much more.

****take money out ****
- to take your money out you just need to go to an Mpesa agent and basically you send them some money, which they then give to you there and then.
- or go to an atm.

http://ourmobileworld.org/post/35349373601/what-is-mpesa-how-does-it-work-how-did-it-start

Section 2.1 – Mobile Payment Systems

1. Sonera MobilePay

FEATURES:

- Operator-independent mobile payment scheme
- A product of Sonera, a Finnish telecommunications company
- Designed for making physical-world purchases from the mobile device, such as vending machine payments
- After registering within the system, payment amounts are debited directly from a user's bank account, or charged to a credit card – like in PayPal
- The mobile user calls/send an SMS to a premium-rate number, which is unique to a specific vending machine

1. Sonera MobilePay

FEATURES:

- The call/SMS is routed to the MobilePay server, and the user is identified using the caller identification feature of GSM – IMSI/IMEI/MSISDN.

- Associated with each number is a fixed purchase price, which is debited from the user's account at the server. For purchases over a certain threshold amount, the server can interactively request a PIN from the user to confirm the transaction.

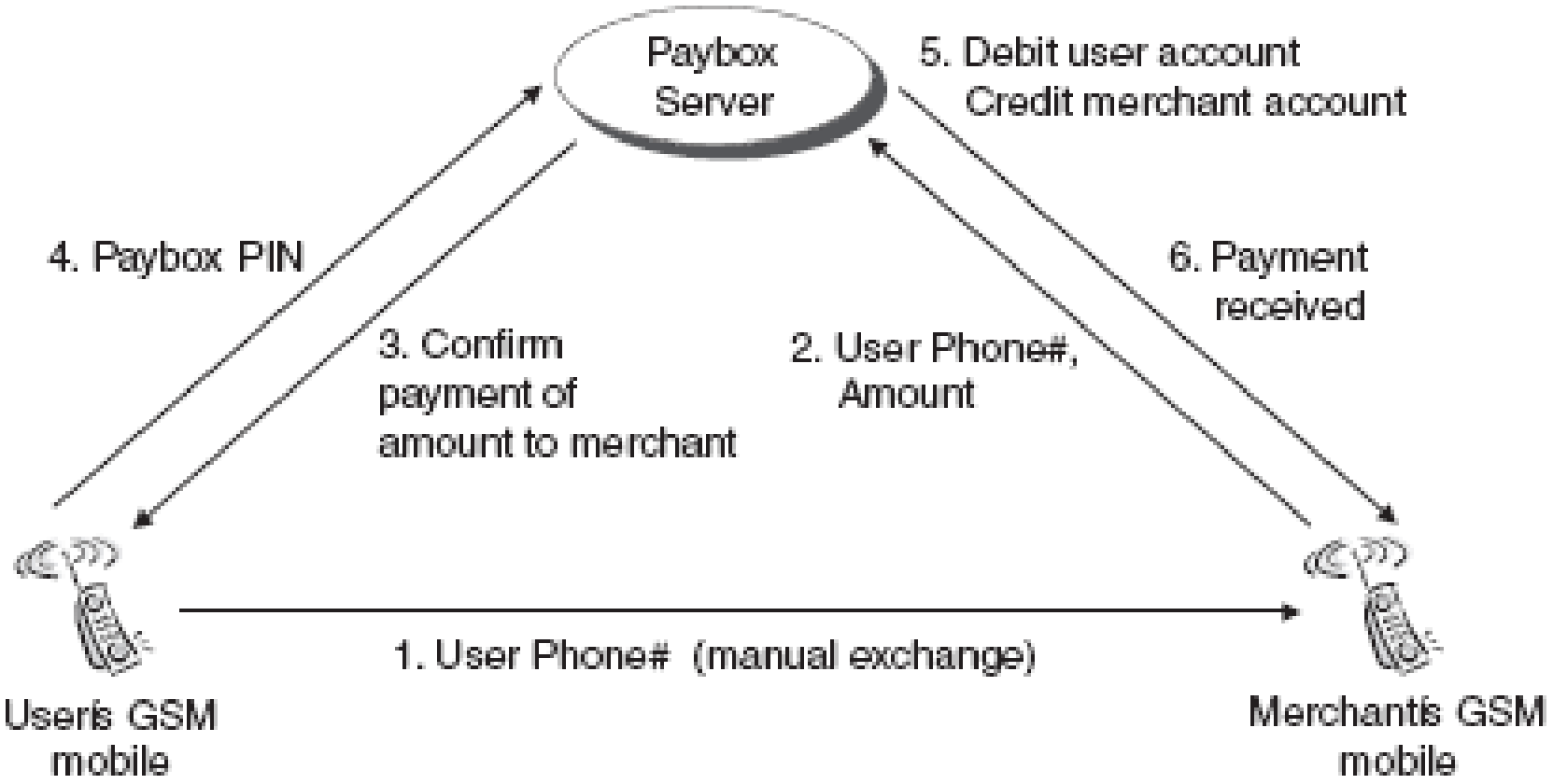
- Based on the called number, the MobilePay server then notifies the merchant that payment has been received and the goods can be released. Such notification can be sent by SMS/e-mail system or web based interface
- Deprecated?

2. PayBox - Sybase mPayment 365

FEATURES:

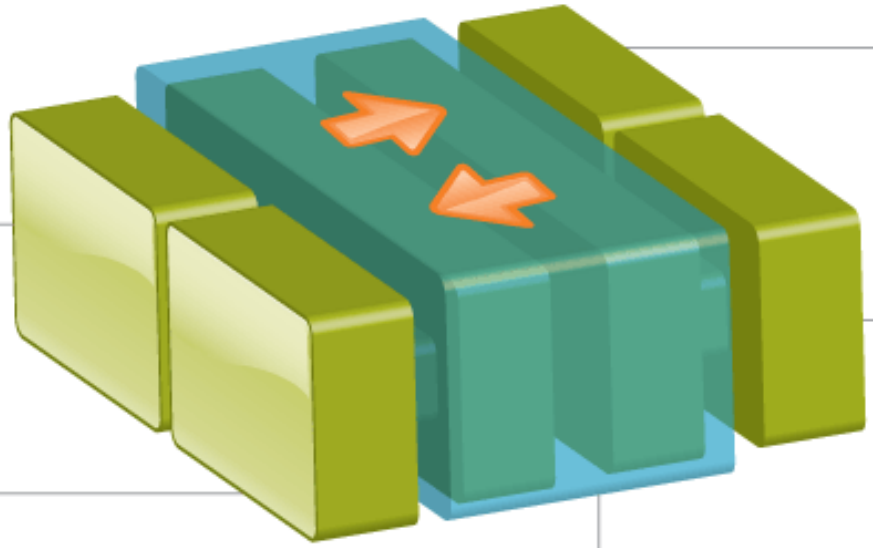
- Mobile 2 Mobile Payments
- Micro & Non-micro Payments, Bill Pay, Loan Payments
- GSM HTTPs over WAP/GPRS/UMTS/HSUPA/HSPDA/LTE – SSL connections on-line dialogues over GSM – additional costs
- PayBox deprecated – Continued by Sysbase mPayment 365 – a SAP company

2. PayBox - Sybase mPayment 365



2. PayBox - Sybase mPayment 365

http://www.sybase.com/mobileservices/financial-services/transactions



Mobile Channels and Services
- m-Banking Including P2P & Remittance
- m-Payment (POS, Mobile POS, Bill Pay)
- m-TopUp Including POS & Streetseller
- m-Content such as Ticketing, ...

