Sos: Accommodation on the fly with ACCESS

Phelan D., Strahan R., Collier R., Muldoon C., O'Hare G.M.P.

Department of Computer Science
University College Dublin
Ireland
Donnacha.Phelan@ucd.ie, Robin.Strahan@ucd.ie, Rem.Collier@ucd.ie, Conor.Muldoon@ucd.ie, Gregory.OHare@ucd.ie

Abstract

This paper introduces Sos, a location aware and context-sensitive accommodation finding service for mobile citizens who require help finding somewhere to stay when they arrive at their chosen destination. Specifically, Sos helps users to find and book hotel accommodation that is most appropriate to their current context. This context combines the users' current location, personal preferences, hotel availability and agenda (e.g. business meeting, tour of city). Sos has been realized as an agent-based application that has been deployed using the Agents Channelling ContExt Sensitive Services (ACCESS) architecture, an open agent-based architecture that supports the development and deployment of multiple heterogeneous context-sensitive services.

1. Introduction

Pervasive Computing is recognised as a key growth area for the computer industry (Pearce, 2003) (Crease, 2003). The vision of Pervasive Computing is often held as one of "smart" devices operating seamlessly and dynamically, forming ad-hoc networks with other related devices, presenting the user with a truly ubiquitous intelligent environment. This vision offers many similarities with the concept of intelligent agents where autonomous entities, known as agents, interact with one another forming ad-hoc alliances and working both reactively, and pro-actively, to achieve individual and common goals.

Perhaps the key distinction between these two visions is that agent technologies are viewed more as enabling technologies than end-user technologies. As such, Pervasive Computing is recognised as one of the key research areas for applications of agent technologies (Luck et al., 2003). This has led to the development of a number of exemplar agent-based systems which showcase how agent technologies might be utilised effectively to realise Pervasive Computing applications (Mihailescu and Binder, 2001) (Keegan and O'Hare, 2002).

A consequence of the emergence of such exemplar systems is the recognition that most location-aware context sensitive applications exhibit a common functional core. These include a location-sensing capability typically Global Positioning System (GPS) augmented with Radio Frequency IDentification (RFID) tags, dynamic generation and acquisition of user specific map fragments, support for management of the users context, and service brokerage.

This paper introduces *Sos*[1], a location-aware and context-sensitive accommodation location service for mobile citizen. The primary focus of Sos is in providing assistance for users who arrive at a given destination without prior bookings and who require help finding somewhere to stay when they arrive at their chosen destination. Specifically, Sos helps users to find and book hotel accommodation that is most appropriate to their current context. This context combines the users current location, personal preferences, hotel availability, and agenda (e.g. business meeting, tour of city). Sos has been realized as an agent-based application that has been deployed using the Agents Channelling ContExt Sensitive Services (ACCESS) architecture, an open agent-based architecture that supports the development and deployment of multiple heterogeneous context sensitive services (Muldoon, 2003).

2. Motivation: Our Intrepid and Disorganized Traveler

There exist a number of scenarios in which a traveller arrives at their destination with either no accommodation booked, or else, finding that their reservation has not been honoured. For example, consider the scenario in which Greg decides at the last minute to attend a conference in Miami and books tickets for the journey. A busy work schedule distracts Greg from the ancillary task of booking hotel accommodation. As a result, Greg arrives in Miami without accommodation. Subsequently, Greg finds out that the UEFA Under 21 Soccer World Cup is also taking place in Miami, and that accommodation is hard to find. Having arrived at 8pm in the evening, Greg's options are limited since the tourist information offices are closed. The option of walking around the city searching for accommodation, is neither appealing nor safe

[1] Our application takes its name from the Irish word for rest– *Sos*, the associations with emergency situations are also not coincidental.

This scenario is compounded by Greg's lack of local knowledge – including a rudimentary command of the local native language, what telephone numbers (if any) to ring, where best to search for a hotel, and more mundane items such as the location of the town center. Dilemmas such as Greg's have inspired the development of *Sos*, a location-aware context-sensitive hotel accommodation finder. Through *Sos*, we are able to marshal this local knowledge and present relevant details using an intuitive map-based interface providing the traveler with a context in which he is able to understand both what his options are, receive assistance in the selection of the hotel most appropriate for his particular needs, together with securing the appropriate reservation.

