



AskMe, Adaptive Context-based Expert Discovery Mechanism for Information Retrieval in Mobile Communities

Alban Hessler
Ch. du Braillon 18
1095 Lutry
Switzerland

SSC

Master Thesis

February 2006

Supervisor
Mr. Miquel Martin
NEC / ISV

Supervisor
Mrs. Oana Jurca
EPFL / LSIR

Professor
Prof. Karl Aberer
EPFL / LSIR

Abstract

This Master thesis presents the design and implementation of AskMe, a new type of service for mobile telephony, within the area of ubiquitous computing. It is a novel approach to collaborative information retrieval, based on using the context of a community to discover and enquire the most qualified experts than can answer the questions of a subscriber. The work is focused on privacy issues, distributed systems, Bayesian networks and incentive-based cooperation. AskMe is built on a highly modular, reusable architecture, accessible through a Web Service interface. A prototype has been implemented in Java to demonstrate the feasibility of the system, which has been tested with simulations, as discussed in the Evaluation section.

Contents

1	Introduction	1
1.1	Background	1
1.2	MobiLife	4
1.3	Business Model	5
2	Related Work	7
3	Requirements & Design Decisions	9
3.1	Motivations	9
3.2	Adaptivity	10
3.2.1	Feedback gathering for machine learning	10
3.2.2	Natural Language Processing	10
3.3	Distributed Architecture	10
3.3.1	Web Services	10
3.4	Privacy	11
3.4.1	Architecture	12
3.4.2	Pseudonyms	13
3.4.3	Cryptography	13
3.5	Incentive	13
3.5.1	Credits	14
3.5.2	Reputation	14
3.6	Context Awareness	14
3.7	Use Cases	14
3.7.1	Looking for a special place	15
3.7.2	Answering a query	17
3.7.3	Receiving an answer and rating it	18
3.7.4	Earning reputation	20
4	Design and Architecture	21
4.1	Overall Architecture	21
4.1.1	Client Software	22

4.1.2	Entity Manager	22
4.1.3	Context Providers Network	23
4.1.4	Context Provider Anonymizer	24
4.1.5	WS broker	24
4.1.6	Bank	25
4.2	The AskMe Modules	26
4.2.1	Orchestrator	27
4.2.2	Query Manager	27
4.2.3	Group Module	28
4.2.4	Interrogator	28
5	Implementation Key Points	29
5.1	Machine Learning	29
5.1.1	Bayesian Network	29
5.1.2	Training	30
5.1.3	Inference & Context Scoring	32
5.1.4	Improving the Structure	33
5.1.5	Implementation	33
5.1.6	Performance Considerations	35
5.1.7	Scalability Considerations	35
5.1.8	Personalisation	35
5.1.9	Natural Language Processing	36
5.2	Incentive system	36
5.2.1	Credits	36
5.2.2	Economic System	38
5.2.3	Reputation	39
5.3	Client Software	39
5.3.1	GUI	39
5.3.2	Web Service on PDA	40
6	Test & Evaluation	42
6.1	Test Environment	42

6.1.1	Simulating Users	42
6.1.2	Metrics	44
6.1.3	Rating Generation	44
6.2	Results	45
6.2.1	Short-circuit training	47
6.2.2	TAN performance	48
6.2.3	Group Module quality	50
6.3	Discussion	51
7	Conclusion	52
7.1	Future Work	52
7.2	Acknowledgements	53
A	Installation and user guide of the software	55
B	Glossary	57
C	Modules Interfaces	58
D	Tree Augmented Naïve Bayes sample	59
	References	60

1 Introduction

As trends show, in the near future, the average cell phones will have the computing power of today's PDA. Coupled with "Always-best-connected" paradigm, the market will be ripe for the roll out of a plethora of new generation services, headed straight into Pervasive Computing era. In this thesis, we propose an original way of using the context of individuals and their devices in pervasive environments to build up an information retrieval system by finding experts able to answer any kind of query. We opted for a peer system, where the user is at the same time a consumer, by issuing queries, and a producer, by answering other users' queries, thus establishing an interaction that pulls users into participation.

