CONTEXT-AWARE COMPUTING FOR BUSINESS TRAVELER CARE

Patrick Hertzog, Marc Torrens, Pearl Pu and Boi Faltings
Institute of Core Computing Science
Swiss Federal Institute of Technology (EPFL)
Lausanne, Switzerland

ABSTRACT

Travel has many situations where context-aware computing can bring important benefits: pointing out notorious delays or bad weather during the planning phase, allowing the user to replan for handling unexpected situations, or suggesting flight alternatives to avoid strikes.

In this paper, we describe an approach for integrating context-aware computing to a mobile travel assistant. Travel plans, generated using booking information and meeting descriptions, are enriched within compact and powerful structures, called User Task Models. These structures are transferred to a mobile device enabling the support for the traveler during his trip. The overall result is what we call an Intelligent Mobile Travel Assistant.

KEYWORDS

travel planner, context-aware computing, pervasive computing, mobile assistants.

1. INTRODUCTION

Selling complex products such as travel requires intensive customer care. Customers have needs such as rebooking in case of missed flights or changes in plans, suggestions of suitable hotels in strange towns, or finding suitable transportation in unfamiliar places. This is an area where information technology, accessible anytime and from anywhere, can provide valuable new services to people. We believe that current IT fails to address this task because it is still geared to providing a uniform service to large user populations. What is required for customer care in travel are differentiated, customer-specific services, taking into account personal profiles, history and the circumstances in which they are invoked. We call these *context-aware* services.

1.1 E-commerce for the Travel Industry

Travel has been identified early on as one of the most promising sectors for electronic commerce. First generation e-commerce platforms consisted basically of a web-based interface to the reservation systems, called Global Distribution Systems (GDS), used by travel agents. These systems carried over many of the limitations of the mainframe systems, for example the fact that it was impossible to go back to earlier steps in the process. Since all users were channeled through a single system, there were frequent crashes due to interaction of different processes. In 2001, two players (ITA software and Expedia) have launched new technology where the search for itineraries and prices is handled separately for each user, similarly to the SmartClient technology [Torrens02a, Torrens02b, Torrens03]. i:FAO is the first company to have announced this new generation technology for the business travel market. In contrast to existing tools, which allow optimizing only a small and predefined set of preferences, *reality* [Torrens03] allows a wide variety that can accurately model the preferences of different customers. It uses constraint satisfaction techniques to compute the solutions and advanced usability techniques to visualize the results in a mixed-initiative system. The new architectures create the potential for greatly improved service to the end-user. These technologies currently offer a great support for the traveler during flight ticket reservation processes.

New emerging technologies and platforms are enabling software to support the user not only during the booking process, but also during the trip. In order to accomplish that, a suitable research area is the context-aware computing paradigm briefly described in the following section.

1.2 Context-Aware Computing Survey

Context-aware computing generally refers to applications (usually running on a handheld computing device) that can discover and take advantage of contextual information [Chen00]. Active context-aware applications can adapt their behavior according to context and take action in an autonomous manner. Passive context-aware applications, only present context to the user and leave them the option to decide what to do with the context. As an example of such passive applications, most airlines now offer to alert travelers of delayed flights through SMS messages to their mobile phones. In the early days of this research area, most of the context-aware systems dealt with only location context, thus they were called location-based systems. Weiser [Weiser91] and Schilit [Schilit94] pointed out some extensions to the context concept, including other contextual information. [Schmidt98], [Chen00] and [Dey00] extended the location context notion to include the following aspects: computing context (connectivity, resources, displays), user context (user profile, current location, social situation), physical context (ambient conditions, lighting, noise) and time context.

Applications of context-aware systems include: Personal Assistants [Asthana94, Fano98, Long96, Abowd97, Youll00], Information Delivery Services [Marmasse00, Pascoe98] and also Office and Conference Assistants [Yan00, Dey99].

2. THE TRAVELER OF THE FUTURE

Our vision for the traveler of the future is that he will be continuously assisted by a mobile device during his trip, as illustrated in the following scenario.

John is a businessman living in Geneva, Switzerland, and he has to attend a conference in Las Vegas. The day prior to the trip, the system warns him through his PDA that pilots have announced a strike. As it is not sure that his flights will not be canceled, the system proposes another flight combination according to the preferences John set during his initial search.

