INTEGRATING MIDDLEWARE PARADIGMS TO SUPPORT A MOBILE SPORT NEWS APPLICATION

Paul Grace, Gordon S. Blair
Computing Department, Lancaster University, Lancaster, LA2 4YR
E-mail: p.grace@lancaster.ac.uk, gordon@comp.lancs.ac.uk
Telephone: *+44 (0) 1524 593141*

1. INTRODUCTION

Mobile applications exist in environments that change (e.g. varying network bandwidth, network connectivity and device location) and may execute on end systems with limited resources such as battery power, CPU speed and memory. In order to provide the best level of service to the end user, the application must be aware of these factors and be able to adapt to deal with them [1]. There are a number of middleware paradigms for supporting mobile applications that include: providing resource discovery, agents and mobile code [3], component-based approaches and adaptation of the middleware platform [2]. All of these are beneficial in providing middleware features to support mobility, but inevitably this leads to the middleware being pieced together in an ad-hoc manner when required, with no integration. Furthermore, there is no overall vision of how they can work together, to provide a platform to support all classes of mobile applications. The authors believe that the way forward is to provide a *generalised component-based middleware solution* that allows the application to be built upon a single platform, preventing the application designer from having to integrate and interact with a set of middleware technologies.

This position paper presents a middleware platform that provides a platform and language independent programming environment that supports a single class of mobile application (we plan to extend its scope later to support a greater range of application types, see section 4), in this case a location-aware, XML content-based mobile sport news application, which requires the integration of middleware technologies to produce a suitable solution. To support it, the middleware platform features: interaction with CORBA services, request-reply communication, agent interaction and provision of system context information. These techniques are integrated together using a component solution to provide a standard service architecture to the application designer. For example, the platform integrates the agent paradigm and the Universally Interoperable Core (UIC) [2] component-based solution (that provides interoperability with CORBA services and other service types from limited end systems) to allow agent interaction with CORBA services. As an additional contribution the work will serve to evaluate state of the art middleware such as UIC.

This paper reports on the design of the sport news application in section 2 and the middleware platform it is built upon is described in section 3. Finally, section 4 draws conclusions and identifies the future direction of the work.

2. MOBILE SPORT NEWS APPLICATION

This section describes a mobile sport news application that was developed upon the middleware platform described in the following section. Figure 1 illustrates the application architecture. The content of the application (sport news stories) is provided as XML documents on a number of separate location based servers, allowing the content of each server to be appropriate to its local area. The mobile PDA devices communicate using GSM. Each device connects to a gateway to allow it to communicate with the application's servers, if a device changes cell then the gateway may change (handled automatically by the system).

The application currently provides the following key features:

- The end user is made aware of stories that are of interest to them based upon their current context. For example, if the user is currently in Manchester and they express an interest in football then the application presents stories about Manchester City (or other teams in the vicinity!). To do this, agents (represented by A in figure 1) are initiated with a profile of the end user describing the sport news relevant to them (for example, football stories). The agent then searches the server appropriate to the device's current location, communicating back any stories of interest.

- The user is kept up to date with the latest news about a current story without constant browsing. Instead of checking the latest score of a football match, the application informs the user when a document has changed. An agent monitors the story and when it is updated the application is signalled.
- The user can browse through sport stories on any server provided by the service.

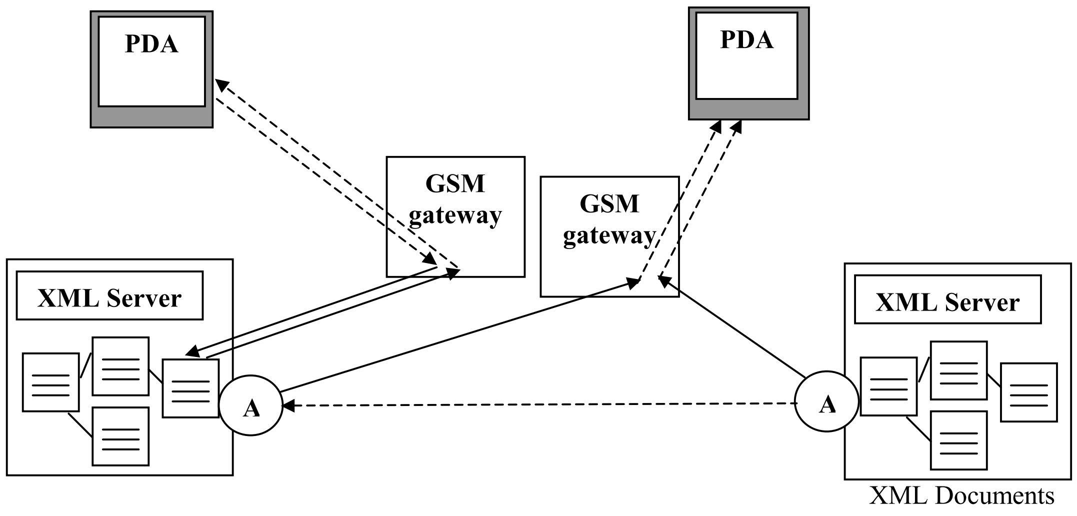

Figure 1 – Architecture of the mobile sport news application

Future work will extend the application to provide the following new features:

