Towards wide-area Ubiquitous Computing

Markus Endler
Laboratory for Advanced Collaboration (LAC)
PUC-Rio, Brazil

February 2008

LAC
Laboratory for Advanced Collaboration

Pontifícia Universidade Católica do Rio de Janeiro

Outline

- Some Facts about PUC-Rio
- Introduction
- Some research challenges
- Overview of MoCA
- Current research work
- Conclusion

Ubiquitous Computing

- Ubiquitous computing = integration of microprocessors into everyday objects + wireless communication among users and interaction with their environments
- Goal (according to *M.Weiser)*: provide users with *customized* and *invisible* communication and information services *at any place, at any time*
- So far, most of the ubiquitous environments have focused at providing localized (intra-domain), “closed-world” and application- or task-specific ubiquity

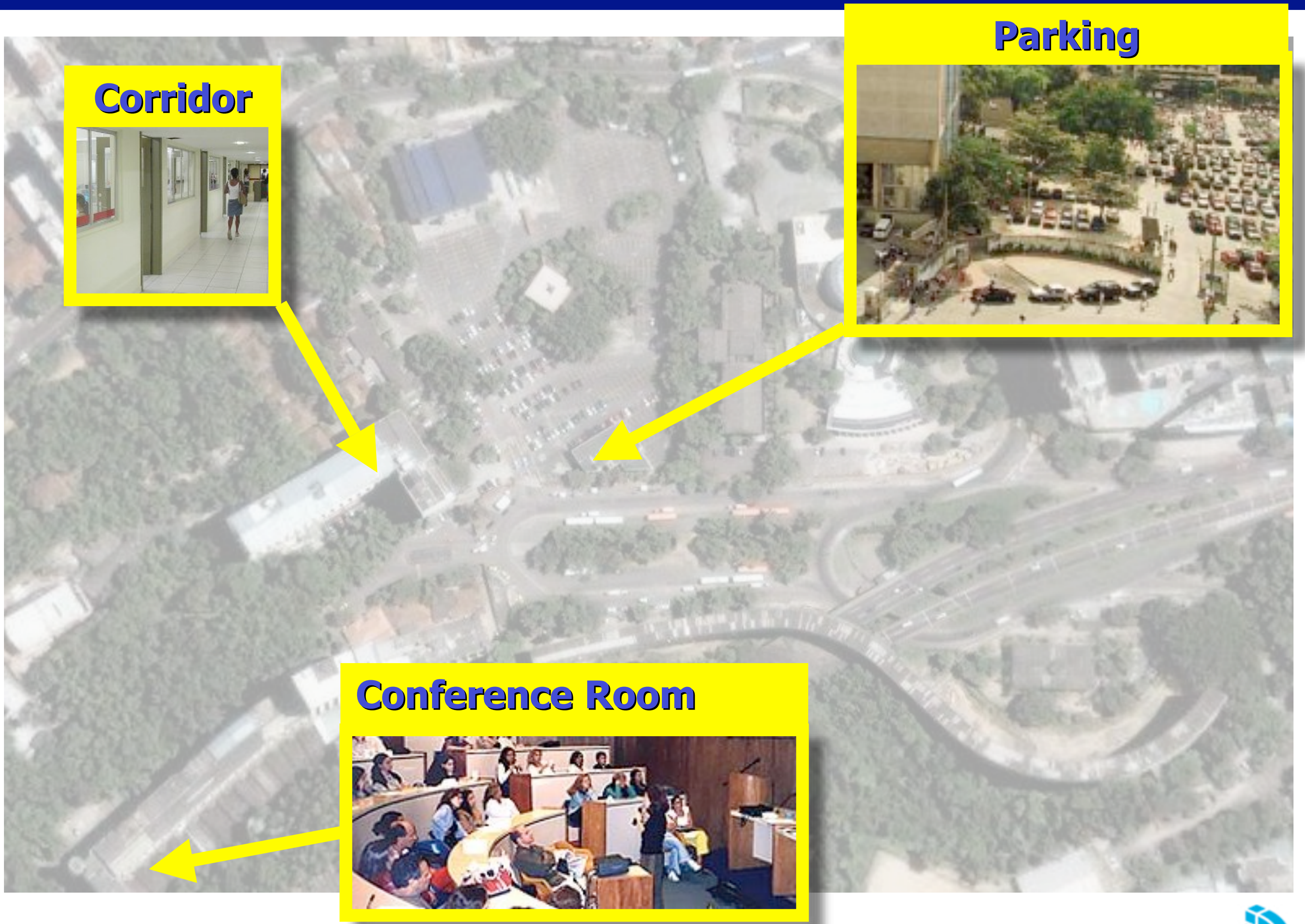
Ubiquitous Computing

But users typically roam between different places.

And at each place they…

- perform specific tasks/activities
- encounter different groups of people
- have connectivity through a specific wireless network
- are surrounded by different sorts of ambient devices (web-cams, projectors, smart boards,etc.)
- are constrained by local rules/ access rights
- have access to place-specific set of services

Scenario



Heterogeneity
Applications and Context

Mobile Matchmaking

- Symbolic Location
- Preferences

Find Directions

- Geogr. Location

Requirements

- **Modeling and use of new contexts (support for evolution)**
- **Application-specific context models**
- **Software interoperability (different MWs for context provisioning,...)**

Conference Assistant

- Preferences
- Activity

The next generation of Ubiquity

NG Ubiquity will be:

- Distributed
- Heterogeneous in several aspects
  - Devices, sensors, platforms, wireless networks
  - Available middleware and application-specific services
  - Context models, semantic descriptions and knowledge bases
  - Concurrent applications with different requirements
  - Users with different demands, backgrounds, roles, and tasks
- Spread over several administrative domains and organizations
- Deployed at work, at home and in public spaces
- Must be prepared for dynamic and unknown user base

- These characteristics translate into a long list of new research challenges…

Reserach Challenges
(a small list)

- How to ensure interoperability in spite of system's and platform's heterogeneity?
- How do context-sensitive applications (of roaming users) interact with local context providers and handle the uncertainty and ambiguity of context information?
- How is context represented/modeled and its semantics shared universally?
- How to give users effective control over disclosure of their context data (e.g. preferences)?
- How to handle service access control and user authentication in an open environment?
- How to design software and protocols for such dynamic and heterogeneous environments?
- How to accurately infer the user's intent and its relation to the visited place, or group of co-located uesrs?
- Etc.

