Building a Generic Broker for Location Retrieval

N. Prigouris, G. Papazafeiropoulos, G. F. Marias, S. Hadjiefthymiades, L. Merakos

Communication Networks Laboratory
University of Athens, Dept. of Informatics & Telecommunications,
Panepistimiopolis, ATHENS 15784, Greece
email: nprigour@noc.uoa.gr

ABSTRACT

Positioning is an essential component for the deployment of the evolving context-aware concepts. Information Society Technologies (IST) project PoLoS investigates existing schemes for Location-Based Services and latest technological achievements in the sector of Geographical Information Systems, positioning techniques and network interfaces (GSM/GPRS) in order to design and implement a platform capable of providing the full functionality needed to design and deploy Location Based Services. PoLoS' Positioning component (POS) offers undependability of the underlying heterogeneous network infrastructures and positioning techniques, establishing a generic, open, modular, and QoS enabled framework. This article illustrates the technical specifications, the design, the functionality and the prototype implementation of the POS component, its features and the services that the POS component offers to the PoLoS platform and middleware applications.

I. INTRODUCTION

The last decade there has been a remarkable development in the field of mobile and wireless computing. The evolution in wireless hardware technology, combined with the rapid proliferation of mobile/wireless Internet prepared the grounds for the introduction of new alluring types of applications. Context–aware applications are a typical paradigm of such types of services. Positioning is treated as an essential component of the evolving context-aware frameworks [1] [2] [3]. Location Based Services (LBS), where the mobile user is provided with information pertaining to its current position, can be characterized as the most rapid expanded area of context–aware computing, on which much research and effort has been carried out. Several services are based on location techniques, such as:

- Emergency services as defined by the E911 mandate and the E112 recommendation
- Point of Interest (POIs)
- Navigation and routing
- Geocoding and reverse Geocoding

LBS services and applications can be offered by commercial, value added service, application and content providers, carriers and network operators, and can be used by the public administration sector, organizations, industry, and citizens. From the providers' and operators' point of view there is a requirement for a generic platform that provides a re-usable environment for rapid development and deployment of LBS services, in an independence from networking platforms, location techniques and terminal technologies (e.g., cellular handset, PDAs) framework. These requirements are addressed by the PoLoS project, which objective is to design, specify and implement an integrated platform, which will cater for the full range of issues concerning the provisioning and delivery of Location Based Services [4]. This platform will enable the specification, creation and fast deployment of LBS services, without any additional requirement in terms of networks platforms or terminal devices.

On the other hand, for the establishment of LBS services, a generic "Positioning" entity is essential, to take advantage of the various underlying mobile and wireless positioning capabilities and to hide the heterogeneity of the location techniques, providing a general service to the upper middleware and platforms. This article introduces the POS component, which is responsible for providing the information pertaining to the position of a mobile or wireless terminal. Section II discusses the location and positioning architectures and techniques as standardized by the international fora or proposed in the literature. Section III provides a brief overview of the architectural design of the PoLoS platform. A general overview of the POS component, summarizing its features and its novel characteristics, is given in Section IV. Section V elaborates on the internal architecture of the component. The open interfaces offered by the POS are described in section VI and the article summarizes on the advantages and possible enchantments that might be applied.

II. LOCATION ARCHITECTURES AND TECHNIQUES

Several location architectures and techniques have been standardized and proposed for cellular (i.e., GSM /GPRS, UMTS) and Wireless Local Area Network (WLAN) infrastructures. Triangulation, scene analysis and proximity [5] can be utilized for obtaining absolute geographical position within a 3-dimension environment. In Global Positioning System (GPS) the timed difference of arrival of signals from distinct satellites is used for performing the necessary computations and inferring an estimation of the objects' position. Network assistance can be used for improving accuracy. Location determination can also be achieved through terrestrial infrastructure by using methods such as Time of Arrival (TOA), Enhanced Observed Time Difference (E-OTD), Angle of Arrival

(AOA) and Cell ID that differ in terms of accuracy, network/handset impact and time to first fix.

