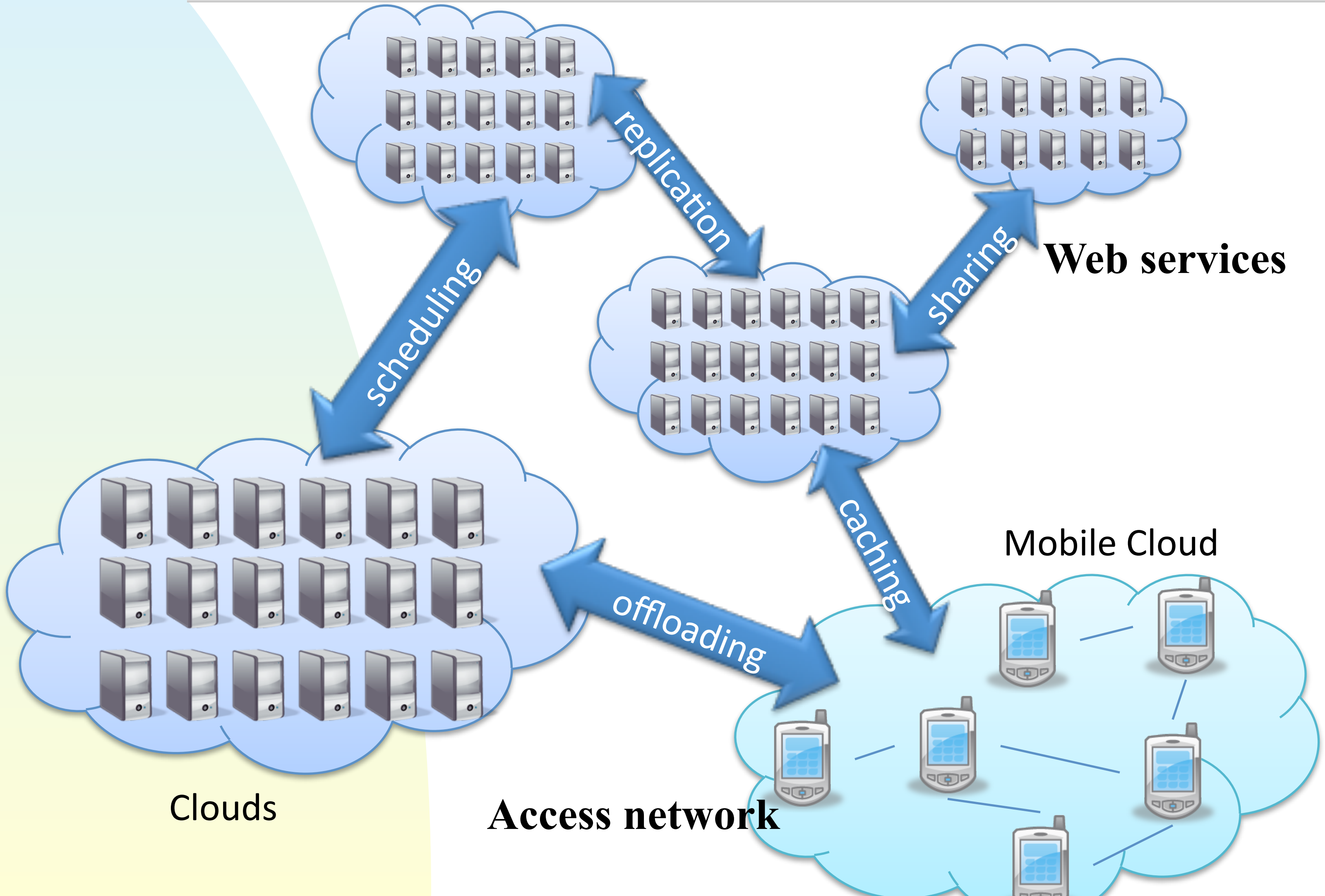
Mobile Middleware

Support Technologies

Contents

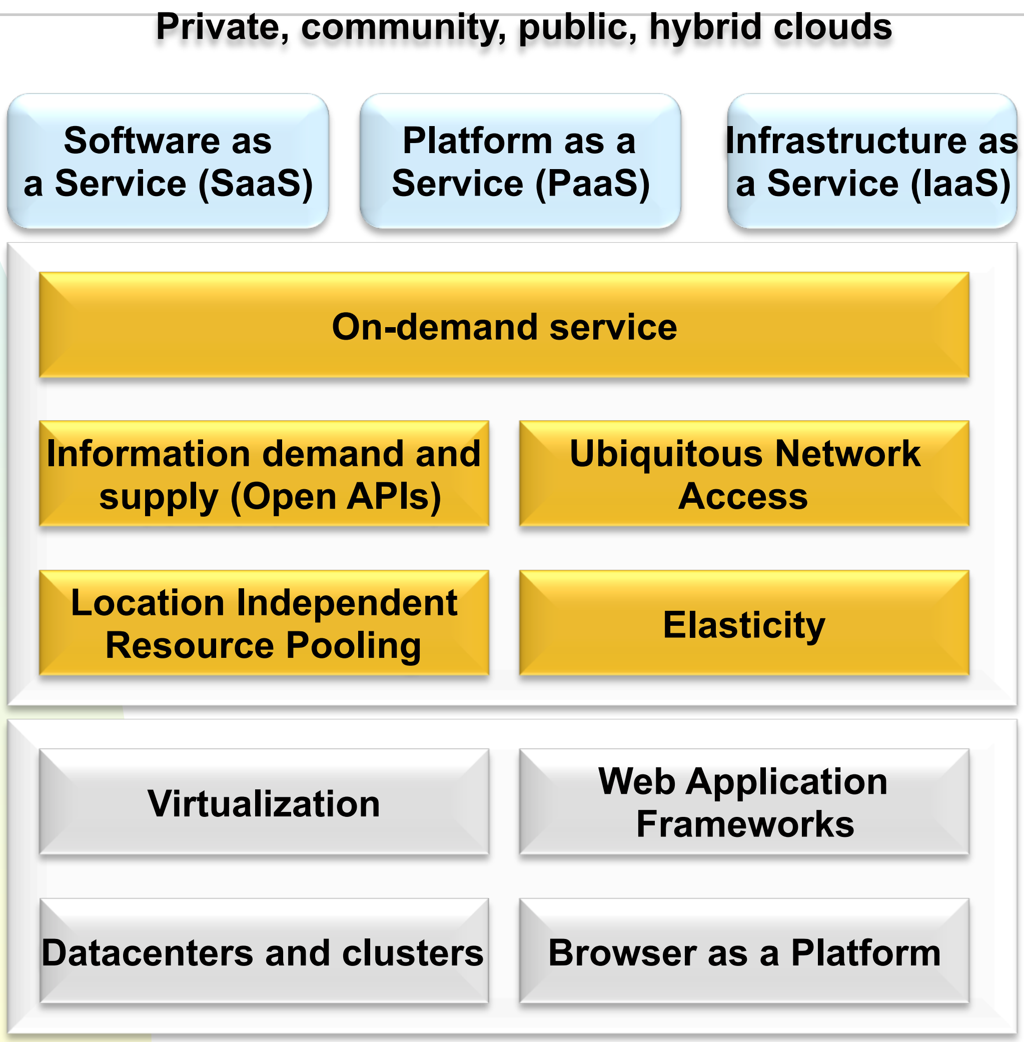
Environment



Cloud Computing

- The services of Cloud computing can be divided into three categories:
- 1. **Software-as-a-Service (SaaS)**, in which a vendor supplies the hardware infrastructure, the software product and interacts with the user using a portal (software on demand, pay-as-you-go).
- 2. **Platform-as-a-Service (PaaS)**, in which a set of software and development tools are hosted by a provider on the provider's infrastructure, for example, Google's AppEngine.
- 3. **Infrastructure-as-a-Service (IaaS)**, which involves virtual server instances with unique IP addresses and blocks of on-demand storage, for example, Amazon's Web services infrastructure.

Cloud Computing


Private, community, public, hybrid clouds
Software as a Service (SaaS)
Platform as a Service (PaaS)
Infrastructure as a Service (IaaS)
On-demand service
Information demand and supply (Open APIs)
Ubiquitous Network Access
Location Independent Resource Pooling
Elasticity
Virtualization
Web Application Frameworks
Datacenters and clusters
Browser as a Platform

Offloading

- The offloading of a mobile computing task is a trade-off between the energy used for local processing and the energy required for offloading the task, uploading its data, and downloading the result, if necessary.

- One can express the offloading energy trade-off as follows:
  - Etrade = Elocal − Edelegate > 0, where Elocal is the energy used for complete local execution, and Edelegate is the energy used if the task is offloaded from the perspective of the mobile device.

- If Etrade is greater than zero, then there is an energy benefit for delegating the task to the cloud.

What can be offloaded?

- Content processing and transformations
  - Example: Javascript processing in OperaMini
- Completion notification and mobile push
- Application execution
  - Google docs, Windows Office
- Connection management
  - BitTorrent!
  - Large downloads
- Speech recognition
  - Siri
- Positioning (A-GPS)

Example of Offloading: Indexing

- Implemented with Dessy: mobile desktop search
- To offload indexing
  - Transmit entire file to the cloud service
  - Wait for response
  - Receive file summary