There are already plenty of information retrieval systems like Google, A9 or Yahoo, and also more specific ones that can recommend a movie or a video game according to people's ratings. We do not want to replace some excellent search engines but rather offer an alternative information system where people can ask human readable questions and get a commercially unbiased answer. No sponsor, no recommended link, just information from people to people. We followed a demographic approach where our system anonymously gathers the context of its users and finds the right experts to answer a query, both interactively or automatically. This is one of the cornerstones of the project, consisting in finding the relevant context of the person that can answer a particular question, so that the people thus found are really able to answer questions satisfactorily.

To incite people to rate and answer the queries a bonus system has also been developed where credits are distributed as would digital money, so that the anonymity of the participants is preserved. Furthermore, a reputation system has been included, so that active people providing quality answers to a lot of questions will be rewarded.

This project aims at designing such an information retrieval system that uses context to find the right experts in mobile communities. We have two clear priorities: first to find the best experts based on a question category and their contexts and secondly to protect the privacy of users. A list of further requirements is exposed later in section 3.

1.1 Background

We are living in a near-future Mobile computing environment, where most of the mobile terminals will be connected to the Internet with flat-rate, and will have as much computation power as a current PDA. We are a step ahead towards pervasive computing, and telephones are equipped with different sensors, whose information can be transmitted to a context network provider

where it will be refined into high-level information, so that it can be further processed at the semantical level. For example, a GPS info like longitude and latitude coordinates will be processed to give a higher-level information like the city, the street name, or even the building and its function (for example, a restaurant or an office). This leads to a better human meaningful abstraction and also allows a computer to act and reason on a semantical level.

Usually, context was used at the user side to improve user experience by proposing a personalised service. We reverse this by not just using a user's own context but by also using the context of all the other users.

One of the bases of our approach is to think that there is a group of people out there that can help us answer one of our own questions and that this group can be defined from its context. This is a strong assumption; so we use the contexts of the users to find the best experts. We expect that answers coming from the people to have a high value, are highly subjective, biased by the personality of the users. This can be much better than a recommendation from a guide, newspaper or popular web search engine, which are often commercially biased. Thus we expect to have the "wiki effect": a collaborative knowledge, working like a word-of-mouth recommendation, but empowered by our system. For example, a query could be "Is there a nice restaurant near the station?" The AskMe engine will find people having the same taste for food as the enquirer and knowing the station area. We expect these experts to give a good recommendation to the enquirer.

The project is composed by three distinct entities:

- The user client, which can be seen as distributed agents collecting and sharing context; launching queries and prompting the user to answer other user's.
- A server side that gathers the contexts of the clients and works as a matchmaker. It also implements all the logic of rewards for the users, which are the query credits and a reputation system.
- A Context Platform that gathers and refines of the clients in order to serve different pervasive environment services with high-level contexts.

The users of our service are mobile and run a client on their wireless connected mobile terminal. By running the service, they offer their context anonymously and get the privilege to ask questions to the system. Our system centralises all the disposable contexts, and upon a query submitted by a user (the *enquirer*), finds out which users (the *experts*) running the client might answer it. The service submits the query to these experts and relays their answer to the enquirer. By running the service and replying queries,

users earn credits. By asking questions, users spend credits. Without credits, a user cannot make queries. Let us already give a short visual example, to illustrate the service.