The 3rd Generation Partnership Project (3GPP) has specified a standard configuration of Location Services (LCS) entities in a GSM and UMTS Public Land Mobile Network (PLMN) [6]. In the defined architecture, the Gateway Mobile Location Center (GMLC) is the first node an external location application (LCS client) accesses in the PLMN. The GMLC performs registration authorization and requests routing information on behalf of the LCS client. The communication interface between GMLC and an LCS client is addressed by the Location Interoperability Forum (LIF), which tries to develop an open specification through the Mobile Location Protocol (MLP). The current release of the MLP specification specifies an application-level protocol for querying the position of mobile stations independent of the underlying network technology [7]. The PoLoS platform acting as an External LCS Client with advanced functionality might interact with the GMLC using MLP.

In WLAN systems, a similar architecture can be considered. The PoLoS project has introduced a novel entity, resembling the functionality of the GMLC, capable of performing location estimation in WLAN environments (e.g., 802.11 series). This entity, referenced as GWLC (Gateway WLAN Location Center) is currently under design. GWLC will utilize commercial indoor location positioning systems (MSR RADAR [8], Nibble [9], Ekahau [10]) and should provide an OSA-based [11] lightweight interface. The service provisioned from the GWLC will be published through a service discovery mechanism (e.g. JINI [12], SLP [16] etc.).

Existing LBS platforms (LocationNet [13], CellPoint [14], Webraska SmartZone Geospatial [15]) utilize a Positioning entity, performing routing of requests/responses to/from a mobile user. Although they support the majority of the positioning techniques (e.g., CellID, TOA, AOA, E-OTD and GPS), most of them lack support for WLAN indoor environments. This feature is addressed in PoLoS POS component, where WLAN communication is achieved through the GWLC.

III. POLOS ARCHITECTURE

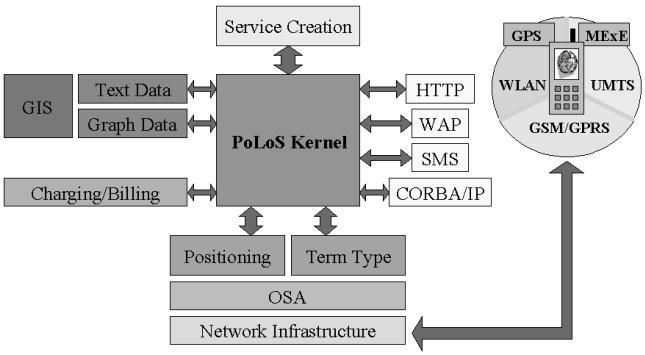

Figure 1: The PoLoS Architecture

The PoLoS project aims to design, specify and implement a generic framework, for the development and deployment of various types of Location Based Services. The platform features a fully component based architecture that is depicted in Figure 1 [17]. As Figure 1 illustrates, the PoLoS platform can be viewed as a unified platform that provides different types of services independently of the underlying technology (WLAN, UMTS, GSM etc.). The main components of the PoLoS platform are:

- The PoLoS Kernel (KRN) comprises the primary component of the pursued platform and is responsible for the co-ordination of communication with the rest components in order to provide the functionality specified for each LBS. It hosts the logic for the deployment, invocation and execution of each service offered by the platform.
- The Geographical Information System (GIS) component interacts with existing deployed GIS repositories retrieving both textual and visual information.
- The POS component is responsible for providing the Kernel with the appropriate information pertaining to the current position of a mobile terminal accessing a particular service. Access to the network is implemented through an OSA /Parlay interface.
- The Interfaces component enables the communication of service specific information to entities external to PoLoS (e.g. mobile terminals) and handles all requests in a device independent way. It supports different transport protocols from SMS and WAP to HTTP/IP.
- The Service Creation Environment (SCE) is an Integrated Development Environment (IDE), which supports the service designer in specifying the service control logic, parameters and additional support elements through the deployment of scripts, written in a XML–based language designed for this purpose.
- The charging/billing module collects and stores invocation information pertaining to end-users in order to perform charging/billing operations.

The PoLoS project, adopts state-of-the-art software tools (EJBs [20] [19], JMS [18], JAIN [21], Open Service Access/Parlay [11]) for platform development. Moreover, it interoperates with the latest transport technologies and techniques in the field of wireless–mobile communications (WAP, SMS, HTTP etc.) in order to offer the end-user seamless access to distinct services.

