AnArchitectureforthEffectiveSupportofAdaptive Context-AwareApplications

ChristosEfstratiou,KeithCheverst,NigelDaviesandAdria nFriday

DistributedMultimediaResearchGroup,Departmento fComputing,LancasterUniversity,
Bailrigg,Lancaster,LA14YRU.K.
e-mail: most@comp.lancs.ac.uk

Abstract. Mobile applications are required to operate in env ironments characterised by change. More specifically, the ava ilability of resources and services may change significantly during a typical period of system operation. As a consequence, adaptive mobile applications need to be capable of adapting to these changes to ensure they offer the best poss ible level of service to the user. Our experiences of developing and evaluating adaptive context-aware applications in mobile environments has led us to b elieve that existing architectures fail to provide the necessary support for such applications. In this paper, we discuss the shortcomings of existing appr oaches and present work on our own architecture that has been designed to meet the key requirements of context-aware adaptive applications.

1 Introduction

Mobile applications are required to operate in envi ronments that change. Specifically, the availability of resources and services may chan ge significantly and frequently during typical system operation [8,11]. As a conseq uence, mobile applications need to be capable of adapting to these changes to ensure t hey offer the best possible level of service to the user [11]. While early research focu sed on applications which adapted to changes in network characteristics, there is now increasing interest in applications that adapt to general environmental and contextual triggers such as changes in a system's physical location, e.g. the GUIDE system [2,3] which supplies users with information tailored to their current location.

Current adaptive mobile applications are built usin g one of two approaches: either the adaptation is performed by the system which und erpins the application (in an attempt to make transparent the effects of mobility) or, the application itself monitors and adapts to change. In some cases, these approach es are combined as, for example, in the MOST system [8] where the middleware platfor m adapts the operation of the network protocol in the face of changes in QoS and, additionally, reports these changes to the application to enable application le vel adaptation. However, in the general case, it has been demonstrated that maintai ning transparency in the face of mobility is not practical and that it is difficult for a system to adapt without support from the application.

Careful examination of current approaches to supporting adaptation reveals two important facts. Firstly, support for adaptation is often fragmented with a range of mechanisms being used to notify applications of changes in different environmental and contextual attributes [4]. Secondly, there is a lack of mechanisms that support coordination of adaptive behaviour across the whole system, according to user requirements. In this paper, we explore the requirements for, and our research into creating, a unified architecture. It can support multiple contextual attributes coupled with a user driven adaptation control mechanism. The benefits of such an approach are clearly illustrated using a set of real-world examples.

2 Drawbacks of Current Approaches

Mobile systems need to be capable of adapting to a wide range of attributes such as network bandwidth, location, power etc. In general, current mobile systems provide support for adaptive applications by notifying applications when certain 'interesting' changes in attributes occur, e.g. bandwidth falls below some specified minimum threshold. It is then the responsibility of the application to adapt in an appropriate way, e.g. by reducing its bandwidth requirements. However, this approach can be shown to lead to inefficient solutions because of the lack of support for enabling coordination between the adaptation policies of multiple applications that may co-exist on the same system. In the following scenarios, we illustrate the kind of problems that could occur as a result of relying upon a simplistic notification based approach and isolated, uncoordinated, application adaptation.

2.1 Scenarios

The Need for Coordinated Application Adaptation for Power Management. This scenario illustrates the need for coordination in order to achieve efficient power management on a mobile system. One existing approach for handling power management, i.e. the ACPI [1] model, is to enable the operating system to switch hardware resources into low power mode when not in use, e.g. spinning down the hard-disk. This approach requires that applications leave hardware resources in an idle state for sufficient periods of time to make the transition between idle and active states worthwhile. Although this approach is suitable when only one application is running on a mobile device, the approach can prove ineffective when multiple applications or system services are sharing hardware resources. In more detail, the lack of coordinated access to hardware resources can result in poor utilisation of the shared resource and therefore sub-optimum power management. For example, consider the case of multiple applications that implement an auto-save feature. In the absence of any coordination between applications each application may choose to checkpoint its state to the disk at an arbitrary time, without considering the state of the disk (i.e. spinning or sleeping). In contrast, if applications are able to coordinate their access to the hard-disk then access to the disk can be clustered, allowing longer periods of inactivity. This latter approach is clearly more power efficient than the situation in which usage of the hard-disk is completely arbitrary and uncoordinated.