3. Related Work

Considerable research has been invested of late in the deployment of ubiquitous and pervasive services, a number of such services have been targeted at the accommodation and tourism sector. Hypergeo[2] is one such system, which developed a prototype tourism service for the delivery of location and user sensitive information taking into account new information technologies, geographic information and new communication infrastructures. Hypergeo includes a hotel information service that presents a list of hotels that is ordered by both user preference and locality. However, the information provided is locational, and does not consider availability or provide mechanisms for enabling the user to make a reservation.

A second system under development is NEXUS (Volz and Klinec 1999). This system aims to develop a generic platform that supports the delivery of location-based services. This includes the implementation of a location aware hotel finding service in which the user specifies the region in which they wish to stay. Unfortunately, no prototype application currently exists.

From the commercial perspective, companies such as TomTom (the Varta Guide)[3] and the Collinson Group (Pocket Travel Plan)[4] offer static hotel information that can be downloaded onto Personal Digital Assistants (PDAs). A third alternative is sino.net, which offers a mobile hotel portal that offers information on Asian hotels and provides an e-mail/phone-based hotel room booking service.

Cheyer and Julia (Cheyer and Julia 1995) have developed a system based around a map-based interface, which provides tourist information about the city of San Francisco. Requests expressed in a variety of modalities can control the scrolling and zoom level of the map, retrieve information about locations and distances and display hotels or attractions meeting a user's preferences. Where appropriate additional information may be obtained from live Internet sources. A similar system is described in (Oviatt, 1996).

[2] http://www.hypergeo.org/
[3] http://www.tomtom.com/
[4] http://www.travelplan.com/

The accommodation finder service that we present in this paper also uses a map-based interface. However in contrast to (Cheyer and Julia 1995), Sos provides strong support for implicit and dynamic user profiling and context sensitivity. Sos also provides functionality, currently lacking in Hypergeo, for checking availability and making reservations. In addition all locational and hotel information is harvested dynamically from preexisting web resources and Sos specific data sources. Consequently Sos offers superior flexibity and adaptivity, to the Varta Guide and Pocket Travel Plan. The provision of such characteristics is primarily achieved through its enabling technology namely ACCESS.

4. The ACCESS Architecture

The Agents Channeling ContExt Sensitive Services (ACCESS) architecture is an agent-based architecture that supports the development and deployment of context sensitive services (Muldoon 2003). ACCESS has been realized as an extension of a pre-existing framework for agent development, known as *Agent Factory* (Collier 2003). Agent Factory is a cohesive framework that delivers structured support for the development and deployment of agent-oriented applications. Specifically, Agent Factory supports the fabrication of a type of software agent that is: autonomous, situated, socially able, intentional, rational, and mobile. It is comprised of four-layers that deliver: an agent programming language, a distributed run-time environment that delivers support for the deployment of agent-oriented applications, an integrated toolkit that delivers a visually intuitive set of tools, and a software engineering methodology that specifies the sequence of steps required to develop and deploy agent-oriented applications with the preceding layers. Additionally, Agent Factory provides FIPA-compliance through an Agent Management System (AMS) agent and a Directory Facilitator (DF) agent. Agent-oriented applications built using Agent Factory use these prefabricated agents to gain access to the infrastructure services provided by the run-time environment (i.e. yellow and white pages services, migration services).

ACCESS provides a membrane of agents that augments the basic infrastructure delivered by Agent Factory through the provision of a collection of agents that facilitate the rapid prototyping of, context-sensitive applications. These agents form a cohesive management layer into which multiple heterogeneous context-sensitive services may be plugged. This enables service developers to focus on the implementation of their service, rather than the infrastructure required to deliver it.

4.1 ACCESS Management Agents

The ACCESS Management agents implement the core functionality of the ACCESS architecture, which includes context management, user profiling, map generation,

content delivery, account management, and location sensing.

The User Profiling Agent provides a mechanism to enable agents request user preferences, which are used to configure user specific services. A distinct user profile exists for every registered user. User profile information is obtained explicitly from the user preference web form and implicitly from the activity analyzer. One advantage of requiring the user to complete a preference form is that it allows the personalisation of services to begin immediately, allowing applications to begin tailoring their services to the needs of specific users prior to observing users' behaviour and preferences. The personalisation process can then further refined as the profile is implicitly developed as the user interacts with the system. Implicit profile information is obtained using data mining techniques on recorded user activity, for instance when examining service usage its noted what was used, when it was used, and where is was used.

