On Service Deployment in Ubiquitous Computing

O. Ardaiz, F. Freitag, L. Navarro
Computer Architecture Department, Polytechnic University of Catalonia, Spain
{oardaiz, felix, leandro}@ac.upc.es

Abstract

Introduction of new services on the Internet is a laborious, time-consuming process. Ubiquitous computing environments also present that same challenge; even augmented due to dinamicity of mobile entities and wireless connectivity, and location and environmental awareness of services. A framework for service deployment will facilitate service introduction in both cases.

In this paper we present our investigations on two of the components of an Internet service deployment framework: service specification interface and mechanisms for service deployment. We present issues we have encounter when trying to extended this framework for service deployment in ubiquitous computing environments. Service deployment on ubiquitous computing environments will require a resource positioning and surrounding functionality; service specification interfaces will be required to incorporate location and environmental service parameters; and new service templates for information aggregation services will appear. Besides mechanisms for service deployment have to adapt to such dynamic environment.

Lastly we present our prototypes and experiments on service deployment.

1. Introduction

"The introduction of new services into existing networks is usually a manual, time consuming and costly process", "there is an increasing demand to add new services to networks to match new application needs" by Campbell et al. [3], "vendors are hesitant to support service before they gain user acceptance, yet the utility of network services is dependant on their widespread availability" by Tennenhouse et al. [17]; this cites expose the relevance of facilitating service introduction into network. That is, enabling an easy, efficient and secure introduction of new services will promote service development, flooding the Internet with new services for the benefit of end users.

Ubiquitous computing [20][7][12] is a paradigm in which networked computing resources are present anywhere. Users augment their computing and communicating capabilities with lots of nearby computing devices, and the network achieves an infinitesimal capillarity reaching everywhere. Resources are mobile and have wireless network connectivity. In such scenario services make use of information, processing capabilities and storage and output resources obtained anywhere.

Ubiquitous computing is around the corner. But ubiquitous computing services will have to be introduced into the system similarly as today's Internet services. We are extending our solution for Internet service deployment to ubiquitous computing environments. It will permit an easy-to-use and cost-effective service introduction in ubiquitous computing environments, thereby allowing ubiquitous services creators to provide its services to more users, and will benefit users with a wider service offer.

1.1. Related Work

The problem of introduction of services has been addressed by different approaches from active networking to open signaling to programmable networks [3]. Most of them have concentrated on providing environments where services can be remotely activated in a secure way. Research on mechanisms for network deployment is being studied at Xbone [18] in ISI and Tempest [16] in Cambridge University. They allow virtual networks to be set up on the Internet to behave as virtual private networks or to isolate experimental services from the Internet. Research on how to dynamically deploy a service is also being conducted at the Darwin project [4] at CMU which attempts to dynamically instantiate virtual meshes, collection of networking resources, to be used for multiparty conferencing with quality of service. Globus project [11], a joint effort of various universities and research centres, has developed a protocol for reservation and co-allocation of resources in computational grids. It allows grid applications to obtain the set of required resources for its execution with QoS guarantees.

Our work concentrates on the development of efficient and cost-effective deployment mechanisms, and easy-to-use framework interfaces. As well we are broadening the scope for service deployment to ubiquitous environments.

2. Internet Service Deployment Framework

2.1. Framework Requirements

To allow for ease creation of services requires the development of a framework that provides basic building blocks with which to construct a service deployment system. This framework should support the automation of every task required to deploy a service. We attempted to reuse the Xbone system for overlay deployment to deploy services [1], however it was not an ideal solution. It required many fundamental changes in its deployment mechanisms, and it was designed for different allocation strategies, so we set ourselves to develop a framework for service deployment from scratch. Our first task was to define which functionality is demanded from this framework.

Deploying a service involves:

1. Obtaining service specifications,
2. Mapping specifications to resources,
3. Discovering resources,
4. Gathering resources,
5. Configuring resources,
6. Activating service,
7. Providing a management interface.

So this framework architecture must be composed at least of resource agents at resource providers nodes and deployment managers at service providers nodes. Resource agents are responsible for publishing resources, mediating between resources and service providers' deployment managers, configuring resources, activating services, and returning management interfaces.

Deployment managers are responsible for obtaining service specifications, mapping specifications to resources, discovering resources, gathering resources, trading with resource agents on behalf of service providers, and managing overall deployment operation.