- High energy savings can be obtained when offloading CPU-intensive tasks
- With N900 and WLAN 700kB/s: 96.5% savings!
  - 200 000 words, 1 MB file
  - With WLAN 100kB/s this is reduced to 83.7%

Virtualization

- Virtualization system is a framework that combines or divides computing resources to present a *transparent* view of one or more environments
  - Hardware/software partitioning (or aggregation)
  - Partial or complete machine simulation
  - Emulation (can be partial or complete)
  - Time-sharing

Mobile virtualization

- Server Based Virtualization of Desktop Infrastructure
- Moving the desktop to a virtualized image in the data center allows the complex components to be protected and componentized
  - Workload isolation and migration
  - Application virtualization
- Virtualization is a possible solution to the fragmentation problem
- http://www.vmware.com/technology/mobile/
- Virtualization for offloading
  - Running applications in a base station, nearby server, in the cloud
  - Research proposals

Session Initiation Protocol (SIP)

- An Application-layer control (signaling) protocol for creating, modifying and terminating sessions with one or more participants.
- Sessions include Internet multimedia conferences, Internet telephone calls and multimedia distribution.
- Members in a session can communicate via multicast or via a mesh of unicast relations, or a combination of these.
- Text based, model similar to HTTP.

- Current system: Signalling System 7 (SS7).
  - Circuit-switched, SIGTRAN with SCTP for IP

SIP History

- Mid-1990s, emerged from the research of Dr. Henning Schulzrinne, Columbia University, on Multi-party Multimedia Session Control (MMUSIC)
- 1996, submitted to the Internet Engineering Task Force (IETF) and developed by the SIP Working Group
- 1999, first published as IETF RFC 2543
- 2000, selected by 3GPP
- 2002, RFC 3261 (with further supplements in other RFCs)

SIP features

- User location: determination of the end system to be used for communication
- User capabilities: determination of the media and media parameters to be used
- User availability: determination of the willingness of the called party to engage in communications
- Call setup: "ringing", establishment of call parameters at both called and calling party
- Call handling: including transfer and termination of calls