On the morning of his departure, an alert is sent to John's PDA advising him to leave earlier for the airport because of a traffic jam due to an accident on the highway going to the airport. As soon as he arrives at the airport, some information is available to him: flight schedule, flight number, airline, terminal, gate, airport maps, facilities, etc.

When arriving in Las Vegas, the system advices John to take a taxi to go to the hotel he booked instead of using public transportation. It displays the hotel name and address, so he can tell the taxi driver the exact destination. Later, it informs John that the following day, he should leave the hotel by 8:30 in order to be on-time for the conference and that it is more convenient to get there by walk as it is close to the hotel. It also gives him the latest weather forecast for the next day. On the morning, shortly before he leaves the hotel, a map displays the route from the hotel to the conference center. During his stay, John can access several types of information: for example, he can ask the system for a restaurant and it proposes some according to his preferences near his hotel or near the conference center.

On the evening before his flight back, he is alerted that heavy snow has been forecasted in Chicago for the following day and that it may cause delays for his connecting flight. The system proposes him another flight to Geneva and asks him if he would like to change his reservation. John prefers to keep his reservation because the risk of delay is not worth the additional cost it implies. It informs him that he should leave the hotel by 8:00 in order to be early enough at the airport for the check-in.

In order to enable technology supporting travelers as described in the above scenario, compact and powerful encoding techniques are needed. Our modeling approach is based on enriched travel plan trees. We call these User Task Models.

3. USER TASK MODEL

When a business traveler plans a trip, he is actually trying to map goal activity requirements such as meeting schedules into a feasible plan, for example a sequence of flights and hotels for each destination. Once the user has selected the preferred flight combination for his trip, the user's plan is stated. In that sense, a travel plan is defined as following:

DEFINITION 1. *A* Travel Plan *is a collection of* n *goal activities* $\{G_1, \ldots, G_n\}$. *A* Goal Activity *G is mainly defined by a location loc(G), a start date/time start(G), an end date/time end(G), and the activity itself activity(G) (*e.g.*, meeting, conference, seminar, etc.). The following restriction is made:* $\forall i,\ 0 \leq i < n$, $end(G_i) \leq_{prev} start(G_{i+1})$, *where* $\leq_{prev}$ *is the time precedence operator.*

Moreover, a goal activity is decomposed in a sequence of required tasks (e.g.*, drive to airport, overnight at hotel, etc.) called means tasks. These tasks are needed to accomplish a goal activity and can be recursively decomposed in sub-tasks.*

Since a travel plan can be seen as a hierarchy of goal activities and means activities, we naturally represent it as a tree. A tree representation of a travel plan, called *travel plan tree* has the following characteristics:

- The root node contains general trip information, such as global preferences, itinerary description, and so on.
- Children of the root node represent the user's goal activities.
- Child nodes of a goal activity node represent its means activities. At the same time, these means activities can be represented recursively by another sub-plan which forms the sub-tree of the node.

Nodes at the same level are chronologically ordered by their *start* and *end* functions. The *start* of the first child of a node *n* cannot be anterior to the *start* of node *n*. Similarly, the *end* of the last child of a node *n* cannot be posterior to the *end* value of node *n*. A travel plan tree would be enough if the traveler's plan does not change, but in practice, this is often not the case. Travel plans change because unexpected situations emerge, and context information is unpredictable. As explained below, the usage of an annotated workflow empowers significantly the support of the traveler.

DEFINITION 2. *An* Annotated Workflow *of a travel plan is the lineal execution of the different tasks following a depth-first traversal of the travel plan tree. The execution of each task is triggered by* time*.*

Each node of the tree includes annotations *with basically two kinds of information related to the corresponding task: a) server requests to retrieve the needed contextual information, and b) personal user preferences.*

Server requests are usually converted by the server into web service calls as indicated in Section 4. Note that annotations in the workflow are crucial for capturing the current context of user's activities. In that sense, the server requests of a task act as sensors of the current user's state, *e.g.*, traffic situation at the highway from home to the airport. Without these annotations the only contextual information is the current user's location and current user's task which can be deduced from the travel plan and current time. On the other hand, personal preferences are used by proactive services to accurately assist the traveler according to his personal needs.