Service providers demand these characteristics from this framework:

- Usability, to allow for an easy service introduction,
- Efficiency and cost-effectiveness, for rapid and cheap service provision,
- Manageability, to govern services once deployed,
- Safeness and fault tolerance, to assure service integrity and availability.

Resource providers demand these characteristics from this framework:

- Efficiency and cost-effectiveness, for highest resource revenue,
- Security, to avoid resource misuse.

There exist much research from active networks to programmable networks and even commercial system that provide security and management functionality to programmable nodes [3], which can be integrated in a framework for service deployment. However we have identified two framework components that provide functionality specific for this problem requiring extensive research to meet every party requirement. On the one hand it is required a service specification interface that allows for easy service deployment requests, on the other hand deployment mechanisms are required to provide for an efficient and cost-effective deployment.

2.2. Service templates

Defining specifications of a service on the Internet is a laborious task. A service provider must calculate which resources capabilities and number of resources to use, sub-networks where service has to be provided, organization among resource, etc.

A service description language can be defined to facilitate service specification as it has been done to define overlay networks in [4] or [16] that would allow to specify service requirements in terms of number and type of basic resources, connections among resources, and service components. However the main goal of the service deployment framework is to facilitate service provider's deployment of their services. Reviewing the evolution of traditional programming environments we find that a programming languages is a complicated technique for many users since it offers more functionality that what an average user requires. For this reason application frameworks such as Microsoft Foundation Classes or communication frameworks such as [8] have been introduce. These programming frameworks allow a service creator to implement their service starting from a skeleton template, which already provides much of the required functionality. They only have to adapt it for their special service requirements. An important benefit of templates is that system components implementing those templates functionality will be reused many times, therefore programming errors can be reduced and systems components stream-lined.

A service deployment framework has to provide a set of predefine service templates. Service providers will select a template and fill it in to create his service specification.

For Internet services we have identified a template that can be instantiated to most Internet services: the dissemination service template. This template allows the creation of content distribution networks, multicasting networks, video on demand services, and mobile adaptation services. Service providers that choose this template know they will obtain a service specifically allocated, configured and organized for information dissemination. Service providers have to select which types of capabilities are require for resource nodes where to activate its services: proxy -caches, streaming capabilities, WAP gateway, etc. They have to select which Internet regions they want their service available. They have to select which is the expected traffic caused by service clients. They have to select which type of service they plan to deploy either a live service or an on -demand service. Deployment managers and resource agents deploying a service that conform to the dissemination template, use specific algorithms for resource allocation, service organization and resource discovery.

2.3. Deployment Mechanisms

A first solution represents the method currently used for provisioning services that require resource allocation at multiple nodes, such as virtual private networks or content distribution networks. It is the SNMP management station based method, in which a centralized entity monitors, chooses and configures resource agents [6]. As every centralized system, it is not the best solution in terms of scalability or fault tolerance. The second method we propose is multicast injection. It considers resource agents as active entities that can decide autonomously whether to accept or discard a service activation request based on local policies. Therefore deployer entities can only inject service specifications (including a reference to application binaries and data) into the system and expect that enough and appropriate resource agents accept it, else they can inject a cancel service request. Obviously injection has to be implementing with some multicast communication scheme.

Per surrogate configuration deployment involves the following steps:

1. Deployer continuously discovers and monitors surrogates resources / surrogates advertise their resources,
2. Provider request service deployment,
3. Deployer calculates resource allocation,
4. Per surrogate configuration,
5. (At timeout), every surrogate response is ok OR deployer sends per surrogate rollback.

It is a centralized system where a deployer has to gather information from every surrogate in order to calculate resource allocations for every deployment requests. It requires computational and bandwidth resources in a centralized location, which is subject to failure. In addition, since more deployers will be contending for surrogates, a deployer can be denied allocation of resources in a surrogate that was thought to be available but another deployer got its resources a little earlier.

Multicast injection deployment has to take the following steps:

1. Provider request service deployment,
2. Deployer injects application service specifications in a global mcast channel,
3. Surrogates map spec -> allocate resource and mcast service match on application channel OR do nothing,
4. Surrogates compare published matches with itself -> service activation OR cancel service and release resources,
5. (At timeout) every region serviced OR surrogate cancels service and release resources.