Our *bottom-up* Approach

- First, learn how to gather & distribute concrete context information*, and develop a robust & useful middleware
- Then, get experience with the design of some context- & location-aware applications
- After this, tackle the more complex problems of handling wide-area and open ubiquitous systems.

(*) We focused on system context data that can be obtained automatically from the device and network, and positioning information that can be automatically inferred.

MoCA’s Research Goals

Main Goals:

- Design and implement a middleware to support the development and deployment of such collaborative applications;
- Experiment with new forms of context/location-aware collaboration, develop applications using the middleware, and evaluate their usability and usefulness.

Target setting:

- Structured wireless network (802.11)
- Users with laptops, palmtops or smart phones
- Intra-domain applications (e.g. for University campus-community, corporation)

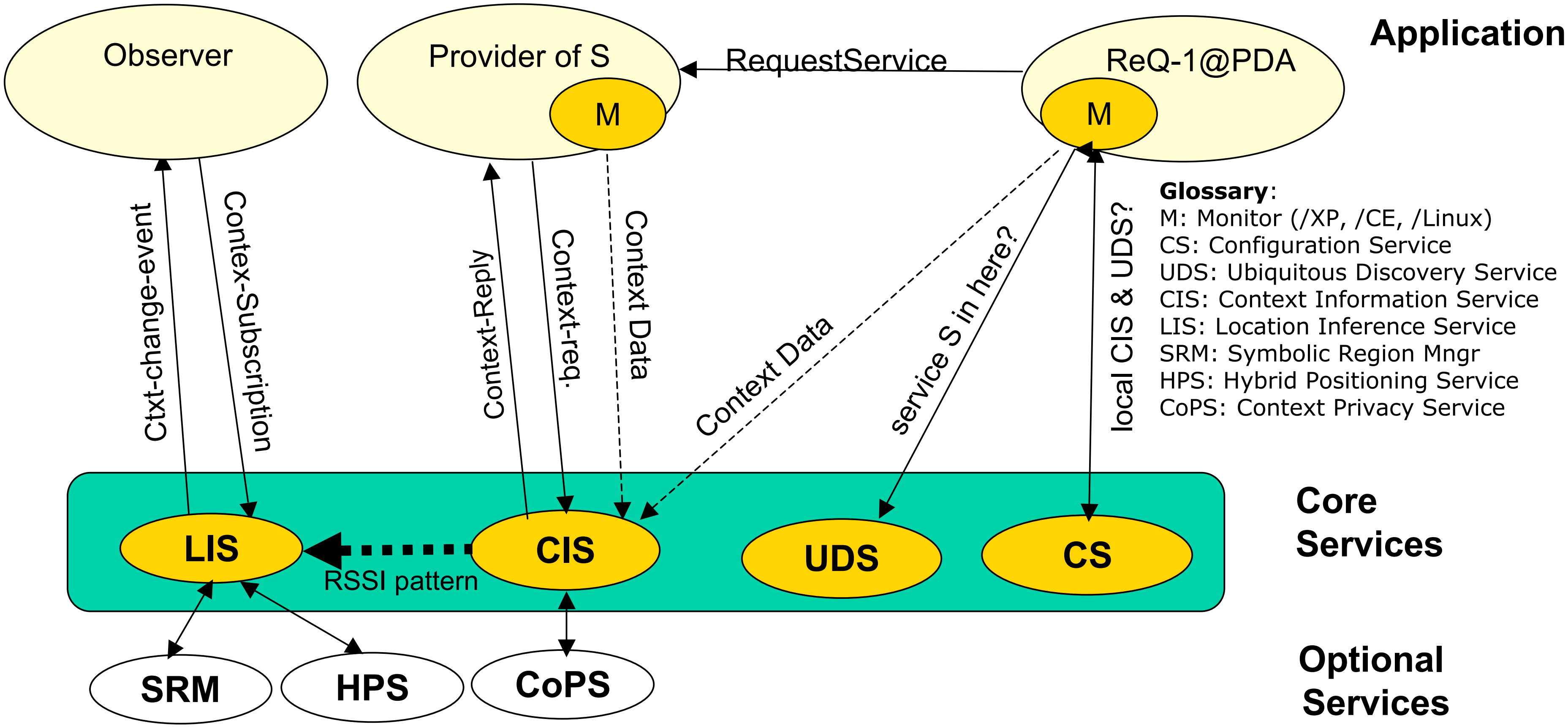
MoCA

MoCA consists of basic services for collecting and processing context information, synchronous & asyncronous communication and client/serverAPIs and a proxy Framework that facilitate the use of these services by developers of applications for mobile networks.

Essentially, the basic services provide

- Distributed monitoring, storage and complex queries about the dynamic execution context (state of mobile device & network resources).
- Symbolic location inference based on IEEE 802.11 signall strength
- Advertisement and location-aware discovery of applications and middleware services

Overview of MoCA



Glossary:
M: Monitor (/XP, /CE, /Linux)
CS: Configuration Service
UDS: Ubiquitous Discovery Service
CIS: Context Information Service
LIS: Location Inference Service
SRM: Symbolic Region Mngr
HPS: Hybrid Positioning Service
CoPS: Context Privacy Service

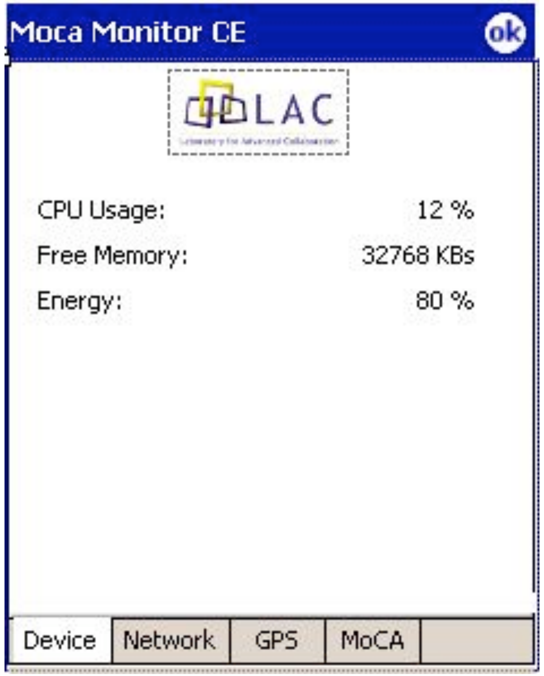
Other tools:
• ContextView - Tool to debugg context-aware systems
• ECI- Event-based Communication

