The Role of Agents in Intelligent Mobile Services

Fernando Koch[1] and Iyad Rahwan[2]

[1] Institute of Information and Computing Sciences
Utrecht University, Utrecht, The Netherlands
fkoch@acm.org
[2] Department of Information Systems
University of Melbourne, Parkville, VIC, Australia
i.rahwan@pgrad.unimelb.edu.au

Abstract. In this paper we argue that the agent paradigm offers promising techniques for dealing with the challenges of building intelligent mobile services. We present Agent Oriented Software Engineering as a solution for the problems in designing a new generation of mobile services. To illustrate our position, we present a brief agent-oriented analysis of a mobile commerce scenario.

1 Introduction

Existing commercial mobile services are only scratching the surface of what is possible. As pointed out by the survey presented in [23], the main problems against technology adoption are lack of interest, unappealing services and communication costs. Hence, in a new generation of mobile services, human-device interaction must become more useful and concise, reducing the number of user interventions while providing appealing value.

The power of pervasive computing is unleashed when the application has the intelligence to process contextual information about the user and the environment in order to provide the user with the *right information at the right time*. A framework for building these applications must provide the means to handle the distribution inherent in the environment and, at the same time, allow the easy mapping of human knowledge into computer applications. We call this class of applications as *Intelligent Mobile Services*.

To address these requirements we must deal with new issues in the field of Distributed Computing and Artificial Intelligence. Mobile computing introduces new elements of complexity [20]: dynamic environments, changes in actual and relative location, constrained computing power, connectivity latency and unreliability, limited battery power, and constrained input and output interfaces. These constraints are not artefacts of current technology, but are intrinsic to mobility.

We argue that agent-based computing [25] is a promising enabling technology for second-generation mobile services. The agent paradigm offers methodologies for creating distributed, intelligent, integrated and cooperative applications. This paper tries to link the requirements of the new generation mobile services to the

solutions provided by agent-oriented software paradigm. This exercise serves as a guide for analysing existing systems and as a mean for identifying new opportunities.

This work is structured as follows: section 2 describes the requirements for mobile services and presents key definitions like context-awareness; the section 3 presents the agent technology and its relationship to the development of intelligent mobile services; section 4 presents the exercise of building an agent-based solution based on the presented ideas, then conclude in section 5.

2 Requirements

Mobile service provision imposes two major challenges: the *infrastructure challenge*, which is concerned with building robust hardware and software technology that facilitate mobile connectivity, location-identification, service discovery, fault tolerance, etc. [9]; second is the *services challenge*, which is concerned with how we can use the infrastructure available in order to provide new and useful services, such as trip planning or and mobile commerce [20]. In this study, our focus is on the latter. In particular, we are interested in high-level *intelligent services*, which take advantage of the processing power of mobile devices in order to provide users with context-aware support.

Zambonelli and Parunak [27] [26] argue that *situadedness*, *openness*, *locality in control* and *locality in interaction* are fundamental characteristics of future software systems. We argue that these characteristics are intrinsically related to known issues in mobile computing:

- **Situatedness**: This property means that the software system is situated in an *environment*, which it can influence and be influenced by. A mobile device aimed at providing support to the user in a dynamic environment would benefit from representing and processing information about this environment in order to provide appropriate support.
- **Openness**: This property refers to the system's ability to accommodate changes in the system structure as when, for example, new components enter the system or existing components leave. In the world of mobile services, new services and devices might appear and disappear due to changes in connectivity, user location, and because users' availability itself changes [16]. The software needs to adapt to such changes appropriately.
- **Locality in control**: This means that software components may be required to operate 'autonomously' based on local policies. In mobile services, this may be a necessity to ensure service robustness, for example if connectivity is lost when outside a coverage area. This may also be required for cost reduction because contacting a service on a centralised server might be expensive. To deal with this, devices must be capable of tracking their execution and interaction states when communicating to external services.
- **Locality in interactions**: This refers to a software component's ability to interact with other components in local geographical or logical neighbourhoods. In mobile services, such interaction is needed for reliability and

quality related reasons. Interaction between different mobile devices or between a device and other services takes place over unstable and unreliable communication channels, and hence must be endowed with the ability to recover such interaction appropriately in case of error.