The Map Agent dynamically generates maps for a specified location. When a generate map request from a service agent is made, the size of the map and its location are specified. The Map Agent contacts its map server, to retrieve the map segments required to generate a map centred on the location specified. The generated maps can subsequently be merged with service specific content overlay. The Map Agents exist as part of a community of peers, this enables the distribution of load and also allows the Map Agents to be geographically bound i.e. Map Agents are not responsible for generating maps for the entire globe, they have specified bounds for their locality.

Within the ACCESS architecture, a user's context is, at this stage of development, merely considered as a combination of their location, previous activities, and preferences. Context is used to help ACCESS understand where and when to provide services, and more specifically, what services the user may need. The Context Agent is responsible for handling context sensitive hotspots within the ACCESS architecture. Specifically, a hotspot is a region of space-time that is limited by specified bounds (e.g. an area of 100 square meters centred around a particular shop, during the shops business hours). Once a user breaches a hotspot the Context Agent informs the Profiling Agent, which may then decipher whether the hotspot breach is relevant to the user.

In ACCESS, hotspots are created in two ways: (1) through requests by specific services that wish to know when a user enters a certain space-time context, (2) through the analysis of past user activity and/or user preferences. The former represents a service-centric approach to context management, while the latter is user-centric.

The Position Agent is responsible for periodically informing the Context Agent of the users position. It is a lightweight agent that resides on the users PDA. It currently uses GPS, however it is envisaged that in the future other position sensing technologies may be used.

The Service Broker manages the advertising of services to the user. The order in which adverts are pushed to the user depends on both the users' spatial/temporal location and user preference (this is their context). In order for a service to be advertised it must register with Service Broker.

The Device Aware Content Delivery Agent (DACDA) performs the roles of system interface and content manager. Specifically, it is capable of determining primitive device characteristics, which it uses to tailor content delivery. All communication with the user is done via the DACDA. It is also responsible for registering the user with the system at start up and registering the user with any services they wish to use. In addition, the DACDA is responsible for monitoring user interaction with ACCESS-compliant interfaces. An ACCESS-compliant interface constitutes a graphical user interface that may be dynamically loaded by the DACDA at runtime. ACCESS-compliant interface components are displayed within the *ACCESS viewer*, a purpose-built interface capable of simultaneously switching between services. These components are described within an interface composition script.

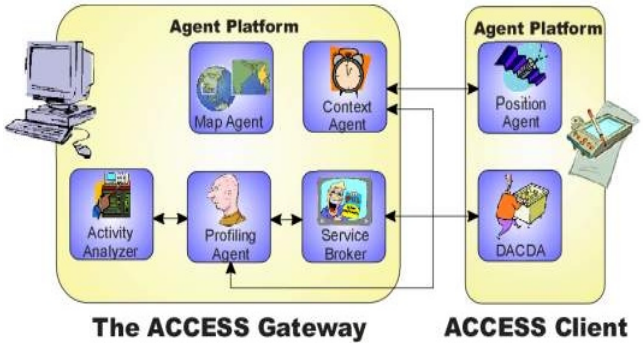

Figure 1. Access Management Agents

4.2 The Service Provider Contract

The Service Provider Contract forms the minimum requirements necessary for a service to be considered ACCESS compliant. Roles that must be filled as part of the Service Provider Contract include the Service Manager and the Service Delivery Agent. These components constitute prefabricated components of the ACCESS development framework and may be extended to provide service specific functionality. Although the Service Provider Contract gives two roles that must be provided, it does not limit the service developer from adding additional roles and agents that are specific to the application they are developing. The Service Manager is responsible for registering the service with the Service Broker. It communicates with the DACDA when a user wishes to use the service. Once the DACDA initiates a conversation the service manager delegates the DACDA to a Service Delivery Agent, which will be the DACDAs communication entry point for the service.

The Service Delivery Agent collaborates with the other service agents to deliver the required content to the user. The functionality of the Service Delivery Agent will primarily be service specific, but from the point of view of

the contract it is viewed as the functional data block that handles or represents the user in the service.
With the provision of the ACCESS architecture, the rapid development of context sensitive services are possible. One such service is that of Sos.

