QoS And Context Awareness For Mobile Computing

Dan Chalmers, Morris Sloman
Department of Computing, Imperial College of Science, Technology & Medicine,
London SW7 2BZ, U.K.
E-mail: {dc,mss}@doc.ic.ac.uk
http://www-dse.doc.ic.ac.uk/

Abstract

Systems must provide for mobile applications to be aware of the context in which they are being used. This is necessary to permit adaptation to heterogeneity of hosts and networks and variations in the user's environment, and so enable applications to be effective in a wide range of devices and situations. While we acknowledge that deterministic QoS management may be unrealistic in these circumstances, we also believe that management of resources provides the most reliable basis for managing the problems giving rise to this non-determinism, especially where over-provisioning to ensure QoS is not possible.

We provide an overview of a system model for context and QoS-aware application support, and the work being undertaken to address the issues this model raises. We then describe an example application to illustrate the need for the integration of the elements in the model.

1 Introduction

The integration of computing and wireless communications will facilitate future mobile multimedia applications in medicine, navigation, entertainment, tourism, information and news distribution, and personal communications. A mobile computing device could range from a notebook size portable computer to a web-enabled telephone or a specialised monitoring device, with considerable variations in processing power and input-output capabilities. Network connections could also range from high-bandwidth local area networks to low and variable bandwidth wireless. Hand-held devices, and devices used for extended periods or in remote locations often exhibit extremes of limitation and variation, and so are of particular interest.

Users often want access to the same information and applications on fixed work-stations in the office, at home and on mobile devices. For instance, a surveyor may wish to check and complete field work in the office. Many web-based applications down-load code which must run on a variety of devices with different screen sizes and processing capabilities. In addition they should adapt to the users' environment. Mobile applications thus need to be aware of the user's *context* which includes:

- Location – accurate to a few cms upwards, depending on the application, as the information made available to a user could be location dependent.
- Relative location – who the user is with might affect security or modes of interaction; proximity to network resources such as printers, databases, or storage servers.
- Device characteristics – processing power; screen size, resolution and colour support; sound output capability; input devices etc.
- Environment – lighting and noise levels, the social situation, which might affect modes of interaction.
- User's activity – activities such as driving a car or walking and any disabilities of the user which affect their modes of interaction.

Wireless communication results in variable Quality of Service (QoS) during movement. Communication QoS characteristics such as bandwidth, delay, jitter, and error rates, and how variable they are could be considered part of the context. We will treat QoS separately in this paper, but will describe the use of context information in QoS specifications and resource management.

The key issue is that mobile applications have to be aware of and adapt to *variations* in the user's context and in the available QoS, as discussed in [1]. Perceived quality will generally be dependant on the context [2]. In general, over-provisioning or a highly controlled infrastructure may not be practical or desirable as a means to enable guaranteed QoS. We believe that adaptive applications running in a context and resource aware environment can provide a powerful degree of autonomous context and QoS management.

The rest of this paper is structured as follows: Section 2 provides an overview of the context and QoS aware application infrastructure we propose. Section 3 illustrates how this can be applied in the provision of location, device and QoS aware Web applications. We describe some of our experiences in implementing this. We compare our approach with others' in section 4 and conclude in section 5.

2 Context Aware QoS Management Architecture

Our basic framework for enabling context aware QoS management and adaptation is shown in figure 1, and described in more detail in [3]. The application has QoS and context management components which hold descriptions of the user requirements and drive an adaptation engine, which interfaces to external data sources.

The context manager is provided with information from a system context manager, which handles generic context descriptions and gathers context data from various sources. Context includes, but is not limited to, descriptions of display; user input devices; processing power; network bandwidth, delay, jitter, cost etc.; location, predicted future location and location relative to other located objects; social situation and user activity. The application context manager will provide the QoS manager with a context dependent specification.

The QoS manager negotiates with the system resource manager for reservation of resources with specified QoS characteristics. The resource manager is responsible for low-level activities such as maintaining throughput, delay, or jitter at agreed levels, while the application is responsible for specifying requirements and performing large scale adaptations affecting the presentation of data.

We support two different approaches to adapting to variations in context:

- Information sources hold (or dynamically create) variants of the information required by the application e.g. different sizes and resolutions on an image. Meta-data is needed to describe the variants in terms of resolution, colour, file size etc. Adaptation is by selecting a information suited to the current context and QoS capabilities.
- System components at source, or within the network, manipulate data by dynamically filtering, compressing or transforming the information, e.g. converting voice to text, as it flows through the network. The changes applied are based on context and QoS information provided by the system and the application. (This is not illustrated in the architecture diagram).