State in SIP

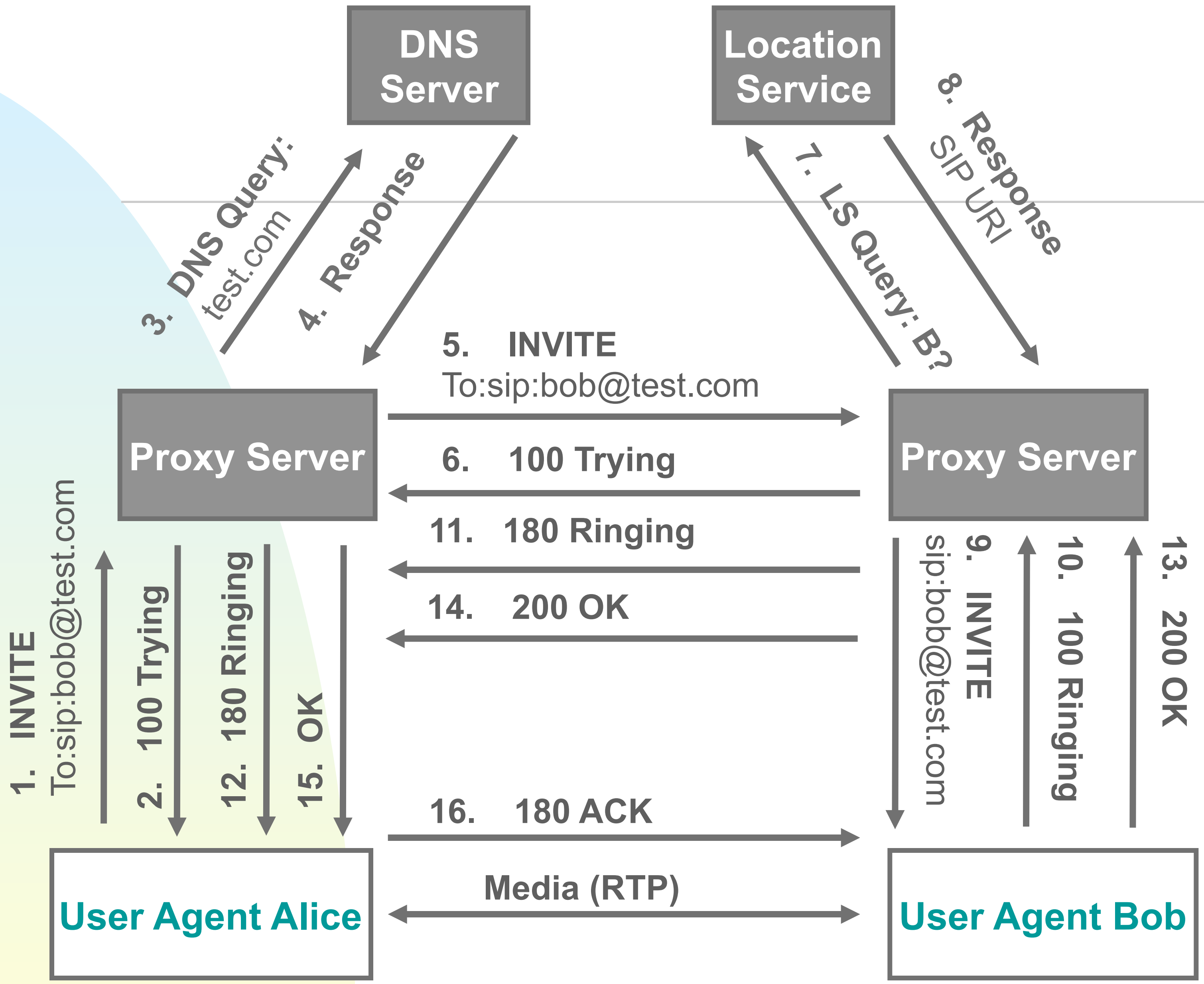
- SIP places state in the end devices
  - End-to-end
  - If end device fails, then the session fails
- Network elements may or may not have state
- Proxies are used for discovery, forwarding, relaying
- SIP uses transactions in message delivery
  - Internal state, timers
  - Can use UDP, TCP, SCTP, etc.

SIP messages

- Start Line (Request Line or Status Line)
  - message type (method type & URI in requests, and response code in responses)
  - protocol version
- Headers
  - fields to convey message attributes
    - To,From, sequence number, timestamp, etc.
  - can span multiple lines, appear multiple times, take multiple comma-separated values
- Body (Content)
  - to describe the session to be initiated
  - to contain opaque textual or binary data

SIP

1. SIP Addressing (SIP URI) and registering SIP addresses
2. Locating a SIP Server
3. Sending SIP Requests: SIP Transactions
4. SIP Methods
5. SIP Responses
6. Subsequent Requests and Responses


DNS Server
Location Service
3. DNS Query: test.com
4. Response
7. LS Query: B?
8. Response SIP URI
5. INVITE To:sip:bob@test.com
6. 100 Trying
11. 180 Ringing
14. 200 OK
Proxy Server
Proxy Server
1. INVITE To:sip:bob@test.com
2. 100 Trying
12. 180 Ringing
15. OK
9. INVITE sip:bob@test.com
10. 100 Ringing
13. 200 OK
16. 180 ACK
Media (RTP)
User Agent Alice
User Agent Bob

SIP and Mobility

- Designed to support various forms of mobility
- IP address changes when DHCP assigns a new one, vertical handoff, roaming between operators
- IP address change requires a re-INVITE with the new IP address, results in 200 OK and ACK. RTP stream will then be established for the correct IP address.
- Can be handled also with:
  - Host Identity Protocol (indirection in the protocol stack), requires changes to kernel
  - SCTP with multihoming
- Details in SIP RFC
  - http://tools.ietf.org/html/rfc3665

IMS

- The **IP Multimedia Subsystem** Provides Multimedia Services Across Networks (fixed & mobile), such as:
  - Instant Messaging, Video Sharing, Push-To-Talk, Gaming, Video Conferencing
- Designed initially for GPRS and circuit switched. Upgraded later for 3G and LTE.
- IMS Uses SIP protocol To Setup Multimedia Sessions Over IP Network
- SIP is a signalling protocol to:
  - Locate user given SIP Universal Resource Identifier (URL) (e.g., sip:jane@isp.com)
  - Set up session and negotiate its parameters