- The end user will be made aware of new stories of interest and the latest updates in current stories independent of their end-system ("*follow-me sport news*") e.g. when they change from PDA to desktop to laptop; this could be extended to include computers, devices or displays close to the user rather than just their own systems.
- Multimedia sports news will be made available different formats (video, audio, pictures, text) at different times depending on the current location, context generally, connectivity etc. An example of this is sending information about a goal scored in football match; if the device is well resourced (screen size, CPU speed, network QoS) and in an environment where noise is acceptable, then a video of the goal is sent. However, if the device in use has less resources or the network QoS decreases then the update could be sent as text only.

3. Middleware architecture

This section presents a middleware platform that supports applications of the type described in the previous section. The architecture follows a component design; each of the middleware services is provided as a component that maps closely to the application functionality described previously. Figure 2 illustrates the service architecture presented to the application designer, who does not need to deal with the problems of integrating the middleware technologies to best support the application, but instead interacts with standard interfaces. The integration of a range of technologies including agents and UIC CORBA personalities [3] to implement the middleware components is described below.

The location context component (shown in figure 2) allows the application to be aware of its current location. In particular, the interface provides a single method to determine the current location and a single event to indicate to the application that the location has changed. To implement this, the middleware reads from the GPS device on request and returns a GPS location; it also monitors the device for changes in location, generating an event when this occurs. The separation of implementation allows the future possibility for the middleware to adapt. For example, the mobile device could have two methods of determining location: GPS or GSM cell. A separate component for each location type could implement the interface, allowing for the middleware to be adapted to use the appropriate component.

The content fetch component provides the ability to fetch XML documents directly from the server. The interface provides a single method, which returns the contents of a named document (identified by a

URL) to the device. To implement this the middleware controls an http socket connection with the XML server, requesting the document and receiving the reply.

The final component (content search) provides the application with the ability to become aware of content that is relevant to the user; for example, each time there is a goal at a football match that the user has specified an interest in, or when a new story becomes available about the user's favourite team. The interface of this component provides two methods and two event types. These are described as follows:

- Search(Profile): A profile describing the interests of the user (in XML) is passed. When a story that matches the profile is found on the XML servers the application is made aware of this story using the StoryMatch event type.
- Watch(URL): The application specifies a document that it wishes to know when it is updated. An event (StoryUpdate) is received each time the story is updated.

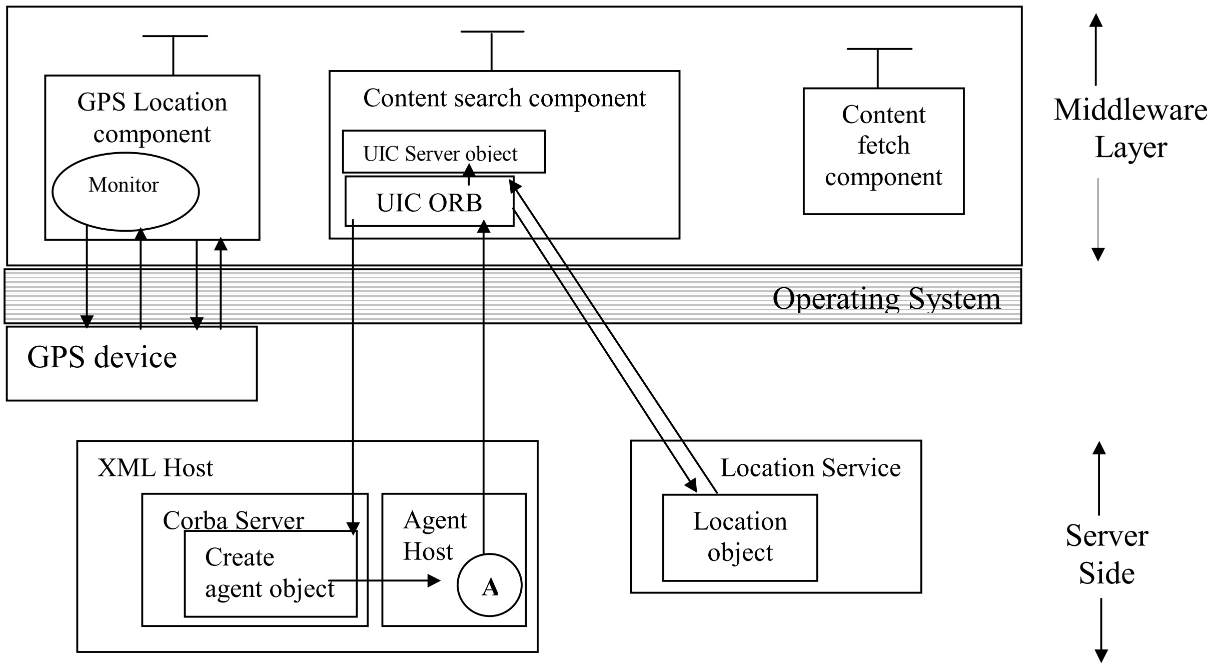

Figure 2 – Middleware architecture to support the sport news application

To implement these features an agent solution was chosen. As shown in figure 2, the agents are not created on the device by the content component but on a remote server. To do this a Universally Interoperable Core CORBA-personality [2] was used for the underlying communication infrastructure, allowing interaction with CORBA services using minimal end system resources. Therefore, the component integrates the use of agents and CORBA interaction to provide the required functionality. Future work will lead to a more principled approach to provide further levels of integration within components (see section 4)