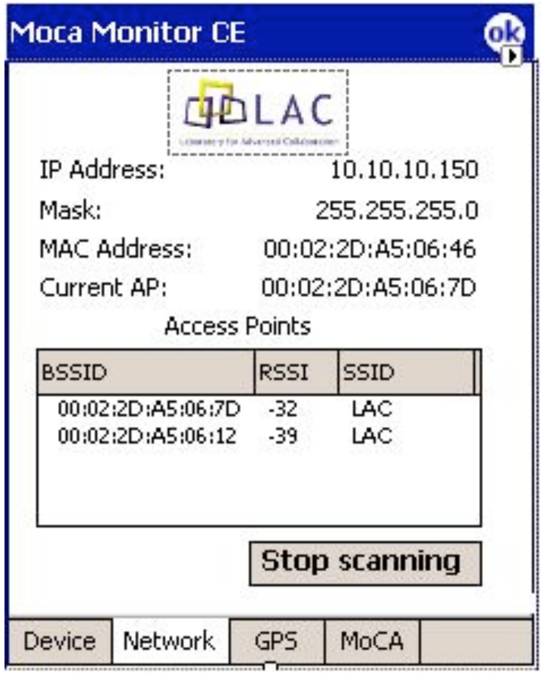
Monitor

- Monitor/ (XP/CE/Sim/Linux/Symbian)
  - is a *daemon* executing on the mobile device;
  - Periodically probes state information about the mobile device's resources and wireless connectivity, sends it to CIS and makes it available to local application clients e.g.:
    - Strength of RF signal received from **all** visible Access Points;
    - CPU utilization, available memory and energy;
    - MAC Address, IP address, IP mask and currently used Access Point;
    - *other data....*
  - RF signals are obtained through WiFi scan operations, in a uniform way, independently of the 802.11 network interface;
  - If the device has also a GPS receiver, GPS coordinates are sent as well
  - The Monitor also reports to the CIS any change of the current **IP address** or **Access Point** of the device (i.e. a handoff)

Monitor for Windows Mobile

- Monitor/CE: Adapted GUI for usage at Palmtops





Context Information Service (CIS)

Delivers context information to any application entity (e.g. server/client/proxies), through direct queries or subscriptions with context expressions.

- Context variables used in expressions
  - **CPU** **(Int – 0 to 100) %**
  - **EnergyLevel** **(Int – 0 to 100) %**
  - **AdvertisementPeriodicity** **(Int – 0 to 100000)**
  - **APMacAddress** **(String)**
  - **FreeMemory** **(Long) in Kbytes**
  - **DeltaT** **(Long)in ms**
  - **OnLine** **(Boolean)**
  - **IPChange** **(Boolean)**
  - **Roaming** **(Boolean)**
  - **RF Signal Strength from each AP** **(dB)**
  - **LinkQuality** **(dB)**
  - **GPS lat, GPS long**
- Usually, applications are interested in:
  - State of the device (e.g. available memory, enery level)
  - Wireless connectivity status
  - Aproximate position of the device

CIS (cont.)

Context Information Service (CIS) can be deployed as a pool of servers (each one collecting the context information from some Monitors);

- Each device has associated context information (index: MAC-Address);
- CIS Clients subscribe to the service informing the devices's MAC-Address (as the Subject) and an SQL92-like expression defining the state of context they are interested in.
- Example:

  Subject="02:DA:20:3D:A1:2B",

  Properties = "roaming = True" OR "FreeMem < 15%" OR "CPU > 90%";

Location Inference Service (LIS)

LIS is a GPS-less positioning service for indoor applications, based on IEEE 802.11 signal strength *finger-printing* that uses a stochastic algorithm to correlate sinal patterns.

Usage is in two phases:

Calibration:

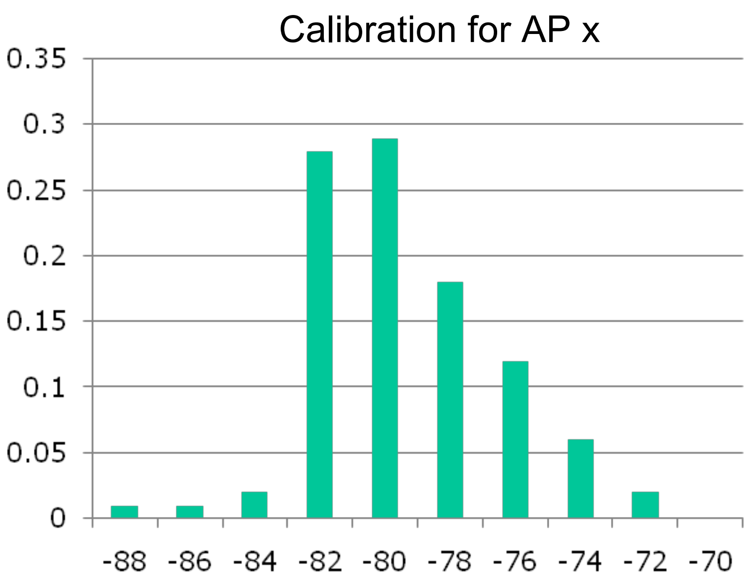
- Within the area of interest (e.g. a building, part of a campus) several **reference points (RP)** are defined (each of which receives an identifier = symbolic region name).
- At each RP, the RSSI from all visible 802.11 Access Points is sampled (e.g. N samples), and stored in LIS's database,

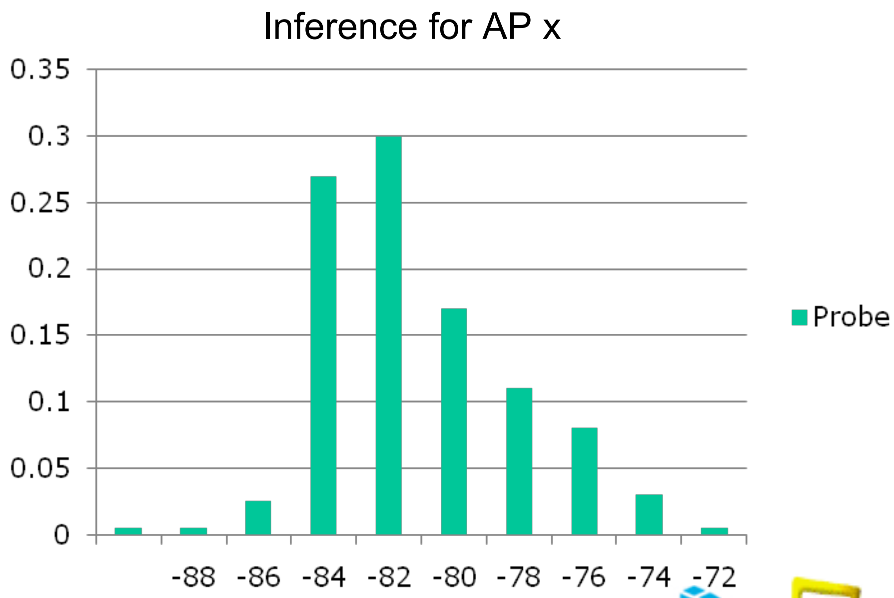
Inference:

- At any time, LIS gets N samples of the signal strength at the current position and identifies the set of *k* RPs in the database that have the most similar signal distributions compared to the current distribution.
- It then infers that the device is located in the symbolic region determined by the majority of RPs.

LIS' Histogram Algorithm

- For each of the visible APs, compares the distribution of the last *N* RSSI probes of calibration *versus* inference
- Let d_i be the distribution difference for AP*i, and* $v=(d_1, d_2, d_3)$ the difference vector (missing APs are not considered)
- If $|v| \leq L$, then the probes refer to same symbolic location





LIS – Sorts of Location Access

LIS supports:

Syncronous access

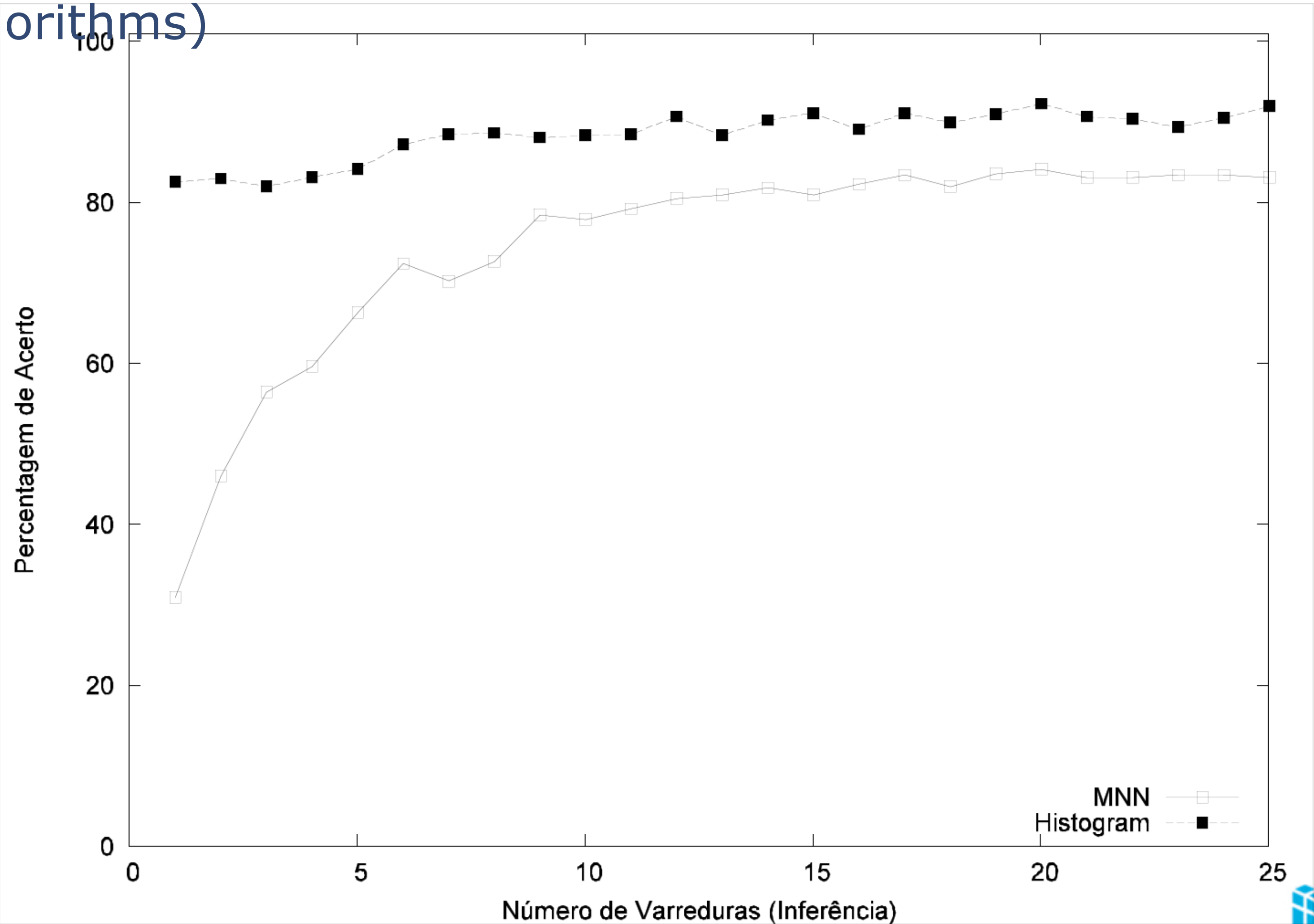
- Symbolic region of a device *D*
- All Devices within a region *R*

- Asynchronous notifications
  - Application instantiates either a ***RegionListener*** or a ***DeviceListener***
  - Whenever a device enters or leaves a region, LIS notifies all subscribed applications about the corresponding ***regionID*** or ***deviceID***

- Definition and Management of hierarchies of symbolic regions

LIS Accuracy

- Percentage of successfull inferences depending on the number of probes (deterministic and Histrogram algorithms)