DEFINITION 3. *A* User Task Model (UTM) *is a travel plan tree with its annotated workflow. A User Task Model is* activated *when current time matches with* start(n)*, where* n *is the first (bottom left) leaf of the tree. On the other hand, a User Task Model is* deactivated *once current time matches with* end(n)*, where* n *is the last (bottom right) leaf of the tree.*

Our notion of workflow implies a lineal execution of tasks, *i.e.*, tasks are executed one by one and never in parallel. In the case we would be interested in modeling some parallelism for a traveler, *e.g.*, two tasks executed at the same time range, two different task models can be considered.

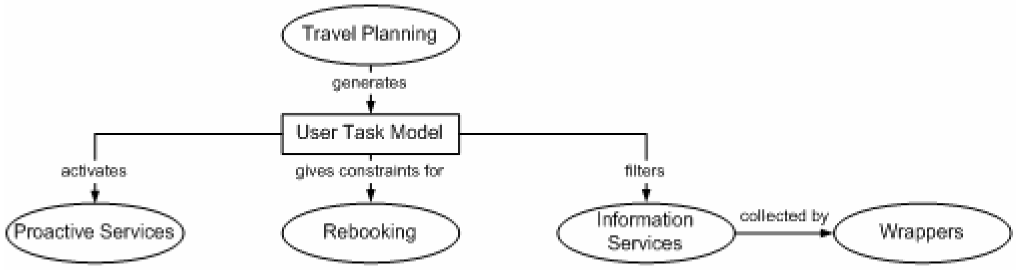

Figure 1. The User Task Model handles the plan execution of the traveler considering time and location contexts, dynamic and personal information.

As shown in Figure 1, each UTM handles the plan execution for a traveler using the system. UTMs are expressed as XML documents according to a XML Schema for better interoperability among internal or external components of the system. Moreover, in this way, UTMs can be originated and used by different software components in a convenient way.

3.1 Generation of User Task Models

The UTM is created by combining booking information, *i.e.*, selected products (flights, hotels, rental cars) with their associated user preferences, and meeting information. This process enables building the travel plan with its tree representation, and also adding the appropriate annotations for the workflow. For example, the task *[Go from Manhattan to JFK Airport]* could be associated with a server request in order to inform the user about traffic jams in New York City. Such server request accesses web services to discover traffic information in New York City. An example of preference for that task could be that the traveler prefers going to JFK Airport by taxi rather than by public transportation.

3.2 Replanning, Proactive Services and Filtering Information

In practice, business travel plans often change because unexpected situations and new requirements occur. For example a meeting can last longer than initially planned, and therefore the user may want to rebook his next flight. In that sense, **replanning** is triggered by the traveler. The User Task Model is then used to select the task of the travel plan that needs to be replanned. Actually, the replanning must take place at the parent node of the selected task, in order to guarantee a coherent resulting model. The annotations of the affected task, especially task's preferences, are then used by the constraint-based engine (same solving engine as used by *reality* [Torrens03]) to propose a small set of possible alternative tasks. Once the user has selected his preferred alternative, the User Task Model is rebuilt consistently considering the appropriate changes.

Proactive services are activated by tasks with annotations representing server requests to those services. For instance, a task *[Flight from JFK Airport to GVA Airport]* could have an annotation indicating that an automatic check-in service must be executed. Other examples include to weak-up the user automatically in order to get on time to a meeting. When the user replans part of his UTM, the system can also alert the concerned people. For example, when the user misses a flight, an email could alert the following meeting participants about the user's delay.

Workflow annotations of UTMs are also used to channel relevant information to the user. For example, one could consider annotations for **filtering information** related to the current user location about: political news, weather forecasts, strike announcements, traffic jams, and so on.

Moreover, UTMs can be used for a variety of different reporting activities. For example, it can be used to generate templates for meeting reports. In this manner, the traveler would have a meeting template (with location, time, description, etc.) for each meeting during his trip. Another interesting usage of the User Task Model would be to ease the production of travel expenses reports. All these usages of UTMs can be simply seen as different views of UTMs, simplifying the amount of different data structures of the system.

4. ARCHITECTURE

The central piece of the whole system is the *reality* server (see Figure 2). It basically acts as a gateway between *Pocket reality* and a set of services and information sources available as web services. One of these services is the i:FAO back-end. It is used to:

1 retrieve availabilities through the GDS. Real-time availabilities are always used to ensure that there is a seat available in the proposed flights.
2 price the suggested alternatives.
3 change the booking of the user according to the alternative he selected.