A key feature of the above characteristics is *context-awareness*. *Context* is any information that can be used to characterize the situation of an entity [6]. Context-awareness involves the means to capture, represent and process context information. In Figure 1, we demonstrate the typical processes involved in a mobile service environment: collecting raw data about the device, user and network, combining this data into context information, and inferring some appropriate action to take.

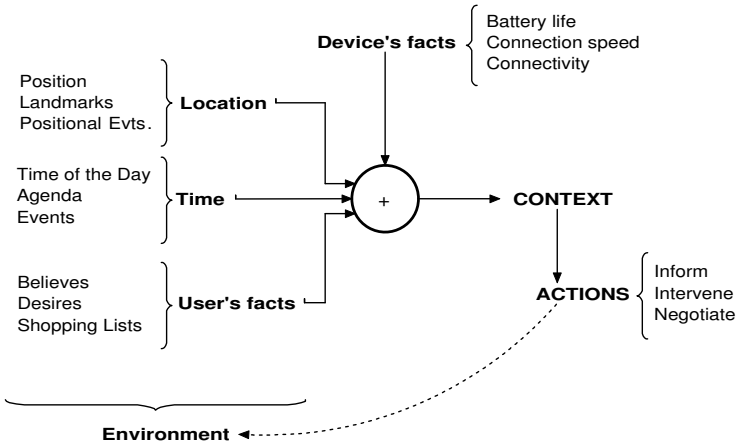

Fig. 1. Context-awareness in mobile services workflow

3 Agents and Mobile Services

Ultimately, the new generation of mobile services must deliver an enhanced user experience. The human-device interactions must become more concise, reducing the number of user interventions. Pervasive software applications should make use of local processing to reason about the user's context and predict user's intents, actions and location, behaving as an always present personal assistant, available anytime, anywhere. A framework for building these applications must provide the means to handle the distribution inherent in the environment and at the same time allow the easy mapping of human knowledge into computer applications.

3.1 The Role of Agents

Agent-based computing is becoming increasingly popular because it enables building modular software systems capable of operating in dynamic, unpredictable environments. The agent paradigm has produced a wide variety of concepts and tools for constructing sophisticated autonomous software and structuring high-level interaction patterns that facilitate cooperative behaviour [12]. In addition, a set of methodologies have been developed that enable system designers to distil domain knowledge and transform it into agent or multi-agent system specifications [10]. Hence, agents seem to offer a set of features that are very closely aligned with the requirements of service delivery challenge in pervasive computing.

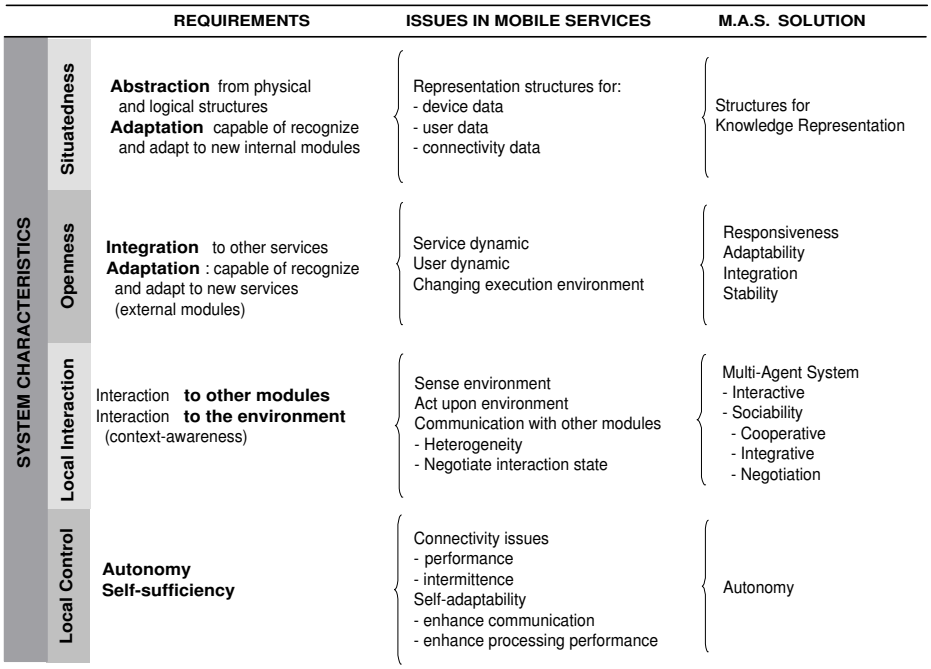

Fig. 2. Role of Agents in Mobile Services

In Figure 2 we summarise the characteristics of mobile services discussed in the previous section, and demonstrate how the *agent paradigm* can provide solutions to support these characteristics, and hence address the requirements of mobile computing. We detail this in the following:

– **Structures for knowledge representation:** Existing agent systems can provide an answer to the situadeness requirement. The agent paradigm has

produced a variety of methods for explicit representation of the environment, and for reasoning about this environment to produce decisions [24]. We denote, however, that agent systems are not the only paradigm to provide this ability. Nonetheless it is an intrinsic problem in multi-agent systems, and hence inherent in agent architectures.
- **Responsiveness and adaptivity:** Jennings and Wooldridge [13] pointed out that responsiveness and adaptivity are inherent features provided by agent systems; agents should be able to adapt to constantly changing execution environments, one of the predictable problems of mobile computing.
- **Sociability and locality of interaction:** As also pointed out by Jennings and Wooldridge [13], agents are able to interact with other agents or humans when needed. Sophisticated interaction mechanisms have been developed in the agent community to facilitate information exchange [4], coordination [8], collaboration [17], and negotiation [1]. Such mechanisms offer great potential to address the local interaction requirement in mobile service delivery.
- **Autonomy:** As we argued in an earlier paper [14], the agent paradigm offers mechanisms that address varying degrees of autonomy, from basic reactive architectures based on a set of pre-determined rules, to mechanisms for proactive behaviour [3] considering the context and user preferences and behaviours. This inherent feature is useful to configure agents to react to and plan for changes in a mobile service environment.

Another important feature of the agent paradigm is that it provides an infrastructure for *decomposition* and *abstraction* [11]. Jennings describes *decomposition* by stating that "the most basic technique for tackling large problems is to divide them into smaller, more manageable chunks, each of which can then be dealt with in relative isolation."

3.2 Relevant Aspects of Agents

Having outlined the different roles an agent can play in mobile service delivery, it would be useful to have a perspective through which to look at existing agent-based mobile applications and to recognise opportunities for further work.

Agent-based services can be analysed on several different dimensions. We find the following classification, due to Jennings et al [13], useful for our purposes. Agent systems can be classified according to the: *sophistication of the application*, from agents work based on well-defined, pre-specified rules and assumptions, to service performing and the more complex predictive/proactive agents; *role of the agent*, like decision support and problem solving automation, and; *granularity of the view*, from single-agents to more complex multi-agent systems. By correlating these aspects of agent-based applications and the roles of agents in mobile service applications, we came up with a taxonomy based on the aspects of *granularity*, *role* and *level of autonomy*, as described below. This taxonomy is one way to describe the different aspects of using agents in mobile service delivery:

- **Granularity**: An agent-based application can provide *single user* support or *multi-user* interaction support. The decision about whether to adopt a

single-agent or multi-agent approach is generally determined by the domain and is similar in nature to decisions about whether monolithic, centralized solutions or distributed, decentralized solutions are appropriate.

- **Role**: Agents running on mobile devices can play different roles, offering varying levels of support to the user. Sheridan [22] presents a taxonomy of four main stages of complex human-machine tasks: (a) *acquiring information*, where the application gathers information and saves it for future processing; (b) *analysing and displaying results* is where the application analyses and displays the collected information, considering the context and tailoring the output information to the user's preferences; (c) *deciding on an action or sequence of actions* means the application suggests actions based on the analysis of environmental information; and (d) *implementing decided actions*, where the application is responsible for carrying out the steps required for the task completion.
- **Autonomy**: An agent application may behave in a *reactive* manner to environment stimuli based on a set of pre-determined rules or user commands. Instead, an agent may act *proactively* considering the context and user preferences and behaviours. Different levels of autonomy exist between these two extremes, and the degree of this autonomy may be adjustable by the user [21].

Using the above taxonomy, we look at some existing agent-based mobile services in the next section.

4 Some Early Attempts

Several efforts in building applications that utilize agent-based techniques to provide services to mobile users have been conduced by other groups. Our aim is to provide the reader with a snapshot of current developments, and to describe these within the above taxonomy.