Clearly this method requires less resource on deployer entities, while permitting surrogates to have more autonomy. A problem of this solution is the level of under-utilization of surrogate resources due to conditional allocations while deployment takes place. On figure 1 we present advantages and disadvantages of both deployment mechanisms, which are discussed next.

	Advantages	**Disadvantages**
Per-node Configuration	-More control	-Deployer computation -Deployer traffic -Stale information -Unused allocations
Multicast Injection	-Faster activation -More adaptable -Simple deployers -Robust system -Loose relations	-More unused allocations -Network traffic

Figure 1 - Mechanisms comparison

Per node configuration presumes more control by the deployment entity over resource agents, since resource agents are passive entities controlled by deployers. In contrast, multicast injection presumes a looser relation between deployers and resource agents. Resource agents subscribe to deployers at will, retaining its autonomy on local configuration actions. It is easy to establish relations with more nodes when least requirements are put on both parties, therefore larger sets of resource agents can be available for multicast injection dynamic deployment. Per node configuration has to implement a centralized resource allocation algorithm, which can require high computational resources when computing on Internet scales. However multicast injection makes use of a distributed allocation algorithm, which makes it more scalable and fault tolerant. Per -node configuration is not as scalable since as the number of nodes rows it requires increased traffic capacity at the deployer site to continually monitor every resource. Multicast injection is more scalable than per node configuration since its traffic and computational requirements do not create a bottleneck at deployers. Stale resource information in per node configuration deployment causes deployer entities to select nodes that have been allocated, and not to consider for allocation nodes that have already available resources. Because multicast injection does not use stale information it is more responsive and adaptable than per node configuration deployment. Multicast injection requires simpler deployers since it just sends a service deployment request: they do not have to be c ontinuously monitoring the state or individually configure every surrogate. It is also more robust since deployers do not have to detect and recover from every surrogate failure. Unused allocations occurs in both cases when surrogates are allocated and released shortly afterwards without providing any service in that period, due to being unable to find enough surrogates for deployment. This effect has to be less important in per node configuration since there allocations are only requested to a subset of surrogates.

3. Service Deployment in Ubiquitous Computing

3.1. Ubiquitous Computing Services

Due to the ubiquity of computing resources in ubiquitous computing environments, current services extend their functionality and new services are enabled.

Services in a ubiquitous computing environment differ from conventional Internet services in that computing resources building those services are also characterized by two new parameters:

- *Location,* since resources can be at any geo -spatial location. Technolog ies such as GPS [15], Cellular Radio Location Tracking Systems [13] or Bluetooth [2] allow for geographical positioning of computing resources.
- *Surroundings,* since resources can be at any environment: outdoor, at the highway, at home, at school, in a car, etc. (vs. Internet computing where resources are only at offices, at data centres, or at home). Technologies such as sensors [5] or geographical information systems GIS [19] allow for determination of resources surroundings.

Many new services are enabled s olely due to these characteristics of ubiquitous computing resources: location-aware services such as positioning systems [9] or proximity services [13], and environment -aware services such as smart offices [14] or navigation systems [10].

We have identifi ed three issues that differentiate service deployment in a ubiquitous computing environment from traditional Internet service deployment:

- Location-aware or position-constraint deployment,
- Environment-aware or surrounding -constraint deployment,
- System dynamics.

In a ubiquitous computing environment services to be deployed can specify their geographical constraints to indicate the geographical area where they wish to provide its service. As well services can specify environmental conditions constraints to indicate environments where they wish to provide its service. Finally inherent dynamics of a ubiquitous computing environment, due to entities mobility and wireless connectivity, will influence mechanisms for service deployment.

Solving these issues requir es extending our service deployment framework in three areas:

- Resource positioning and surrounding systems, that allow for location and environment aware deployment,
- New service templates, that take into account new services and location and environment p references of services,
- Adaptative deployment mechanisms, to allow for efficient deployment on very dynamic environments.

3.2. Resource Positioning and Surrounding

Resource discovery is one of the main challenges of ubiquitous computing since in ubiquito us computing environments resources providing a better service come and go becoming visible any time [7]. Resource discovery is also one of the fundamental mechanisms for service deployment. Deployment managers need to discover resources that matches servi ce specification before attempting to configure them. Service deployment in a ubiquitous computing environment has to deploy services on resources that are location or environment constrained. Resource discovery for service deployment in a ubiquitous compu ting environment has to be extended to support location and environment constrained selection.