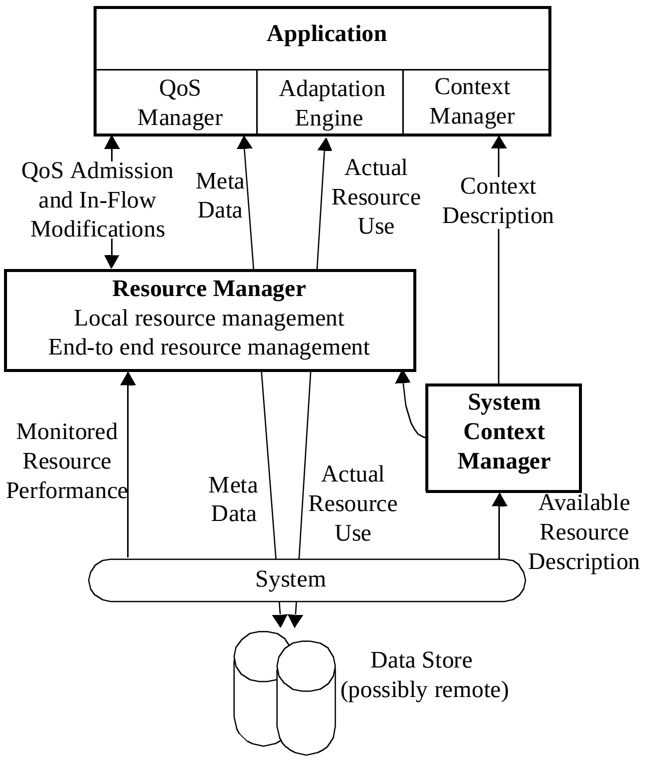
Application
QoS Manager
Adaptation Engine
Context Manager
QoS Admission and In-Flow Modifications
Meta Data
Actual Resource Use
Context Description
Resource Manager
Local resource management
End-to end resource management
System Context Manager
Monitored Resource Performance
Meta Data
Actual Resource Use
Available Resource Description
System
Data Store (possibly remote)

Figure 1: QoS Management Architecture

We shall use a map navigation application as an example. The map indicates current location as well as various types of information about the locality – street names, traffic restrictions, shop descriptions, topological data, and may provide hyper-links to other information of interest such as historical data. The map data should be generally available, via the internet and local wireless LANs, and be used from a variety of systems, e.g. PDAs, information kiosks, lap-top and desktop PCs. The update of the map requires transactional semantics by areas and feature types.

2.1 Context Aware QoS Specification

As discussed in [2,4] perceived quality will generally be dependant on the context. We describe our support for context based specification of QoS requirements, and system level resource management. Application adaptation due to context is discussed in section 2.4.

The provision of generic descriptions of what user's preferences are in various contexts, using various applications is a large task involving many specialisations. We do not propose to solve this problem for the general case. However, by limiting our initial set of users, contexts and QoS parameters to be considered we believe we can formulate manageable specifications. Our initial interest is in surveyors, including those both updating and adding new types of data to existing maps. This scenario can involve loading data in order to meet a schedule; sharing data; cooperation and transfer of data between mobile users; and mobile hosts updating shared databases. The scenario can be extended so that many users may be contributing data to a highly dynamic map. For instance cars by being on a road might indicate traffic on the shared map.

The user requirements for QoS can be specified as authorisation policies which define the services or resources a user is permitted to access based on constraints in terms of the current context. In addition, obligation policies define event triggered actions (which could be a specific adaptation) to be performed, also based on context constraints [5]. We have concentrated on application adaptation to differences in network characteristics, the device being used, the user's current activity, and location.

The user perceived QoS specification must map onto system-level QoS specifications. A formal notation can aid this by describing a translation between parameters at various levels, and a standardised formatting of the constraints to be applied. [6,7,8] provide examples of work in this area, which we intend to investigate further.

An interesting model of environment monitoring designed to manage changes in devices or device characteristics due to mobility is described in [9]. A model such as this which provides monitoring of system events and selects significant events to be notified to the system would be a crucial part of an effective context management system for mobile hosts.