- **MyCampus** [19] is a Semantic Web environment for context-aware mobile services at Carnegie Mellon University. In this project, the agent-based approach provides autonomous discovery of services and personalized services based on users' preferences and contextual attributes. According to the taxonomy we have given above, MyCampus would be classified as *single-user support*, with the role of *acquiring information* to *analyse and display*. Essentially, MyCampus is a *reactive system*, as it works by responding to environment changes.
- **AbIMA** [18] is an agent-based intelligent mobile assistant that runs on a hand-held device and assists the user through the execution of individual tasks. AbIMA uses a set of pre-programmed plans for executing different tasks. These plans can be distilled from domain knowledge, or learned through observation of user behaviour. AbIMA offers agent-based support for a *single user*, acting by performing *analysis and displaying suggestions* to the user. It exhibits *proactive behaviour* by providing proactive advice to the user when things go wrong and initial plans cannot be executed.

– **Paurobally et al** [15] proposed an agent-based framework for providing personalised mobile services. Producers and consumers of services are viewed as software agents, some of which are located on users' mobile devices. These agents use negotiation as a mechanism for reaching agreement on the terms of a transaction in a mobile-commerce scenario. As per our taxonomy, this infrastructure would be classified as *multiple-user* support, as it supports a community of users and their inter-operations with the environment; the role would be *implement action* and on autonomy it would classify as *pro-active*.
– **The Electric Elves project**[2] exploits agent technology to support human organisation. Teams of agents help users conduct routine, well structured tasks, such as organising meetings. Each person has their own proxy agent running on a mobile device. The Electric Elves project is based on a multi-agent approach to coordinating multiple mobile users. Agents' roles range from *deciding what information to acquire*, to *analysing situation*s using decision-theoretic planning, to making suggested actions. Agents exhibit varying degrees of autonomy based on the type of situation at hand and learned user preferences

	Granularity		Role				Autonomy	
	Single	Multi	Acquire	Display	Decide	Implem.	React	Proact
Mycampus		X	X	X			X	
ElectricElves		X	X	X			X	X
Abima	X			X				X
Paurabally's		X				X		X

Fig. 3. Classification of early attempts

In Figure 3 we present which depicts the classifications given above.

5 Engineering Agents-based Mobile Services: Mobile Computing Example Scenario

Our long-term aim is to provide an agent-based approach to specifying and constructing intelligent mobile services. In this section, we take a preliminary step in this direction through a brief case-study. Our aim is not to give a comprehensive *methodology* but rather to outline a coherent sequence of steps for building agent-based intelligent mobile services and discuss the agent-based techniques

that can prove useful. The steps start from scoping the problem to defining the requirements and functional specifications, to defining the various processing components, processing rules and their interaction. We go through these steps through an illustrative scenario (depicted in Figure 4).

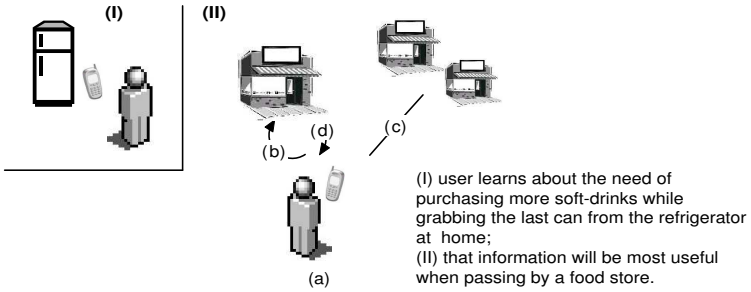

Fig. 4. Mobile commerce scenario

Define the problem

Purpose: *Provide a clear definition of the problem to guide subsequent steps.*

The problem in the proposed example is to create a context-aware mobile commerce application that utilises pervasive computing technologies in order to deliver the right information (shopping list quotation) at the right time (while nearby a grocery store).

Define the requirements and functional specification

Purpose: *Scope the aimed level of automation, assess the available enabling and support technology, what context-awareness support infrastructure is available, and describe the requirements from the service with regards to feedback loops, human-computer integration and group of participants.*

The requirements for running this mobile service are: (1) a location-based system, with landmark event generation, which detects the users device nearby a setup landmark and sends a notification; (2) the interface definition between the device-based application and the grocery store server; (3) connectivity; (4) plan resolution while the user is walking nearby the store; (5) interfacing to present the solution (e.g. price quote) to the user.

Considering these requirements, this applications architecture can be model as either client-server or local-processing applications, depending on the available device connectivity.