Note that the last two functionalities are not yet available in the back-end and therefore have not been included in the prototype described in Section 5.

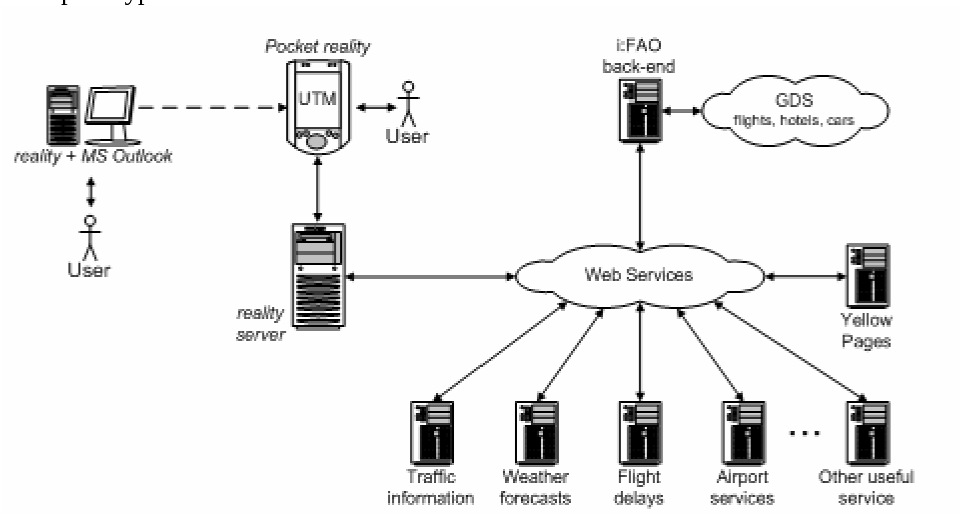

Figure 2. *reality* and *Pocket reality* access a variety of web services through the *reality* server.

Other information sources may include weather forecasts, flight delays, traffic jams and so on. The main advantage of accessing them as web services is that the information is provided in a structured way which is understandable by software. Another advantage is the automatic service discovery: one can imagine that, in the future, the *reality* server can automatically search for the needed services through a kind of *Yellow Pages* directory.

The communication between the server and *Pocket reality* uses a proprietary format which has the advantage of being very compact. Moreover the stream is compressed in order to minimize the transferred amount of data. This is especially important when working with mobile device that may use low bandwidth networks (*e.g.*, GSM). It also implies a significant cost reduction when the transmission is charged by amount of data (*e.g.*, GPRS).

5. THE PROTOTYPE

The goal of this prototype is to take care of the user during his trip. Therefore a logical way to interact with him is to use a mobile device. Many different mobile devices are hitting the shelves nowadays; they can be divided in four main families: 1) Notebooks, 2) Mobile phones, 3) Pagers, and 4) PDAs.

Some recent products (*e.g.*, Sony Ericsson P800) indicate that the mobile phones and the PDAs may converge to a single device and this will become especially true with third generation phones. It has been decided to focus the development of the prototype on the PDA for the following reasons: a) it offers a great tradeoff between size and performance, b) It is well spread among the target users, *i.e.*, the business travelers, c) it emulates quite well the capabilities of next generations of mobile phones, and d) it offers multiple connectivity possibilities, *e.g.*, WiFi, GPRS/GSM via a Bluetooth phone.

5.1 Challenges and Goals

Developing an application for the PDA raises both technical and usability challenges. Firstly, the performance and the memory size of a PDA cannot be compared with a desktop PC. Secondly, the screen is much smaller and the interaction model is slightly different.

Besides those obvious issues, another usability factor has to be taken into account: the social context. It is not conceivable to suppose that the user is in the same environment as if he was in front of his desktop PC. Let's imagine a business man wanting to rebook his next flight because his current meeting is going to last longer. It is probable that he will do it during a short break or even during the meeting. Therefore his attention is less focused and there is a much bigger probability that he is disturbed by an external element. This means that the interaction between the user and the application should be minimized: this way, the user will spare a considerable time and the risk he makes a mistake is minimized.

