<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>DGPS for mobile users: a mobile agent approach</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Authors(s)</strong></td>
<td>O'Grady, Michael J.; O'Hare, G. M. P. (Greg M. P.)</td>
</tr>
<tr>
<td><strong>Publication date</strong></td>
<td>2004-12-08</td>
</tr>
<tr>
<td><strong>Conference details</strong></td>
<td>The 2nd European Space Agency (ESA) Workshop on Satellite Navigation User Equipment Technologies (NAVITEC '2004), 8th-10th December 2004, Noordwijk, The Netherlands</td>
</tr>
<tr>
<td><strong>Item record/more information</strong></td>
<td><a href="http://hdl.handle.net/10197/4476">http://hdl.handle.net/10197/4476</a></td>
</tr>
</tbody>
</table>
INTRODUCTION

As the computational power of mobile devices increases, so does the demand for more sophisticated applications and services. In particular, the demand for location-aware services is likely to increase substantially. Such services will vary in how accurate a position solution they will require. However, there can be little doubt that some will require an accuracy that can only be provided through DGPS. In this paper, we propose a solution based on mobile agents. In particular, we envisage agents migrating to applications hosted on mobile devices and delivering a DGPS solution. It is assumed that GPS is available locally on the device and that a wireless connection to SISNet server is available.

MOBILE COMPUTING

Mobile computing has grown in importance over the last decade to become a significant research discipline in its own right. A number of visions have been proposed as to how mobile computing should evolve and how best the paradigm itself may be realised. Two diametrically opposed paradigms have been proposed which may be imagined as existing at opposing ends of the mobile computing spectrum. The first, Ubiquitous Computing [1], envisages an environment saturated with sensors and associated computational infrastructure. In such a scenario, a user would not normally be expected to carry computer hardware. Rather, they would access any desired services through appropriately endowed objects dispersed throughout the environment. In contrast, Wearable Computing [2] anticipates the user carrying the required computational resources about their person. Such an approach offers potential users several advantages regarding security and privacy issues. Between these two extremes is the classic mobile computing vision of a laptop or PDA, augmented with a wireless communications facility for remote data access.

One particular variety of mobile computing that has grasped the imagination of business and academic alike is that of location-aware computing. In this case, the location of the end user is the essential component considered when delivering a service. However, determining the location of the user accurately enough can present significant difficulties. GPS remains the preferred choice for many people. It is free, available worldwide and gives a reasonably good position reading to within 20 metres or so. Alternative techniques based on the topologies of cellular telecommunication networks have been proposed; some of which have been ratified by the appropriate telecommunications standards body. In the case of the third generation (3G) Universal Mobile Telephone System (UMTS) [3], three techniques have recommended: Cell-ID, Observed Time Difference of Arrival (OTDOA) and Assisted-GPS (A-GPS). The difficulty with such techniques is that the accuracy of the recovered position can vary widely. A-GPS is comparable with conventional GPS but has not been implemented widely. If a more accurate position reading is required, then a DGPS solution must be considered. Traditionally, this has been impractical due to the nature and expense of the hardware involved. However, ongoing developments concerning the broadcasting of differential corrections via the internet has led to some intriguing possibilities for suitably equipped mobile users.

Though technological developments in the mobile computing field have been impressive, much work remains to be done on the less glamorous but essential system engineering issues. Quite simply, "best practice" principles have not yet been identified for potential designers and developers of mobile computing applications. Mobile Human Computer Interaction (HCI) being one area that is receiving increased attention from the research community. One radical alternative to traditional software development methodologies is that of one based around intelligent agents, and, in the next section, such agents are examined in further detail.
Recent advances in PDA and smart-phone technologies have opened up a new frontier in intelligent agent research. Until recently, the computational demands that a Multi-agent System (MAS) required rendered their use on lightweight devices impractical. This situation has changed in recent years and a number of projects concerning the practical deployment of agents on mobile devices have been documented in the literature. Examples include the Lightweight and Extensible Agent Platform (LEAP) [4], MicoFIPA-OS [5] and Grasshopper [6]. However, as the term intelligent agent has not been standardised, its interpretation can vary and the resultant agents can display widely varying capabilities and characteristics. One particular family of agents are those that adhere to the Belief-Desire-Intention (BDI) paradigm [7]. Such agents use an intrinsic combination of beliefs, commitments and commitment rules to maintain a mental state. The mental state, at any moment in time, represents a snapshot of the agent's status and will determine its immediate future actions. By continuously monitoring its environment, the agent can update its mental state accordingly thus determining what activities it must undertake.