Decompose the problem

Purpose: *to define the autonomous entities in the problem and how they can be distributed. Decomposition is the most basic technique for tackling large problems is to divide them into smaller, more manageable chunks, each of which can then be dealt with in relative isolation, as described by Jennings [11].*

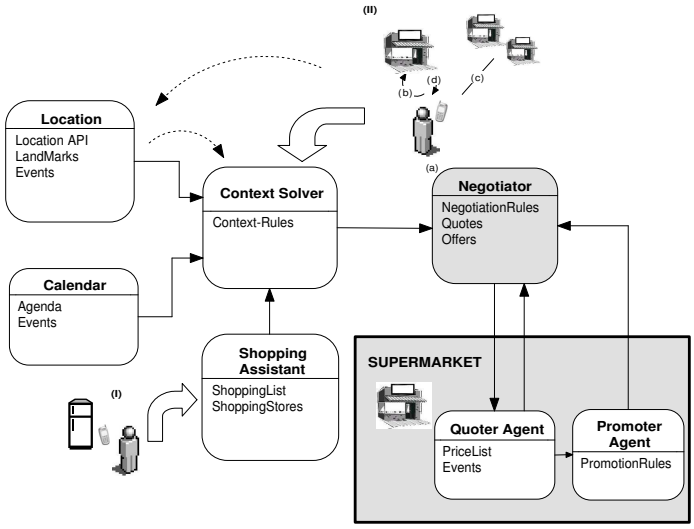

Fig. 5. Application architecture

By decomposing the problem into the *minimum* parts, we came up with an abstract sketch of a multi-agent system where each module addresses one specific part of the problem. The modules are presented in Figure 5 and described below.

- **Location Agent**: knows about the user's position and processes landmark events. Although at first glance it may look unnecessary to have an agent dedicated to location-based services, for the sake of this exercise we are decomposing the problem to the minimum components. This agent would contain the skills necessary to interface with location-based services, to handle the landmark table and to related positioning information (coordinates) to landmarks proximity.
- **Calendar Agent**: keeps track of user's appointments and provide the functions to check user availability;
- **Shopping Assistant Agent**: keeps track of user's shopping lists and preferred stores;
- **Context Solver Agent**: implements the context inferencing and triggers the selected actions;
- **Negotiation Agent**: implements the negotiation strategies and functions.

Define the processing rules

Purpose: *describing the process events and, specially for Mobile Services, the relationship between events, context and actions. For context-aware computing, these are the context inference rules, as described by Dey [5, 7].*

The *Context Solver Agent* receives the landmark event and must process it accordingly. For example, it should check if there is a shopping list available, if the user's agenda allows a visit to the nearby shopping store and, if possible, compare the received quote with the quotes provided by the user's preferred stores. Finally, it should present the quote along with a suggest of action to the user. This context resolution rule is presented in the pseudo-code below.

Event: Near a supermarket

EVENT -
location(near, supermarket(SUPER-A))
CONTEXT -
location(near, supermarket(SUPER-A)) AND
Agenda(available, next(30mins)), AND
ShoppingList(supermarket, SHOPPINGLIST))
ACTION -
setContext(NegotiatingList, SHOPPINGLIST),
Negotiate(SHOPPINGLIST, SUPER-A, QUOTE),
Store(quote(SUPER-A,QUOTE)),
ShoppingStores(supermarket, STORESLIST),
Negotiate(SHOPPINGLIST, STORESLIST, LISTQUOTES),
Store(quotes(STORESLIST, LISTQUOTES)),
Compare(QUOTE, LISTQUOTE, RESULT),
Display(RESULT)

Define the modules interdependencies and interoperations

Purpose: *This phase leads to the distribution problem and the developer must decide where each module will execute (server-based, client-server, local-processing and/or mobile code) and the interactions between the modules.*

In this environment, when the user passes by a food store the *Location Agent* detects the landmark and throws an event to the *Context Solver Agent*. The system checks for time availability and presence of a shopping list. If there is a positive context (i.e. if a certain condition is satisfied based on the observation of the environment), the system triggers the negotiation process, which will interact with the supermarkets' agent, bargain and display the best price quote it could find. The modules interdependencies are presented in the Figure 5.

Exercise conclusions

Other steps include lower level definitions such as the specification of the graphical user interface, aspects of human-computer interactions, application program-

ming interfaces to access contextual data and connectivity and networking issues. These stages are essential for the development of the final product but as they enter into lower-level details and technological issues we regard them as beyond the scope for this paper.