The Problem of Conflicting Adaptation. In this scenario, we illustrate the potential problems that can occur in a system that utilises s eparate adaptation mechanisms for different attributes. We consider a hypothetical mo bile system which utilises two independent adaptation mechanisms, one for managing power and the other for managing network bandwidth. The two mechanisms can conflict with one another as the following example illustrates. If the system ne eds to reduce power consumption, the power management mechanism will request those a pplications that are utilising network bandwidth to postpone their usage of the ne twork device in order to place the network device in to sleep mode. As a consequence o f applications postponing their use of the network, the available network bandwidth increases. However, the network adaptation mechanism will detect this unused bandwi dth and notify applications to utilise the spare bandwidth. In this way, there reque st to utilise available bandwidth is in direct conflict with the request to postpone netwo r kusage.

This example highlights the problem of relying on i ndependent and uncoordinated adaptation mechanisms. Clearly, coordination or har monisation is required in order to detect and avoid potentially conflicting adaptation mechanisms. In the example presented, the instruction given to applications to utilise more bandwidth should have been withheld if conserving power was the system's primary goal.

Utilising Alternative Location Sensing Mechanisms. This scenario considers the case of supporting multiple services for providing similar contextual information. In this case, we consider a location aware system that is capable of sensing its current location through two different mechanisms: a local GPS device and using beaconing in a cell-based wireless network. Using the latter mechanism, the system can identify the current cell in which it operates and thus spec ify its location. Both mechanisms deal with the same problem but they both have diffe rent characteristics. The GPS mechanism is typically the more accurate (accuracy in the region of 5m) but does require extra power to operate. Alternatively, the network-based solution is generally less accurate (depending on the size of a cell, e.g . approximately 200m for WaveLAN). However, the fact that the network device is already in use by the system for communications means that the addition power co nsumption required for identifying the location of the base-station would be minimal.

It follows that the mobile system would select to u se the GPS based solution if accurate location information was required and conc ern over additional power usage was not an important issue. Alternatively, if the l ife time of the system's batteries (and therefore operation) was more important than achiev ing greater location accuracy then the network location mechanism would be the mo reappropriate mechanism. The adaptive strategy that would be most appropriate de pends on both the user's requirements and the context of other attributes, s uch as power. In order for the system to make such decisions there is a basic requ irement for system-wide adaptation policies. Without such policies, coordin ated adaptation on the use of alternative context retrieval mechanisms is difficu lt because each application relies on a single mechanism without being able to identify t heimplications of its operation on other system resources and, consequently, other app lications.

3 Analysis

Multiple Attributes. The previous scenarios illustrated a number of potential problems with current approaches to developing adaptive context-aware mobile systems. In this section, we generalise on these findings to present a critique of existing mobile systems and their suitability for supporting adaptive applications.

Based on the ideas of ubiquitous computing [20] future mobile systems should be able to discover changes in both the user and system environment and adapt to these changes. Current context-aware applications handle context in an improvised fashion. Application developers usually bundle the application with specific mechanisms for accessing context. However, this approach does not allow coordinated adaptation on context changes leading to the problems presented. Dey [5] has addressed this problem and suggested a general platform to support context-aware application. Our belief is that though a general platform for supporting context-aware application is necessary, this platform should also be capable of addressing the problems of coordinated adaptation.

The situation is complicated still further by the fact that the adaptive behaviour triggered by one attribute can cause side-effects so nother attributes. These side-effects could, in-turn, trigger adaptation requests to other applications that result in conflicting actions (as illustrated in the conflicting adaptation scenario in section 2.1.2). Moreover, current research [4,6,7,12,13] has identified the need to provide adaptation solutions based on the combination of different attributes.

Existing architectures do not provide the necessary support to enable programmers to construct applications, which can adapt to multiple attributes and identify and cope with conflicts in adaptation strategies.