2.2 Resource Management

The system context manager provides information to the resource manager on expected resource characteristics and preferences. This enables the resource manager to provide reservations against more accurate resource models. A change in resource model will cause a corresponding change in the adaptations performed by the application in order for requests to gain admission, e.g. when moving from wired to wireless networks, or areas of differing wireless network coverage. Context aware resource management also allows for intelligent selection of external resources at a system level, for instance mapping connections to mirrored web site to the local site when moving between countries.

An interesting model of environment monitoring designed to manage changes in devices or device characteristics due to mobility is described in [9]. A model such as this which provides monitoring of system events and selects significant events to be notified to the system would be a crucial part of an effective context management system for mobile hosts.

Resource reservation is required for QoS management of mobile applications [10,11]. Firstly, to make the most efficient use of the resources which the system expects to have, while attempting to limit the application's expectations. Secondly, to provide a starting point for in-flow adaptation in the event of resource capacity deviating significantly from the system's expectations.

From experiments with a web browsing application [3] we have found that advance reservations of bandwidth are needed to cater for typical protocol set-up delays to web servers. We have used a simple time sliced model, as in [12]. The network connection is modelled in terms of throughput (capacity per unit time). A connection request describes a rate to be met for a period (this may be derived from a total requirement and a deadline). The requirement is compared to availability for each time period. A shortfall in availability in one period may be compensated for by reserving further resources in other periods. Concurrent requests then share the resource by each reserving a portion of the total resource for each period in time. Some management may be required to enforce this, or to update the reservations due to usage deviating from the reservations. The reserved bandwidth cannot be reserved by another application, unless the application holding the reservation is willing or can be forced to adjust it's reservation during the data flow. The reservation should be timed to start at the point when the flow of data is expected to start, so that the reservation more accurately models the real requirement. When the reservation cannot be met, the admission function returns a failure which should be passed to the application so that it can adapt the request to use fewer resources, by modifying it's deadlines or else cancel it.

Reservations should be made on an end-to-end basis, in order to guarantee resources. This has been discussed in some depth in the QoS literature, and the resource manager should undertake this, or the use of DiffServ [13] or some other end-to-end mechanisms, where possible. Where end-to-end reservations cannot be achieved we believe that the use of local reservations based on previous resource characteristics is more effective than a blind best-effort system, and avoids the unnecessary adaptation characteristic of a purely reactive adaptation model. Local resource reservation is especially useful in mobile systems where the local network interface is often the limiting factor in the network connection, and where wireless connections can render end-to-end guarantees meaningless.

Configuarable protocol stacks [14] permit dynamic protocol configuration between end points, tailored to the context or required QoS. For example, dynamically inserting multiplexing, compression or encryption components. This enables security to be treated as a QoS parameter, with a possible trade-off between security and cost due to protocol overhead. Fine grained QoS management may be provided with the aid of dynamic protocol layers by the application of filters, buffers etc. Protocol layers enabling QoS management may be dynamically deployed at any point in the network, and not just at the end points of the communication.

2.3 Application Adaptation Due To Resource Variation

Applications incorporate an adaptation engine which addresses issues of perception of quality [6,15]. This could be generalised to implement adaptation functions for media types and/or application classes. An adaptation engine must aim to provide the best *achievable and acceptable* delivery of the data flow possible. By using media and/or application specific adaptation engines, the task of making the adaptations can be broken down, and an intelligent selection between alternatives where a trade-off or selection between adaptation strategies must be made. Note that this adaptation is in conjunction with system-level adaptation and QoS management functions, which are designed to manage more minor variations in resources, while the application is responsible for large grained QoS management. We now outline our resource, requirement and data description model, developed from [16], which enables adaptation by both the application and resource management functions.

The QoS specification describes an area of acceptable performance as an n-dimensional model for n QoS managed parameters, which in turn may be combined into an f-dimensional model of f flows. A flow might be separate media streams, or elements (individual files) in a web page request. Composite flows can combine multiple flows for particular applications. The parameters for a flow will generally be interrelated and have boundaries of acceptable performance. The possible parameter combinations for each media flow or application describe a separate bounded performance space. Each QoS parameter model may in turn be presented at various levels of QoS parameter abstraction. A parameter must be describable as some directed continuous or discrete value series.

An *achievable* performance region defines the capability of a resource in terms of a region of possible performance trade-off among the QoS managed parameters (see figure 2). An *acceptable* performance region defines the application requirements in terms of the same QoS parameters. The intersecting space then describes the "achievable and acceptable" region in the performance space, which the flows should occupy.