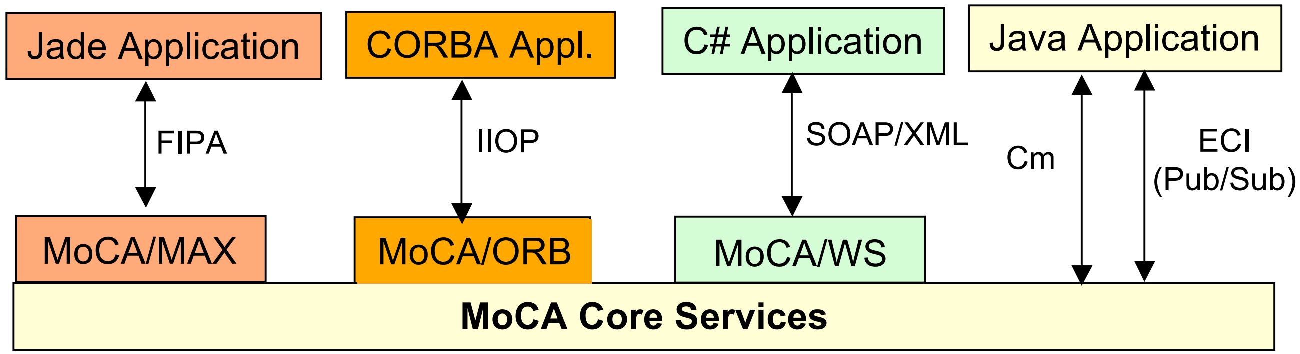


Some Prototype Applications

Some location-aware prototypes developed using MoCA:

- W-Chat *chat-tool with connectivity awareness*
- Notes in the Air (NITA) *notes to locations (virtual whiteboards) and location-based chat*
- Mobile Matchmaking Service *location-based matching of user interests*
- BuddySpaceLive *on-line tracking of friend's locations*
- Wireless Marketing Service *location-based discount coupons*
- Virtual Lines *location-based reservation of a position in a line*
- Who Are You? (WAY) *proximity-based exchange of business cards*
- uGuide *allows to open region-specific URLs in any browser*
- iPH *Interactive Presenter for Handhelds, co-edition options are enabled depending on location*

MoCA´s *Personalities*



In order to widen the usability of MoCA’s core services, we have developed three additional programming interfaces ➔ personalities

- MoCA/MAX – for use with Agent Framework Jade
- MoCA/WS – a proxy that works as a Web Service
- MoCA/ORB – a proxy talks IIOP

Some uses of MoCA

MoCA has been used as the fundamental building block for other specialized middleware, developed by other groups and universities in Brazil. Some examples:

- MoGrid: a middleware for mobile Grids (LNCC/RJ) – *PhD thesis*
- Context-aware Exeption Handling for Ubiquitous environments (LES/PUC-Rio) - *M.Sc. thesis*
- ContextTV: a context-aware middleware for interactive digital TV on mobile devices (Fed. U. of Pernambuco) – *M.Sc. thesis*
- MAG/MoCA: Interface for Grid access through handheld devices (Fed. U. of Maranhão) – *Integrade Research project*

And MoCA is also being used in foreign universities:

- SMA group at LIP6, Paris
- L3S Institute Hannover, Germany

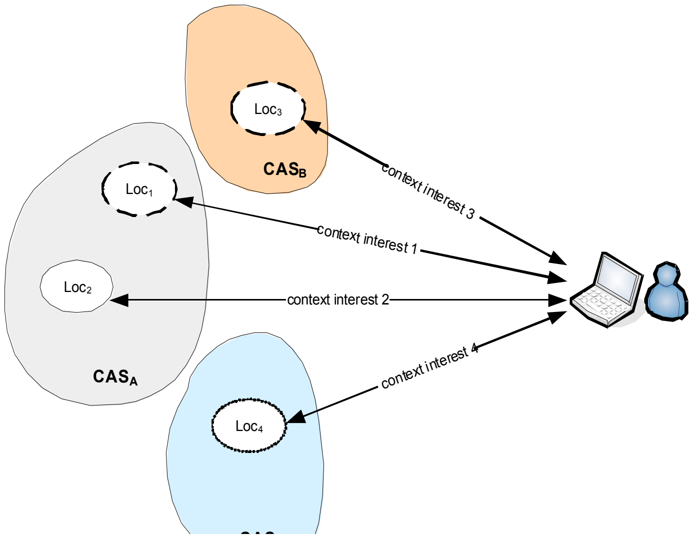
Context Management in Heterogeneous and Evolving Environments - by Ricardo Rocha

Middleware Requirements:

Distributed Management of Context	▪ Efficient dissemination of context information ▪ Requires context service discovery ▪ Transparent context access
Support for Seamless Evolution of Context-Aware Systems	▪ Creation and change of context instances/types without compromising the consistency of global context system types ▪ Seamless evolution for current clients, avoiding disruption
Dynamic Context Discovery	▪ Transparent resolution of dynamic context conflicts, accordingly to client interests (e.g. "location") ▪ Transparent resolution of the context instance that satisfies client interest ▪ Semantics of "Here"
Domains of Context Perception	▪ Context perception (types, instances) according to domains and client location ▪ Distribution does not guarantee this requirement

State of the Art

- Distributed approach

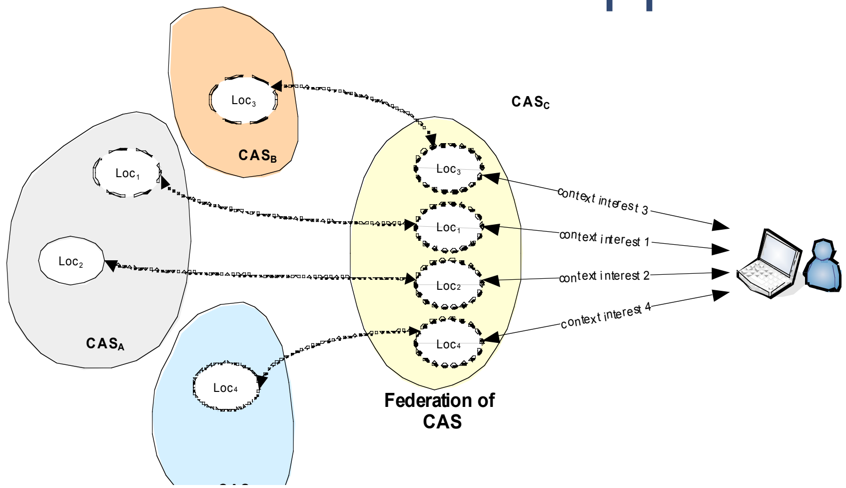


Examples: Confab, Gaia, AURA CIS, PACE, Contory

Limitations

- Client must have explicit knowledge of each CAS
- CAS addresses cannot be solved dynamically