Several other factors could be improved and extended in this architecture sketch, such as perception of device and network conditions. For example, if the battery level is low or the network is expensive or unreliable then the agent may not conduct the time-consuming price bargaining process. This would address an extra requirement in mobile computing: *local control*.

In our classification system, the *Personal Assistant* described in the above scenario would be classified as a *single-user support* for granularity, has the role of *suggesting action* and is *pro-active* in terms of autonomy. Several other possibilities are feasible, and here are some examples:

- *Acquire information, analyse and display* and *reactive*: The agent could be reactive in the sense that it only responds to an explicit request for quotes by the user based on the given shopping list and, once processed, the quote information is acquired and displayed to the user. This is a rather simplistic agent and would not require location-based context-awareness infrastructure.
- *Decide action* and *reactive*: The user can delegate the act of bargaining and finalising a deal to the agent. For instance, the agent collects the quotes from the nearby and remote store. It also checks for alternative brands that the user configured as acceptable and compile the best quote from every store, based on what user preferences. The agent will work to find the best deal possible to the customer and direct him to a particular food store to shop. This solution requires connectivity but, since the action is initiated by the user, does not require a location-based service that proactively initiates interaction via detecting user location. This scenario can be engineered using a client-server architecture with the context rules processing in the server.
- *Decide action* and *proactive*: In this variant, the user's agent starts to bargain by itself when it detects the food store's proximity and if the stored shopping list and other scheduled activities permit (it wouldn't make sense to spend processing power and communication if the user doesn't have time availability for shopping at that time). Moreover, the proativiness could also take place on the *merchant side*, for example, as the merchant's agent detects the customer's proximity and the amount it intends to shop, it could provide a discount coupon valid for a period of time, teasing this customer to step in. Of course, this raises privacy issues worthwhile of serious separate investigation.

Although simplistic, this exercise demonstrates some solutions provided by the agent-oriented approach, and provides the reader with a hands-on feel for building a mobile service using agent-oriented software engineering.

6 Conclusions

We hope to have demonstrated that the agent paradigm provides useful tools and abstractions for addressing the requirements of the new generation of mobile services. We also hope that our classification would help future research in the area by enabling the identification of opportunities to extend existing efforts. We described a recipe for engineering agent-based mobile applications and presented a quick hands-on exercise utilizing agents for mobile services.

The use of context information will be important for mobile services. Agent-based software development could provide the structures for building context-aware systems. It is important to create better methods to collect, represent and process contextual information. Agent-oriented approaches can supply the tools for such development.

Our future work includes extending our classification model and use it to analyse other existing systems. We also plan to engineer more elaborate mobile service architectures based on these classifications. Finally, we will work on the implementation of the proposed examples, creating proof-of-concept applications.

Acknowledgement

This work was conducted while Fernando Koch was a visitor at the Department of Information Systems, University of Melbourne.

References

1. M. Beer, M. d'Inverno, M. Luck, N. Jennings, C. Preist, and M. Schroeder. Negotiation in multi-agent systems. *Knowledge Engineering Review 14*, pages 285–289, 1999.
2. H. Chalupsky, Y. Gil, C. A. Knoblock, K. Lerman, J. Oh, D. V. Pynadath, T. A. Russ, and M. Tambe. Electric elves: Applying agent technology to support human organizations. In H. Hirsh and S. Chien, editors, *Proceedings of the 13th International Conference of Innovative Application of Artificial Intelligence (IAAI-2001)*. AAAI Press, 2001.
3. M. Dastani, F. Dignum, and J.-J. Meyer. Autonomy and agent deliberation. In M. Rovatsos and M. Nickles, editors, *The First International Workshop on Computatinal Autonomy - Potential, Risks, Solutions (Autonomous 2003)*, pages 23–35, Melbourne, Australia, July 2003.
4. F. de Boer, R. M. van Eijk, W. van der Hoek, and J.-J. Meyer. A fully abstract model for the exchange of information in multi-agent systems. *Theoretical Computer Science*, 290(3):1753–1773, 2003.
5. A. K. Dey. *Providing Architectural Support for Building Context-Aware Applications*. PhD thesis, Georgia Institute of Technology, November 2000.
6. A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. Technical Report GIT-GVU-99-22, College of Computing, Georgia Institute of Technology, June 1999.