5. The Sos Hotel Finder Service

Sos is a location-aware and context-sensitive hotel finding service for mobile citizens who require help finding somewhere to stay when they arrive at their chosen destination. Specifically, *Sos* helps users to find and book hotel accommodation that is most appropriate to their current context. This context combines the users current location, personal preferences, hotel availability, and agenda (e.g. business meeting, tour of city).

5.1 The Sos Hotel Finder Architecture

Sos has been realised as an ACCESS-compliant service. That is, the *Sos* application uses the generic common core of functionality that is deployed through the ACCESS architecture and augments this with service specific components. Specifically, Sos makes use of 5 aspects of this common core of functionality:

- The *ACCESS Viewer* is used as a medium through which the Sos Service is able to interact with the user.
- The *user location sensing* capability is used to identify the users current location.
- The *context management* functionality is used to set up hotspots that act to advertise the service to the user.
- The *map generation* functionality is used to generate maps that display both the users position as well as hotel locations.
- The *service brokerage* functionality is used to register the service with the ACCESS gateway, allowing users to manually select the service.

This functional portfolio is delivered via the implementation of the roles specified in the Service Provider Contract (section 4.2) in conjunction with a number of custom service agents. A diagrammatic overview of the Sos Hotel Finder Service Architecture can be found in figure 2.
In this architecture, the Sos Manager agent implements the Service Manager role, and is responsible for registering the service with the service broker, and setting up the relevant hotspots for triggering the service. Conversely, the Sos Delivery agent implements the Service Delivery Agent role. Its implementation defines how agents deliver the *Sos* service. This includes contacting the Map Agent at relevant times, and interacting with the Matcher Agent whenever the user initiates a search for a list of hotels.
The remaining functionality of *Sos* is realised through the implementation of a number of custom service agents. Specifically, the Matcher Agent is responsible for handling user requests for hotels. The request received by the matcher agent combines a username, target search area, and a key that is used to uniquely identify the ACCESS Viewer Panel (see section 4.1) that initiated the request. Based upon this request, the Matcher Agent retrieves a set of hotels from the relevant target area, and then ranks those hotels based upon availability, cost, location, and similarity to the users preferences (see section 5.2). Availability and pricing information is obtained through collaboration between the Matcher agent and a number of Wrapper agents that interface directly with relevant websites / web services. Finally, a web-based interface is provided that enables administrators to update the hotel listings, and users to register and modify their profiles.
To illustrate the operation of this architecture, we now describe the principle usage scenario where the system retrieves a personalized list of hotels for the user. As stated previously in section 4.1, all interactions with the user are handled through the ACCESS Viewer. Specific events relating to this interaction are sensed by the