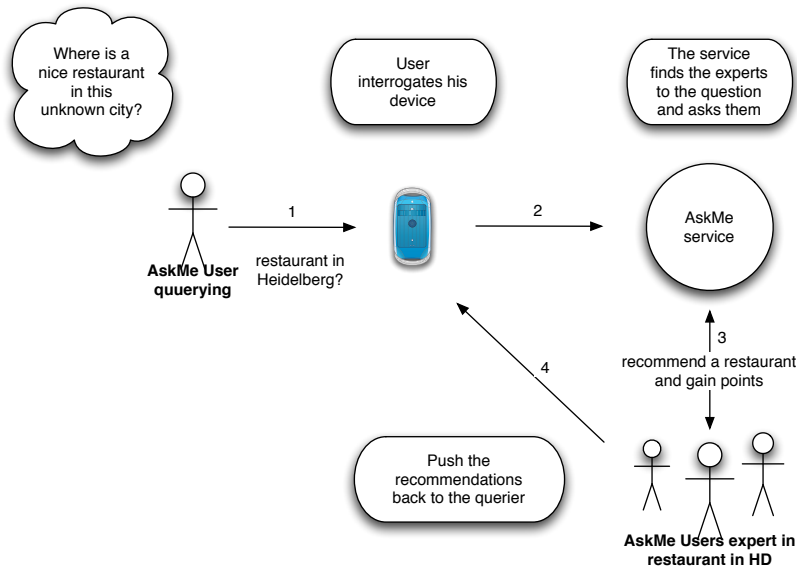


Figure 1: An overview of the AskMe service

An individual can at anytime and anywhere ask questions from his mobile device. The service finds the right people to answer and forwards the query. The experts are signalled that they received a question and can answer at any time.

Imagine a user in a city, Heidelberg, who looks for an Italian restaurant. The system will ask all the inhabitants (*user static context*) of Heidelberg and that like Italian food (*user preference context*). There are of course more context variables taken into account like the age, the budget or the music preference. There are about twenty context variables in AskMe, but there could be even more.

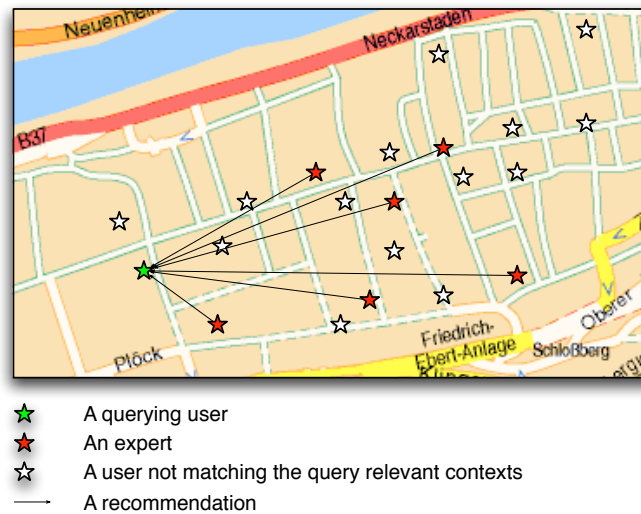


Figure 2: A user being recommended a restaurant by his peers in a city

1.2 MobiLife

MobiLife¹ is a European project² aiming at developing new applications and services for ubiquitous devices. The AskMe project itself is not part of MobiLife, but is influenced by how context is produced and consumed. In the MobiLife framework[10], we imagine that many services would like to use users' contexts. So the framework can serve more than one service. The most important thing to retain from it is that there are context consumers like AskMe, and context providers, like a mobile user sharing his context or a Web service serving the high level contexts from many users.

¹www.ist-mobilife.com

²MobiLife IST-511607

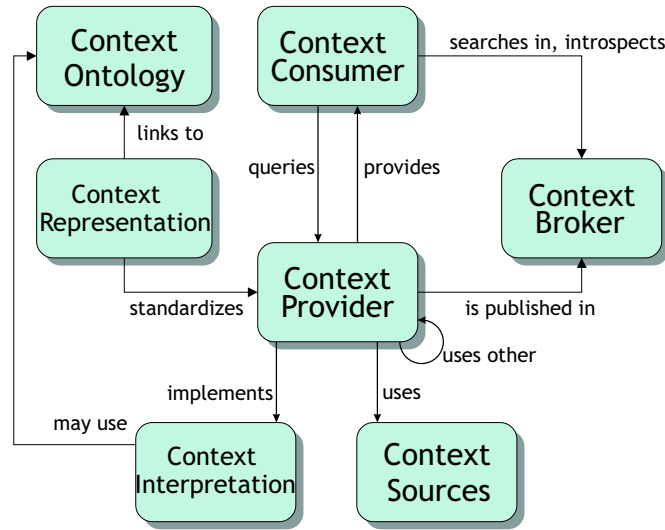


Figure 3: The context management framework of MobiLife

AskMe would be a context consumer in the MobiLife environment. Also it is important to know how a context is seen in MobiLife.