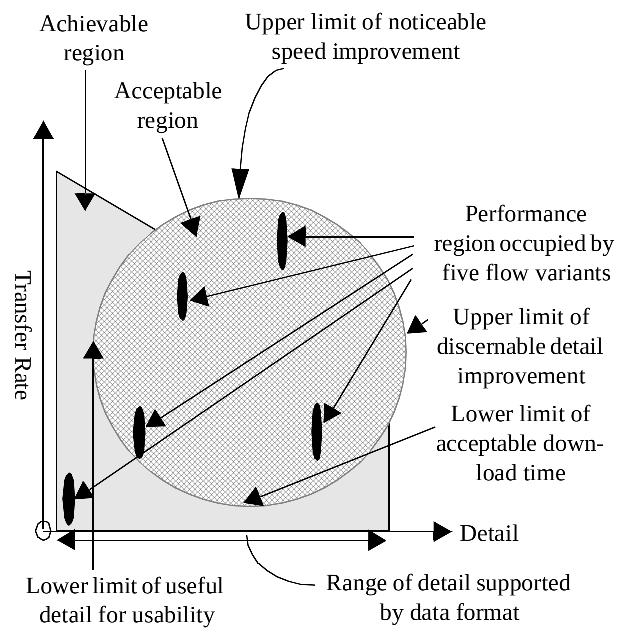
Achievable region
Upper limit of noticeable speed improvement
Acceptable region
Performance region occupied by five flow variants
Transfer Rate
Upper limit of discernable detail improvement
Lower limit of acceptable download time
Detail
Lower limit of useful detail for usability
Range of detail supported by data format

Figure 2: Parameter Spaces for a Flow

A *flow variant* defines a performance region for a specific level of compression, data resolution or MPEG enhancement layers etc. By trading parameters off against each other a flow's position in the performance space may be transformed. For instance, placing a greater demand on bandwidth, in order to restrict jitter. These variations may be achieved by some combination of pre-generated variants, and by filters etc. being applied on an *ad-hoc* basis to the flow. Note that a flow variant is a region and not a point, due to unavoidable variations in the infrastructure or inherent in the data stream, for instance between scenes in a video.

A data flow may be provided in several variants. For instance, a map segment may be available at various scales, showing various types of feature, and covering differing areas. These changes in the data will affect it's resource usage: by changing the size of the data, the display size and resolution required to show the map segment etc.

One of those flow variants which is within the intersection of the acceptable and achievable performance regions should be selected to implement the flow based on goals such as maximising QoS, maximising overall resource availability or minimising cost. [16,17,18] are examples of similar models.

Figure 2 shows an example of this for a single flow, e.g. a map segment image. The available resource (network bandwidth) has been mapped onto a region of plausible download time/detail combinations. Note that the media encoding also limits the achievable performance space. Five flow (image) variants are shown, each taking a space in the time/ detail space and corresponding to particular scale, segment size and level of detail for a map segment. One of the variants is outside the achievable region and one is outside the acceptable region leaving only 3 viable choices.

A change in resource availability would shift the edges of the achievable region. Changes in context might alter the boundaries of the acceptable region. If the current flow variant was still achievable and acceptable no change would be needed, else a different flow variant would be selected, or if none are in the new achievable/acceptable intersection region then a failure would be signalled.

The adaptation engine must at least address the following issues: Trade-off between parameters of a single element; trade-off between elements in a multi-media stream; selection of the most appropriate in-flow adaptations to be made; and trade-off between the responsiveness, or agility, in implementing these dynamic adaptations against the intrusiveness of frequent adaptation to the user. The treatment of adaptation in [9] is the basis for this model, although we expect our guarantee-less reservation model to provide extra information to the resource management model on the QoS requirements for concurrent flows, and so allow adaptation with a more complete understanding of system wide resource requirements. For instance, this would allow data which is not part of a real time stream to have it's reservation deferred to free resources for real time streams during periods of resource shortfall; also flows which are nearly complete need not have adaptation initiated where the overhead this generates would have an excessive impact in relation to the benefits gained.

Annotating flows with a relative importance rating can assist in selecting a flow for adaptation, although we found in [3] that the variation in resource requirements between flow variants will tend to dominate over a preference indication unless the gains due to each adaption stage for each flow are quite similar.

2.4 Application Adaptation Due to Context Change

In addition to changes in the presented media through QoS management's response to resource variation, the architecture supports adaptation due to context variation [2,16]. We consider that context is unlikely to vary rapidly in most cases, but the changes caused are likely to be more significant. We will focus on user interface differences across heterogeneous devices, location awareness, and social and activity based preferences.