There exist several system for positioning computing resources. GPS allows for geographical positioning of a device any place on Earth with accuracies ranging f rom 100 mts to 10 mts. Cellular mobiles networks allow positioning of wireless phones with accuracies of cell size. A very promising technology "bluetooth" allows for wireless discovery of peers within a range of 10 to 100 mts, thereby establishing their relative position. Sensors or geographical information systems GIS provide determination of resource surroundings or environmental conditions.

Deployment of services in a ubiquitous computing environment requires a resource discovery system that is location and environment aware. Due to the wide range of location accuracy and environmental conditions possible there will not be a unique system. I.e. bluetooth will be an efficient technology for resource discovery to deploy services on very nearby location. Cellular Location Tracking systems can be used to discover resource for service deployment on a cellular operator network, and GPS should be employed to discover resources to deploy services on a global scale. Surroundings -constraint services which require d real time information such as temperature, traffic state, or light conditions will need to use on site sensor information to discern where to deploy a service. More stable surrounding constraints such as building, road or mountains surroundings can be pr ovided by geographical information systems GIS.

3.3. Ubiquitous Services Templates

In ubiquitous computing environments a new kind of services will be very common: services that aggregates information from a number of information sources and processes i t according to some user rules. Examples of such services are alarm systems and portal services. This kind of services requires a new service template: the aggregation service template. It differs from dissemination services in that it requires different a llocation algorithms and service organization structures; and communication bottlenecks appear at aggregation nodes vs. at edge nodes on dissemination services.

As well services in a ubiquitous computing environment require specification of two new parame ters: geographical location where services have to be deployed, and environmental conditions where services have to be deployed.

So we can summarize the two service template we have identify in an ubiquitous computing environments as:

- *Dissemination serv ice* is an output type service that obtains information from a source or an aggregation service and disseminates it to a number of output resources (proxy servers, video / audio clients, mail servers) scattered over a geographical area or environment. Examples of such services are broadcast events and notification services,
- *Aggregation service* is an input type service that obtains information from a number of input resources (sensors, web cams, audio inputs, web servers,..) scattered over a geographical area or environment. That information, after some initial processing, is transported following an aggregation communication scheme to an aggregation entity where it is processed with an input filter such as an alarm engine, a directory, or custom made.

3.4. Ubiquitous Deployment Mechanisms

A ubiquitous computing environment is characterized by its high dinamicity. Mobile computing entities come and go providing their resources, wireless connection have abrupt variations. Therefore selection of resources where to deploy a service is a much more complicated task than on the Internet where resource availability and connectivity is quite stable. The adaptability of service deployment mechanisms to these changes is the main property to be looked for.

For a start both deployment mechanisms considered for Internet service deployment are valid. However centralized configuration has worst adaptability properties since it requires all information to be available at a centralized point. Centralized configuration in an ubiquitous computing environment will make use of much stale resource information thereby failing to deploy many of its services, unless it increases its monitoring rate to very high rates causing a lot of traffic. It is an option that has little chances of be ing used for ubiquitous deployment. Multicast injection on the contrary has better adaptability properties since it is an asynchronous solution that does not require full synchronization. Nodes have to take decisions autonomously whether to activate a ser vice or not, being able to withstand connection drops. As a last advantage multicast is easier to implement in wireless environments.

4. Experiments

4.1. Prototype

We have developed the Xweb framework prototype that allows for deployment of web services. A web service deployed operates as a set of interconnected service nodes called surrogates that provide service to several web client regions scattered all over the Internet.

It provides service creators a dissemination service template. Web application creators have to fill in web service characteristics such as number of service points, expected total clients traffic, Internet regions where service has to be provided, maximum distance between clients and surrogates, etc.

Dynamic deployment mechanisms make deployment managers and resource agent interact for efficient and cost-effective service deployment. There are two possible deployment mechanisms: centralized configuration or multicast injection. We show a performance evaluation of both mechanisms below.

Resource agents configure web surrogates at resource nodes to start servicing a web application on service provider behalf. As well they control resource usage for appropriate service responsiveness by limiting number of services provided at each surrogate and monitoring service response time.

Once a web service is deployed an SNMP management interface is returned to its creator. Currently services location is expressed as a list of Internet autonomous systems numbers. More accurate location constraints and environmental constraints required integration of some positioning service such as GPS or mobile tracking systems and surrounding services such as GIS or sensors at resources agents. We are evaluating implementations of those technologies for integration into our prototype.