Context: A recent definition of context is due to Dey & Abowd (1999) [1] who defined it as:

any information that can be used to characterise the situation of an entity, where an entity can be a person, place, physical or computational object

In MobiLife, a context can be either static (eg. slowly changing preferences like Chinese food) or dynamic environment dependent (e.g. being right now at the Chinese restaurant). The only influence this has on the system is that dynamic contexts must be refreshed often from a Context Provider Source, while static ones can be updated once upon login from the Profile Manager. We will also see the context of a user as an aggregate of his own context and his device's context. For the project, we consider the context gathering done by some other application. In order to simulate this information and inject it into AskMe, a context provider has been implemented for the simulation.

1.3 Business Model

This service requires the cooperations of users, the roll out of new services and an increased complexity in the terminals. It is not reasonable to further develop if no place in the market is ever envisioned. Such vision must satisfy the three stake holders in this market: users, network operators and mobile terminal manufacturers.

The advantage for the user is to have access to premium recommendations and answers from the on-line community experts. He can expect highly targeted answers without any commercial bias. Furthermore, systems like e-blog, plates.com or flicker show that users are eager to engage in electronic community services.

Network operators would benefit from:

- Users spending more time on their phone. Users have an attractive entry into the new gamma of services available in this next generation.
- The service can be chained to other services such as SMS, interactive chat or a PoC session (Push to Talk over Cellular), so that the enquirer can get more precision about an answer from the expert. The enquirer would have to pay for the communication.
- Access to a large database of anonymous but valid user behaviour models, which be used to further enhance other services.

Mobile terminal manufacturers would obtain a two-fold benefit: firstly, users would upgrade their terminal, since increased context information means increased chances of being deemed an expert and getting one's questions rightly answered. Secondly, the interests of manufacturers are often linked to those of network operators, which support the user by paying a large percentage of the terminal cost.

2 Related Work

There are many works done in the field of information retrieval, expert finders and context-awareness. We are presenting diverse works that we found insightful in particular areas of the AskMe project.

SHOCK One similar work done was SHOCK: Communicating with Computational Messages and Automatic Private Profiles [20]. It proposes a strong privacy architecture to broadcast messages according to a profile matching. Thus, one can ask a question to the experts only, but the profile is not based on the user's context. Due to its peer-to-peer hybrid architecture, each message is received by all the clients, and processed against the client profile. If it matches with a score higher than a set threshold, then the question is shown to the user. The advantage is that the profile stays at the user at any time, but it does not scale well. We think it is an excellent alternative to AskMe, but can only work inside a small group.

Google Answers This is a good example of how to build an *incentive* based collaborative information system [13]. The company first selected a certain number of experts, around 500 people in numerous domains. The enquirer can ask almost anything, but must choose the category of his question himself. The enquirer also chooses how much money he wants to pay the expert. According to that amount, he will get a faster and more complete answer. If he is too stingy, he might get no answer. If he is unsatisfied with his answer, he might claim his money back. He will rate his answer and can ask for a refinement of the answer. The expert will gain a good reputation by having received good ratings for his answers.

This is the kind of incentive system that we would like to have in our information system, whereas in our case the money does not have to be real, but rather exchangeable for service usage.

Lycos iQ Lycos has developed a new service called iQ[21], which is available in a Beta stage in Germany (no French or English version yet). It offers a social network system where users can pose and answer questions. Then the enquirer can rate the answers and good experts get points and climb up a scale of reputation that goes from "student" to "Einstein". For the moment there is no other reward for getting points besides the reputation aspect. There is also the possibility to search in a database for questions and answers; additionally each question is categorised, hence people can answer questions of their field, eg. one can feel as an expert in computer science and try to answer questions in that category.

The system is very similar to ours, since it's also a peer system with some incentive, but there is no utilisation of context, since it does not take advantage of a mobile environment. Moreover, the question it focuses on are rather static or problem solving oriented, and not a step forwards ubiquitous and mobile environment. People have to look for a query to answer, they are not explicitly solicited, since there is no expert finding algorithm.