User Interfaces

The key elements here are the *variation* in device capabilities and the variation in specification due to the context. A user may use a PDA, a lap-top and a high power work-station in the course of a day. These are likely to have displays ranging from small and monochrome to large, crisp true-colour. Similarly, input devices on kiosks, PDAs and lap-tops tend to be less precise, or harder to manipulate, due to space and ruggedness restrictions.

It is quite possible that the same binary application may be installed on different device types, for instance using Java, but it would seem clear that the user interfaces must differ. A map displayed on the work-station would provide full colour and much detail which would not display effectively on a more restrictive device such as a PDA, while an interface tailored to the PDA would be unnecessarily restrictive for use on the work-station.

Location Awareness

We have been investigating a location service which provides a general framework for determining geographical location using a variety of different sensors [19,20]. We discussed above the possibilities for selecting between copies of resources depending on location, which might be implemented as part of system QoS management. A location aware application modifies its behaviour based on it’s current physical location. We have experimented with the design of a web browser with an integrated location service which enables the presentation of data relevant to location, e.g. street maps, weather forecasts etc., see section 3.2. A survey tool might display editable map data in a similar way, providing the data for the current site.

Location aware services, and the techniques we have been experimenting with for location aware presentation of data are discussed further in section 3.

Social Situation And Activity

This is a somewhat general class of preferences and responses to sensed context. Simple preferences might include resolution or font changes when vibration is sensed, for instance when travelling; audio data may be undesirable or inaudible in public or noisy environments. Activity and interests can influence actions: a PDA might be used in the same street by both a pedestrian and a car driver. The pedestrian will not be very interested in one-way streets, while the driver is unlikely to require detailed information about the local shops or history. Selection of data from speed of movement is described in [21]. More advanced interaction and proximity based functions are described in [2,22].

3 Adaptation In A Map Application Due To Context And QoS

We shall now extend our map example to describe the use of context in application based adaptation and QoS management through the model described above. We shall discuss the adaptation of the displayed map by the application, then summarise some implementation work based on the ideas presented here.

3.1 QoS For Location Dependant Information

Activity and location information both need to be considered so a user moving rapidly, for instance in a car, will see a less detailed road map than the detailed street plan presented to a pedestrian. Additionally, a user is likely to want information relevant to where they are now, rather than where they have just been. This can be facilitated by the use of prediction for pre-fetching data [21,23]. To ensure timely delivery the map’s attributes, such as detail, scale, and segment size, may be varied.

We assume that the volume of data delivered to an application for a given level of detail, map scale etc. is described in meta-data, or can be estimated (especially for adjacent locations from one source). The network QoS management functions can provide an estimate of the achievable bandwidth. We now have a performance space including data transfer time (from network bandwidth and server response time) and map segment size, scale and level of detail. It is assumed that map scale limits the map details. The user’s requirements describe a space with dimensions of timeliness and relevant map detail, which may vary due to context - from the device display characteristics, speed of motion, and social situation etc. The intersection of these is the achievable and acceptable performance space.

The predicted “next location area” can be obtained from the location service, along with an estimate of when it will be required. This translates to one or more map segments and a deadline for loading. The total data to be loaded can be established, and compared with the expected available bandwidth in the period between now and the expected required display time by applying to the resource manager for these resources. If the down-load can be satisfied then resources can be reserved and the down-load initiated. If the resource request cannot be satisfied then various approaches to adapting may be possible:

- A map with less detail and hence requiring shorter load time can be provided. Scaling might be achieved by classifying types of feature on the map, and requesting only a subset e.g. contours or county boundaries might be omitted.
- The map scale may be adjusted to provide a map covering a greater area for the same segment display area. This approach can be particularly useful if the movement prediction potentially covers several segments.

- Different sized segments can be retrieved, while maintaining the detail and scale, to reduce the number or frequency of segment loads needed. Segment size must be traded off against overheads, and ensure the user's field of interest is covered.
- Segments may be omitted to give a discontinuous but still timely map display. Where multiple segments are required to cover the predicted location, the most likely locations may be down-loaded by preference to reduce load.

Scaling through feature sets is most flexibly applicable with vector encoded maps. Raster encoded data would require the storage of pre-generated variants with predefined feature sets, to be at all scalable. This type of scaling is in any ways similar to selection of compression ratios, e.g. as in [15].