IV. POS OVERVIEW

The positioning component (POS) of the PoLoS platform is responsible for providing location information to the kernel. It integrates the underlying positioning technologies, providing a unified interface, overriding the differences that are introduced by the diversity of the underlying location tracking techniques. POS offers a well-defined service to the KRN via a communication interface. This interface supports four (4) types of requests forwarded by the KRN to the POS component:

- Request-Response (RR)

- Event-driven Request (ER)
- Periodic Request (PR)
- Generic Request Response (GR). Similar to a Request Response except that there are no attributes associated with it. The POS should use appropriate criteria to define a best-fit response.

Requests of ER type contain certain attributes and result in position reporting based on certain location related–events occurring in the underlying network (e.g. a user enters or exits a specific geographical area). In PRs, the KRN explicitly defines the mean time between the resulting location reports.

POS combines a number of enhanced features that differentiate it from other similar approaches. The adopted POS design goals are:

- **Generality**: by supporting various types of request (RR, ER, PR, and GR) and offering location information retrieval, based on either an IP address (Ipv4, Ipv6) or an E.164 address [22], through several types of network infrastructure.
- **Portability**: through the usage of the Java language and the features offered by the J2EE technology (EJBs, JMS, JMX [23]).
- **Modularity**: due to its modular architecture (see section V) and the discrete functionality that is supported by each of the sub-modules.
- **Efficiency**: by constructing asynchronous method calls between its sub-modules, the introduction of micro-Wrappers (see section V.E) and the incorporation of multiple Message Queues that enable the concurrently processing of multiple requests.
- **Extendibility**: due to POS modular design as well as the use of Java technology features such as RMI, EJBs and JMS. The POS design minimizes the relationships between sub-modules and allows their deployment in a fully distributed environment.
- **QoS orientation**: by disposing mechanisms for QoS support based on certain attributes provided during a request, such as time to respond and priority level.
- **Openess**: due to the adopted OSA compliant interface with the underline network infrastructure.

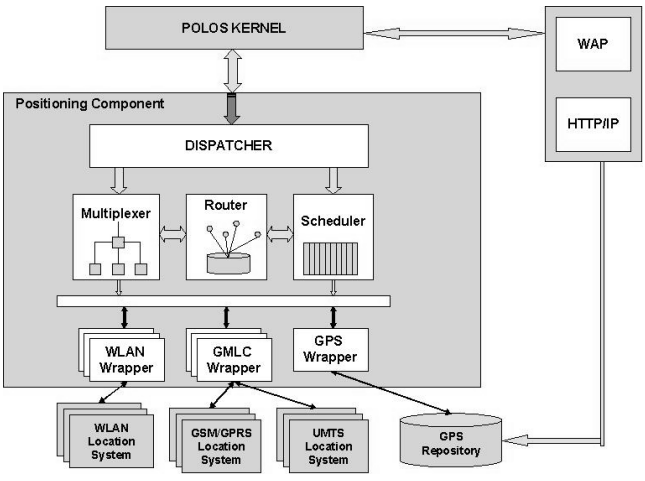

Figure 2: POS Architecture

V. POS ARCHITECTURE

The POS internal architectural design is presented in Figure 2. Most of the modules comprising POS are implemented in Java as simple EJBs with the exception of the Wrapper component, which features a more sophisticated design.

A. Dispatcher module

The Dispatcher module is responsible for communicating with the KRN. It receives requests from the KRN, pre-parses them and forwards them to the appropriate entity (Multiplexer, Scheduler). Requests of RR and GR type are routed to the Scheduler while the Multiplexer module handles the ER and PR requests.

B. Router module

The Routing module incorporates an internal database, with proper mappings between IP addresses and Wrapper IDs, which are used in order to decide the wrapper that will handle a request. The Router encapsulates the logic for resolving the IP address of the GMLC/GWLC entity based on the users' address (IP or E.164) that is passed from KRN during the request. The Router module also provides a well-defined administrators' interface for configuration management (insertion/deletion) and monitoring of the database's routing information entries. Through this interface, the POS administrator controls the creation/destruction of Wrapper objects pertaining to specific database entries.