- Federation-based approach



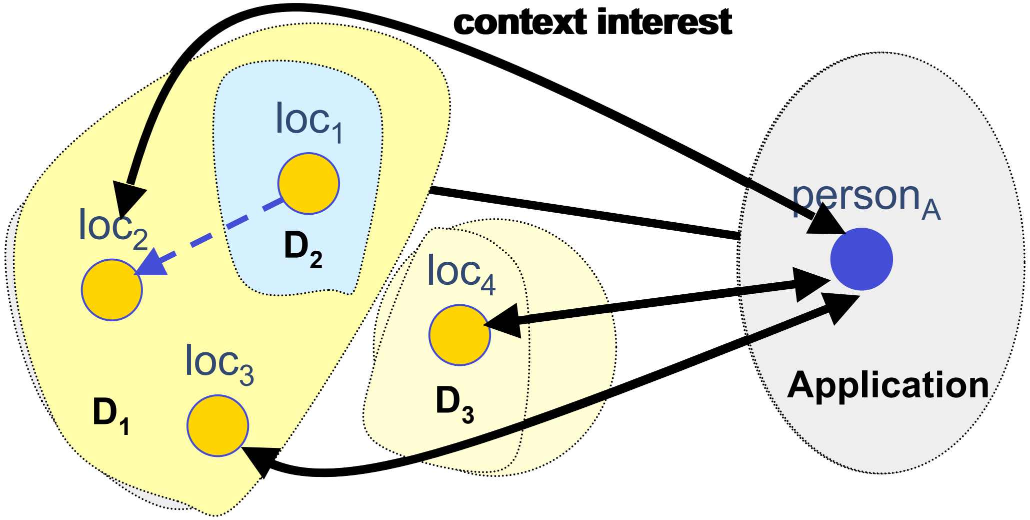
Examples: CAMUS, SCaLaDE, Nexus, CoCo

Limitations

- Reduces the problem to an interoperability and context dissemination issue
- Does not implement context scoping and domain discovery
- Limits the type of abstraction sharing

CAS = Context-Aware System

Proposed Approach: Domain-based Context Management



CAS ➔ Context-Aware System

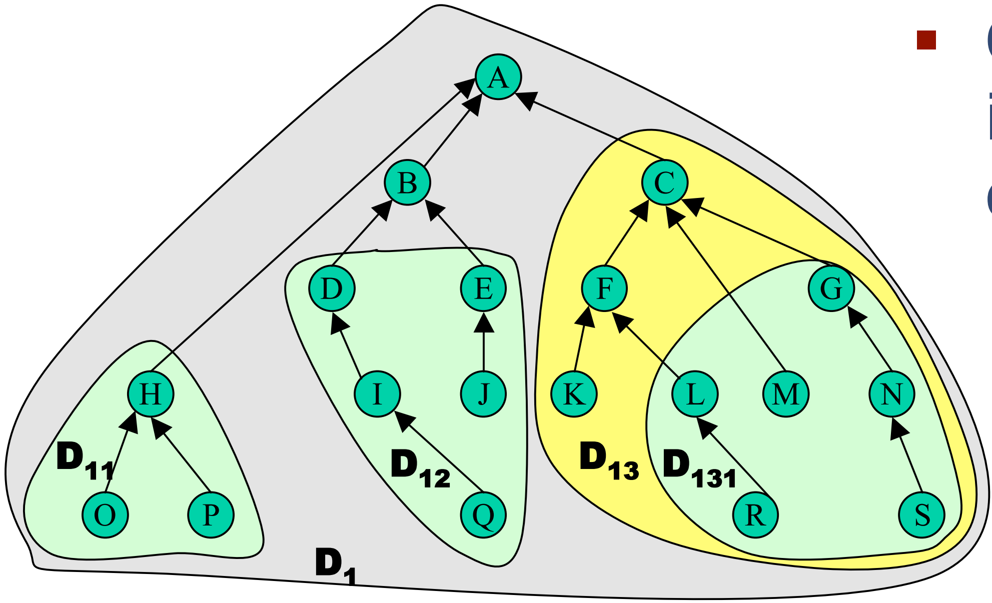
Person's app subscribes to context "location"

- Key concept ➔ **Context Domains**
  - Network segment that establishes responsibility for managing and distributing context, as well for serving a group of context consumers.
- A context domain establishes
  - A set of context types
  - Storage of instances for context
  - Storage of instances for context, when applied to domain entities.
  - Set of context-aware services
  - Sub-domains

Design Rationale and Approach's Strategies

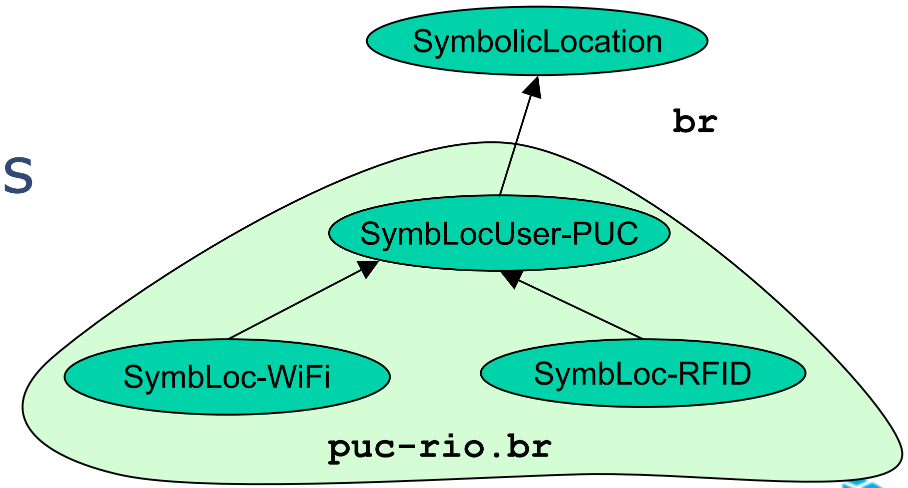
- Appropriate context modeling approach
  - Concepts (context types) of different domains may be interrelated through inheritance and association relationships ➔ **contextual interoperability**
  - Decoupling between context models and inference mechanisms
- Deployment mechanism for context models that allows access to strongly typed context (types solved at development time), using OO concepts
- Middleware architecture supporting context domain concept
  - Distributed architecture
  - Distributed provision and storage of context
  - Transparent resolution and access to context repositories (local vs. remote)
  - It implements discovery of domains
- Programming abstractions for handling context interest based on domains (APIs for specific form of context subscriptions/queries)