IMS Proxies I

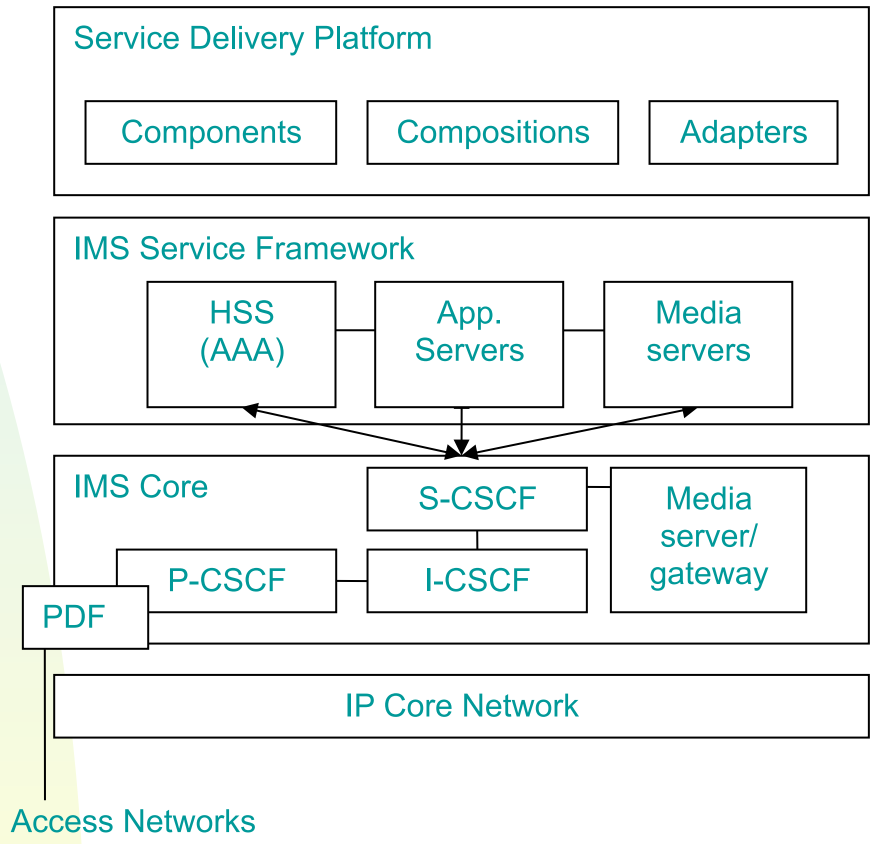
- CSCF = Call State Control Function
- A *Proxy-CSCF* (P-CSCF) is a SIP proxy that is the first point of contact for the IMS terminal
  - it is assigned to an IMS terminal during registration, and does not change for the duration of the registration
  - it sits on the path of all signalling messages, and can inspect every message
  - it authenticates the user and establishes an IPsec security association with the IMS terminal.
  - it can also compress and decompress SIP messages using SigComp, which reduces the round-trip over slow radio links
  - it may include a Policy Decision Function (PDF), which authorizes media plane resources e.g. quality of service (QoS) over the media plane. It's used for policy control, bandwidth management, etc. The PDF can also be a separate function.
  - it also generates charging records

IMS Proxies II

- A *Serving-CSCF* (S-CSCF) is the central node of the signalling plane
  - it is always located in the home network. It uses Diameter Cx and Dx interfaces to the HSS to download and upload user profiles — it has no local storage of the user data. All necessary information is loaded from the HSS
  - it handles SIP registrations, which allows it to bind the user location (e.g. the IP address of the terminal) and the SIP address
  - it sits on the path of all signaling messages, and can inspect every message
  - it decides to which application server(s) the SIP message will be forwarded, in order to provide their services
  - it enforces the policy of the network operator

- An *Interrogating-CSCF* (I-CSCF) is another SIP function located at the edge of an administrative domain

IMS

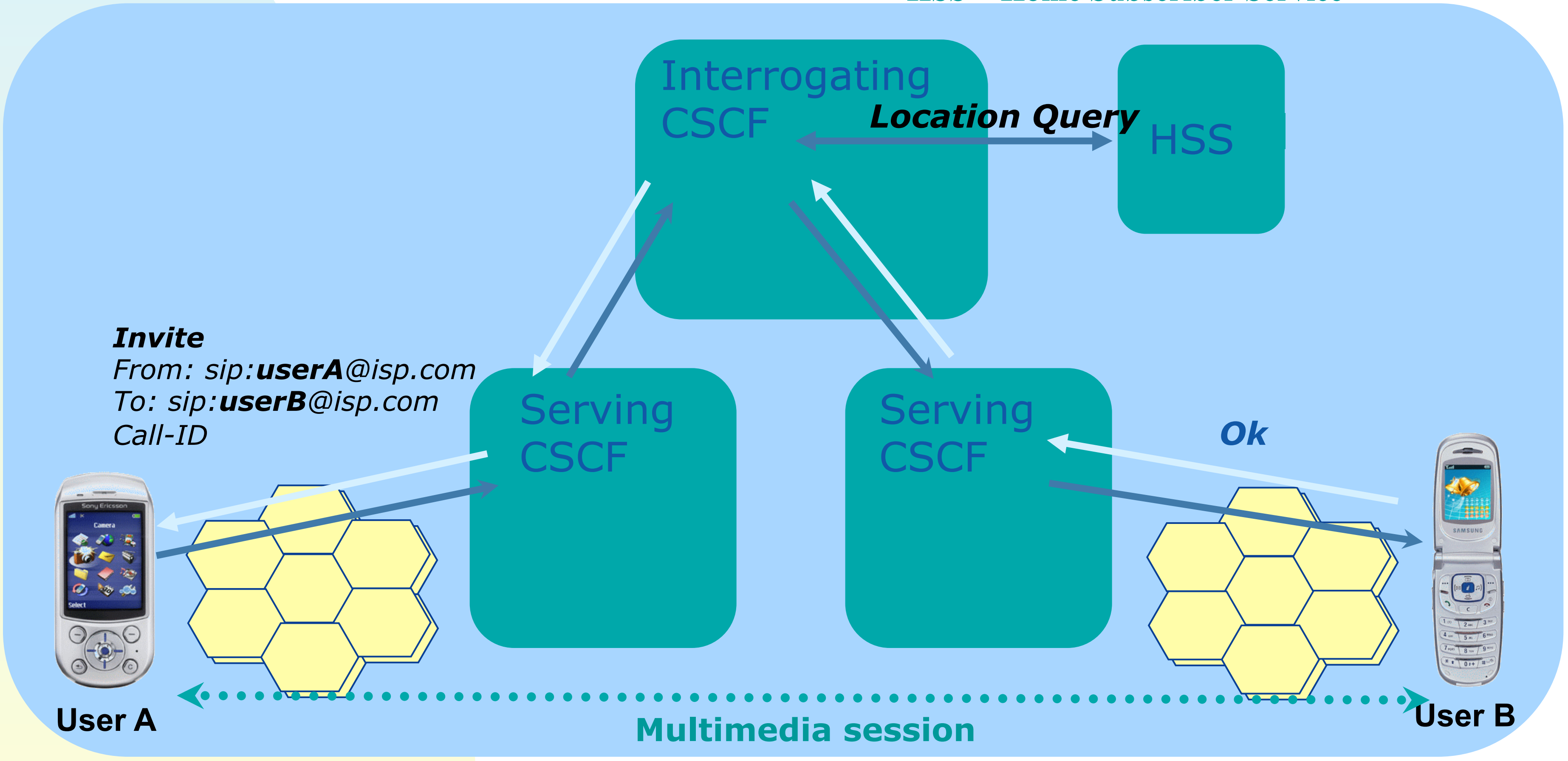


Service Delivery Platform
Components
Compositions
Adapters
IMS Service Framework
HSS (AAA)
App. Servers
Media servers
IMS Core
S-CSCF
Media server/ gateway
P-CSCF
I-CSCF
PDF
IP Core Network
Access Networks