C. QoS Scheduler module

The QoS scheduler handles requests of RR or GR type forwarded by the Dispatcher. Requests are firstly stored in the queue that corresponds to their QoS class and then they are scheduled based on various criteria such as response time, priority level etc. Each class is served currently in a round robin fashion, whilst other scheduling disciplines (such as WFQ [24]) will be applied depending on efficiency and performance goals.

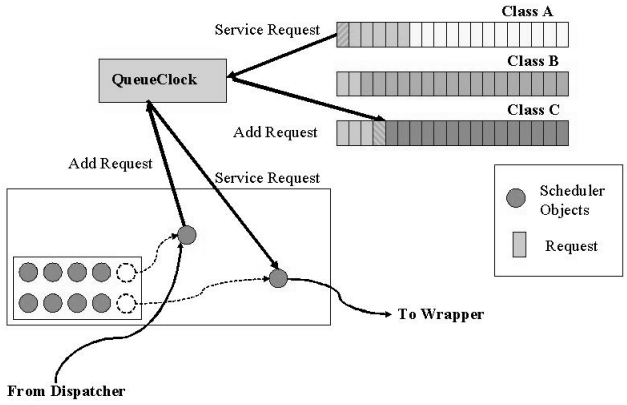

Figure 3: Design and Architecture of Scheduler Component

The QueueClock (QCl) element can be considered the heart of the module. It is the entity responsible for the maintenance of the queues and co-ordinates the insertion/removal of requests to/from each Class queue. The Scheduler Objects (see Figure 3) act as communication points with the neighbor entities and their

role is twofold. When invoked by the Dispatcher unit they are used for adding requests to queues through the QCl. The QCl can also request a Scheduler object in order to service a request. The Scheduler forwards requests to Wrappers based on information obtained from the Router. When a Wrapper accepts a request, the Scheduler removes it from the queue. Each service request can be retried for a number of times before a time out occurs.

D. Multiplexer module

The Multiplexer module handles requests of ER and PR type. It communicates with the Router obtaining information relatively to the Wrapper that will handle the request. Its role consists in multiplexing service requests of the aforementioned types emanated from the same user/terminal. The multiplexing can significantly reduce the signaling overhead produced from the ER or PR messages, since the position information is now retrieved only once.

E. Wrapper modules

Wrappers are entities that exist at the bottom level of the POS component and are created or destroyed by the Router module during the process of insertion or deletion of routing entries. They act as communication points with the underlying network infrastructure, and are designed to serve multiple requests/responses simultaneously. Three types of Wrappers exist:

- **GMLC Wrapper**: handles communication with the GSM/UMTS location system (i.e. a GMLC).
- **WLAN Wrapper**: handles the interface between the POS and WLAN location system (i.e. GWLC).
- **GPS Wrapper**: handles the interface between the POS and the GPS Repository. ER requests are not supported.

Figure 2 illustrates that multiple instances of each Wrapper type may exist. Each Wrapper instance pertains to a distinct operator of a certain network type and is recognized within the PoLoS system by a unique Wrapper ID. There is also the possibility that an operator should have multiple Wrappers of a particular type.

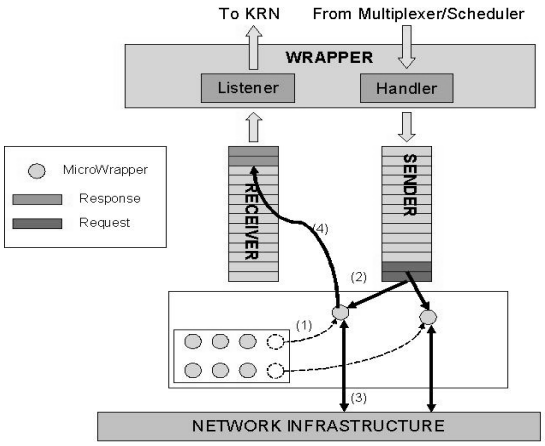

Figure 4: Internal Design & Architecture of the Wrapper component

Figure 4 elaborates on the internal structure and functionality of the Wrapper module presenting the actual processing of a received request. Every request is intercepted by the Handler, assigned a unique id and then forwarded to a proper SENDER Message Queue (JMS queue). Certain micro-wrappers are activated (1) and remove the request from the queue (2) in order to process it. The micro-Wrappers communicate with the underline network infrastructure (3) through a well-defined OSA API. The obtained results (i.e. current user location, periodic location reports) are forwarded to another JMS queue (4). The Listener entity that exists within the Wrapper poles this RECEIVER queue for possible messages and forwards the results upwards to the KRN.

