Chapter IX – Presence Applications and Services

http://users.enucs.concordia.ca/~glitho/
Outline

1. Basics
2. Interoperability
3. Presence service in clouds
Basics

- 1 - IETF abstract model
- 2 - An example of implementation
Telecommunication Services Engineering (TSE) Lab

IETF Abstract model

Presence service

- Enable the publication and the retrieval of context information (e.g. online/offline, willingness to communicate) or more generally
  - Space
  - Environment
  - Physiology

- Note: The information is collected by sensors that are not part of the presence framework
IETF Abstract model

- Clients
  - Presentities: publish information
  - Watchers: Retrieve the information (pull, push)
- Presence service: accept the information (maybe centralized or distributed)
IETF Abstract model

- Clients
  - Presentities: publish information
  - Watchers: Retrieve the information (pull, push)

- Presence service: accept the information (maybe centralized or distributed)
IETF Abstract model

Presence service

Presentitites

Watchers
IETF Abstract model

Two types of watchers

- Fetcher
  - Fetch the current value of some presentity presence information
- Subscriber
  - Request the notification of changes in some presence entity presence information
An example of implementation

Two standard IETF implementations

- XMPP
- SIP SIMPLE

Focus in this course: SIP SIMPLE
An example of implementation

SIMPLE (SIP for Instant Messaging and Presence Leveraging Extensions)

- Set of specifications for presence and instant messaging with SIP
- Cover among other things:
  - Core protocol machinery
  - Presence documents (XML based)
  - Privacy and policy
  - Provisioning
An example of implementation

SIP SIMPLE

Protocol machinery

- Extended SIP event framework
  - Publish (New message)
  - Subscribe
  - Notify

SIMPLE presence documents

XML based
An example of implementation

SIP SIMPLE

- Entities
  - Presence User Agent (PUA):
    - client (presentity or watcher)
  - Presence Agent (PA)
    - : stores presence information
- Presence Server (PS):
  - Acts as a PA or a proxy to forward requests to the appropriate destination
An example of implementation

SIP SIMPLE

- Related messages:
  - SIP Publish, Subscribe and Notify
  - Information model: XML based known as PIDF

Note: A plethora of other implementations exist (standard and proprietary)
Examples of applications

Examples of applications areas

- Instant Messaging (IM)
  - Rely on a SIP extension (i.e. MESSAGE)
    - Do not require SIP sessions establishment although they may be sent within on-going SIP sessions.
Examples of applications

Beyond instant messages

- Presence enabled conferencing (i.e., Start a conference when a quorum is online)
- Unified Communications (e.g., Email, SMS, FM radio)
  - Use most appropriate communication mean
- Social networks
Examples of applications

Illustration

- Draw a sequence diagram with participants, P1, P2, P3, P4, P5, a conference server, and a presence server where the conference starts automatically when at least 3 participants are on-line and let us assume, P2, P4, and P5 get successively on-line
Interoperability

- 1 - Problem statement
- 2 - The state of the art
- 3. Our recently proposed solution
Interoperability
Telecommunication Services Engineering (TSE) Lab

Interoperability
Problem statement

The problem

- Several non interoperable standards (e.g. SIP SIMPLE, XMPP, Wireless Village)
- Several non interoperable proprietary implementations (e.g. Facebook, Skype)
- How to bridge the different worlds?
Problem statement

The problem

- More concretely
  - Users may have several accounts with several providers with non interoperable implementations
  - How can a presentity with multiple accounts publish context information via the account she/he is using at a given time, with the possibility for watchers to retrieve this information via any of the other accounts?
The state of the art

- Bidirectional mapping (e.g. IETF spec for XMPP / SIP SIMPLE mapping)
  - Exponential growth with number of standard and non standard presence implementations
  - Few specifications exist (XMPP / SIP SIMPLE only known standard)
The State of the Art

- Open source SIP communicator
  - Support all protocols
  - Not realistic
The State of the Art

- SOAP based Web services used as common denominator / glue
  - Example: YooHoo
    - Enable end-user to access different presence services through a Big Web services
  - Problem: small footprint devices
Our proposed solution

Our proposed centralized solution

Centralized solution

Our design goals:

- No restriction on the business model
- Avoid one to one mapping
- Presence service (e.g. XMPP, SIP) interface neutral
- Unique and lightweight interface to clients
- Accommodate clients behind firewalls
- Deployment on Web servers with no additional server
- No modification to existing presence servers