IMS

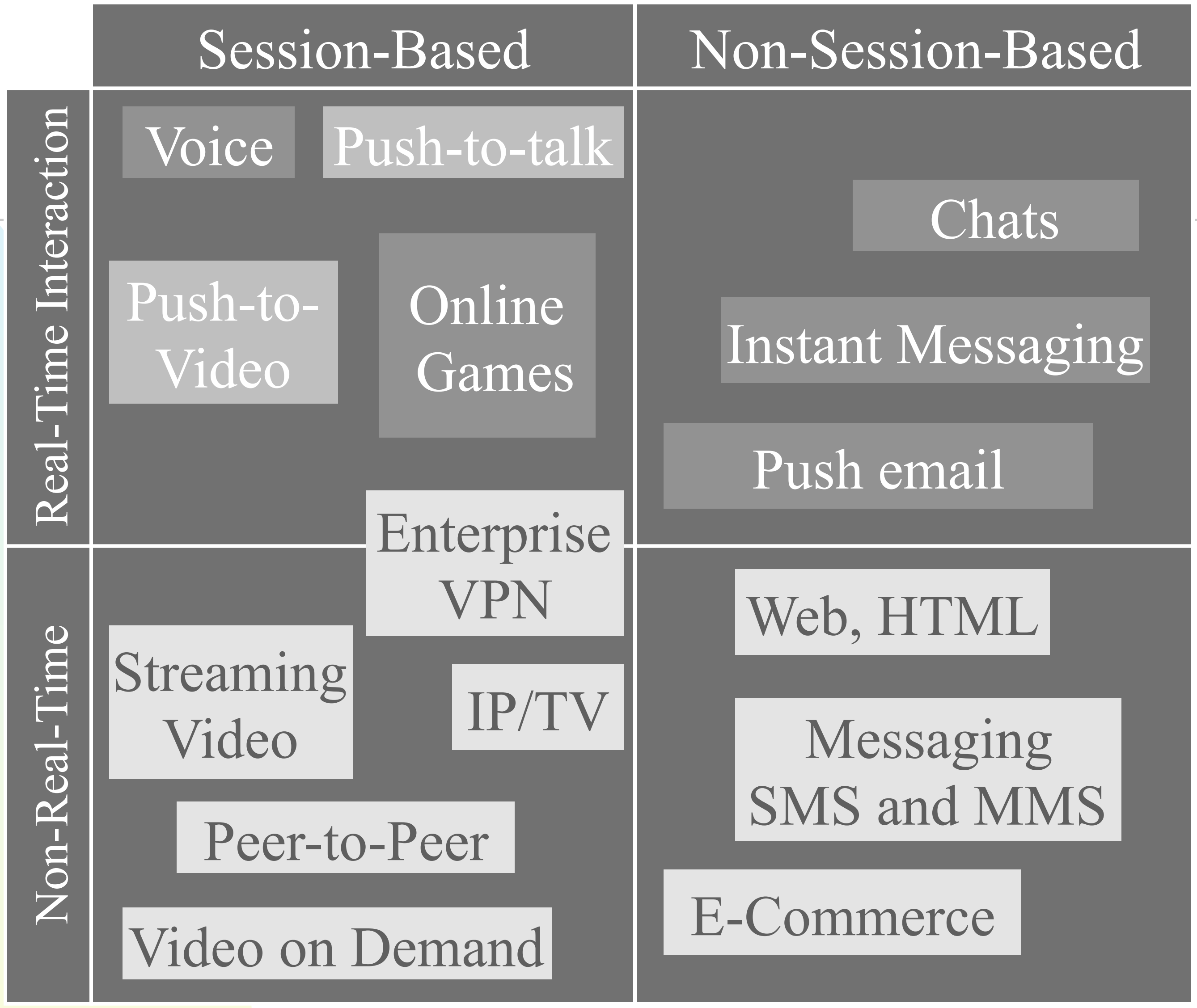
- Example of call routing

CSCF = Call State Control Function
HSS = Home Subscriber Service



LTE and IMS

- LTE is all-IP, but 3G and prior systems have used circuits for voice
- How to handle voice with LTE without radical modifications to the access network
- Solutions:
  - CSFB (Circuit Switched Fallback): In this approach, LTE just provides data services.
  - SVLTE (Simultaneous Voice and LTE): In this approach, the handset works simultaneously in the LTE and CS mode
  - **VoLTE (Voice Over LTE): IMS-based, all-IP, SIP**
- Operators have to choose a solution
- http://news.verizonwireless.com/OneVoiceProfile.pdf


Session-Based
Non-Session-Based
Real-Time Interaction
Non-Real-Time
Voice
Push-to-talk
Push-to-Video
Online Games
Chats
Instant Messaging
Push email
Enterprise VPN
Web, HTML
Streaming Video
IP/TV
Messaging SMS and MMS
Peer-to-Peer
E-Commerce
Video on Demand
SIP (IMS) only Applications
SIP or Non-SIP Applications
Non-SIP Only Applications

Web Service Architecture

- The three major roles in web services
  - Service provider
    - Provider of the WS
  - Service Requestor
    - Any consumer / client
  - Service Registry
    - logically centralized directory of services
- A protocol stack is needed to support these roles

Web Services Protocol Stack

- Message Transport
  - Responsible for transporting messages
  - HTTP, BEEP
- XML Messaging
  - Responsible for encoding messages in common XML format
  - XML-RPC, SOAP
- Service Description
  - Responsible for describing an interface to a specific web service
  - WSDL
- Service discovery
  - Responsible for service discovery and search
  - UDDI

What is SOAP?