Although not depicted in Figure 4 each queue may serve multiple Wrappers simultaneously. New pairs of SENDER, RECEIVER queues can be introduced by the system administrator and assigned to specific Wrapper instances for load balancing reasons.

VI. POS INTERFACES

This section presents the interface between POS and its neighbor entities. We elaborate only in the communication of POS with the KRN component, since POS interface with the underline network infrastructures is based on the OSA/Parlay API specification [11].

POS provides two method calls for the communication with the KRN. GetLocation() is the generic method used for issuing any type of request to POS while stopLocationRequest() is used explicitly by the KRN to signal the termination of ER and PR notifications. Using these calls, the KRN passes to POS the Service ID and the User ID, as well as a set of attributes in the case of getLocation() (Table 1). The first two arguments uniquely identify a request within the PoLoS platform. The attributeSet is a structure that contains information necessary for POS operations. Table 2 summarizes these attributes, and gives details on their types and possible values.

Table 1: Interface KRN → POS (Methods Description)

Methods	Arguments	Description
getLocation()	serviceID, userID, attributeSet	Generic request method for retrieving the user's location
stopLocationReporting()	serviceID, userID	Stops the sending of location requests (ER & PR only)

Table 2: POS Request Attributes (AttributeSet Description)

Attribute Name	Type	Details
requestMethod	String	RR, PR, ER, GR
priorityLevel	Integer	Level of Priority to be issued by Kernel
Period	Float	For PRs only. Values in msecs
timeToRespond	String	Values: NO_DELAY, LOW_DELAY, DELAY_TOLERANT, USER_DEFINED
posWrapper	String	The KRN explicitly defines the Wrapper. Values: GMLC, GWLC, GPS
altitude	Boolean	The KRN explicitly defines if it needs the Altitude.
type	String	Values: LAST_KNOWN, INITIAL, CURRENT, DON'T_CARE
method	String	Values:E-OTD, AGPS, CellID, ToA etc.
accuracy	Integer	Meters
area	Hashtable	Defines the desirable area in the case of an ER positioning request.

Figure 5 depicts a sequence diagram that provides details on the various method calls and message exchanges between POS sub-modules for RR type of requests. In the diagram below, userID is a structure that contains addresses that uniquely identify an end user (e.g. E.164 or IP) while serviceID is a discriminator for a KRN service.

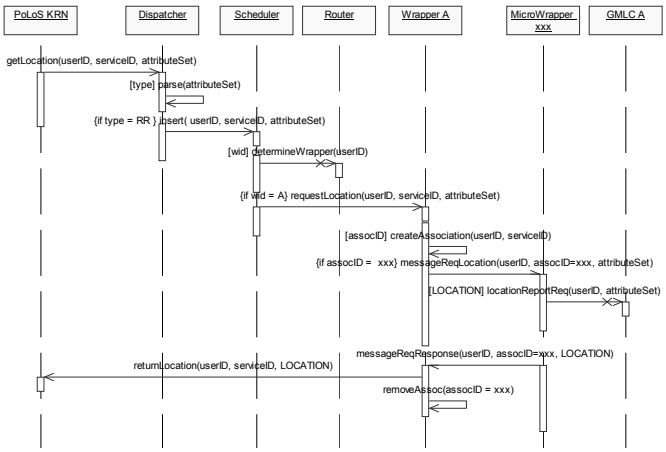

Figure 5:POS Request Response (RR) Sequence Diagram

VII. CONCLUSIONS

PoLoS Positioning Component deploys a generic framework for data positioning retrieval. Based on state-of the-art tools (such as J2EE) it consolidates a number of characteristics that are suitable for deploying LBS applications. Although designed and implemented within the PoLoS project, it is completely decoupled from the PoLoS platform, due to the generic Positioning API offered and the modular architecture that is adopted. POS component can be deployed either as an integral or as a plug-in module of other core systems. Its most pursued feature is the provision of a unified interface that hides the heterogeneity of the underlying network infrastructure, supporting position data retrieval by mobile and wireless networks. Work is still under consideration, and concerns the implementation of scheduling and multiplexing schemas that can be used within the Scheduler and the Multiplexer module respectively, in order to maximize efficiency and performance of the POS component.