The bandwidth experienced is the QoS characteristic we have focused on, as this has the greatest impact on non-continuous media, such as electronic map and web page display. This can simply be derived from the total size of the components of a page divided by the time taken to load the page, with some adjustment for connection and protocol set-up time. However, in a mobile environment the link quality may vary considerably, and this variation should also be measured and modelled. It is important that the next map segment should be ready in time. If segments overlap there is a strong case for displaying the next segment before absolutely necessary, rather than using a latest possible display model when calculating the data volume that may be down-loaded. The down-load time available should be compared against both the expected bandwidth, and an estimate of the worst likely bandwidth. The average case should provide for the use of any overlap in advance, and the worst case for just-in-time loading. An effective resource reservation and monitoring function should provide for these needs.

3.2 Overview Of Implementations

We have undertaken some implementation work exploring the ideas described above. Two parts of this are outlined below, although this is still a work in progress.

QoS And Context Adaptable Web Browser

A plug-in for the Netscape browser, was developed to enable adaptation of web pages by selecting among versions of page elements [3]. The meta-data described the relative importance (as judged by the author), dimensions (pixels) and size (bytes) of each element of the page [11]. The adaptation engine then restricted the set of possible element variations by only allowing those which could be completely displayed: thus adapting requested data to the screen size. Then a request for the sum of the sizes of the selected element variations within a user specified down-load time was placed with the resource manager. Down-load deadlines were variable with page importance rating and context, e.g. allowing data-rate to be defined as more important when using an expensive connection. The resource manager was implemented as a static function, which managed competition between browser instances. If the request could not be satisfied the size of the adjustment required to meet the deadline was returned. The adaptation engine then made a new selection of element variations to fit within the available resources. The selection was made by minimising the sum of (adaptation stages taken from the best version of the element * importance weighting of the element) across all elements. Once a satisfiable request had been made the HTML of the (invariant) base page was rewritten to reflect the element variations to be loaded, and then passed to the browser for rendering.

Note that the adaptation is performed within the application, and can thus be more aware of the user and application than a proxy or server side adaptation, such as mentioned in [7,24], which would have to either perform a very generic adaptation, or impose a complex protocol with the associated overhead. It is also distinct from the model in [11], where the adaptation is performed between the application and lower layers in the communications stack. Again, this allows a generic model, where the application is responsible for how it prefers to adapt. If adaptation engines are reused between applications a similar effect to stack based adaptation occurs, however.

Experiments with this system showed that as a basic model it was acceptable, but required the use of advance reservations to model the request set-up time between the client and web server. It also did not provide for in-flow modifications, but treated a sequence of page visits as a flow, with each page load being fixed once requested. Resource management would have been improved by implementing monitoring of actual bandwidth, but used assumed network bandwidths. This was due to API and time constraints.

Location Aware Web Page Display

We are working on a Java application which will select web pages according to sensed location. A menu selected service allows for different web pages to be loaded according to location, or for a URL to have location dependant arguments appended to it.

The location service fuses location descriptions from multiple sources to aid accuracy. We describe location as a rectangular area containing the actual location. This simplifies the use of location data sources with varying degrees of accuracy. A tree based user interface working from a text file defining locations, and a parser to read data from a GPS receiver on the serial port in a Windows environment have been implemented to date.

Location aware web "applications" were dynamically loaded into a standard application management framework with an application set-up data file argument. Location dependant web pages were selected from a list of URLs, or by generating URLs with attributes according to the current location and displayed on an HTML rendering pane [25].

The system described in section 3.1 is an extension of this application. We have found that the definition of applications is quite simple, and requires little code. We have implemented the presentation of national and regional weather forecasts from a list of URLs in a data file, and the presentation of a street map site [26] using functions to convert longitude and latitude to Ordnance Survey Grid coordinates [27]. The grid reference is used as arguments to the URL, which then causes a page including the map tile to be returned. Scaling of data and prediction functions to demonstrate QoS management within the framework have not yet been implemented.

4 Related Work

The Guide project at Lancaster [28] includes many similar aims to those presented above, in particular the support of data display dependant on location, and the ability to work within environmental restrictions. However there seem to be some engineering design decisions such as one way broadcast data delivery to restrict the complexity of applications which removes much of the need for QoS management, and a singe hardware platform removes the need for hardware context awareness.