Adaptation Mechanism. Current mobile systems supporting adaptive applications tend to rely heavily on integrating QoS feedback and adaptation with network bindings. Examining the architecture of such systems allows us to identify a framework for analysing the architectural model of existing adaptive systems. The framework comprises two layers, the upper application layer and the lower representing the adaptation support platform. Between these two layers we can identify four distinct flows of control and information (see figure 1).

Flow A. Represents the requirements set by the application concerning the resources or attributes supported by the underlying infrastructure. For example, in the case of network adaptation this flow could represent the application's network QoS requirements.

Flow B. Represents the ability of the application to control the functionality of the underlying infrastructure. In the case of accessing a GPS device this could represent, for example, the control of the device by the application.

Flow C. Represents an information flow from the platform to the application. This could be used, for example, as a notification mechanism to inform the application when certain requirements cannot be met. Such notification could then trigger the application to adapt.

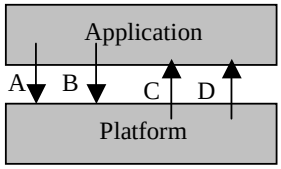

Fig.1. Directedflowsbetweenapplicationsandplatform

Flow D. Represents the ability of the underlying platform to actually control the operationoftheapplication.Morespecifically,th isflowrepresentsanexplicitrequest from the system for the application to perform a sp ecific adaptive behaviour. For example, the application might be requested to redu ce its demand for network bandwidthordiskusage.

Consideration of this framework enables a classific ation of current systems according to the types of flows supported. For exam ple, network based adaptive systems such as BAYOU [18], Odyssey [16], MOST [8] and Rover [10] support flowsAandC.

Contextaware applications likeGUIDE[2,3],Stick- eNotes[17]andCyberguide [14] are based on flows B and C. In more detail, fo r these applications, flow B represents the access to the various context-sensor s while flow C represents the informationflowingfromthesensorstotheapplica tion.

According to our knowledge no platform supporting c ontext-aware adaptation provides a flow of control from the platform to the application. Indeed, although examples of systems providing this type of flow can be found in the distributed systems community, e.g. ISIS-META [15], it should b e noted that these systems consideronlynetworktriggeredadaptation.

4 ArchitecturalRequirements

The previous sections have described the limitation s of current approaches for supporting adaptive mobile applications. In particu lar, these approaches do not provideappropriatesupporttoenableapplications toadapttomultipleattributesinan efficientandcoordinatedway.Thissectionconside rsasetofrequirementsthatcould be used to develop an appropriate architecture for supporting adaptive mobile applications.

4.1 SupportingaCommonSpaceforanExtensibleSe tofAttributes

The first key requirement of the architecture is to provide a common space for handling the adaptation attributes used by the syst em. It is important that new attributescanbeintroducedintothesystemasand whentheybecomeimportant,e.g. thecostofspecificservicesformobileusersori nformationabouthumanphysiology for wearable computers. The fact that new contextua l attributes for triggering adaptationcanariseimpliesthat:

- thesetofattributesthatcantriggeradaptationneed stobeextensible,
- thecharacteristicsofalltheseattributesmayvar y.

4.2ApplicationControlandCoordination

A second architectural requirement is the need to s upport the control of adaptative behaviouracrossallcomponentsinvolvedintheint eraction.Asdescribedearlier,one of the main limitations of current approaches is th at applications themselves are responsible for triggering an adaptive mechanism wh en the underling infrastructure notifies them about any changes. In order to suppor t flexible and coordinated adaptation there is a requirement for the triggerin g of adaptation on a system-wide level. Given this approach, the decision about when and how an application should adapt is pushed into an external entity, with cross -application knowledge, while the adaptivebehaviourisstillapartoftheapplicati on'scharacteristics.

4.3SupportforSystemWideAdaptationPolicies

A further requirement is to support the notion of s ystem-wide adaptation policies. Morespecifically,suchpoliciesshouldenableamo bilesystemtooperatedifferently giventhecurrentcontextandtherequirementsoft heuser.

The specification of adaptation policies should be goal-oriented. Two kinds of goalscanbeidentified:

1. effects on resources. The policy specifies a spe cific aim for the use a specific resource. Example policies include reducing the req uired network bandwidth and maximisingthedurationofoperationofthesystem.
2. effects on applications. The policy specifies th e mode of operation for specific applications. Example policies include defining pri orities on applications which determine the order in which they are allocated res ources and maximising the duration of operation of the system while having a specific application operating withfullfunctionality.