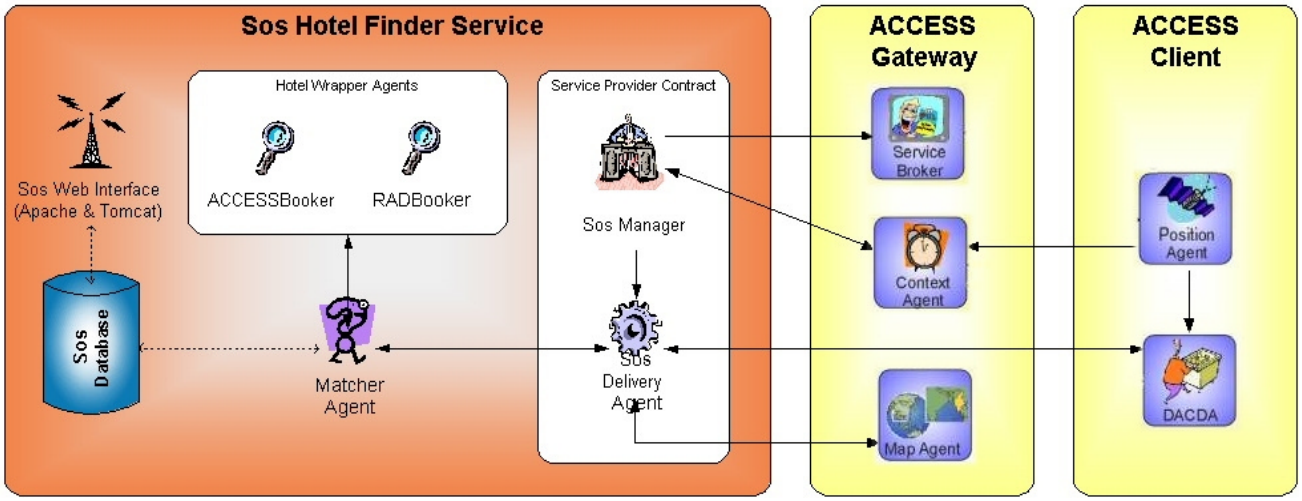

Figure 2. The Sos Hotel Finder Architecture

associated DACDA and routed to the appropriate Service Provider Agents (in this case the Sos Manager agent and Sos Delivery agent). In the scenario where the user employs Sos to locate a suitable hotel, the users accommodation requirements (e.g. desired target area, length of stay, date of arrival) are transmitted to a pre-assigned Sos Delivery agent via the DACDA. Upon receipt of these requirements, the Sos Delivery agent contracts a Matcher agent to generate a personalized list of hotels. The Matcher agent searches the database for hotels located in the target area that meet the user's preferences, instructing the relevant Hotel wrapper agents to retrieve hotel availability and pricing information that is relevant to the users accommodation requirements. Upon completing this search, the Matcher agent passes the personalized list to the Sos Delivery agent, which, in turn, routes the list to the relevant DACDA for presentation to the user.

5.2 Matching Hotels To Users in Sos

Matching of users to hotels is achieved through a similarity metric that combines location information, user preferences, hotel availability, and price. The location of the hotel is evaluated with respect to the target area chosen by the user (see section 6 for an illustration of this). User preferences are based upon profiles of both the user and the hotels. Both profiles combine information on the desired / achieved Start Rating of the hotel (1 star to 5 star) and the amenities offered by the hotel (e.g. whether there is a pool, bar, restaurant, baby changing facilities, etc.). Hotel availability is used to filter the list of hotels (i.e. hotels that do not have availability are removed). Finally, the price is used as the final differentiator between hotels (cheaper is seen as better). Purpose-built Wrapper agents provide an agent-oriented interface to hotel web site booking forms through which they are able to retrieve hotel availability and pricing information. The similarity Matcher Agent collates hotel availability and pricing information and uses this information in conjunction with the similarity metric to generate a list of hotels.

6. So, what happened to Greg?

Upon arriving in Miami, Greg switches on his Personal Digital Assistant (PDA), and receives an advert from the *Sos* Hotel Finder Service. Clicking on the advert brings Greg to the welcome screen. Detecting that Greg is an existing user of *Sos*, the screen instructs Greg to continue. This leads on to a second screen (figure 3a) in which Greg is asked to enter various details about his stay in Miami. This includes the arrival date (the current date by default), the number of nights and the number of people. Greg is also prompted to select an area of interest. This drop down list contains a number of places that may be of interest to Greg. In this case, Greg is presented with a personalised list of options that includes the railway station, downtown, and the venue of the conference he will attend. A fourth

Figure 3. (a) The Hotel Requirements Panel (left)
(b) The Select Map Panel (right)

option in the drop-down list is the "Select from map" option. It is this option that Greg selects.

Upon choosing this option, Greg is presented with a screen that contains a map that displays his current position. Various options allow Greg to change the zoom level, to pan the map, and to select an area on the map. However the principal purpose of this screen is to get Greg to identify the area of Miami in which he wishes to stay. This circular area and is identified by two clicks over the map. The first click identifies the centre point of the circle, and the second identifies the radius of the circle. Figure 3b shows a screen shot of the area that Greg selects. Once an area has been selected, Greg clicks on the continue option. *Sos* now has sufficient information to start a search for available hotels. This information is sent to a pre-assigned Sos Delivery agent via the DACDA as described in section 5.1.

Once this search has been completed, a third screen is presented to the user containing a list of hotels that have been found (see figure 4a). This list of hotels is filtered for availability, and is ordered by similarity, price, and proximity to the centre of the selected area.