[21] describes the provision of maps using similar techniques to those we have presented of scaling the map data in an appropriate manner for the user's activity, through a specifically tailored application. The effects of wireless links are countered using intelligent prefetching based on predicted location, as we do. The map scaling presented however assumes that a fixed volume of data can be downloaded, scaling the map detail and segment size according to the activity only, rather than due to the prevailing network characteristics. Their study also describes the utility of providing "islands" of high bandwidth wireless network coverage over a general low bandwidth provision. It would seem that while these can afford a significant benefit for the application described (from results of a simulation), the real life provision of these would require specific tailoring to patterns of user-application activity. Our approach does not assume any special network provisions, although it may make use of these where available.

Another system providing for scalable map data is described in [22]. The system they describe is very general, allowing contextual presentation of data through the use of SGML. We see the techniques we describe as being complementary to this kind of application. They describe problems with location sensing, which we believe could to some extent be alleviated by the use of a multi-sensor location service such as we describe, based on [19,20]. There is also no provision described for scaling data based on the degree of network connectivity. It seems to be assumed that either prefetching based on all possible contexts is possible, or that response speed is not crucial. The provision of data adaptation through context based resource management could extend the utility of the applications described.

Another approach to managing the different needs of wireless and wire-line network protocols, and data selection due to network conditions is the use of agents and/or proxies, as in [24]. They propose that wireless users connect to a mobile-connection host using a protocol suited to wireless hosts, which re-sends data over TCP/IP to wire-line hosts. A similar effect is achieved in our approach by using dynamically configuarable "plug-in" protocol stack layers. The host communicates with the network on whatever protocol is mutually supported and agreed to be the most suitable, rather than requiring connection to a statically defined protocol on a different socket.

Their approach also provides for the use of unmodified browsers and servers through the use of agents and proxies. They use pre-fetching, compression, filtering (omission of components of a web page) and caching to improve the transfer speed. The use of web caches is commonplace, and often improves data transfer, and can still be used in our approach. Our approach also allows the use of compression and omission, however we place the onus on the server to provide variants of the data, and the application to make selections. We believe that providing pre-defined data variants at the server is more efficient than frequent on-the-fly modification distributed across the network, in many cases. This approach to data provision also provides the content provider with a greater degree of control.

5 Conclusions

We have presented an infrastructure designed to enable application aware adaptation to both context variation and dynamic resource characteristics, typically found in mobile applications. The infrastructure also provides reservation, admission and filter based QoS management which insulates the application from fine-grained resource characteristic fluctuations. Some researchers contend that the uncertainties of mobile use render these QoS management techniques meaningless. However we believe that management of uncertainty is preferable to no management, and mobile computing will continue to exhibit uncertainties in resource availability.

The management of QoS has been integrated with context management to enable improved resource and requirements management. We consider that these qualities are necessary to enable effective application adaptation and resource management for heterogeneous systems and varying user requirements, particularly where mobile devices are being used. We believe this combination of context sensitivity in applications, system management and requirements management is unusual, and can improve application usability.

We have also presented an example application making use of these integrated facilities to illustrate the concepts.

The map display application is quite generic, and might be the basis of many more specialised applications. The example application described showed that QoS management effectiveness may be improved by context awareness. We also showed that the application of context sensitivity beyond user interface and location can usefully affect behaviour in our example application, although these continue to be the most immediately applicable context parameters. We are continuing to investigate the design of adaptive applications, QoS, and context management for mobile and distributed systems, in particular for cartography.

Acknowledgements

We gratefully acknowledge financial support from the EPSRC (Grants GR/L 06010, and GR/M 26428) and the contribution of our colleagues to the ideas described in this paper, in particular Ulf Leonhardt, Emil Lupu and Nat Pryce.