Proposed Approach



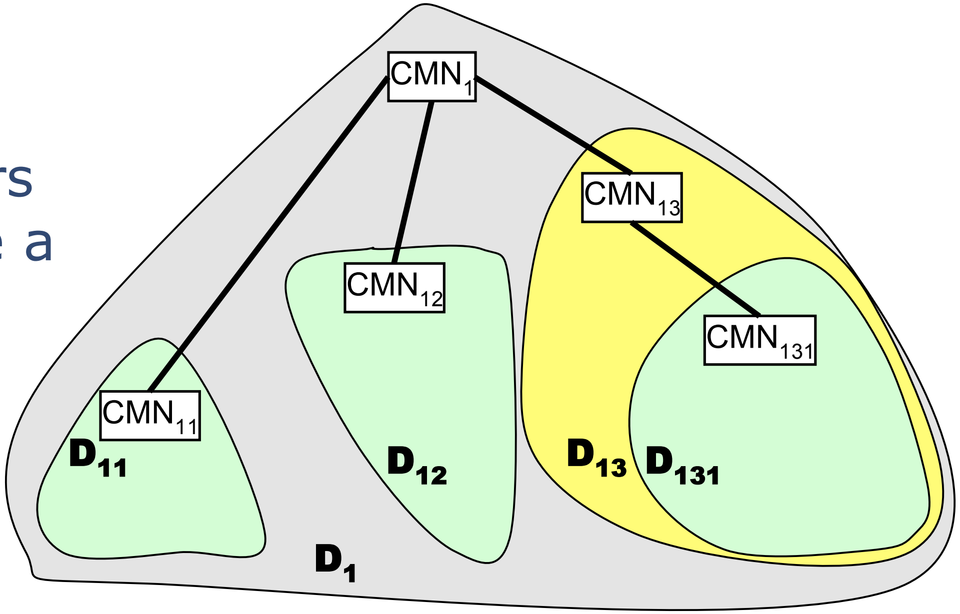
- Context type system is segmented through domains

- Examples for domains `br` and `puc-rio.br`.



Architecture of the Proposed Solution

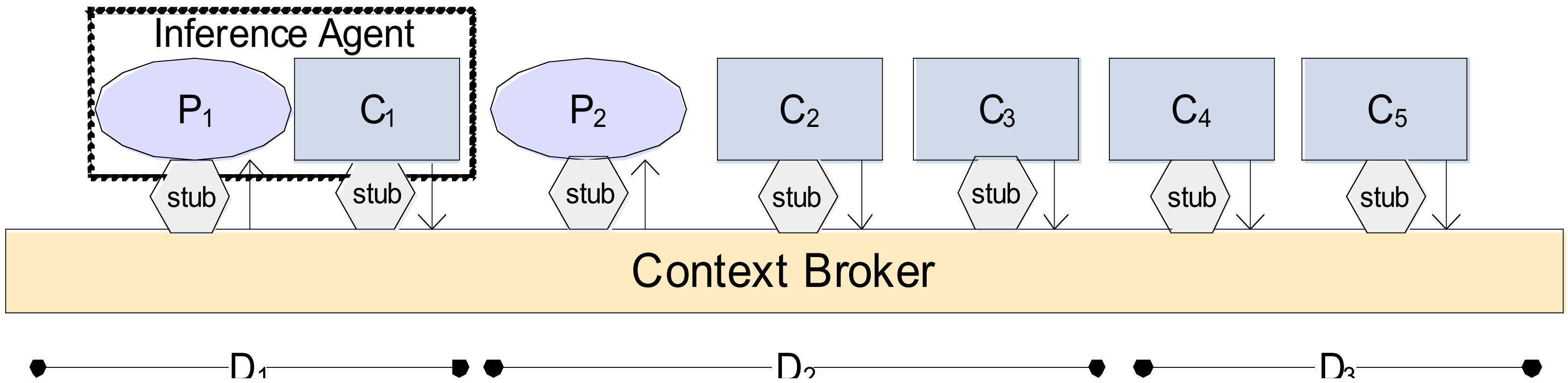
- Context Management Nodes (**CMN**) are responsible for managing context, consumers (clients) and providers inside a domain.



- Services in a node:
  - Middleware for context management
  - Node discovery service
    - Discovers and registers clients in a domain, executing inter-domain hand-off.
  - Context Delivery Service
    - Delivers context to domain consumers (local entities)
    - Manages client mobility

Component Interaction

- Three components interact:
  - Context provider
  - Context consumer
  - Context Broker → an abstraction for distributed context management nodes
- Supports the decoupling among models and inference mechanisms



Hybrid Positioning Service
by Victor Fusco

Motivation:

- Symbolic location is well-suited for most UbiComp applications
- Geographic location is needed for map-based applications and can be obtained from an increasing number of mobile devices (with GPS)
-