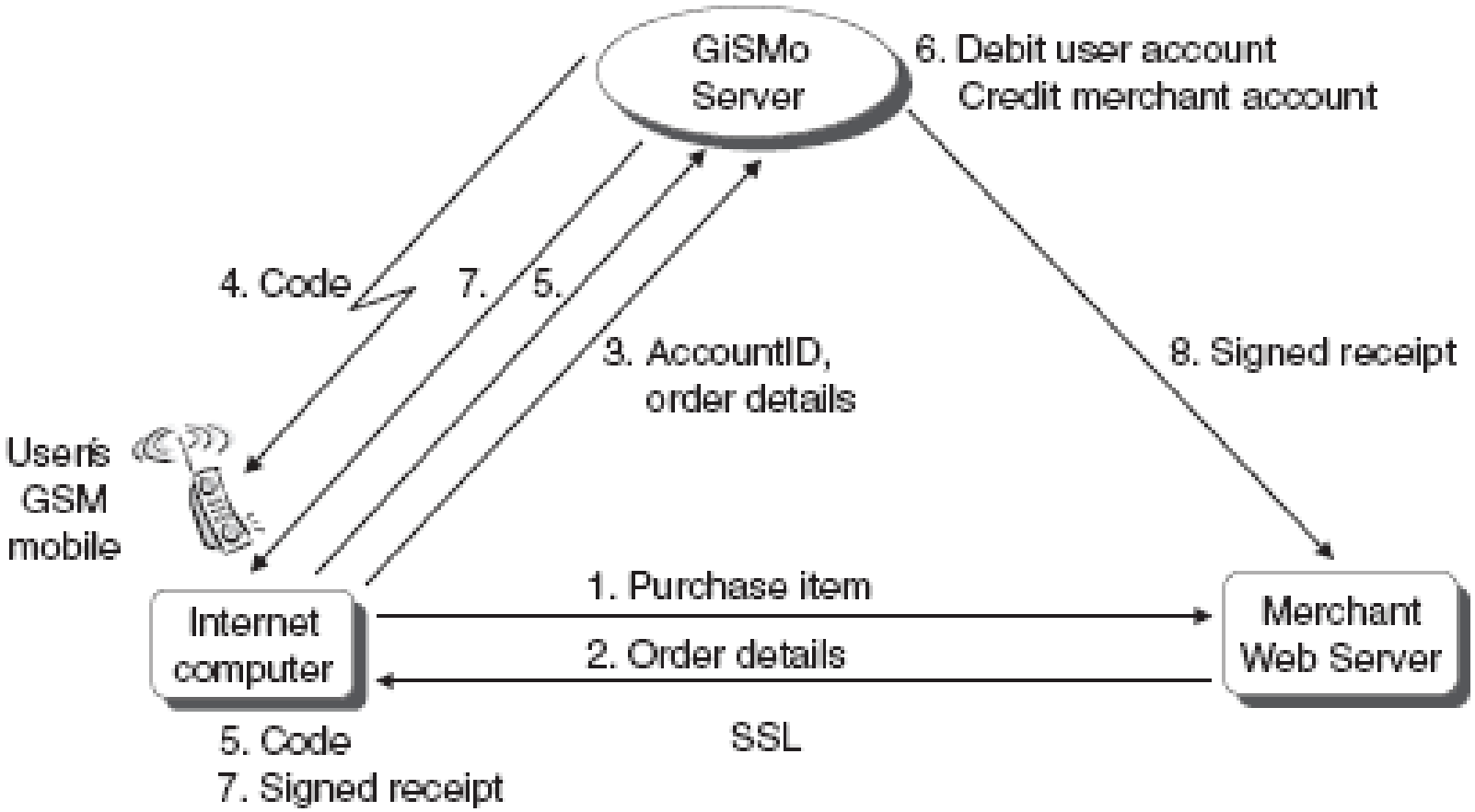
Electronic Payments
- Voucher
- POS Cash-In
- Bank Transfer
- Credit Card TopUp
- Transfer PayPal, ...

3rd Party Acceptance
- Bank Merchants
- Acquiring Partners
- Incoming Remittance

Card Payments
- Voucher Cards
- Credit Cards
- m-Moneybox Prepaid Card
- Salary or Gift Cards, ...

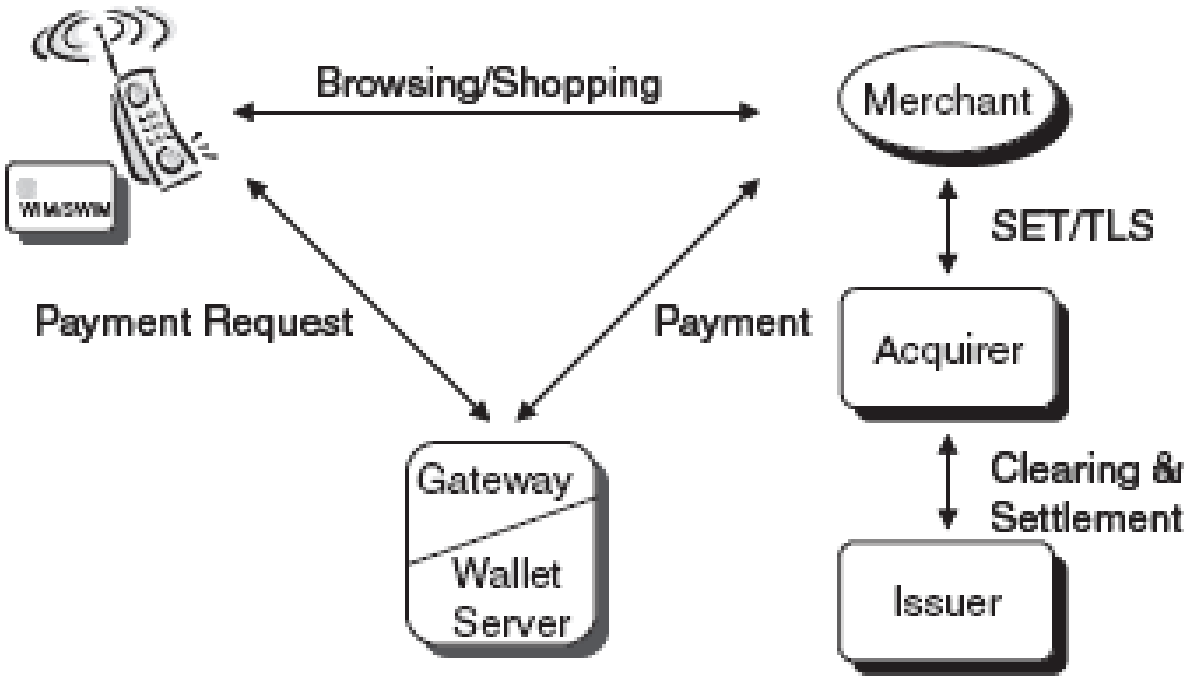
Sybase 365 Mobiliser Core
- Merchant Accounts
- Transaction Processing
- Consumer Accounts
- Data Warehousing and OLAP

3. GiSMo


GiSMo
Server
6. Debit user account
Credit merchant account
4. Code
7.
5.
3. AccountID,
order details
8. Signed receipt
User's
GSM
mobile
Internet
computer
1. Purchase item
2. Order details
Merchant
Web Server
5. Code
7. Signed receipt
SSL