7. A. K. Dey, G. D. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human Computer Interaction Journal*, 16:97–166, 2001.
8. E. H. Durfee. Practically coordinating. *Artificial Intelligence Magazine*, 20(1):99–116, Spring 1999.
9. K. Henricksen, J. Indulska, and A. Rakotonirainy. Infrastructure for pervasive computing: Challenges. In K. Bauknecht, W. Brauer, and T. A. Mck, editors, *Informatik 2001: Wirtschaft und Wissenschaft in der Network Economy - Visionen und Wirklichkeit*, volume 1 of *Tagungsband der GI/OCG-Jahrestagung*, pages 214–222, Universitt Wien, September 2001.
10. N. R. Jennings. Agent-oriented software engineering. In F. J. Garijo and M. Boman, editors, *Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World : Multi-Agent System Engineering (MAAMAW-99)*, volume 1647, pages 1–7, Spain, 30– 2 1999. Springer-Verlag: Heidelberg, Germany.
11. N. R. Jennings. An agent-based approach for building complex software systems. *Commun. ACM*, 44(4):35–41, 2001.
12. N. R. Jennings, K. Sycara, and M. J. Wooldridge. A roadmap of agent research and development. *Journal of Autonomous Agents and Multi-Agent Systems*, 1(1):7–38, 1998.
13. N. R. Jennings and M. J. Wooldridge. Applications of intelligent agents. *Agent technology: foundations, applications, and markets*, pages 3–28, 1998.
14. F. Koch and I. Rahwan. Classification of agents-based mobile assistants. In *Proceedings of the AAMAS Workshop on Agents for Ubiquitous Computing (UbiAgents)*, New York, USA, Jul 2004.
15. S. Paurobally, P. J. Turner, and N. R. Jennings. Automating negotiation for m-services. *IEEE Trans. on Systems, Man and Cybernetics (Part A: Systems and Humans)*, 33(6):709–724, 2003.
16. G. P. Picco, G.-C. Roman, and A. L. Murphy. *The future of Software Engineering*, chapter Software Engineering for Mobility: A Roadmap, pages 241–258. ACM Press, 2000.
17. D. Pynadath and M. Tambe. Multiagent teamwork: Analyzing key teamwork theories and models. In C. Castelfranchi and L. Johnson, editors, *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2002)*, pages 873–880, New York, USA, 2002. ACM Press.
18. T. Rahwan, T. Rahwan, I. Rahwan, and R. Ashri. Agent-based support for mobile users using agentspeak(l). In P. Giorgini, B. Hederson-Sellers, and M. Winikoff, editors, *Agent Oriented Information Systems*, Lecture Notes in Artificial Intelligence. Springer Verlag, Berlin, Germany, 2004.
19. N. Sadeh, T.-C. Chan, L. Van, O. Kwon, and K. Takizawa. Creating an open agent environment for context-aware m-commerce. In Burg, Dale, Finin, Nakashima, Padgham, Sierra, and Willmott, editors, *Agentcities: Challenges in Open Agent Environments*, LNAI, pages 152–158, Heidelberg, Germany, 2003. Springer Verlag.
20. M. Satyanarayanan. Pervasive computing: vision and challenges. *IEEE Personal Communications*, 8(4):10–17, 2001.
21. P. Scerri, D. Pynadath, and M. Tambe. Towards adjustable autonomy for the real-world. *Journal of AI Research (JAIR)*, 17:171–228, 2002.
22. T. B. Sheridan. Rumination on automation. In *Proceedings of 7th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems*, Kyoto, Japan, 1998. Oxford: Pergamon Press. Plenary address.

23. M. Uncapher. M-commerce e-data: Jupiter media metrix says mobile transactions to comprise only a sliver of all online shopping. *ITAA E-LETTER*, page 2, July 2001.
24. G. Weiss. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. The MIT Press, 1999.
25. M. J. Wooldridge. *An Introduction to MultiAgent Systems*. John Wiley & Sons, Chichester, England, 2002.
26. F. Zambonelli and H. V. D. Parunak. Towards a paradigm change in computer science and software engineering: a synthesis. *The Knowledge Engineering Review*, 2004. (to appear).
27. F. Zambonelli and H. Van Dyke Parunak. From design to intention: Signs of a revolution. In *Proceedings of the First International Joint Conference on Autonomous agents and Multiagent Systems*, pages 455–456. ACM Press, 2002.