5. Architecture

5.1StructureofthePlatform

We propose that future mobile adaptive applications should adopt an architecture in which mechanisms and policies are decoupled and, fu rthermore, mechanisms can be exposed and externalised in order to enable control by independent entities. Our architecturehasbeendesignedtoaddressthesereq uirements.

Figure 2 shows the relationship between the main co mponents of the architecture of our proposed platform. The basic functionality o f the architecture is two-fold, namely: thediscoveryandcontrolofservicesoffer ingcontextualinformationandthe coordination of adaptive behaviour of the system ba sed on changes in context. More specifically, the platform discovers available cont ext information in the system's environment and manipulates the contextual informat ion in the *context database* .

Context aware applications expose their adaptive mechanism to the platform by registering with the *application database*. The *adaptation control* driven by *adaptation policies* (as specified by the user) coordinates the coexisting applications according to changes in context.

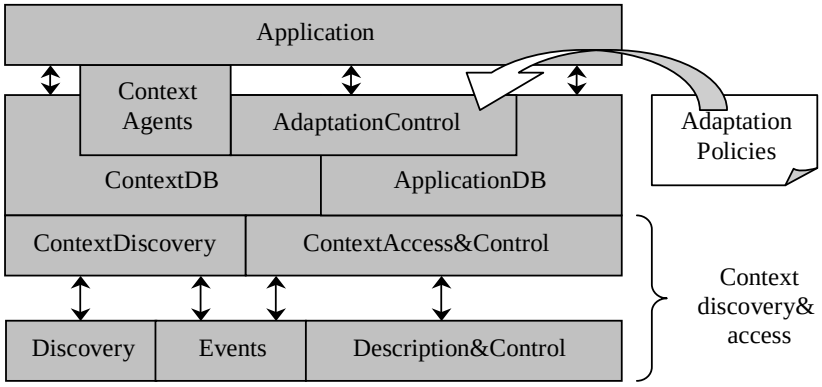

Fig.2. The overall architecture of the system

All communications between the main components of the architecture are performed using HTTP as the transport protocol and XML to represent the format of the messages. The reason for this decision was based on the fact that these technologies support the lightweight integration of distributed components. Other alternatives, such as CORBA and RMI, are too heavyweight, and would require additional infrastructure, i.e. an ORB or RMI daemon respectively.

5.2 Context Discovery and Access

This part of our architecture is responsible for locating services that provide contextual information. These services can either be sensors embedded within the system (e.g. a body temperature sensor) or services in the surrounding environment (e.g. active devices in the user's environment). In a mobile environment the availability of context information can change rapidly. Thus, it is important to be able to discover context services when they are available and receive notification when they are not. For this reason, a considerable part of the architecture is based on the UPnP architecture [19]. In more detail, the mechanism for discovering available context services is based on services advertising themselves (using multicasting) to any interested context-aware applications.

Once a service is discovered the platform retrieves the XML description of the device, an abbreviated example of which is shown figure 3. The XML description provides the access points for sending control messages to the object and subscribing for event notifications. In addition, it defines the types of messages that can be sent to the service and the types of information that the service can offer. Given this information, the platform is capable of using the service and receiving notification events when the state of the service changes.

```
<service>
   <category> Location </category>
   <type> GPS </type>
   <action>
      <name> getXCoord </name>
   </action>
</service>
```

Fig.3. SampleoftheXMLdefinitionofaservice

5.3ContextDatabase

Thiscomponentservesasaregistryforallthosec ontextservicescurrentlyavailable. An important part of the context database is a clas sification of the context services into context types. More specifically, each context service has to specify the type of context that it supports, e.g both the GPS and the network based locator provide location information. The specification of context type is achieved using an XML template that defines the kind of information this type offers. Using this approach, an application can retrieve the specific contextual da ta in a way that is decoupled from the service used for acquiring the data.

This method of hiding the actual mechanism for retr ieving contextual information, allows the platform to coordinate the access to con text for different application. Moreover it offers our platform with the potential to switch between different services of the same type depending upon the predefined adap tation policies that have been specified.