A number of characteristics are synonymous with intelligent agents including autonomy, proactivity reactivity, social ability and so on, all of which may be incorporated to varying degrees. One particular facet of agents is of particular interest, namely mobility. An agent moves, or in agent parlance migrates, by moving their code from one computer platform to another, assuming of course that the destination platform is willing to receive the agent and that all security issues have been addressed. Two kinds of migration strategy can be adopted:

1. **Strong migration** where the agent's object state, code and control state is captured. Upon migration, the agent can resume execution on the destination machine at the exact point it left off.
2. **Weak migration** where only the agent's object state and code is captured. Upon migration, a known point in the code is called to restart the agent on the destination machine.

The strategy adopted depends on the prevailing circumstances and the objective of the agent. If an agent had to migrate unexpectedly, possibly for load balancing reasons, a strong migration approach would be appropriate. If an agent needed to migrate so as to avail of, or indeed provide, a certain service, a weak migration approach could be adopted.

**GULLIVER'S GENIE: A CLASSIC MOBILE COMPUTING APPLICATION**

Gulliver's Genie [8] [9] is a classic example of a mobile computing application that integrates several apparently disparate technologies to deliver services to mobile users. In this case, the tourist domain is the focus of the delivered services. The Genie delivers navigation information as well as multimedia presentations on attractions encountered by the tourist while they roam some area. All presentations are context-sensitive, that is, they are sensitive to the tourist's position as well as their personal and cultural interests.

**Architecture**

In essence, Gulliver's Genie comprises a suite of agents residing both on the client and on the server as illustrated in Fig. 1. All the agents collaborate to deliver the necessary services to the tourist. Each agent is now briefly reviewed:

**Spatial Agent**

To determine a tourist’s spatial context, this agent autonomously monitors the GPS signal and interprets it accordingly. It periodically broadcasts the position to other interested agents. The Spatial Agent is unique in that it harnesses its capacity to migrate. Though GPS is the de facto standard for position determination at present, systems using cellular network techniques are envisaged in the future. By using a mobile agent, the Genie can be deployed on those devices that utilize cellular techniques when they become available, as the appropriate agent encompassing the logic for handling cellular network positioning may be dispatched to the device. Should there be a need to avail of a Satellite Based Augmentation System (SBAS), for example EGNOS, WAAS, this could also be facilitated quite easily.

**Cache Agent**

Intelligent precaching is one of the Genie’s defining characteristics and the Cache Agent is responsible for implementing this strategy on the client. An environmental model is provided by the GIS Agent on the server. By considering this model in light of the tourist’s movements, it identifies possible attractions that the tourist may visit.
multimedia presentation is requested from the Presentation Agent in anticipation that it will be downloaded by the time the tourist encounters the attraction in question. Should the tourist not encounter the attraction, the presentation is simply discarded.

**Fig. 1. Architecture of Gulliver’s Genie**

**GUI Agent**
Controlling the interface on the tourist’s device is the main task of the GUI Agent. In normal navigation mode, an electronic map is displayed with the current position and orientation highlighted. This is, of course, continuously updated as the tourist moves, courtesy of updates from the Spatial Agent. Should the tourist encounter an attraction for which a presentation has been precached, the Cache Agent prompts the GUI Agent to display the presentation, monitor the tourist’s interaction and provide feedback to the Profile Agent on the server.

**Registration Agent**
Tourist must first register for Genie services. The Registration Agent takes care of this process as well as assigning Tourist Agents to individual tourists in response to requests for Genie services.

**Tourist Agent**
All tourists registered for Genie services are assigned their own individual agent, termed Tourist Agents, on commencing a session. Such agents are cloned from the tourist agent template. Essentially, this agent is the tourist’s interface to the services offered by the Genie. Acting on prompts from the Spatial Agent, it arranges the construction of environmental models in conjunction with the GIS Agent and prompts the Presentation Agent to maintain an updated list of presentations in anticipation of requests from the Cache Agent. Though there is some computational overhead in assigning an agent to each tourist, such an approach ensures the future scalability of the Genie as the number of concurrent tourists increase and a wider variety of services are offered.

**GIS Agent**
Accurate environmental models are necessary for the successful operation of the Genie. Such models are provided to the Cache Agent on the client as well as to the Presentation Agent for dynamic presentation pre-assembly on the server.

**Profile Agent**
User profiles are essential to realizing adaptivity and personalization. The Profile Agent is responsible for maintaining user profiles and updating them in light of ongoing tourist interactions with the Genie.

Presentation Agent

The provision of personalized multimedia presentations is a core tenet of the Genie’s raison d’être. Such presentations are assembled in light of the tourist’s profile and their current environmental model. This server-side presentation repository is continuously updated in light of tourist movement and refinements of their individual profiles.