- Fundamentally stateless one-way message exchange paradigm
  - More complex interactions may be implemented
- Exchange of structured and typed information
  - Between peers in decentralized fashion
  - Using different mediums: HTTP, Email, ..
- Request-reply and one-way communication are supported
- Note that XML infoset is an abstract specification
  - On-the-wire representation does not have to be XML 1.0!
- SOAP 1.2 ”HTTP Subset”. SOAP as HTTP extension
- Specifications
  - SOAP Version 1.2 Part 0: Primer
  - SOAP Version 1.2 Part 1: Messaging Framework
  - SOAP Version 1.2 Part 2: Adjuncts
  - SOAP Version 1.2 Specification Assertions and Test Collection

REST

- REST (Representational State Transfer) (Roy Fielding, PhD thesis)
  - Architectural style of networked systems
  - Applications transfer state with each resource representation
    - Representations of the data are transmitted
  - State is a property of a resource
- Resources
  - Any addressable entity
  - Web site, HTML page, XML document, ..
- URLs Identify Resources
  - Every resource uniquely identifiable by a URI

REST II

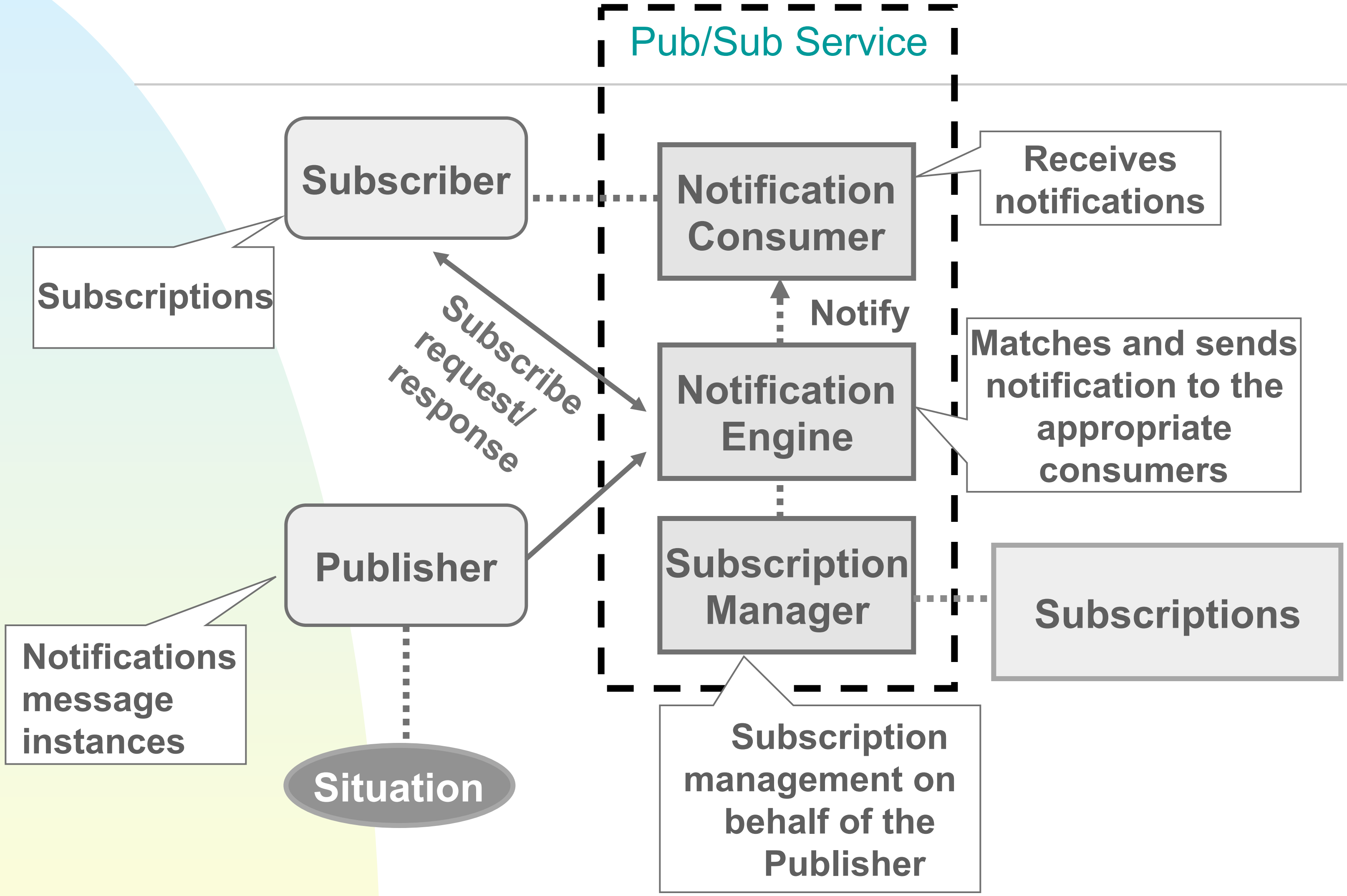
- Uses standards
  - Addressing and naming: URI
  - Generic resource interface: HTTP GET, POST, PUT, DELETE
  - Resource representations: HTML, XML, GIF,..
  - Media types: MIME
- Loose coupling
- Stateless transactions
- Self-descriptive messages
- **Hypermedia is the engine of application state**
  - **Just resources and URIs**

Event-based Systems and Publish/ subscribe

- Event delivery from publishers to subscribers
  - Event is a message with content
  - One-to-many, many-to-many
  - Builds on messaging systems and store-and-forward
- A frequently used communication paradigm
  - Decoupling in space and time
  - Solutions from local operation to wide-area networking
  - Proposed for mobile/pervasive computing
- The event service is a logically centralized service
  - Basic primitives: subscribe, unsubscribe, publish
- Various routing topologies and semantics

Event Systems

- Traditional MoM systems are message queue based (one-to-one)
- Event systems and publish/subscribe are one-to-many or many-to-many
  - One object monitors another object
  - Reacts to changes in the object
  - Multiple objects can be notified about changes
- Events address problems with synchronous operation and polling
- In distributed environments a logically centralized service mediates events
  - anonymous communication
  - expressive semantics using filtering


Pub/Sub Service
Subscriber
Notification Consumer
Receives notifications
Subscriptions
Subscribe request/ response
Notify
Notification Engine
Matches and sends notification to the appropriate consumers
Publisher
Notifications message instances
Subscription Manager
Subscriptions
Situation
Subscription management on behalf of the Publisher