5.4ApplicationDatabase

The application database serves as a repository for the adaptation mechanisms of all applications running on the system. The application developer is responsible for actually implementing the adaptive behaviour of his /her application. In addition, the application developer is also responsible for expos ing this behaviour to the platform by ensuring that the application registers all of i ts adaptation mechanisms with the application database.
The description of the adaptation mechanisms should specify the type of context that can trigger this mechanism. This information is use d by the adaptation control in order to coordinate the triggering of the applicati ons based on changes in context. In figure 4, we illustrate an example of the informati on that may be provided by an application in XML, concerning its adaptive functio nality.

5.5ContextAgents

A context agent is a piece of code that can be plug ged into the platform in order to perform the application specific manipulation of co ntextual information.

```
<application>
   <name> WebBrowser </name>
   <adaptationMode>
      <name> lowBand </name>
      <trigger>
         <context> availableBand </context>
         <condition> lessThan-9600 </condition>
      </trigger>
   </adaptationMode>
</application>
```

Fig. 4. Sample of the XML definition of application's operation modes. The XML based description provides the different operational modes of the application, coupled with the contextualtriggerthatwouldmaketheapplication switchintothatmode.

A common case of context manipulation is the combination of primitive context information for constructing a complex type of contextual data. For example, a contextagentpluggedintotheplatformcancombine locationdataandcurrenttimein ordertoprovidelocation-and-timetrackinginformationsimilartothedatausedbythe Stick-eNotesystem[17].

To present even more clearly the operation of a context agent we will present an example based on the GUIDE system [2, 3]. The GUIDE system is a mobile electronic context-aware tourist guide. As part of its functionality it provides informationabouttouristattractionsinHTMLformat,triggeredbythelocationofthe user. In order to introduce the GUIDE system into our platform we need to split the application into two parts: an ordinary web-browser and a location-triggered HTTP proxy. The HTTP proxy operates as a context agent which is plugged into the platform and has direct access to the location information provided by the platform. Triggered by changes in location, the agent can request the appropriate HTML data fromthecontentserver.

Thekeymotivationbehindintroducingthenotionof contextagentsistoenablethe developer to distinguish the functionality of the application from the acquisition and manipulation of context data. Importantly, this allows context agents to be used for integrating non-context-aware applications (like an ordinary web browser) into a context-awaresystem.

5.6AdaptationControl

This module is responsible for monitoring the status of contextual triggers and making decisions about the behaviour of the platform and the applications. The decision taking procedure is based on a set of adaptation policies specified by the user. These adaptation policies are specified by defining priorities both among the applications running on the system and among the resources of the system. This prioritisationrepresentstheimportanceofthesee ntitiesaccordingtotheuserneeds.

Theadaptationcontrolisfurtherdividedintotwo sub-modules: *internaladaptation* and *externaladaptationas* explainedbelow .

Internal adaptation. This module coordinates the context monitors that are required for all applications running on the system and coordinates potential adaptation within the platform itself. The adaptation actions that can be performed on context acquisition are tightly coupled with the context classification that has been described earlier (in section 5.3). Context services are clustered into context types according to the type of information they provide. For example, a GPS device and a network based location mechanism would be members of the same type of context, i.e. location. Both these mechanisms can provide similar types of information but have different specifications and different requirements. When the system gets into a state whereby one of the two mechanisms is favoured (according to the adaptation policies specified by the user) then the platform will switch to the mechanism that is preferred (as illustrated in figure 5).

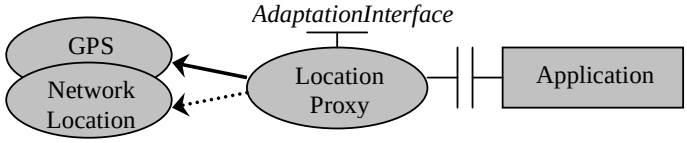

Fig.5. Internal adaptation: switching between different location mechanisms.