Implementation

At present, the Genie runs on a standard PDA, namely an iPAQ 3850. GPS, which gives a position reading to within 20 metres on average, is used for determining location. Orientation can also be derived from GPS, albeit in an approximate manner. For data communications, the standard 2.5G technology, the General Packet Radio Service (GPRS) is used. In each case, a corresponding PCMCIA card was procured. Both cards were then incorporated into the iPAQ via a dual-slot expansion sleeve. From a software perspective, the client components are implemented in Java and a commercial JVM, namely Jeode, is used as the runtime environment on the IPAQ. All communication with the server takes place over a standard HTTP connection.

On the server side, the Agent Factory [10] runtime environment is deployed. This is augmented with a sophisticated database that supports multiple data types including geospatial data and multimedia-related data. A toolkit for populating this database forms an indispensable component.

Modus Operandi

Before the Genie can be used, an information space must be defined for the area in question. A prospective tourist must then register and provide some information concerning their personal profile and cultural interest profile. They can then avail of Genie services. Given the importance of navigation, a map with their current position and orientation highlighted provides the default interface (Fig. 2.). As they roam around, the multimedia presentations are downloaded automatically and displayed once the tourist encounters the corresponding attraction. This process of intelligently precaching data in an almost just-in-time basis is one of the Genie’s distinguishing features. Such a strategy disguises the poor data-rates available on cellular data networks as well as maximizing the memory resources on the PDA.

THE NEED FOR BETTER POSITION RECOVERY

Feedback from user trials indicated that while users appreciated the Genie’s capacity to provide data when and where it was needed, it was noted that the timing was occasionally off. Given the average 20 metre error margin associated with
GPS, this is not surprising. However, while users may appreciate this, it is not likely that they will pay for a service that does not meet their expectations. The issue facing us then is to improve the performance of the pre-caching mechanism. Improvement in telecommunications technologies, particularly 3G, will aid this. However, as timing is all important, the need to improve the accuracy of the recovered user position becomes critical.

**Proposed Solution**

DGPS is the logical solution when improved user position readings are required. But how should a DGPS solution be implemented? The traditional approach involving the user carrying additional antennae and equipment is not practical for most people, and certainly not tourists. The current generation of Compact Flash (CF) GPS receivers have the capacity to receive SBAS signals. While this will improve the situation in the coming years, a problem will still exist in so-called urban canyons; exactly the kind of places that tourist are likely to congregate. In an effort to address this, ESA conceived the SISNet [11] initiative in which the differential corrections are broadcast over the internet. Thus anybody with a wireless connection can access this service.

In the case of the Genie, we are considering incorporating SISNet both on the server side and on the client side: each with its own advantages and disadvantages.

**SISNet on the Server**

In this case, an agent, residing on the Genie server, would interface with the SISNet server and, in this way, provide a DGPS service to the Spatial Agent on the client. The Spatial Agent would send the necessary GPS data, gleaned from the NMEA sentences, and the agent interfacing with the SISNet server would take this and calculate an improved position. This improved position reading could then be used to improve the services provided by the Genie. For standard location-aware services, this scenario would work reasonably well.

**SISNet on the Client**

Realising a solution based on SISNet on the client is a more difficult prospect. A wireless communications channel must be continuously available, and this resource may be needed by other agents. The effective management of the communications channel becomes a critical issue. Considering the limited computational resources available, anything that adds to the load on the system could potentially have an adverse effect on system performance. In the case of the Genie however, this is the only viable approach. Intelligent precaching is dependent on the user's position being available. Obviously, the more accurate this is, the more effective the precaching will operate, and ultimately, the more rewarding the end-user experience.

Incorporating the mobility characteristic of agents offers a potentially elegant solution. When an application needs local access to the SISNet, the appropriate agent could migrate from the server to the device and provide this service. In some cases, this might only be for a short time while the user is availing of some service that demands the availability of corrected position readings immediately. In the case of the Genie however, this agent would migrate at the start of a session and remain for its entire duration.

**FUTURE WORK**

Ongoing research concerns the investigation of autonomic computing principles [12] and how they might be usefully employed in the implementation of mobile computing applications. Part of this involves the construction of a suite of agents that provide specialised services and can collaborate to deliver services. As part of this, it is intended to complete the development of a agent that can interface with SISNet from a mobile device. As well as being used by Gulliver's Genie, such an agent could be used in other applications that could beneficially use the SISNet service.

**ACKNOWLEDGEMENTS**

Michael O'Grady gratefully acknowledges the support of the Irish Research Council for Science, Engineering & Technology (IRCSET) through the Embark Initiative postdoctoral fellowship programme. Gregory O'Hare acknowledges the support of Science Foundation Ireland through Grant No. 03/IN.3/1361.
REFERENCES