Selected “bridging” technology: RESTful Web services
Centralized solution

Our centralized approach: The business model

Figure 1: Business Model
Centralized solution

Our centralized approach: The architecture

Figure 2: Overall Architecture
Centralized solution

Our centralized approach: Universal presence server

Figure 4: Software architecture of the UnviPS
Telecommunication Services Engineering (TSE) Lab

Centralized solution

Our centralized approach: Scenario (Alice off-line in IMS but online in Google)

1: POST(alice@ericsson.com)
2: SUBSCRIBE
3: SUBSCRIBE
4: 200 OK -chunk1-(http://www.univps.com/users/1)
5: SUBSCRIBE
6: 200 OK
7: 200 OK
8: 200 OK
9: NOTIFY(offline)
10: NOTIFY(offline)
11: NOTIFY(offline)
12: XMPP subscribe alice@gmail.com
13: XMPP Notify(alice online)
14: 200 OK -Chunk2-(alice's presence info @ gmail.com)

Alice is a UnivPS user, his UnivPS profile shows that he has also a Google XMPP account.
Interoperability
Our de-centralized approach: A P2P Approach

Our decentralized solution

Differences with previous approach:

- Service offered by a federation of providers instead of a single universal presence service provider

Selected “bridging” technology: P2P overlays

Overlay built on top of real presence servers and real services nodes (e.g. protocol translators)
Our decentralized solution

On P2P overlay

- Current way of implementing P2P computing
  - Application layer virtual networks that provide storage, processing, connectivity and routing
    - Network built by peers that federate to offer storage and processing capabilities to applications
      - Built on top of existing networks, thus the name of overlay
        » Applications running on top of transport protocols of real network
        » Real network nodes become virtual nodes in the overlay
Structured P2P overlays vs. unstructured P2P overlays

P2P overlay

Overlay (Above Transport)

Real network (PHY, link, IP, Transport)
Our decentralized solution

Our de-centralized approach: A P2P Approach
Our decentralized solution

Our de-centralized approach: A P2P Approach

Alice

Provider-A E-PS
(Protocol P1)

Provider-B E-PS
(Protocol SIP)

Provider-C E-PS
(Protocol P3)

Provider-D E-PS
(Protocol SIP)

Overlay Network

SN 1 @ a.com

Provider-A E-PS does not have
the accounts list of Emý

Subscribe(emy @ b.com, use = convergedps)

FindPSA by URI(emy @ b.com)

FindSN(from = P1, to = SIP)

ForwardMessage(sub1, b.com, B-PSA_address)

FindPSA by Domain(b.com)

FindSN(from = SIP, to = P3)

ForwardMessage(subscription, b.com)

Subscribe(emy @ c.com)

Notify(emy @ c.com - is online)

FindPSA by Domain(c.com)

FindSN(from = P3, to = c.com)

FindSN(from = SIP, to = P1)

FindSN(from = SIP, to = P3)

ForwardMessage(subscription, c.com)

Subscribe(emy @ c.com)

Notify(emy @ c.com - is online)

FindPSA by Domain(c.com)

FindSN(from = P3, to = c.com)

FindSN(from = SIP, to = P1)

FindSN(from = SIP, to = P3)

ForwardMessage(subscription, c.com)

Subscribe(emy @ c.com)

Notify(emy @ c.com - is online)

FindPSA by Domain(c.com)

FindSN(from = P3, to = c.com)

FindSN(from = SIP, to = P1)

FindSN(from = SIP, to = P3)

ForwardMessage(subscription, c.com)

Subscribe(emy @ c.com)

Notify(emy @ c.com - is online)

FindPSA by Domain(c.com)

FindSN(from = P3, to = c.com)

FindSN(from = SIP, to = P1)

FindSN(from = SIP, to = P3)

ForwardMessage(subscription, c.com)

Subscribe(emy @ c.com)

Notify(emy @ d.com - is online)

FindPSA by Domain(d.com)

FindSN(from = P3, to = d.com)

FindSN(from = SIP, to = P1)

FindSN(from = SIP, to = P3)

ForwardMessage(subscription, d.com)

Subscribe(emy @ d.com)

Notify(emy @ d.com - is online)

FindPSA by Domain(d.com)

FindSN(from = P3, to = d.com)

FindSN(from = SIP, to = P1)

FindSN(from = SIP, to = P3)

ForwardMessage(subscription, d.com)

Subscribe(emy @ d.com)

Notify(emy @ d.com - is online)

FindPSA by Domain(d.com)

FindSN(from = P3, to = d.com)

FindSN(from = SIP, to = P1)

FindSN(from = SIP, to = P3)

ForwardMessage(subscription, d.com)