Event Systems

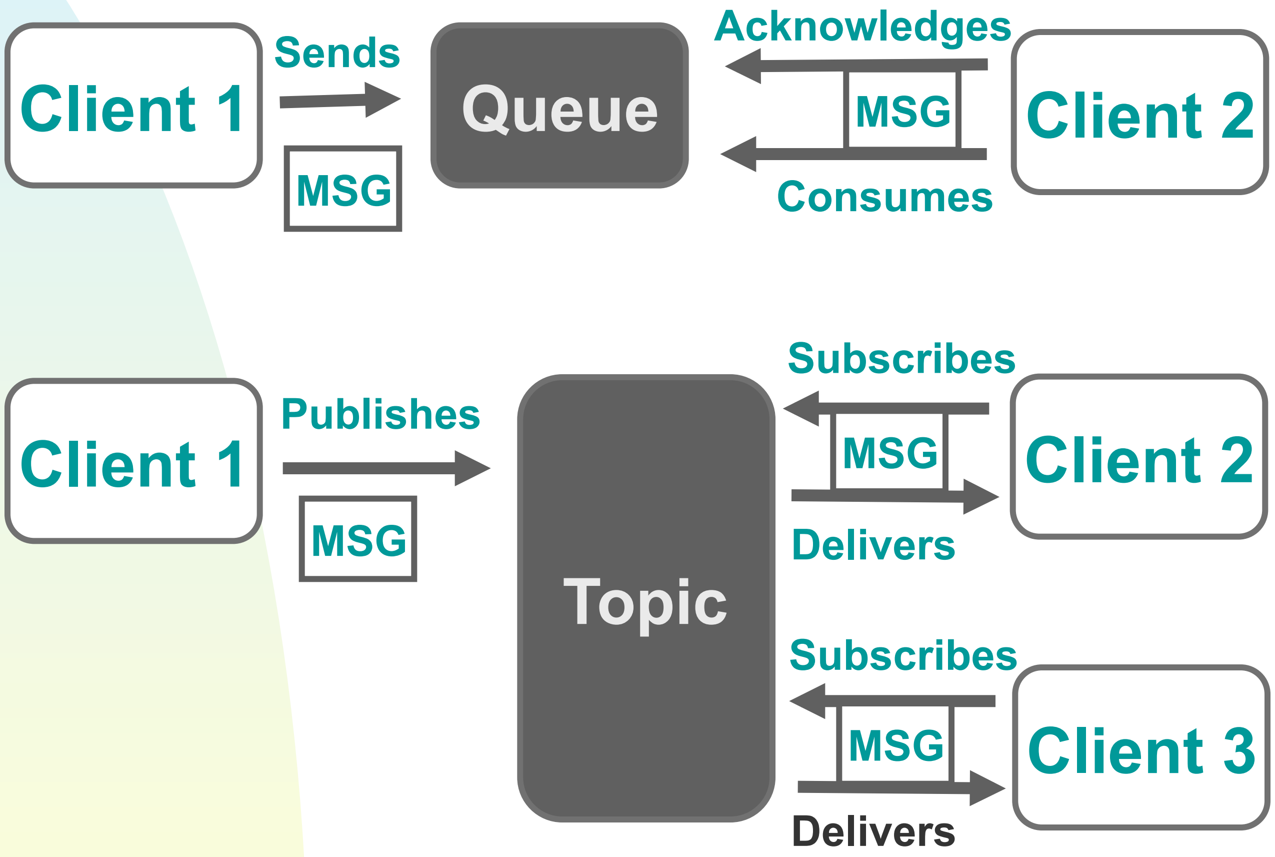
- Push versus Pull
- May be implemented using RPC, unicast, multicast, broadcast,..
- Three main patterns
  - Observer design pattern
    - Used in Java / Jini
  - Notifier architectural pattern
    - Used by many research systems
  - Event channel
    - Used in CORBA Event/Notification Service
- Filtering improves scalability / accuracy
  - Research topic: content-based routing

Tuple Spaces

- Tuple-based model of coordination
- The shared tuple space is global and persistent
- Communication is
  - decoupled in space and time
  - implicit and content-based
- Primitives
  - **In,** atomically read and removes a tuple
  - **Rd**, non-destructive read
  - **Out**, produce a tuple
  - **Eval**, creates a process to evaluate tuples
- Examples: Linda, Lime, JavaSpaces, TSpaces

Java Message Service (JMS)

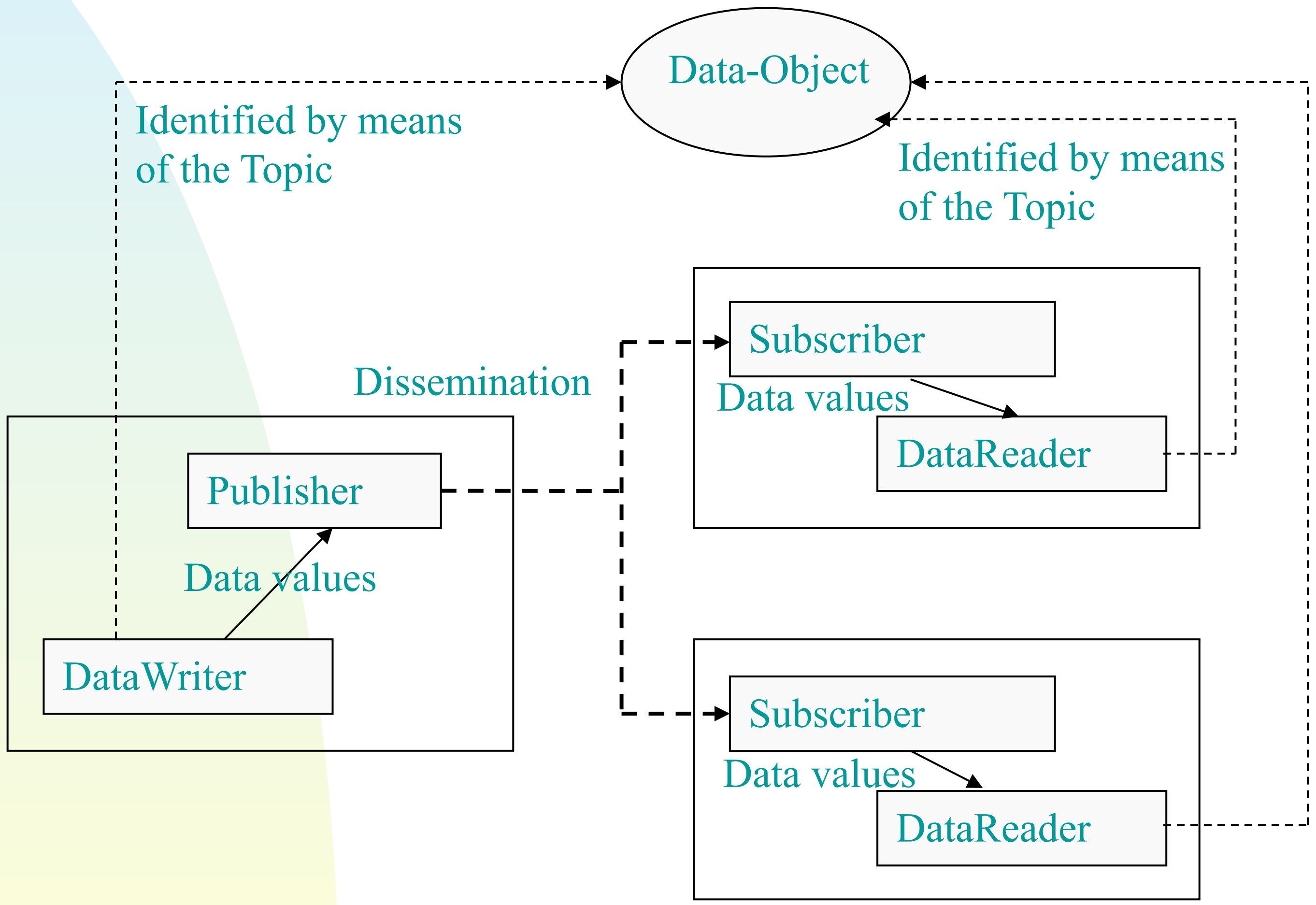
- Asynchronous messaging support for Java
- Point-to-point messaging
  - One-to-one
- Topic-based publish/subscribe
  - SQL for filtering messages at the topic event queue
  - One-to-many
- Message types:
  - Map, Object, Stream, Text, and Bytes
- Durable subscribers
  - Event stored at server if not deliverable
- Transactions with rollback