References

1. Chalmers, D., Sloman, M,: A Survey of Quality of Service in Mobile Computing Environments *IEEE Communications Surveys, 2nd Quarter 1999*
2. Schilit, B.N., Adams, N., Want, R.: Context-Aware Computing Applications *Proc. IEEE Workshop on Mobile Computing Systems and Applications,* Santa Cruz, 1994
3. Chalmers, D.: Quality of Service in Mobile Environments *MSc Dissertation* Dept. of Computing, Imperial College, London, 1998
4. Katz, R.H.: Adaptation and Mobility in Wireless Information Systems *IEEE Personal Communications* vol1 no1 p6-17, 1994
5. Sloman, M., Lupu, E.: Policy Specification For Programmable Networks *IWAN Workshop, Berlin, June 1999.*
6. Noble, B.D., Satyanarayanan, M., Narayanan, D., Tilton, J.E., Flinn, J., Walker, K.R.: Agile Application-Aware Adaptation for Mobility *Proc. 16th ACM Symposium on Operating Systems Principles* p276-87, Saint-Malo, France, October 1997
7. Frølund, S., Koistinen, J.: QML: A Language For Quality Of Service Specification *Hewlett-Packard Research Report HPL-98-10,* 1998
8. Loyall, J.P., Schantz, R.E., Zinky, J.A., Bakken, D.E.: Specifying And Measuring Quality Of Service In Distributed Object Systems *Proc. ISORC '98,* Kyoto, Japan 1998
9. Welling, G., Badrinath, B.R.: An Architecture for Exporting Environment Awareness to Mobile Computing Applications *IEEE Trans. Software Eng.* vol24 no5 p391-400, May 1998
10. Srivastava, M., Mishra, P.P.: On Quality of Service in Mobile Wireless Networks *Proc. IEEE 7th Intl. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'97)* p147-58, 1997
11. Bolliger, J., Gross, T.: A Framework-Based Approach to the Development of Network-Aware Applications *IEEE Trans. Software Eng.* vol24 no5 p376-389, May 1998
12. Ferrari, D., Gupta, A., Ventre, G.: Distributed Advance Reservation for Real-Time Connections *Multimedia Systems* vol5 p187-198, Springer-Verlag 1997
13. IETF Network Working Group: RFC 2475 An Architecture for Differentiated Services ed. Blake, S., et al., 1998
14. Pryce, N., Crane, S,: A Uniform Approach to Configuration and Communication in Distributed Systems *Proc. 3rd Intl. Conf. on Configuarable Distributed Systems* Annapolis, Maryland, USA, January 1996
15. McIlhagga, M., Light, A., Wakeman, I.: Towards a Design Methodology for Adaptive Applications *Proc. 4th Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networks, (MobiCom'98)* p133-144, Dallas, Texas, USA, 25-30th October 1998
16. Gecsei, J.: Adaptation in Distributed Multimedia Systems *IEEE Multimedia* vol4 no2 p58-66, April-June, 1997
17. Koistinen, J., Seetharaman, A.: Worth Based Multi Category Quality Of Service Negotiation In Distributed Object Infrastructures *Hewlett-Packard Research Report HPL-98-51,* 1998
18. Waddington, D.G., Hutchison, D.: A General Model for QoS Adaptation May 1998. *Proc. 6th Intl. Workshop on Quality of Service (IWQoS'98)* Napa, California, USA, May 1998
19. Leonhardt, U., Magee, J.: Towards A General Location Service For Mobile Environments *Proc. 3rd IEEE Workshop on Services in Distributed and Networked Environments* p43-50, Macau, June 1996
20. Leonhardt, U.: Supporting Location-Awareness in Open Distributed Systems. *Ph.D. Thesis* Dept. of Computing, Imperial College London, May 1998
21. Ye, T., Jacobsen, H.A., Katz, R.: Mobile Awareness In A Wide Area Wireless Network of Info Stations *Proc. 4th Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networks, (MobiCom'98)* p109-120, Dallas, Texas, USA, 25-30th October 1998
22. Brown, P.J., Bovey, J.D., Chen, X.: Context-Aware Applications: From the Laboratory to the Marketplace *IEEE Personal Communications* October 1997 p58-64
23. Levine, D.A., Akyildiz, I,F., Naghshineh, M.: A Resource Estimation and Call Admission Algorithm for Wireless Multimedia Networks Using The Shadow Cluster Concept *IEEE/ACM Trans. on Networking* vol5 no1 February 1997
24. Alanko, T., Kojo, M., Liljeberg, M., Raatikainen, K.: Mowgli: Improvements for Internet Applications Using Slow Wireless Links *Proc. 8th IEEE Intl. Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'97)* p1038-1042 1997
25. ICESoft AS.: ICE Browser Swing: http://www.icesoft.no/ICEBrowserSwing/index.html
26. The UK Street Map Web Page: http://www.streetmap.co.uk/
27. Ordnance Survey: The Ellipsoid and the Transverse Mercator Projection, Geodetic information paper No 1, version 2.2, February 1998
28. Davies, N., Mitchell, K., Cheverst, K., Blair, G.: Developing a Context Sensitive Tourist Guide *Proc. 1st Workshop on HCI for Mobile Devices* p64-68, Glasgow, UK, May 1998