Each remote XML host provides a single CORBA object (create agent) that offers one method to create a new agent. The component first needs to find one of these objects; to do this it uses its current location. A CORBA request is made to a known Location Service, which provides methods to translate the location into the IOR of a suitable create agent object. When the create agent method is called it communicates with an agent host to create the new agent, the user profile and the IOR of a server object on the device are passed as parameters and stored for use. The component contains a single UIC server object that provides two one-way methods: StoryUpdate(String URL) and NewStory(String URL) that allow the agents to communicate back to the device. An agent calls the appropriate method using the IOR it has stored, passing the URL it has found to inform of story updates or new stories of interest. Each method generates an event to inform the application of new stories or updates. In this version of the prototype, the agents are implemented using the Aglets toolkit [4].

4. CONCLUSIONS AND FUTURE WORK

The application and middleware platform were implemented on Handspring Visor PDAs, communicating via GSM using VisorPhone plug in modules. The code was developed using the CodeWarrior development environment for the Palm operating system.

The middleware platform and sport news application presented have identified the benefits of creating a component-based solution that integrates middleware paradigms for mobility. The application designer is presented with a service architecture that provides the functionality to implement the given application. They do not need to piece together platforms providing context awareness, service interaction and agent interaction; instead the implemented platform provides a standard service architecture for them to communicate with.

The middleware presents a set of components each providing a specific middleware functionality to support the implementation of the application. These are integrated to provide the service architecture. However, the content search component provides a deeper level of integration in that it uses two middleware techniques in its implementation (UIC and agent paradigm). In this prototype, the integration is hidden from the application and is implemented as a monolithic component. Therefore it is not suitable for a generalised approach to allow middleware technologies to be easily integrated. Future work, will consider further componentisation and integration to provide middleware techniques that better support mobility.

One of the goals of the work is to evaluate the use of agents and the UIC technology. Agents provide the following advantages for the type of mobile application implemented. Browsing for information is less suited to mobile devices, because fetching of information may be slow and wastes resources if the information is unwanted; therefore, agents can search for and send only information of interest to the user to overcome this. Agents can also control communication with the device to overcome periods when disconnected [4]. The use of UIC provided a minimal CORBA communication platform to limit the use of system resources. Its ability to adapt and interoperate with a range of service types will be investigated further (see later).

The work is originally targeting one application type and is therefore too specific to be considered as a middleware platform for mobile applications in general. Hence, future work will involve creating a more generalised solution to encompass a range of application styles (e.g. multimedia, virtual communities etc.) and context information (network QoS, nearby hosts, system resources, etc.).

The middleware solution at present is static and the application designer is simply presented with a set of functionalities to help implement a mobile application. A more general solution will provide a greater range of middleware services and therefore, a percentage of it may be unwanted. Also, the application may also wish to change the middleware implementation dynamically, as described for the location changes. Therefore, future work will investigate how to create a configurable and adaptable middleware platform that is suited to mobile devices by considering dynamic aspects and further componentisation. The component model will be extended to support configuration and re-configuration i.e. the platform will composed of a configuration of components selected at build time, which can be re-configured at run-time to respond to environment changes. These components will be described in terms of a set of required and provided interfaces. The work will also examine the benefits of dynamic downloading of components during re-configuration, to limit the resources used on the end system. Finally, existing adaptable middleware technologies such as UIC and OpenOrb [5] will be investigated as solutions to provide re-configurable communication platforms.

5. REFERENCES

1. Katz, R. “Adaptation and Mobility in Wireless Information Systems”. IEEE Personal Communications. 1(1), pp. 6-17, 1994.
2. Ubicore. “Universally Interoperable Core Homepage”. http://www.ubi-core.com/
3. Jacobsen, K. and Johansen, D. “Ubiquitous Devices United: Enabling Distributed Computing Through Mobile Code.” In, Proceedings of the Symposium on Applied Computing (ACM SAC'99*)*, February 1999.
4. IBM. “IBM Aglets Software Development Kit – Homepage”. http://www.trl.ibm.com/aglets/
5. Blair, G.S., Coulson, G., Andersen, A., Blair, L., Clarke, M., Costa, F., Duran-Limon, H., Fitzpatrick, T., Johnston, L., Moreira, R., Parlavantzas, N., Saikoski, K., “The Design and Implementation of OpenORB v2”, To appear in IEEE DS Online, Special Issue on Reflective Middleware, 2001.