As well we are currently developing the interface template for aggregation service to allow for deployment of event services and multimedia information portal services. This template will allow service creator to choose type of information source: web server, audio input, temperature sensor and location and environmental conditions of resources, to create a service that combines all that information to be provided to clients.

4.1. Simulated performance evaluation

We have evaluated both deployment mechanisms in a simulation on the ns-2 network simulator. We simulated deployment of web services, demanding resources on 5 random nodes, over 25 resource nodes on a typical WAN network topology. Resource demand range from 20 average application per second to 60 average application per second (an average service is a service with media resource demands and media service duration). On figure 2 we show which allocation algorithms were used to allocate resources in each simulation. These are very basic algorithms that try to allocate resources on the nearest surrogate possible or they fail.

Centralized Configuration Allocation Algorithm	**Multicast Injection Allocation Algorithm**

Deployer {
 	
:: Select_nodes {
Foreach service region {among those with enough resources, allocate on nearer surrogate}
}
 	
::Deploy_timeout {
If some region not serviced {send cancel-request}
}
}
	Resource Agent {
	
	::receive_mcast_injection {
	If enough resources && near some service regions {multicast_service_match}
	}
	
	::receive_service_match {
	If peer_agent nearer than this to some service region {stop service in that region}
	}
	
	::deploy_timeout {
	If not every region serviced {stop-local-service}
	}
	}

Figure 2 - Allocation algorithms implemented in simulation

On figure 3 and 4 we show which is the success rate and percentage of unused allocations of both deployment mechanisms. Multicast injection has a high success ratio of deployed application that centralized configuration deployment mechanisms (SNMP). Both mechanisms depart from the maximum number of possible deployed services as the number of resources demanded increases due to requesting already allocated resources. However centralized configuration deployment suffer a success ratio decrease at high loads due to resource contention among deployers. Multicast injection avoids that contention by allocating resources as soon as they are freed, however some resources can be allocated and not used leading to thrashing and deadlock situations.At figure 4 we see that this percentage is very low even at high loads. This is a consequence of service times being much longer that allocations intervals, therefore undesirable thrashing and deadlock shall not occur during normal operation of deployment systems.

These figures tell us that multicast injection mechanisms adapts to resource availability variations better that centralized configuration deployment, and shall be more effective in using resources in ubiquitous computing environments.

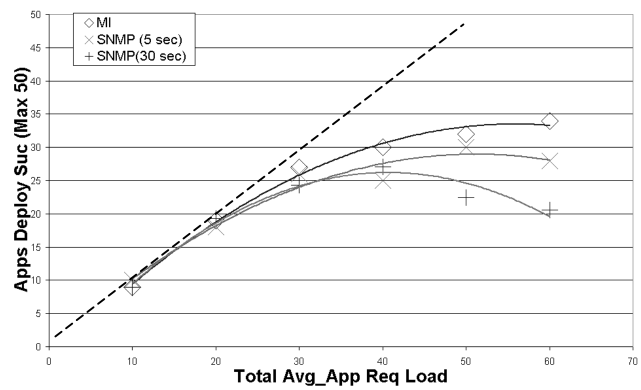

Figure 3 - Deployed services vs. resource demand

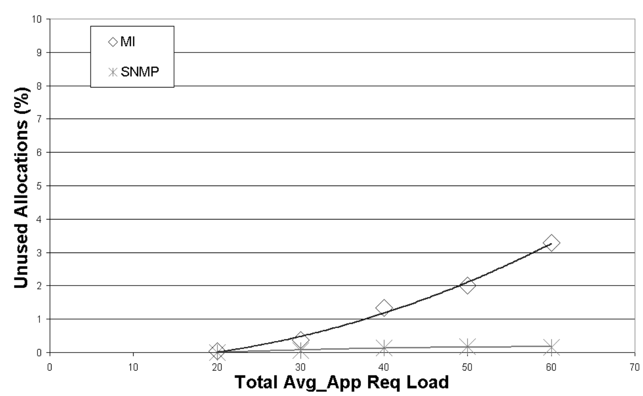

Figure 4 - Unused allocations vs. resource demand

5. Summary

In this paper we have exposed our current framework for Internet service deployment and have characterized which are some of the most relevant issues that service deployment will present on ubiquitous computing environments. Among them we present which new deployment functionality requires the new services of ubiquitous computing, and the new conditions for deployment in ubiquitous computing environments.