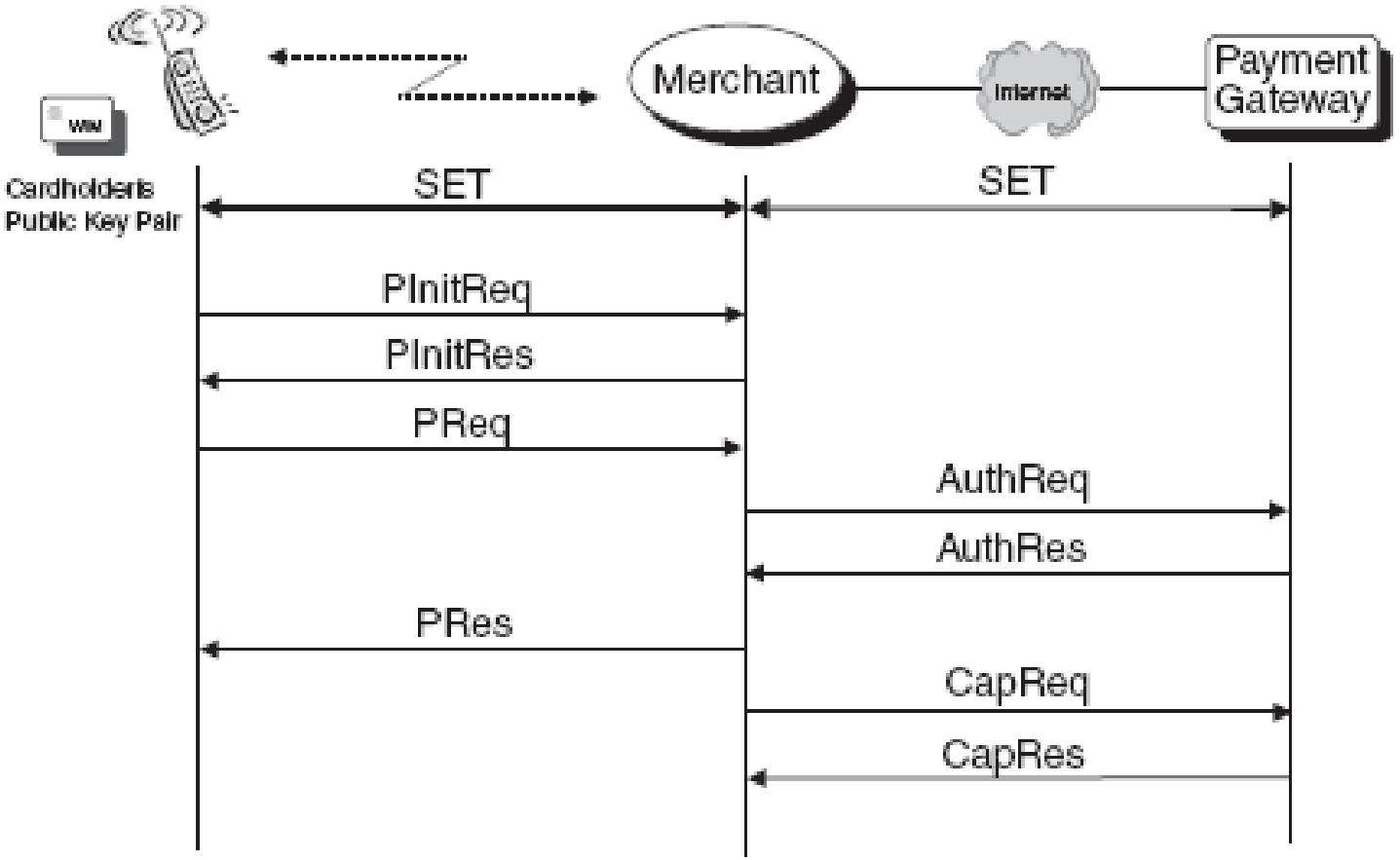
4. Mobile SET

SET Deprecated?, too much GSM connections and phone processing?