Subscribe(emy @ d.com)

Notify(emy @ d.com - is online)

FindPSA by Domain(d.com)

FindSN(from = P3, to = d.com)

FindSN(from = SIP, to = P1)

FindSN(from = SIP, to = P3)

ForwardMessage(subscription, d.com)

Subscribe(emy @ d.com)

Notify(emy @ d.com - is online)
Presence services in clouds

- 1 - Cloud basics
- 2 - Virtualization
- 3 - Problem statement
- 4 - Virtualized presence service
Cloud basics

Motivations / expected benefits

- Cost reduction (hardware/software resources and their management)
- Consumption/usage-based pricing
- Elasticity
- Flexibility in service deployment
Cloud basics

Figure 1: Cloud Computing Architecture

- **End Users**
- **Software as a Service (SaaS)**
- **Platform as a Service (PaaS)**
- **Infrastructure as a Service (IaaS)**

**Resources Managed at Each Layer**

**Application**
- Business Applications, Web Services, Multimedia

**Platforms**
- Software Framework (Java/Python/.Net) Storage (DB/File)

**Infrastructure**
- Computation (VM) Storage (block)
- CPU, Memory, Disk, Bandwidth

**Hardware**
- Data Centers

**Examples:**
- Google Apps, Facebook, YouTube, Salesforce.com
- Microsoft Azure, Google AppEngine, Amazon SimpleDB/S3
- Amazon EC2, GoGrid, Flexiscale
Cloud basics

Software as a Service (SaaS)
- Accessible via GUI but also via programmatic interfaces (i.e. APIs)
- Alternative to running services locally

Platform as a Service (PaaS)
- Development and management of services hosted in the cloud (e.g. Google Apps Engine)

Infrastructure as a Service (IaaS)
- Computing resources (e.g. storage and processing) that are dynamically split, assigned and re-sized.
  Key technology: Virtualization
Virtualization
Virtualization

Basics of virtualization

- Enable the co-existence of entities on same substrates/building blocks - e.g.
- Several operating systems on a same machine
- Several networks on a same set of routers / links

Advantages

- Efficient usage of resources
- Easy introduction of new functionalities
Virtualization

**Figure 2:** Non virtualized environment vs. non virtualized environment: An example from the computer world
Virtualization

Figure 2: 2 virtual networks using a same substrate network
Virtualization

Basics of virtualization

- Differences between virtualization and VPN
  - VPN does not allow heterogeneous networks on a same infrastructure because based on a same technology
- Differences between virtualization and overlay networks
  - Overlays are designed independently of each other and may interact in an harmful way
  - Clear separation between substrate and virtual networks in case of virtualization
Problem statement

The problem

- Presence service is expected to grown significantly in the future (e.g. social networks), thus the need to:
  - Re-use presence substrates / building blocks across standard and proprietary implementations for cost sharing
  - Easily introduce new presence functionality

Potential technological basis for a solution: Virtualization
Problem statement

Our ultimate goal: Presence service in the cloud

- On-going step: Infrastructure as a Service (IaaS)
  - Virtualized presence service

- Next steps
  - Platforms as a Service (Paas)
    - Multi-level abstraction APIs
      - Experienced developers
      - Novice developers
  - Software as a Service (SaaS)
    - Access by other applications and access by end-users
Virtualized presence service

Our proposed solution for presence service virtualization

F. Belqasmi, N. Kara, R. Glitho, A Novel Virtualized Presence Service Architecture for Future Internet, accepted, IEEE ICC 2011 Workshop on Future Internet
Virtualized presence service

Our terminology

- Virtual Presence Service (VPS)
  - Presence service as deployed by service providers (e.g. SIP SIMPLE, XMPP, Facebook)

- PSS (Presence Service Substrate)
  - Sharable building blocks that could be re-used across standard and non-standard VPSs
Virtualized presence service

Our proposed solution: Scenarios

- **c. A SIP SIMPLE VPS that relies on several PSSs**

- **a. Several SIP SIMPLE VPSs share a same PSS**

- **b. A SIP SIMPLE VPS and an XMPP VPS that share the same PSS**
Virtualized presence service

Our design goals

- Use of a same PSS by several VPS that may be in separate domains
- Use of several PSS by a same VPS for scalability purpose
- VPS and PSS should interact dynamically
- PSS should enable the rapid deployment of a very wide range of VPS.
Virtualized presence service

Our proposed solution: architecture
Telecommunication Services Engineering (TSE) Lab

Virtualized presence service
Our proposed solution: architecture (SIP VPS instantiation and use)
References