Having a User Task Model as described above can help a lot to minimize the interaction. It contains elements, *e.g.*, time constraints, preferences, that the user has already stated. Therefore, when he would like to change the plan, he does not have to care to state them again. They will be automatically taken into account.

The developed prototype, called *Pocket reality*, tries to overcome those challenges. It basically allows the user to replan parts of his trip, *i.e.*, meetings or flights, in an easy and efficient way. The interaction is based on three key phases that are detailed in the following sections. First of all, the User Task Model is visualized to give the user a global overview. Then the user can trigger a replanning depending on what has changed in his plan. Finally, if significant changes have to be made to respect the new plan, a set of alternatives are proposed and the user is then invited to choose his preferred one.

5.2 Visualizing the User Task Model

The visualization designed for the User Task Model is based on schedule. Each state of the task model is represented by an item displayed chronologically from top to bottom (see Figure 3). To improve the readability, the items are grouped by days. As it looks like visualizations found in agenda software, the users are used to it and understand it well.

Figure 3. The User Task Model is displayed chronologically. Past states are grayed out while the current one is highlighted. The user *tap-and-holds* on a meeting in order to change its end time. *Pocket reality* proposes him alternatives matching the new end time.

There are two types of items, *i.e.*, one for the flights and one for the meetings. They can be easily identified by their background color (respecting the same color schema as in *reality*) and both are structured in a similar way:

- On the left part, start and end times are displayed respectively at the top and bottom of the item.
- The right part of the item displays essential information about the corresponding state. For instance, on the items representing flights are displayed information like airports, terminal, airline and flight number.

To facilitate the navigation through the task model, the past items are grayed out and the current one is highlighted.

5.3 Triggering the Replanning

If something changes in the user's plan (*e.g.*, missed flight, canceled meeting and so on), he has to tell it to the system, so *Pocket reality* can replan the corresponding part of the trip and take the appropriate actions. This is done by using contextual menus that are displayed by *tap-and-holding* on the concerned item (described in Section 5.2). The menus are dynamically generated to take into account the context and propose changes that make sense (see Figure 3). For example:

- if the user is going to fly with British Airways, he may want to rebook in order to fly with another airline (*e.g.*, in case of strike).
- if, according to the schedule, the user should be in a plane and he uses *Pocket reality* to modify the flight, it is probable that he wants to tell that he missed it.
- if a meeting has already begun, it makes no sense to allow the user to change its start time.

Once the user expressed the needed change, *Pocket reality* replans the corresponding part of the trip. If that change implies modifications of further activities or conflicts with other, the user will be warned. For instance, a meeting lasting longer may prevent the user to catch his next plane. In that case, he will be asked if he want to take a flight leaving later. If the answer is positive, *Pocket reality* will retrieve the current availability information and propose a set of suitable alternatives.

5.4 Selecting the best Alternative

Once the current availabilities are received from the GDS, the best alternatives are computed by a Constraint-Based engine, as described in [Torrens02a]. It takes into account all the necessary information available from the User Task Model such as time constraints and user preferences. The engine ensures that the suggested alternatives fully respect the time constraints (*e.g.*, the user will arrive early enough for his next meeting) and match the user preferences. Therefore it is highly probable that those alternatives satisfy the user. However, if the user is not satisfied with the suggested alternatives, he has the ability to refine the search by adding (or removing) preferences. This process uses the same philosophy as *reality* [Torrens03]: the user can criticize each attribute of the flights, *i.e.*, time, airline, connecting airport, and new alternatives are automatically computed and proposed. Unlike in *reality*, the critic is done via contextual menus that are displayed by *tap-and-holding* on the corresponding attribute. As shown in Figure 3, the screen is divided in two parts. The best alternatives are displayed on the upper part and on the lower part are shown the preferences. There are two types of preferences: the ones given by the user either via his User Task Model or by criticizing the alternatives and the ones coming from the corporate profile. Once the user is satisfied, he can definitively replan his trip by tapping on the *Book it* button. The resulting User Task Model is then displayed as described in Section 5.2.

6. CONCLUSION

Techniques from Artificial Intelligence have been applied in the travel planning domain [Linden97, Torrens02a, Torrens03] in order to better assist travelers during the booking process. The next step is to actively support the traveler *en-route*. Current mobile devices (such as PDAs) enable advanced software development for such purpose.