VIII. ACKNOWLEDGMENTS

The work presented in this paper has been performed in the framework of the project IST-2001-35283 "PoLoS", which is partly funded by the European Community and the Swiss BBW (Bundesamt fur Bildung und Wissenschaft). The authors would like to acknowledge the contributions of their colleagues from CSEM, ALCATEL SEL AG, Telefonica I+D, Epsilon Consulting Ltd, INTRACOM SA, EPSILON SA and the University of Athens.

REFERENCES

[1] U. Leonhard, J. Magee, P. Dias, “Location Service in Mobile Computing Environments,” Computer & Graphics Special Issue on Mobile Computing, Vol. 20, Nr. 5, Sept/Oct ‘96

[2] P. Beadle, B. Harper, G.Q. Maguire and J. Judge, “Location Aware Mobile Computing,” Proc. IEEE Intl. Conference on Telecommunications, Melbourne, Australia, Apr. 1997

[3] A. Smailagic, and D. Kogan, "Location Sensing in a Context Aware Computing Environment," IEEE Pervasive Computing, Jul.-Sept. 2002

[4] IST-2001-35283 Project PoLoS, “Integrated Platform For Location Based Services”, Public Deliverable D011, “Project Presentation”, Apr. 2002

[5] J. Hightower and G. Borriello, “A Survey and Taxonomy of Location Systems for Ubiquitous Computing,” University of Washington, Computer Science and Engineering, Technical Report UW-CSE 01-08-0, Aug. 24, 2001

[6] 3GPP TS 23.002 V5.6.0 (2002-03), 3GPP; Technical Specification Group Services and Systems Aspects; Network architecture (Release 5)

[7] Location Interoperability Forum Technical specification TS101 v2.0.0, Mobile Location Protocol, Nov. 2001

[8] P. Bahl and V. Padmanabhan, “RADAR: An in-building RF based user location and tracking system”, Proc. IEEE INFOCOM, Mar. 2000

[9] P. Castro, P. Chiu, T. Kremenek, and R. Muntz, “A Probabilistic Room Location Service for Wireless Networked Environments,” Proc. Ubicom 2001

[10] Ekahau’s Positioning Engine, Jan. 2002

[11] Open Service Access (OSA); Application Programming Interface (API); Part 1: Overview, Final draft ETSI ES 202 915-1 V1.1.1 (2002-11), www.parlay.org/specs/es_20291501Overview.zip

[12] Sun Microsystems, Jini™ Architecture Specification, Version 1.2 Dec. 2001, www.sun.com/software/jini/specs/jini1_2.pdf

[13] http://www.locationet.com

[14] http://www.cellpoint.com

[15] http://www.webraska.com

[16] E. Guttman, et. al, “Service Location Protocol Version 2,” RFC 2608, Jun. 1999

[17] IST-2001-35283 Project PoLoS, “Integrated Platform For Location Based Services”, Public Deliverable D111, “PoLoS Platform Architecture and Services Specification”, Sept. 2002

[18] Sun Microsystems, Java™ 2 Platform, Enterprise Edition (J2EE™), http://java.sun.com/j2ee

[19] Sun Microsystems, Enterprise JavaBeans 2.1 Specification, Public Draft

[20] E Roman, “Mastering Java Beans”, 2nd Edition, John Wiley & Sons 2002

[21] JAIN White paper, java.sun.com/products/jain/WP2002.pdf

[22] P. Faltstrom, “E.164 Numbers and DNS”, RFC 2916, Sept. 2000

[23] Sun Microsystems, JMX Framework, http://java.sun.com/products/JavaManagement

[24] A. Demers, S. Keshav and S. Shenker, “Analysis and simulation of a fair queueing algorithm,” Proc. ACM SIGCOMM’89