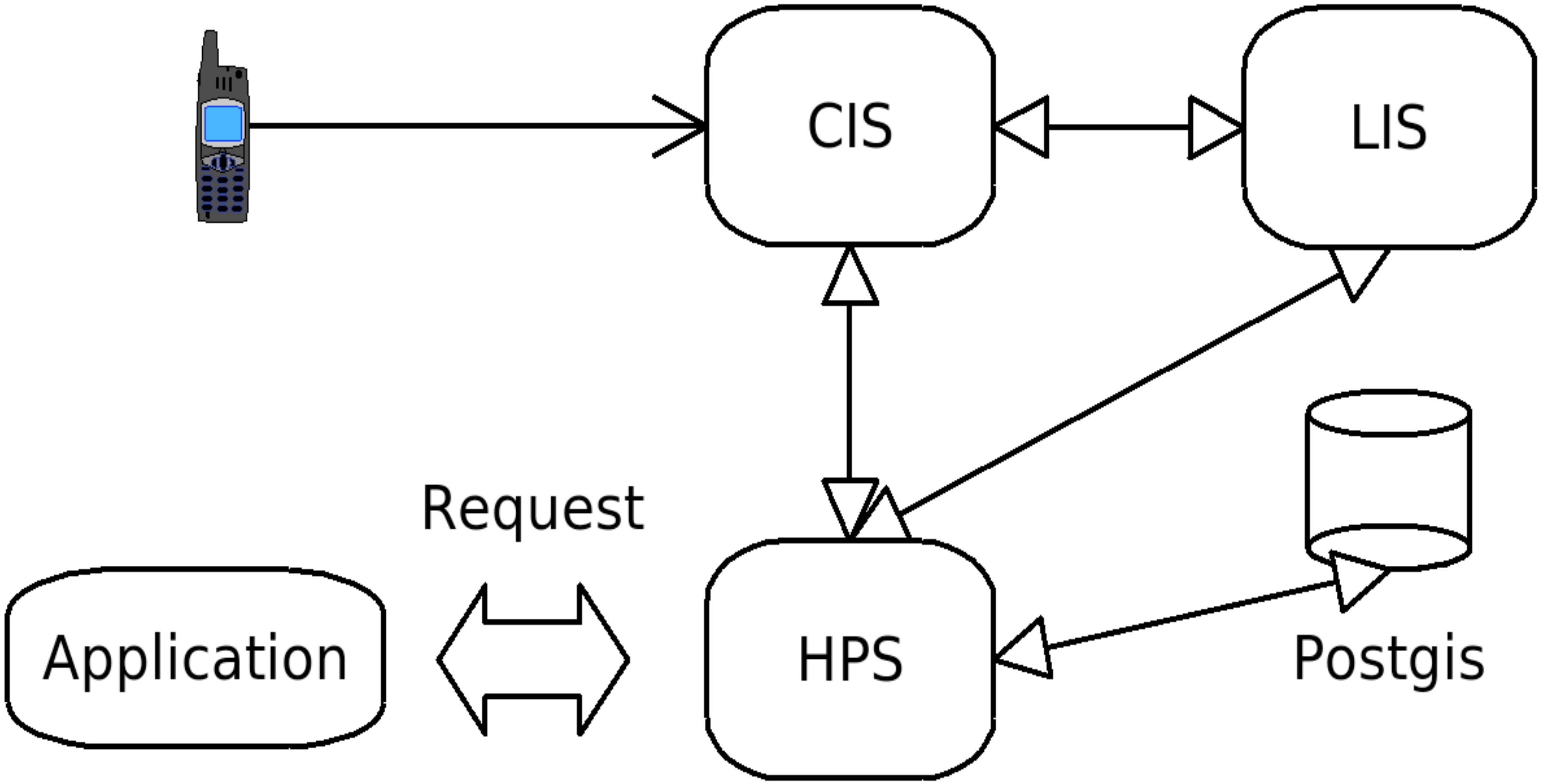
Main Goal: Implement a Positioning Service that

- Converts symbolic regions into latitude and longitude coordinates and converts GPS coordinates into symbolic regions.
- Provides uniform access to the mobile device´s positioning data

HPS Overview

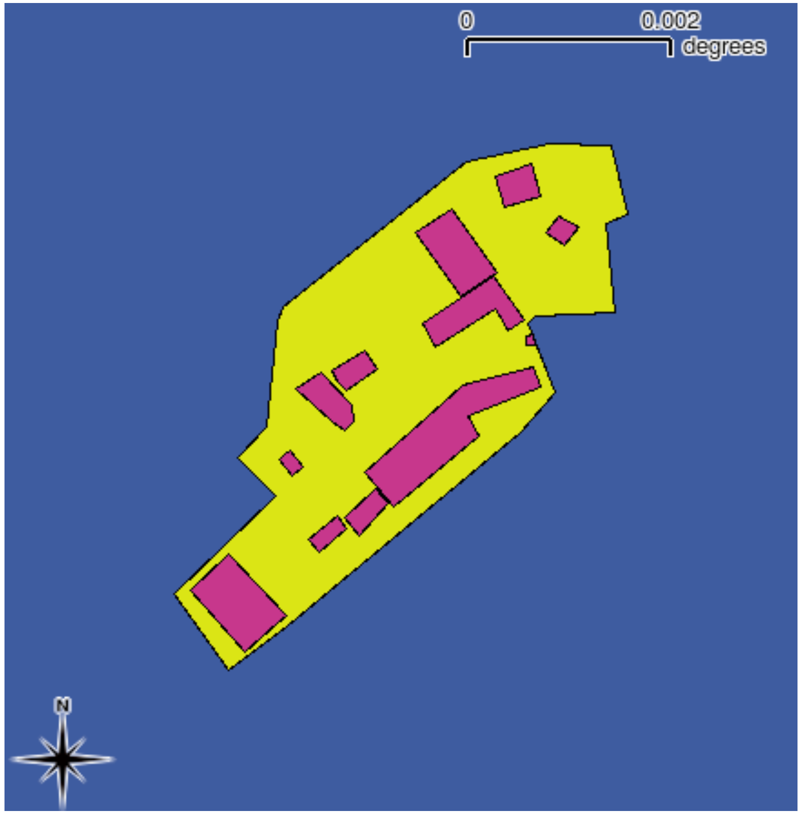
- On request, the HPS collects the device´s localization data from the MoCA's LIS/CIS, if available.
- This data can be symbolic regions (semantic position) or GPS coordinates (geographic position).
- With that information it queries a GIS database to find the missing data.
- Then, it provides the client both types of data and the source it came from.

Main components and their interaction



Main components and their interaction

- HPS gets either the symbolic location name from LIS, or GPS coordinates from CIS.
- With either such information, HPS does a query at vectorial data stored on the PostGIS database (geographic objects in PostgreSQL Object DB) "asking":
  - For (lat,long) gets polygon which contains that point.
  - For (symb.name) gets the centroid of the corresponding polygon



Example: Geo-referenced polygons representing PUC-RIO and some of its buildings.

Collaborative Research Projects

Project Campus (with SMA/LIP6)

Goal: Use Multi-Agent systems for developing inter-domain Ambient Intelligence

Main issues: collaborative reasoning, ontology mediation

Target application: Facilitating user collaboration in smart buildings and homes

Project Mobilis (with UFMG and TU Dresden)

Goal: Support context sharing as means of enriching user's collaborative activities and social networking

Main issues: uniform interface for accessing and sharing heterogeneous context and awareness information

Target application: Tourism and Elderly care

Conclusion

In order to enable wide-area Ubiquity many issues related to context distribution, interoperability and heterogeneity have to be solved.

- Our approach:
  - Support distributed context management, where the context type system is segmented through domains
  - Each domain is independently managed w.r.t. context types, context sources, places and services
  - Provide specialized services (e.g. context providers) which aggregate, synthesize, or infer new context information
    - e.g. LIS, HPS
  - Use ontologies for modeling and reasoning about context types & properties (precision, accuracy), as well as user and ambient properties and states
  - Employ agent technology for decentralized context reasoning, and regulating interaction between users and places

Thank you!

For documentation and download of MoCA visit:

www.lac.inf.puc-rio.br/moca

Or Email to: info@lac.inf.puc-rio.br

For more information, please contact me:

endler@inf.puc-rio.br