However, some challenges must be faced in order to truly offer a real support for the traveler. These challenges are related to context-awareness and usability issues. Our approach to overcome such difficulties is based on data structures, called *User Task Models*, providing an explicit record of user activities and intentions that serves as a basis for: a) filling in context in user-initiated actions, and b) activating relevant proactive services.

Thus, User Task Models include travel plan information while providing context-awareness in a succinct way. Finally, we have presented a working prototype which uses UTMs for replanning purposes. Future work for the prototype includes: proactive services integration, relevant information channeling and UTM views development to ease travel reporting.

REFERENCES

Abowd, G. D. et al, 1997. Cyberguide : A Mobile Context-Aware Tour Guide. *ACM Wireless Networks*, Vol.3, No. 5, pp. 421-433.

Asthana. A. et al, 1994. An indoor wireless system for personalized shopping assistance. *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*. Santa Cruz, California, USA, pp. 69-74.

Chen, G. and Kotz, D., 2000. A Survey of Context-Aware Mobile Computing Research. *Technical Report TR2000-381*. Department of Computer Science, Dartmouth College, USA.

Dey, A. K. et al, 1999. The Conference Assistant: Combining Context-Awareness with Wearable Computing. *Proceedings of the Third International Symposium on Wearable Computers (ISWC 1999)*. San Francisco, California, USA, pp. 21-28.

Dey A. K. and Abowd G.D., 2000. Towards a better Understanding of Context and Context-awareness. Proceedings of the CHI 2000 Workshop on “The What, Where, When, Why and How of Context-Awareness”.

Fano, A. E., 1998. Shopper’s Eye: Using Location-based Filtering for a Shopping Agent in the Physical World. *Proceedings of the Second International Conference on Autonomous Agents (Agents 1998)*. Minneapolis/St. Paul, Minnesota, USA, pp. 416-421.

Linden, G. et al, 1997. Interactive Assessment of User Preference Models: The Automated Travel Assistant. *Proceedings of the Sixth International Conference on User Modelling*. Chia Laguna, Sardinia, Italy.

Long, S. et al, 1996. Rapid prototyping of mobile context-aware applications: the Cyberguide case study. *Proceedings of the Second Annual International Conference on Mobile Computing and Networking*. White Plains, New York, USA, pp. 97-107.

Marmasse, N., 2000. Location-aware information delivery with commotion. *Proceedings of the Second International Symposium on Handheld and Ubiquitous Computing (HUC)*, Bristol, UK, pp. 157-171.

Pascoe, J. et al, 1998. Developing personal technology for the Field. *Personal Technologies*, Vol. 2, No. 1.

Schilit B., et al, 1994. Context-Aware Computing Applications. IEEE Workshop on Mobile Computing Systems and Applications.

Schmidt, A. et al, 1998. There is More to Context than Location: Environment Sensing Technologies for Adaptive Mobile User Interfaces. *Workshop on Interactive Applications of Mobile Computing (IMC 1998)*, Rostock, Germany.

Torrens, M. et al, 2002. SmartClients: Constraint satisfaction as a paradigm for scalable intelligent information systems. *CONSTRAINTS: an international journal*, Vol. 7, pp. 49-69.

Torrens, M., 2002. Scalable Intelligent Electronic Catalogs. PhD thesis, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland.

Torrens, M. et al, 2003. *reality*: a Scalable Intelligent Travel Planner, Decision Support for the Business Traveler. *Proceedings of the Eighteenth Annual ACM Symposium on Applied Computing (SAC 2003)*, Melbourne, Florida, USA, pp. 623-630.

Want, R. et al, 1992. The Active Badge location system. *ACM Transactions on Information Systems*, Vol.10, No. 1, pp. 91-102.

Want, R. et al, 1996. *Mobile Computing*, chapter The ParcTab Ubiquitous Computing Experiment, Kluwer Academic Publishers.

Weiser, M., 1991. The computer of the 21st Century. Scientific American 265(3):66-75.

Yan, H. and Selker, T., 2000. Context-aware office assistant. *Proceedings of the 2000 International Conference on Intelligent User Interfaces*, New Orleans, Louisiana, USA, pp. 276-279.

Youll, J. et al, 2000. Impulse: Location-based Agent Assistance. *Proceedings of the Fourth International Conference on Autonomous Agents (Agents 2000)*, Barcelona, Catalonia, Spain.