Highlighting a hotel, and clicking on the Location option presents the user with a fourth screen (figure 4b) containing a map, the Greg's location, and the available hotels. The map is centred on the selected hotels location, but is set at a zoom level that also shows Greg's current location. However, if Greg is outside the scope of the maximum zoom level, then his location is not displayed. Clicking on one of the hotel icons brings Greg to a screen that displays details (name, price, description and picture) of the selected hotel. Once Greg is happy with one of these hotels, he returns to the Available Hotels screen, and selects the desired hotel. This causes Sos to generate an offer that is presented on a screen. This screen allows Greg some level of customisation of the offer. Once happy, Greg accepts the offer. A short time later, an Authorise Payment screen appears asking Greg to confirm

Figure 4. (a) The Available Hotels Panel (left)
(b) The View Hotel Location Panel (right)

the payment. Upon confirming the payment an appropriate transaction occurs (e.g. a VISA transaction), and a booking number is generated. This number is displayed to Greg, and stored in his "My Bookings" container.

7. Conclusions

This paper has introduced Sos a context aware, just-in-time accommodation location service for mobile users. Sos distinguishes and differentiates itself from other systems in several important respects:

- It delivers system adaptivity through the use of a rich user context yielding a personalized hotel finding service.
- It yields a just-in-time accommodation location and reservation system.
- It adopts a multi-agent system metaphor facilitating intelligent push service delivery, dynamic profile updates, agent migration and system load balancing across the client server architecture.
- It provides wrapper functionality whereby Sos can access preexisting web based information sources.
- It provides a support envelop that ensures utmost ease in the uploading of new hotel entries together with associated pricing, promotional materials and preferred subscription tariff bands.

A fully functional prototype has been developed and demonstrated at (Muldoon 2003). On going research is investigating the scalability of the service together with usability trials these will be reported in later work.

Acknowledgements

We gratefully acknowledge the support of Enterprise Ireland (grant ATRP/01/209) and Science Foundation Ireland through Modelling Collaborative Reasoners (SFI Investigator Award).

References

Creese, S. 2003, Future Challenges in Pervasive Computing Environments, *SC Infosec*, Mar 5 2003

Keegan, S., and O'Hare, G.M.P. 2002, EasiShop: Context sensitive Shopping for the Mobile User through Mobile Agent Technology, In *Proceedings of PIMRC 2002 13th IEEE International Symposium on Personal Indoor and Mobile Radio Communications*, Lisbon, Portugal.: IEEE Press.

Mihailescu P., and Binder W. 2001, A Mobile Agent Framework for M-Commerce, In Proceedings of Agents in E-Business, Vienna, Austria:

Pearce J., 2003, IBM: Pervasive Computing is the future, *ZD Net*, Jan 30 2003

Luck, M., McBurney, P., and Preist, C., 2003, Agent Technology: Enabling Next Generation Computing. *AgentLink* 12:1465-3842.

Cheyer A., and Julia L 1995, Multimodal maps: An agent-based approach. In Proceedings of the International Conference on Cooperative Multimodal Communication, Eindhoven, The Netherlands: Springer

Oviatt. S. L. 1996, Multimodal interfaces for dynamic interactive maps. In *Proceedings of Conference on Human Factors in Computing Systems 1996*, Vancouver, Canada: ACM Press

Collier, R.W., O'Hare G.M.P., Lowen, T., and Rooney, C.F.B. 2003, Beyond Prototyping in the Factory of the Agents, In Proceedings of 3rd Central and Eastern European Conference on Multi-Agent Systems, Prague, Czech Republic.

O'Hare, G.M.P. 1996, Agent Factory: An Environment for the Fabrication of Distributed Artificial Systems, in O'Hare, G.M.P. and Jennings, N.R. (Eds.), Foundations of Distributed Artificial Intelligence: Sixth Generation Computer Series, Wiley Interscience.

Volz, S., and Klinec, D. 1999. Nexus: The Development of a Platform for Location Aware Applications, In Proceedings of Third Turkish-German Joint Geodetic Days – Towards A Digital Age, Istanbul, Turkey.

C. Muldoon, G.M.P. O'Hare, D. Phelan, R. Strahan, R.W. Collier 2003. "ACCESS: An Agent Architecture for Ubiquitous Service Delivery", In Proceedings Seventh International Workshop on Cooperative Information Agents (CIA), Helsinki, Finland.

Collier, R. 2001, Agent Factory: A Framework for the Engineering of Agent-Oriented Applications, *Ph.D. Thesis*, Computer Science Dept., University College Dublin, Éire .