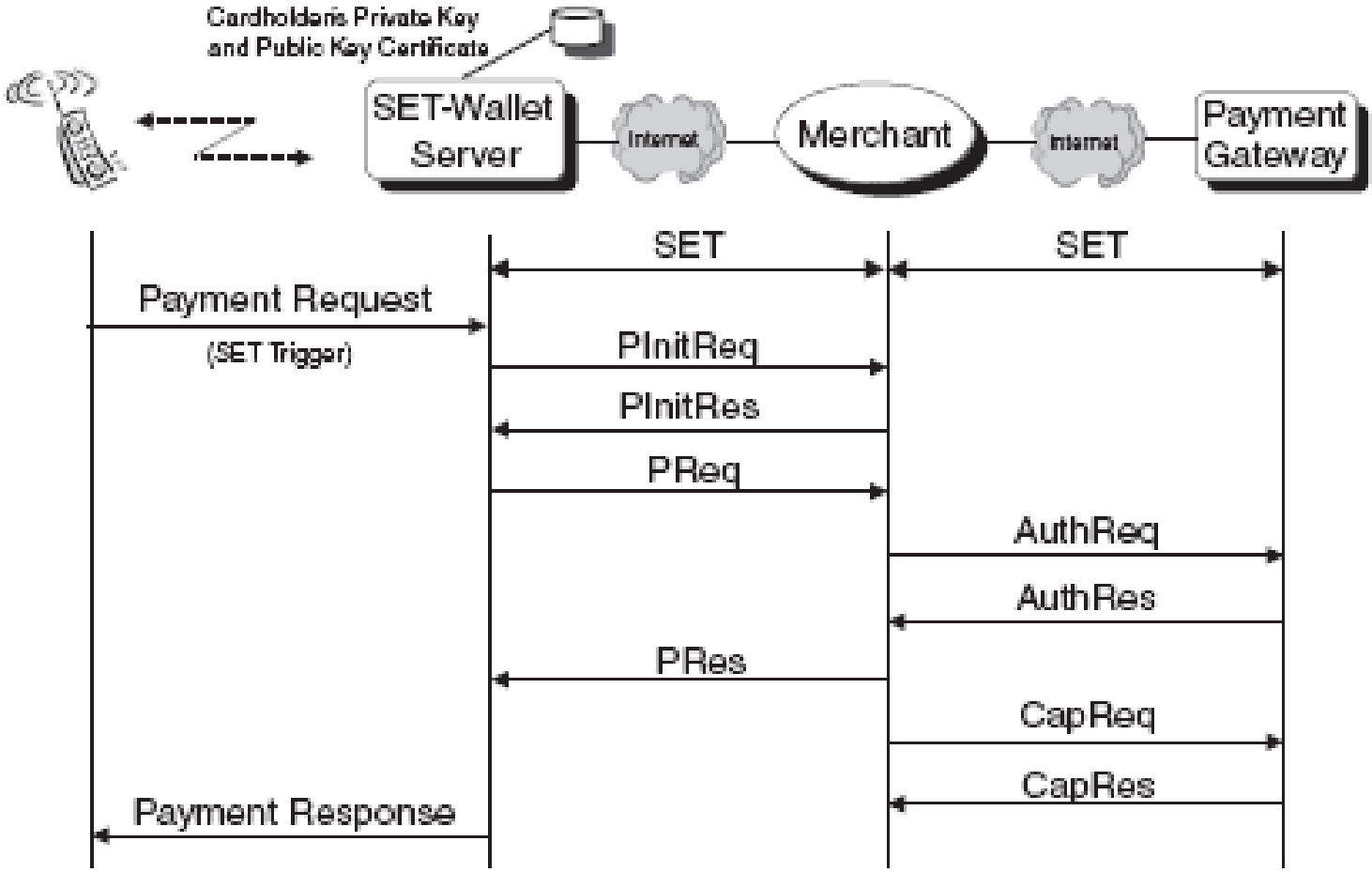
4. Mobile SET

4.1 Mobile Handset based SET Wallet



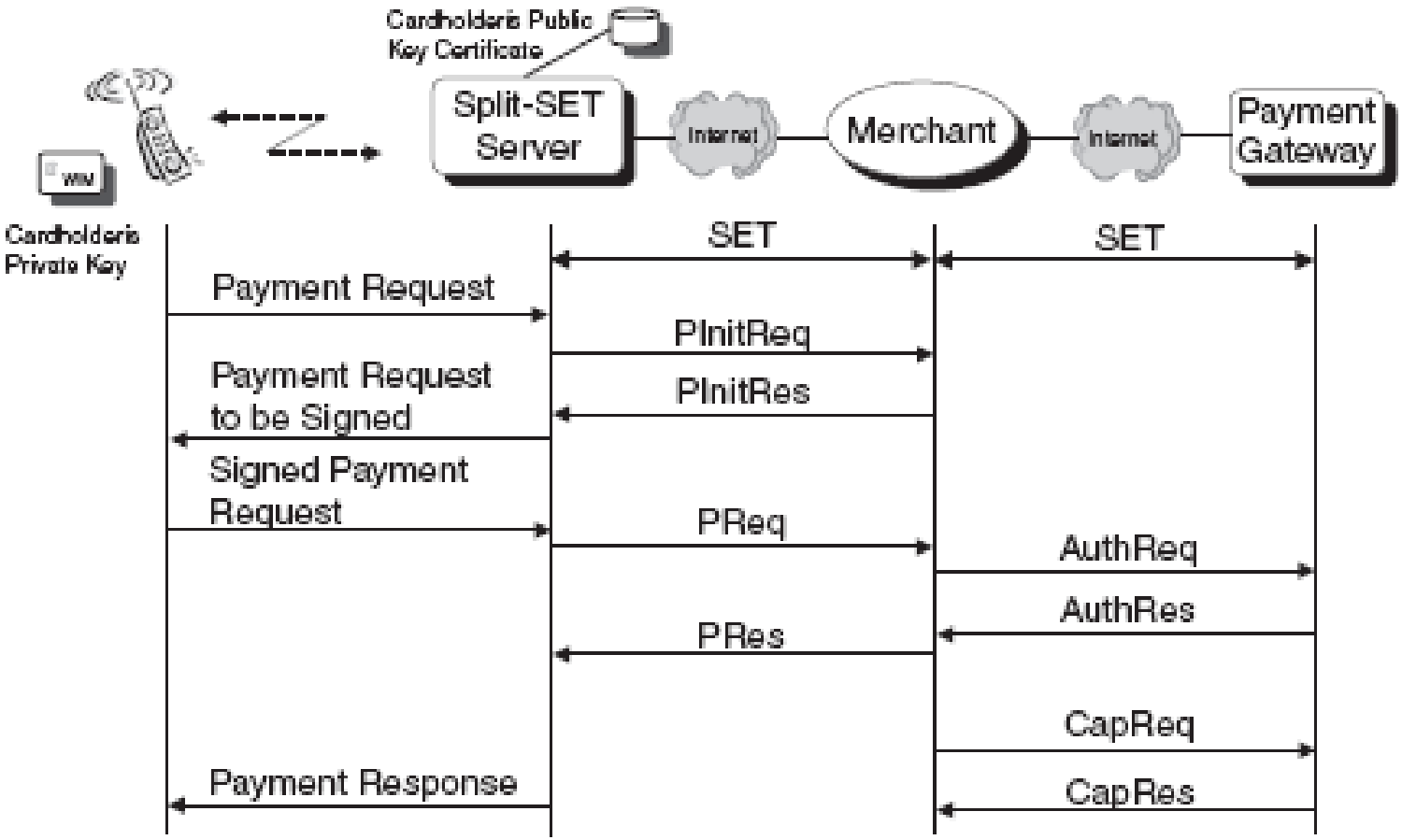
4. Mobile SET

4.2 Split-SET – SET Wallet SERVER



4. Mobile SET

4.3 Split-SET – SET Wallet SERVER + Handset Private Keys ⇔ 4.1 + 4.2



5. GeldKarte – Mobile Evolution?

http://www.geldkarte.de - SIM Cards ?

Account-linked Card

With the account-linked card, the microchip is fitted to the girocard (former ec card) or bank card. It is linked to the holder's personal bank account, which makes loading and unloading the card very simple.

Account-linked Cards

To load the GeldKarte, the holder transfers the desired amount to the card from his account. As with withdrawals from cash dispensers, the process is secured by a PIN number. The PIN must be typed in after the GeldKarte is inserted into the ATM. Only when the transaction has been authorized is the amount requested transferred to the GeldKarte chip.

5. GeldKarte – Mobile Evolution?

http://www.geldkarte.de - SIM Cards ?

Stand-alone GeldKarte or white card

The stand-alone GeldKarte is not linked to the holder's bank account. These cards are often issued by banks on behalf of third parties (e.g. public transport corporations), or as special editions (decorative cards, rather like telephone cards).

Example of a white card

These so-called white cards can be loaded at special terminals in bank branches, usually against payment in cash.

It is also possible to load the GeldKarte using other cards for authorization. To do this, the holder inserts the white card into one slot in a special terminal, and then inserts an ec or bank card into the other slot. The money is then transferred from the account linked to the girocard (former ec card) to the GeldKarte chip.

5. GeldKarte – Mobile Evolution?

http://www.geldkarte.de – Load & Unload e-Money

Loading and unloading an account-linked card

Loading is that easy!

If you want to load your account-linked card (i.e. the chip on a girocard (former ec card) or bank card), just look out for a cash dispenser (ATM)!

Loading is done in just the same way as withdrawing cash from that machine:

- Insert the card into the cash dispenser.
- Select "Load GeldKarte".
- Select amount
- Type in your PIN, and confirm.
- The amount selected will be transferred automatically from your bank account to the chip on card. It's as easy as that.

(Un)loading an account-linked GeldKarte at a cash dispenser

5. GeldKarte – Mobile Evolution?

http://www.geldkarte.de – Load & Unload e-Money

Unloading an account-linked card at the cash dispenser

You can unload your GeldKarte only at your own bank or Sparkasse – again, using the cash dispenser. It's just as simple:

- Insert your card.
- Select "Unload".
- The credit on your GeldKarte is transferred back to your bank account. That's it!

Loading stand-alone GeldKarte cards

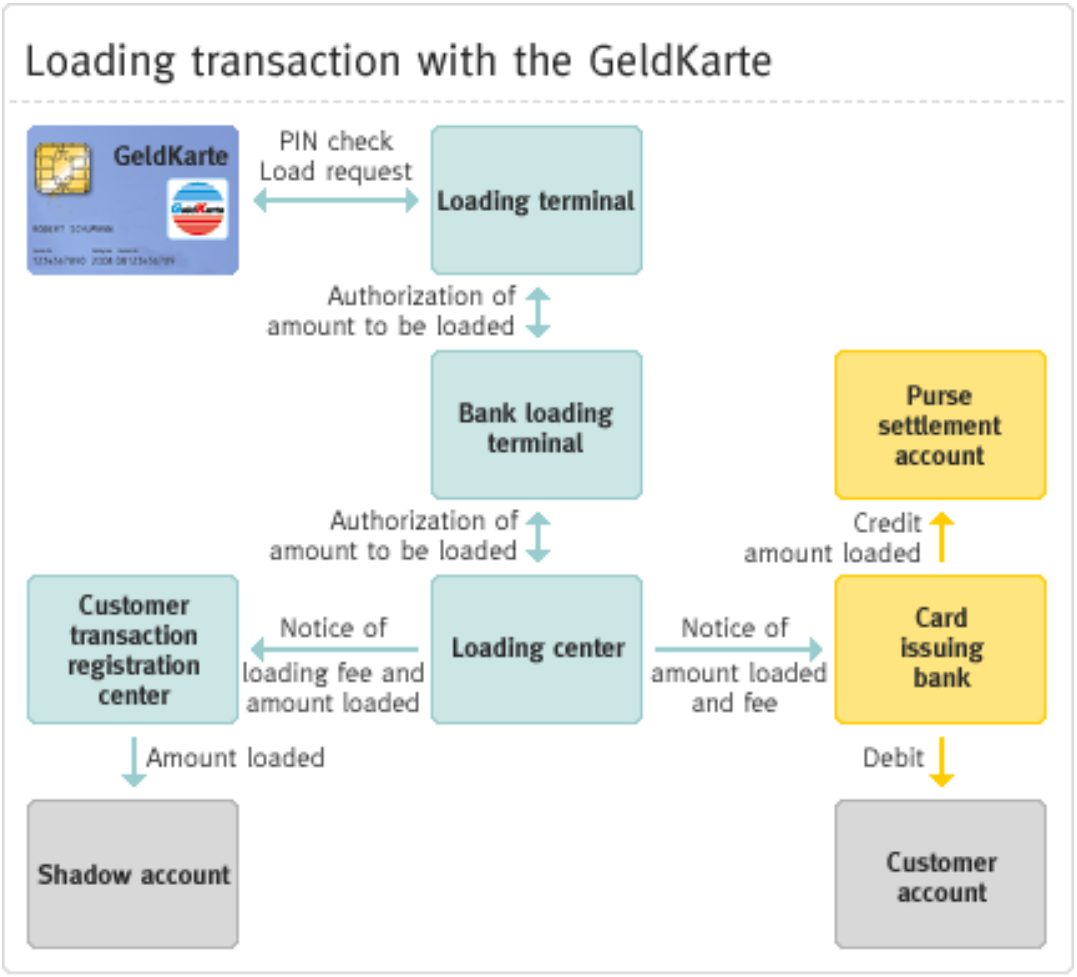
Loading against other cards

There are two ways a stand-alone GeldKarte card can be loaded:

Either at the bank counter against cash payment. Or you can use special loading terminals that accept other cards (girocards, bank cards). However, these terminals are not very common in Germany yet.