In the example discussed, all applications that require location information do so via the location proxy object. Thus, the actual mechanism for retrieving contextual information is hidden from the application. This transparent access to context enables the platform to switch between mechanisms without affecting the applications involved. To illustrate this, consider the example described in section 2.1.4 and a scenario in which it is crucial to reduce the aggregate level of power consumption. In such a scenario, the platform could reduce power consumption by switching off the GPS device, and using the network locator in order to retrieve location information.

External adaptation. This module is responsible for coordinating the adaptive behaviour of the applications running on the system. Its operation is driven by both a set of adaptation policies defined by the user and the XML description of each application specifying its various operation modes (as described in section 5.4). Recall that the XML description of each operation mode is marked according to its effect on resource utilisation (e.g. power, network) and the use of context services. The adaptation control module can use this information in order to decide which operation mode has to be triggered under each potential set of circumstances. In more detail, when resources (power, network, etc.) become unavailable the adaptation control picks the adaptation mechanism with the lowest resource requirements.

In order to clarify this approach, consider the following example (which is illustrated in figure 6). Two adaptive applications run on a mobile device: a web browser and a video player. The adaptation policies specified by the user define priorities between the applications and the resources (such that the lower the number the greater the priority). The adaptation control module is aware of the adaptive modes that these applications can support by accessing the application database. It also knows the status of all the contextual data that is available on the system. When

anyofthecontextualtriggersreachavaluethattrig gersareactionbythesystem,the adaptationcontrolhastodecidewhichadaptationm echanismshouldbeinvoked.In the example presented, both power and available ban dwidth are low. However, the adaptation control would choose to overcome the pow er problem, because the user hasspecifiedthatpoweristhemostimportantreso urce.Notethatenablingtheusertospecify priorities has in this case enabled the pla tform to overcome an area of potentialconflict(ashighlightedinsection2.1.2).

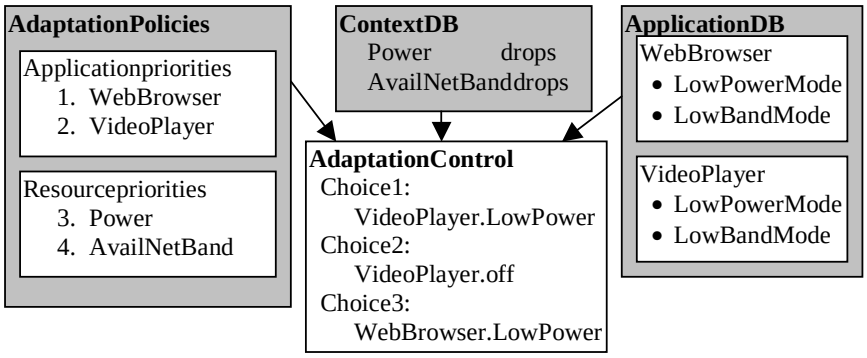

Fig.6. Externaladaptation:makingadaptationchoicesacc ordingtoadaptationpolicies

In order to decide which application to invoke, the adaptation control component checks the prioritisation of the applications and p icks the application that is less important to the user (i.e. the video player). The appropriate adaptation mode would then be triggered causing the video player to switc h to low power mode. If the reductioninpowerresultingfromthisactionisin sufficientthentheadaptationcontrol component would proceed with the next course of act ion, i.e. turning off the video playerandtriggeringthewebbrowsertoenterlow powermode.

6.Conclusions

In this paper, we have argued that existing archite ctural approaches for supporting adaptive mobile applications have a number of short comings. Furthermore, analysis of these shortcomings has led to the identification of a set of architectural requirements. Namely: support for a common contextu al space, mechanisms to support co-ordinated adaptations between multiple a daptive applications and support for user defined adaptation policies. We have also described the architecture of our platform, which has been designed to meet these req uirements and which enables mobile systems to extend their awareness of all rel evant contexts that might affect overallsystemadaptationpolicies.Fundamentalto ourapproachistheideaofhaving system-wide decision making policies that consider the most efficient adaptation outcome from a number of possible adaptations. Thi s is achieved by requiring the applications to provide information about themselve s, their adaptation mechanisms, andthecontextualtriggersthatcanaffecttheirb ehaviour.