Client 1
Sends
MSG
Queue
Acknowledges
MSG
Consumes
Client 2
Client 1
Publishes
MSG
Topic
Subscribes
MSG
Delivers
Client 2
Subscribes
MSG
Delivers
Client 3

OMG Distributed Data Service

- The Data Distribution Service for Real-Time Systems (DDS)
- The specification defines an API for data-centric publish/subscribe communication for distributed real-time systems.
- DDS is a middleware service that provides a global data space that is accessible to all interested applications.
- DDS uses the combination of a Topic object and a key to uniquely identify instances of data-objects.
- Content filtering and QoS negotiation are supported
- DDS is suitable for signal, data, and event propagation.

DDS


Data-Object
Identified by means of the Topic
Identified by means of the Topic
Dissemination
Subscriber
Data values
DataReader
Publisher
Data values
DataWriter
Subscriber
Data values
DataReader

Data synchronization

Synchronization models

- WHEN to sync?
  - manually
  - Automatically
- Synchronization and replication
  - store data in a number of locations
  - fault tolerance, but...
  - most copies unavailable most of the time

Synchronization models

- HOW to sync?
  - permit only one modifiable copy of data
    - lock on data
    - hub-and-spoke model
  - multiple copies can be independently modified
    - more flexible
    - more complex to implement the sync process

Synchronization models
Sync process

- Update detection
  - recognition that a data has been modified
- Update propagation
  - transmission of changes among all the data replica
- Reconciliation
  - combination of all the updated data to build a synchronized version

Synchronization models
Sync process

- Update detection
  - recognition that a data has been modified
  - triggers the start of sync process
  - clean / dirty status (modification flag)
  - modification timestamp
  - hash of the content

Synchronization models
Sync process

- Update detection
  - modification timestamp
    - store only the timestamp of last modification?

      (e.g. in file system)
      - comparison of all the timestamps
        - time / resource consuming!
      - monitoring file system for changes
      - use dir timestamp if equal to last modified contained file
  - semantics of modification time
    - when a file timestamp is actually modified?
      - content modifying, file renaming, file relocation?

Synchronization models
Sync process

- Reconciliation
  - combination of all the updated data to build a synchronized version
    - opaque data (e.g. binary, pictures)
      - ask the user which version to use
    - structured data (e.g. XML)
      - edit logs
      - state comparison
      - both (can) use the latest common ancestor as comparison aid

Synchronization models
Sync process

- Reconciliation
  - type of modifications (inside a single file):
    - insertion
    - deletion
    - moving
    - changing

    - moving often as deletion + insertion
    - use of unique ID per piece of structured content

Synchronization models
data sync algorithms & tools

- rsync
- Remote Differential Compression (RDC)
  - Microsoft Windows Server 2003 R2 / 2008
  - allows data to be synchronized between two or more computers, using compression techniques to minimize the amount of data sent across the network.

Synchronization models
data sync algorithms & tools

- Remote Differential Compression (RDC)
  - files (to be synced) divided into chunks of data
    - chunk bounded using an incremental fingerprint function
  - MD4 hash calculated for each chunk
  - comparison of MD4 lists (signature, one per file)
  - transfer only of missing / different chunks
  - can be applied recursively!
    - original file size 9GB
    - signature 81MB
    - signature of signature 6MB

Sync in middleware

- Publish / subscribe
  - for update propagation - data channel
    - each edit gets immediately published as a event
    - continuous reconciliation
    - multiple users
  - for update detection
    - an update is advertised to all subscribers

Case studies: SyncML

- Synchronization Markup Language
- Open Mobile Alliance (OMA) Data Synchronization and Device Management
- Interoperable protocol to sync data

Case studies: SyncML

- Interoperable protocol to sync data
  - update propagation
    - transfer of updates among devices
  - client-server architecture
  - based on “edit log” model
    - addition, delete, replace of objects
    - unique ID
  - data type & data store independent
  - Notification of conflicts

Pervasive computing middleware

Projects	***Key Issues***
UIC	**Heterogeneity of devices and networks:** It helps users to specialize to the particular properties of different devices and network environments
X-Middle	**Disconnected operations in mobile applications:** It allows mobile users to share data when they are connected, or replicate the data and perform operations on them off-line when they are disconnected; data reconciliation takes place when user gets reconnected
Gaia	**Dynamic adaptation to the context of mobile applications:** It supports the development and execution of portable applications in active spaces
Lime	**Programming constructs which are sensitive to the mobility constraints:** It explores the idea by providing programmers with a global virtual data structure and a tuple space (Tspace), whose content is determined by the connectivity among mobile hosts
Tspaces	**Asynchronous messaging-based communication facilities without any explicit support for context-awareness:** It explores the idea of combination of tuple space (Tspace) and a database that is implemented in Java. Tspace targets nomadic environment where server contains tuple databases, reachable by mobile devices roaming around
L2imbo	**QoS monitoring and control by adapting applications in mobile computing environment**: It provides the facilities of multiple spaces, tuple hierarchy, and QoS attributes
Aura	**Distraction-free pervasive computing:** It develops the system architecture, algorithms, interfaces and evaluation techniques to meet the goal of pervasive computing