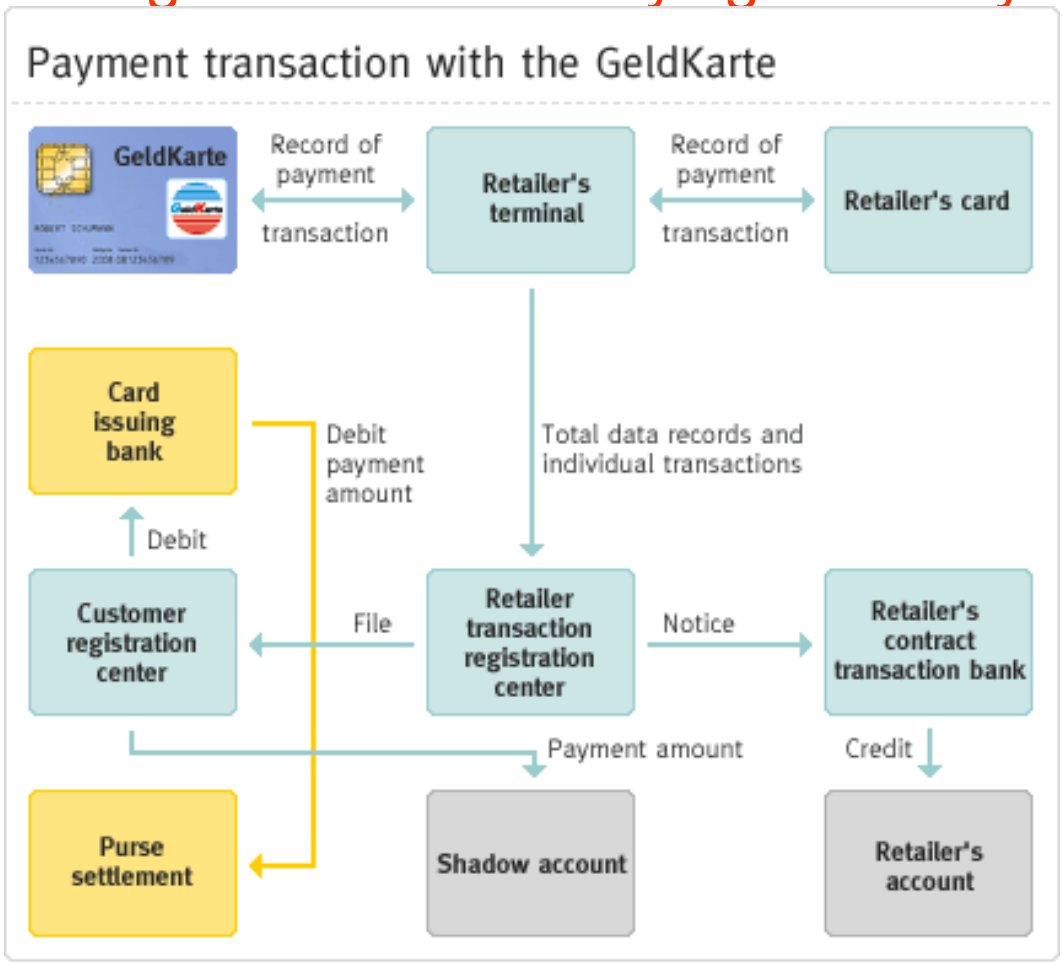
5. GeldKarte – Mobile Evolution?

http://www.geldkarte.de – Loading e-Money Procedure



5. GeldKarte – Mobile Evolution?

http://www.geldkarte.de – Paying e-Money Procedure



Source: Chipkarte d. dt. Wirtschaft / VÖB

2.6 – Mobile DRM Subway Payment - Case Study

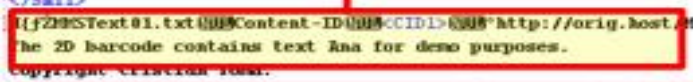
MMS with DRM

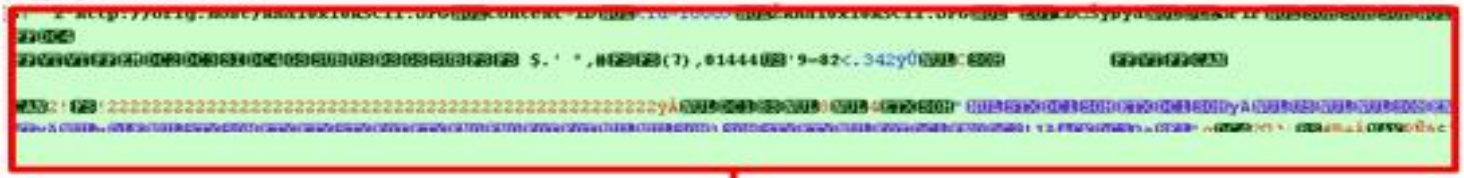
0. OMA DRM Part:
- simple "Forward Lock" type

1. MMS Header Part:
From: cristian.toma@nokiatoolkit
To: +40747012345/TYPE=PLMN
Subj: Subway 1 Journey Ticket

2. SMIL Part:
- text in MMSText01.txt
- image in Ana10x10ASCII.jpg

3. Text File Part:
This is a ticket with DataMatrix...

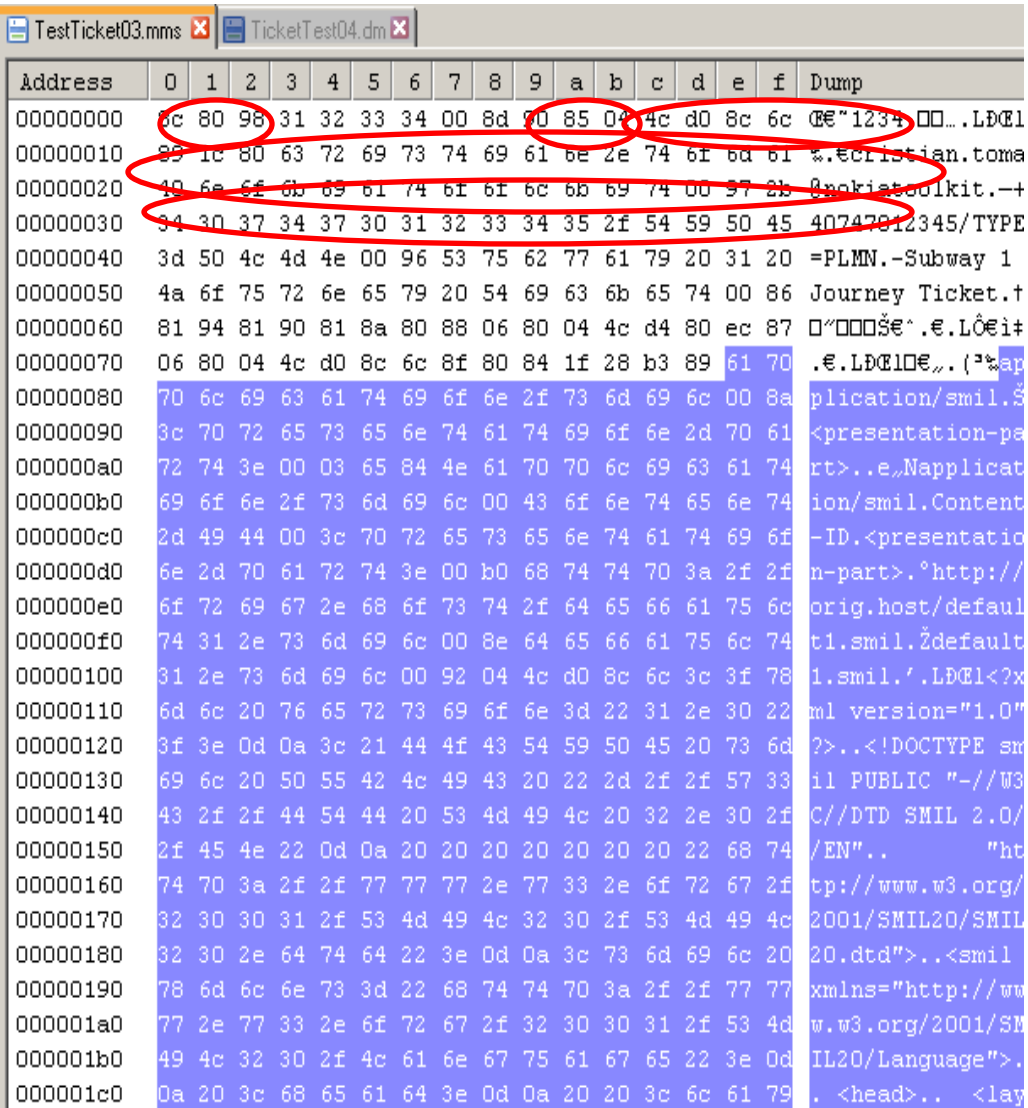




4. Image File Part:
Contains in JPEG compression DataMatrix 2D barcode which encodes "Ana"

2.6 – Mobile DRM Subway Payment - Case Study

Plain MMS Byte Level



8C 84 .... Message type - MMS Message of type: *m-retrieve-conf:* A message received by an MMS device containing MMS media content. For *m-send-req:* A message sent from an MMS device containing MMS media content should be the value: **0x8C 0x80**

....