References

1. Advanced Configuration and Power Interface Specifi cation, Revision 1.0, Intel/Microsoft/Toshiba(1999).
2. Cheverst,K.,N.Davies,K.Mitchell,A.Friday. :ExperiencesofDevelopingandDeploying a Context-Aware Tourist Guide: The GUIDE Project. I n: Proc. of MOBICOM'2000, Boston,ACMPress(2000)
3. Davies N., K. Cheverst, K. Mitchell, A. Friday.: Caches in the Air: Disseminating InformationintheGuideSystem.In:Proc.ofthe2 ndIEEEWorkshoponMobileComputing SystemsandApplications(WMCSA'99)(1999)
4. Davies N., A. Friday, S. Wade, G. Blair.: L 2imbo: A Distributed Systems Platform for MobileComputing.In:ACMMobileNetworksandAppli cations(MONET),SpecialIssue onProtocolsandSoftwareParadigmsofMobileNetwo rks,3(2)(1998)143-156
5. DeyA.,AbowdG.,SalberD.:AContext-BasedInf rastructureforSmartEnvironments.In: Proc.ofthe2000ConferenceonHumanFactorsinCo mputingSystems(2000)
6. ElisC.:TheCaseforHigher-LevelPowerManagem ent.In:Proc.ofHotOS(1999)
7. Flinn J., M. Satyanarayanan.: PowerScope: A Tool for Profiling the Energy Usage of Mobile Applications. In: Proc. of the Second IEEE W orkshop on Mobile Computing SystemsandApplications(1999)
8. FridayA.,N.Davies,G.Blair,K.Cheverst.:De velopingAdaptiveApplications:TheMOST Experience.In:JournalofIntegratedComputer-Aide dEngineering,6(2)143-157
9. Goland, Y., Cai T., Leach P., Gu Y., Albright S. : Simple Service Discovery Protocol, Version 1.03. IETF Internet-Draft. http://www.ietf. org/internet-drafts/draft-cai-ssdp-v1-03.txt
10. Joseph A., J. Tauber, F. Kaashoek.: Mobile Comp uting with the Rover Toolkit. In: IEEE TransactionsonComputers:SpecialissueonMobile Computing,43(3),(1997)
11. Katz R.: Adaptation and Mobility in Wirless Inf ormation Systems. In: IEEE Personal Communications,1(1)(1994)6-17
12. Kravets R., P. Krishnan.: Application-Driven Po wer Management for Mobile Communication. In: Fourth ACM International Confere nce on Mobile Computing and Networking(MOBICOM'98)(1998)
13. KunzT.,J.Black.:AnArchitectureforAdaptiv eMobileApplications.In:Proc.ofthe11th InternationalConferenceonWirelessCommunications (Wireless'99)(1999)
14. Long, S., R. Kooper, G.D. Abowd, C.G. Atkeson.: Rapid Prototyping of Mobile Context-Aware Applications: The Cyberguide Case Stu dy. In: Proc. of the 2 nd ACM InternationalConferenceonMobileComputing(MOBIC OM)(1996)
15. Marzullo K., R. Cooper, M. Wood, K. Birman.: To ols for Distributed Application Management.In:IEEEComputer,24(8)(1991)42-51
16. Noble B., M. Satyanarayanan, D. Narayanan, J. E . Tilton, J. Flinn, K. Walker.: Agile Application-Aware Adaptation for Mobility. In: Proc of the 16 th ACM Symposium on OperatingSystemPrinciples(1997)
17. PascoeJ.:TheStick-eNoteArchitecture:Exten dingtheInterfaceBeyondtheUser.In:Proc. oftheInternationalConferenceonIntelligentUser Interfaces(1997)
18. TerryD.B.,M.Theimer,K.PetersenA.J.Deme rs.:ManagingUpdateConflictsinBayou, aWeaklyConnectedReplicatedStorageSystem.In:P roc of the 15 th ACM Symposium on OperatingSystemPrinciples(1995)
19. Universal Plug and Play Device Architecture, Ve rsion 0.91, Microsoft Corporation, March 2000.http://www.upnp.org/download/UPnP_Device_Arch itecture.mht
20. Weiser M.: Some Computer Science Issues in Ubiq uitous Computing. In: Communications oftheACM,6(7)(1993)75-84