Some kind of resource positioning and surrounding functionality will be required for effective deployment of new services on ubiquitous environments. Service templates will be required to incorporate location and environmental service parameters. And new service templates for information aggregation service will be commonly employed to deploy new services that will come about on ubiquitous computing environments due to the large number of information sources that will become available in such scenarios. Mechanisms for service deployment will be required to be even more adaptable to communication and computational availability caused by variable wireless connectivity and mobility of devices. Our multicast injection for service deployment is a good solution that provides good adaptability on Internet environment simulations; it should provide to be valid also in ubiquitous computing environments.

We are complementing the Xweb prototype for aggregation template service deployment. As well we are currently evaluating positioning and surrounding implementations to integrate into the prototype to provide for ubiquitous service deployment.

6. References

[1] Ardaiz O., Touch J. "Web Service Deployment Using the X-bone", In Proceedings of Spanish Symposium on Distributed Systems SEID 2000, Sept 2000.

[2] Bluetooth SIG, Bluetooth Technology, 1998. (http://www.bluetooth.com/technology/)

[3] Campbell A.T., De Meer H.G., Kounavis M.E., Miki K., Vicente J.B., Villela D., " A Survey of Programmable Networks", ACM SIGCOMM Computer Communication Review, April 1999.

[4] Chandra P. et. Al. "Darwin: Customizable Resource Management for Value-Added Network Services",. 6 th IEEE International Conference on Network Protocols (ICNP'98), 1998.

[5] Chen G., Kotz D.. "A Survey of Context-Aware Mobile Computing Research", Department of Computer Science Dartmouth College, Technical Report TR2000-381.

[6] Cisco Inc. "Cisco Provisioning Center White Paper" July 2000, http://www.cisco.com/warp/public/cc/pd/nemnsw/pvcr/tech/provs_wp.htm

[7] Esler M., et al. , "Next century challenges: data-centric networking for invisible computing: the Portolano project at the University of Washington", Proc. 5th ACM/IEEE Conf on Mobile Computing and Networking, Seattle 1999.

[8] Fayad M., Schmidth D. " Object-Oriented Application Frameworks" Guest Editorial, Communication of ACM Vol 40 N 10 Oct 1997.

[9] Feuertein M., Parttt T. "A Local Area Position Location System", IEE Conference Publication n 315, 1989 pp 79-83.

[10] Figueroa F., Mahajan A. "A Robust Navigation System for Autonomous Vehicles using Ultrasonics" Control Engineering Practice, Vol 2 N 1 1994 pp. 49-59.

[11] Foster I., Kesselman C., Lee, C. Lindel B.l, Nahrstedt K., Roy A. "A Distributed Resource Management Architecture that Supports Advance Reservation and Co-Allocations", In International Workshop on Quality of Service, 1999.

[12] Garlan D., "The Aura Project", OOPSLA'99, Ubiquitous Computing Panel, October 1999, Denver CO.

[13] Hellebrandt M., Matha R., "Location tracking of Mobiles in a Cellular Radio Networks", IEEE Trans. on Vehicular Technology, Vol. 48, No. 5, pp. 1558-1562, February 1999.

[14] Hodges S., Louie G. "Towards the Interactive Office" Proceedings of SIGCHI'94, Boston April 1994.

[15] Hofmann-Wellenhof B., Lichtenegger H., and Collins J.. "Global Positioning System: Theory and Practice", Springer-Verlag, second edition, 1993.

[16] Isaacs R.,. Leslie I., "Support for Resource-Assured and Dynamic Virtual Private Networks" IEEE Jrl. on Sel. Areas in Communication Vol. n. 2001.

[17] Tennenhouse D.L., Wetherall D.J., "Towards an active networks architecture", Proceedings, Multimedia Computing and Networking, San Jose CA, 1996.

[18] Touch J., Hotz S.,"X-bone: a System for Automatic Network Overlay Deployment" Third Global Internet Mini Conference in conjunction with Globecom´98, Nov. 1998. http://www.isi.edu/x-bone.

[19] Walter V.,. "Automatic classification of remote sensing data for GIS database revision" IAPRS, Vol. 32, 1998 Part 4, Stuttgart, Germany, pp. 641-648.

[20] Weiser M., "Some Computer Science problems in Ubiquitous Computing", Communication of ACM, July 1993, pag. 73-84.