8D 90 .... MMS Version

85 04 4C D0 8C 6C – time and date in TLV ASN.1 DER format with values in secs. from 1970 => aprox. 357982 hours => 14915 days => 40.86 years => year 2010 sometimes in November

89 1c 80 63 72 69 73 74 69 61 6e 2e 74 6f 6d 61 40 6e 6f 6b 69 61 74 6f 6f 6c 6b 69 74 – 0x1c bytes length of 'From' field – 0x89 with value: cristian.toma@nokiatoolkit

00 – the fields separator

97 2b 34 30 37 34 37 30 31 32 33 34 35 2f 54 59 50 45 3d 50 4c 4d 4e – tag 0x97 is field 'To' with the value: +40747012345/TYPE=PLMN

00 – the fields separator

2.6 – Mobile DRM Subway Payment - Case Study

Plain MMS Byte Level

TestTicket03.mms | TicketTest04.dm

Address	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	Dump
00000000	8c	80	98	31	32	33	34	00	8d	90	85	04	4c	d0	8c	6c	Œ€˜1234.□□….LÐŒl
00000010	89	1c	80	63	72	69	73	74	69	61	6e	2e	74	6f	6d	61	‰.€cristian.toma
00000020	40	6e	6f	6b	69	61	74	6f	6f	6c	6b	69	74	00	97	2b	@nokiatoolkit.—+
00000030	34	30	37	34	37	30	31	32	33	34	35	2f	54	59	50	45	40747012345/TYPE
00000040	3d	50	4c	4d	4e	00	96	53	75	62	77	61	79	20	31	20	=PLMN.–Subway 1
00000050	4a	6f	75	72	6e	65	79	20	54	69	63	6b	65	74	00	86	Journey Ticket.†
00000060	81	94	81	90	81	8a	80	88	06	80	04	4c	d4	80	ec	87	□″□□□Š€ˆ.€.LÔ€ì‡
00000070	06	80	04	4c	d0	8c	6c	8f	80	84	1f	28	b3	89	61	70	.€.LÐŒl□€„.(³‰ap
00000080	70	6c	69	63	61	74	69	6f	6e	2f	73	6d	69	6c	00	8a	plication/smil.Š
00000090	3c	70	72	65	73	65	6e	74	61	74	69	6f	6e	2d	70	61	<presentation-pa
000000a0	72	74	3e	00	03	65	84	4e	61	70	70	6c	69	63	61	74	rt>..e„Napplicat
000000b0	69	6f	6e	2f	73	6d	69	6c	00	43	6f	6e	74	65	6e	74	ion/smil.Content
000000c0	2d	49	44	00	3c	70	72	65	73	65	6e	74	61	74	69	6f	-ID.<presentatio
000000d0	6e	2d	70	61	72	74	3e	00	b0	68	74	74	70	3a	2f	2f	n-part>.°http://
000000e0	6f	72	69	67	2e	68	6f	73	74	2f	64	65	66	61	75	6c	orig.host/defaul
000000f0	74	31	2e	73	6d	69	6c	00	8e	64	65	66	61	75	6c	74	t1.smil.Ždefault
00000100	31	2e	73	6d	69	6c	00	92	04	4c	d0	8c	6c	3c	3f	78	1.smil.'.LÐŒl<?x
00000110	6d	6c	20	76	65	72	73	69	6f	6e	3d	22	31	2e	30	22	ml version="1.0"
00000120	3f	3e	0d	0a	3c	21	44	4f	43	54	59	50	45	20	73	6d	?>..<!DOCTYPE sm
00000130	69	6c	20	50	55	42	4c	49	43	20	22	2d	2f	2f	57	33	il PUBLIC "-//W3
00000140	43	2f	2f	44	54	44	20	53	4d	49	4c	20	32	2e	30	2f	C//DTD SMIL 2.0/
00000150	2f	45	4e	22	0d	0a	20	20	20	20	20	20	20	20	22	68	/EN".. "h
00000160	74	70	3a	2f	2f	77	77	77	2e	77	33	2e	6f	72	67	2f	tp://www.w3.org/
00000170	32	30	30	31	2f	53	4d	49	4c	32	30	2f	53	4d	49	4c	2001/SMIL20/SMIL
00000180	32	30	2e	64	74	64	22	3e	0d	0a	3c	73	6d	69	6c	20	20.dtd">..<smil
00000190	78	6d	6c	6e	73	3d	22	68	74	74	70	3a	2f	2f	77	77	xmlns="http://ww
000001a0	77	2e	77	33	2e	6f	72	67	2f	32	30	30	31	2f	53	4d	w.w3.org/2001/SM
000001b0	49	4c	32	30	2f	4c	61	6e	67	75	61	67	65	22	3e	0d	IL20/Language">.
000001c0	0a	20	3c	68	65	61	64	3e	0d	0a	20	20	3c	6c	61	79	. <head>.. <lay

96 53 75 62 77 61 79 20 31 20 4a 6f 75 72 6e 65 79 20 54 69 63 6b 65 74 – tag 0x96 is field 'Subject' with value: Subway 1 Journey Ticket

00 – the fields separator

…

SMIL Part - Synchronized Multimedia Integration Language to control the presentation of the parts of MMS message.

…

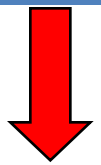
TEXT Part – from file 'MMSText01.txt' with content: "This is a ticket with DataMatrix as test.
The 2D barcode contains text Ana for demo purposes.
Copyright Cristian Toma."

…

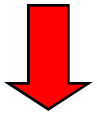
2.6 – Mobile DRM Subway Payment – Case Study
2D BATS - Barcode Automatic Ticketing System

QR Code Content

METICKET:**N:**-;**ADR:**-;**TEL:**-;**EMAIL:**-; **ENCINFO:***Base64(**Encrypt(line:blue;sdate:20101101Z112345;edate:20101201Z112345;nojrns:1)**)*;**PTSYS:**Boston Subway;**URL:** http://www.mbta.com/;

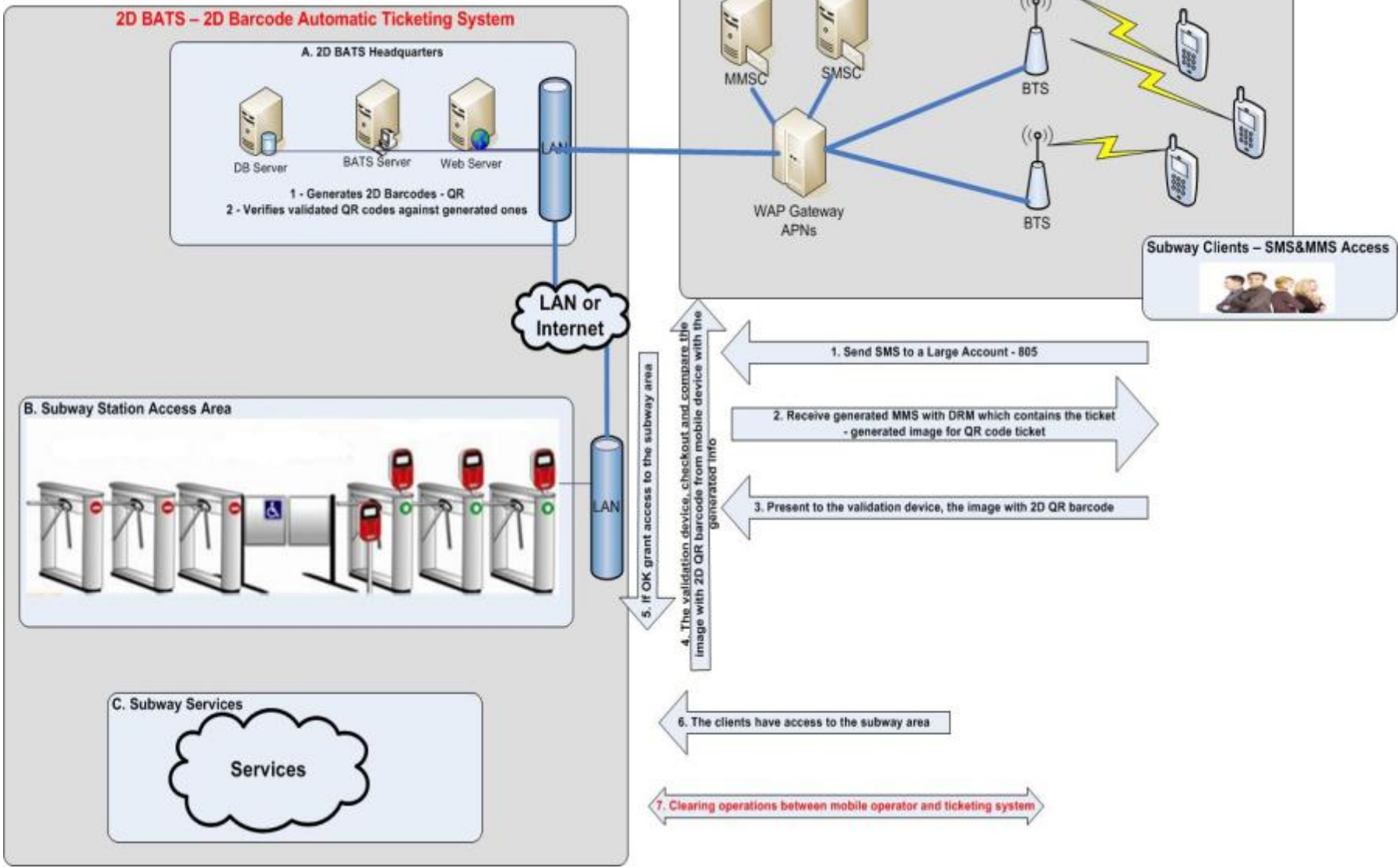


METICKET: **N:**-; **ADR:**-; **TEL:**-; **EMAIL:**-;
ENCINFO:bGluZTpibHVlO3NkYXRlOjIwMTAxMTAxWjExMjM0NTtlZGF0ZToyMDEwMTIwMVoxMTIzNDU7bm9qcm5zOjE=;**PTSYS:** Boston Subway;**URL:** http://www.mbta.com/;



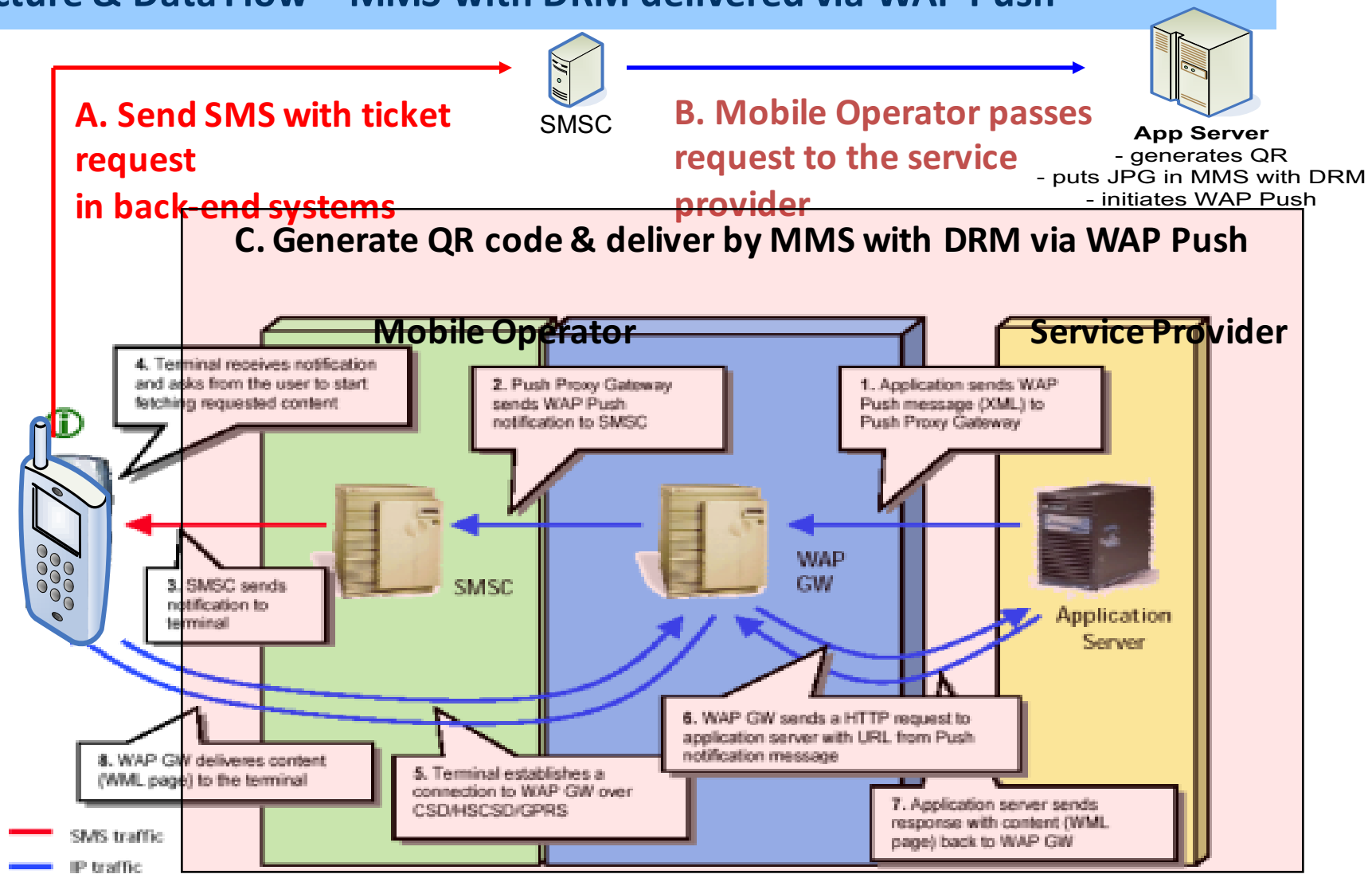
2.6 – Mobile DRM Subway Payment – Case Study
2D BATS - Barcode Automatic Ticketing System

Architecture & Data Flow



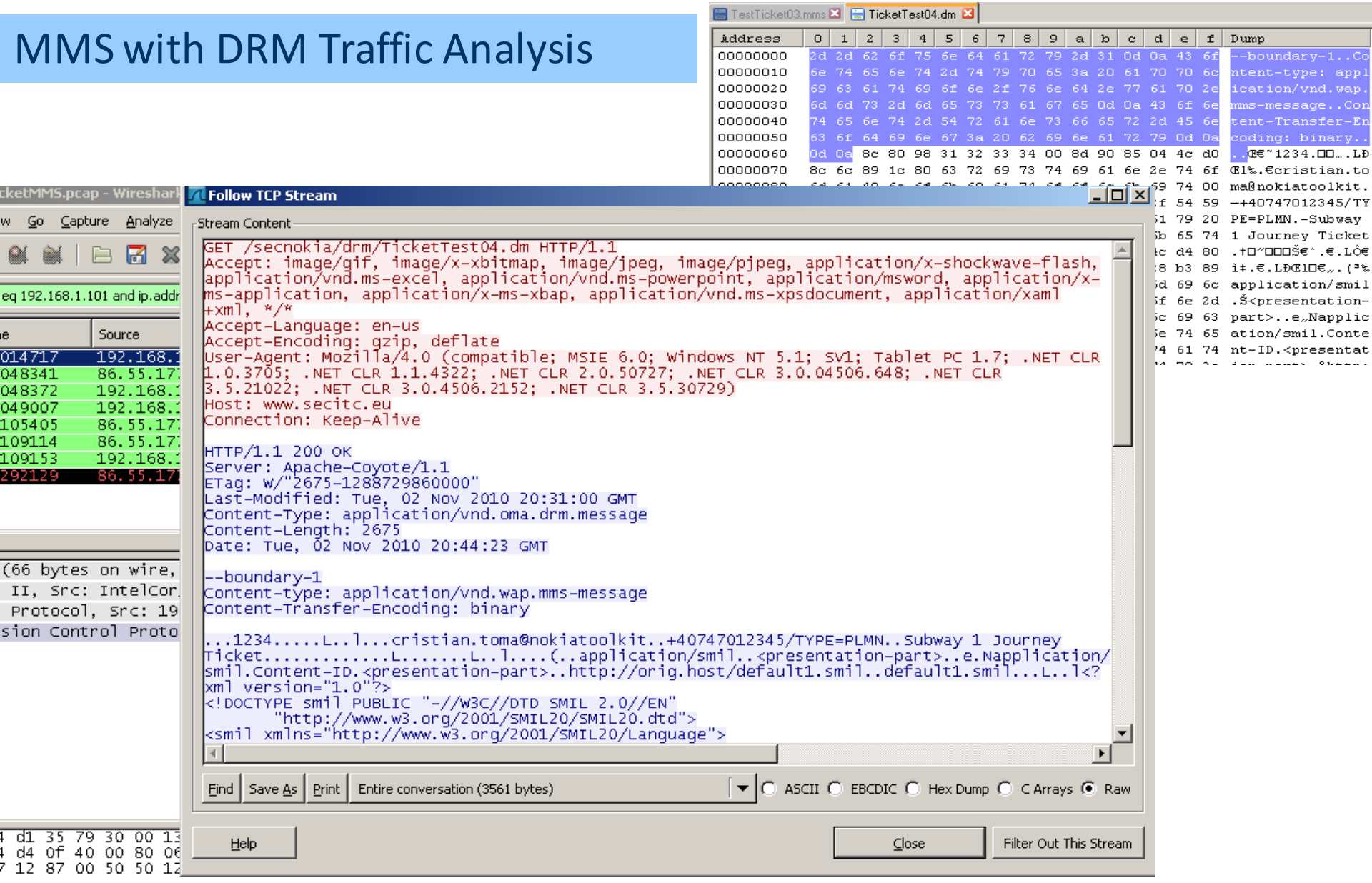
2.6 – Mobile DRM Subway Payment – Case Study
2D BATS - Barcode Automatic Ticketing System

Architecture & Data Flow – MMS with DRM delivered via WAP Push



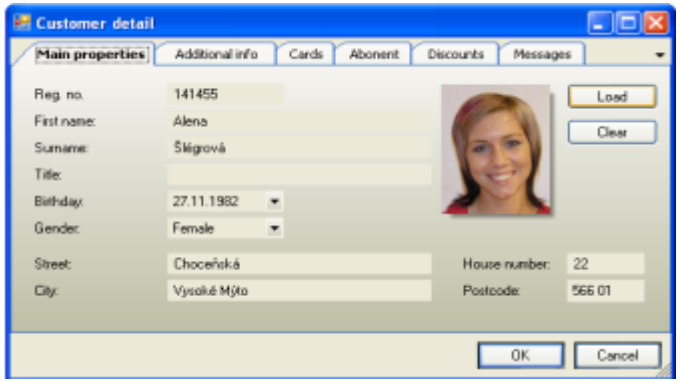
2.6 – Mobile DRM Subway Payment – Case Study
2D BATS - Barcode Automatic Ticketing System

MMS with DRM Traffic Analysis



2.6 – Mobile DRM Subway Payment – Case Study
2D BATS - Barcode Automatic Ticketing System

POS – Distribution Chain



POS Application

Vending Machines – Necessity?

NXP Mifare 1K/4K RFID Cards

Public Transport Subscriptions & Tickets

QR

Data Matrix

Public Transport Subscriptions & Tickets

2.6 – 2D BATS - Barcode Automatic Ticketing System

POS – Distribution Chain & Payments

1. Online Web – print 2D barcodes from the e-mail

2. Online m-Phone – 2D barcodes in WAP Push/MMS

3. ATM – 2D barcodes print on the receipt

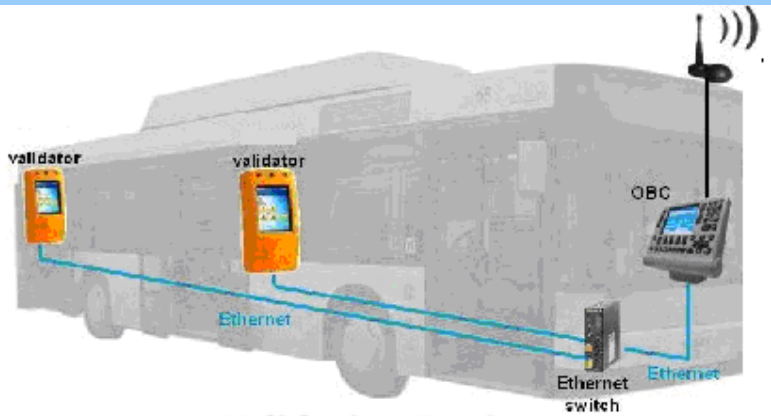
QR

Data Matrix

Public Transport Subscriptions & Tickets

2.6 – 2D BATS - Barcode Automatic Ticketing System

POS – Validation & Inspection – Extension to the Vehicles



Validation Device

- Reader RFID
- CMOS Camera
- RFID Cards & 2D barcodes reader application
- Ethernet – via OBC (Wi-Fi, GPS, GSM)

NXP Mifare 1K/4K RFID Cards

Public Transport Subscriptions & Tickets

Inspector

- Reader RFID
- CMOS Camera
- RFID Cards & 2D barcodes reader application
- GPS, Wi-Fi, GSM

QR

Data Matrix

Public Transport Subscriptions & Tickets

2.6 – Mobile DRM Subway Payment – Case Study
2D BATS - Barcode Automatic Ticketing System

Security Issues

QR Code – Automatic Data Acquisition – RAM/EEPROM of Validation Devices + Reed Solomon vs. BCH for error correction code

Encryption/Decryption + Encoding Scheme – QR JPEG contains AES-Rijndael + Base64 Encoding

NFC - Security

Security of Delivery – MMS with DRM contains SMIL with JPEG – QR code, sent via WAP Push => App Security + Mobile Operator Security

Security of Billing & Clearing – between Subway Subsidiaries, Mobile Operator & Applications Service Provider

Security of Communications – security of the communication between:

- SMSC and App Server using SS7 signaling
- App Server and Push Proxy Gateway via WAP GW – for WAP Push
- App Server and MMSC via WAP GW

2.6 – Mobile Payment Systems – Improvement for BATS

Mobile Ticket - How Mobile Ticket delivery works?

Delivery of tickets to mobile phones can be done in a variety of ways:

- Text messaging (SMS) - visual inspection or OCR
- USSD/Text messaging with WAP Push - visual inspection or OCR
- Picture messaging (SMS, EMS, WAP Push and MMS) - usually uses a barcode – 2D barcode – QR
- ***Dedicated Mobile Application*** - which can store and render barcodes delivered via SMS, GPRS, Bluetooth, IRDA or RFID. Barcodes rendered on the device by a dedicated application have the advantage of being full screen without clutter, meaning faster and more successful scanning. A dedicated mobile application can also help the user to organise and sort their tickets better than when an SMS or MMS inbox is full of similar tickets, which is especially useful for transport tickets.
- Device RFID - This is the method proposed under the Near Field Communication (NFC) specification but not yet in general use, except of Japanese Osaifu-Keitai

http://www.intercom.ee/mobile-ticketing-works

2.6 – Mobile Payment Systems – Improvement for BATS

Mobile Ticket - Delivery?

QR Code

DOV

NFC

Security Issues Summary
for easy sharing

Section Conclusions

QR Code – Automatic Data Acquisition – RAM/EEPROM of Validation Devices + Reed Solomon vs. BCH for error correction code

Encryption/Decryption + Encoding Scheme – QR JPEG contains AES-Rijndael + Base64 Encoding

NFC - Security

Security of Delivery – MMS with DRM contains SMIL with JPEG – QR code, sent via WAP Push => App Security + Mobile Operator Security

Security of Billing & Clearing – between Subway Subsidiaries, Mobile Operator & Applications Service Provider

Security of Communications – security of the communication between:

- SMSC and App Server using SS7 signaling
- App Server and Push Proxy Gateway via WAP GW – for WAP Push
- App Server and MMSC via WAP GW

Share knowledge, Empowering Minds

Communicate & Exchange Ideas

Questions & Answers!

But wait…

There's More!

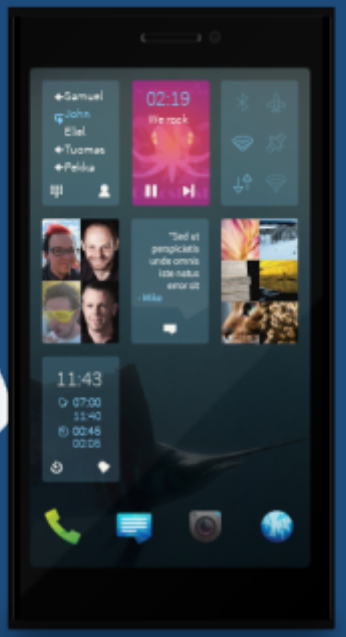
What about the M-Payment Services standards?

» Check out the
m-Payments Specs

2D Barcode URL

http://acs.ase.ro | **http://www.ism.ase.ro**

Scan the Tag
to get the web
Mobile
Address



Thanks!