Stick-e notes This is one of the first context-aware application[5] not long after the founding paper from Schilit on context-aware applications in 1994[28]. It allows a user to let geographically bound sticky notes, so that any users passing by the location of the sticky note can then read it. This is a good example of collaborative knowledge sharing, where the user, which can be seen as an expert in AskMe, can let his opinion about a restaurant or any place. So a geographical recommendation system can be put in place. The system is passive, no one can ask a question, but it is a nice example of possible collaboration among users.

PILGRIM This system offers the user a list of recommended URL links according to his position[6]. It is initially trained according to other users' Web surfing at a particular position using their mobile handheld. This is a simple way to use other people's contexts (in this case limited to location only) to have a recommendation about Web sites. This is not an information retrieval system, but presents a good collaboration system among the users without the need of an incentive based system. However the service is very restricted in comparison to AskMe.

These different examples have shown that there is, to the best of our knowledge, no equivalent service to AskMe. In the next section, we will see how the service is built.

3 Requirements & Design Decisions

In this section we will present the different aspects of the project and how the latter will influence the design of the service.

3.1 Motivations

- *Privacy*
 - Privacy is a major concern among the public about such services. We have to ensure that data cannot be tracked back to a real person. This has strong implications for the architecture and the protocols of the system.
- *Adaptivity and Precision*
 - The expert finder should promptly adapt itself to new types of query and contexts, with minimal operator work and effectively find the right experts. To have a good success rate in our recommendation, we will first limit our system to a few question categories, because finding the right expert for any question is a Herculean task.
- *Responsiveness*
 - We would like the users to get their answer as fast as possible. Efforts have to be made in the user interface as much as in the implementation of the service, to optimise this parameter. Also, we would see later that some calls can be made in advance, thus speeding up the system.
- *Mobility*
 - We expect our users to move, changing IPs and possibly getting disconnected and reconnect. The system must be tolerant to such behaviours by implementing reliable and stateless mechanisms.
- *Scalability*
 - As many people have a mobile terminal, the number of users can rapidly grow. Hence the service must be scalable by distributing itself and optimising resources.

- *Usability*
 - The service must be as simple to use as possible, as any software should be. But the Graphical User Interface (GUI) is even harder to design on a handheld, due to limited space and clumsiness of the input mechanisms.

3.2 Adaptivity

3.2.1 Feedback gathering for machine learning

A feedback from a user can be retrieved so that the relevant contexts finder unit improves its decisions. There are several techniques available. One is to directly ask the user about what he thought of the recommendation. There are more subtle techniques, where the system guesses by itself if the user liked the recommendation (Did the user effectively go to the recommended place? Did the user go again to the recommended place again?). The system has to learn by itself how to find the best experts according to the question category. This can be done by using machine learning algorithm like Bayesian networks or decision rules learning.

3.2.2 Natural Language Processing

We would like AskMe to automatically processes queries and infer what they are about, through Natural Language Processing (NLP). This idea was superficially investigated, and currently it is the user who defines the category of his question.

3.3 Distributed Architecture

3.3.1 Web Services

We want our architecture to be modular and distributed, so we can distribute the load on several machines and separate the different actors. As the project was written in Java, several possibilities were at hand to have a distributed system, like Enterprise Java Beans.

Web services (WS) makes the development of the client fairly easy. By just having knowledge of the WSDL file, which defines the service entirely, it allows rapid generation of the stubs and skeletons and it is also possible to make a call on the fly to a Web service. This offers extreme interoperability, as it is not language or platform dependent. As SOAP messages are encapsulated in HTTP messages, the message will have no trouble traversing firewalls in a deployed environment. Furthermore, the MobiLife context

management framework was done in Web Services, thus using the same technology makes it easier, as the know-how was already present in the group.

A drawback to Web Services is that both standards and the tools are not sufficiently mature. The Web Service Security (WSS) standard developed by OASIS[24] is still poorly implemented, like WSS4J that has released its 1.0.0 version recently. There is a lot of effort to improve Web services, but open source implementations are still limited, such as in WS-reliability. In Java, we expect the situation to improve with the final release of Axis2, which can integrate easily WSS and WS-R modules.

3.4 Privacy

Privacy is a common problem with context aware and collaborative recommender systems. Defining privacy is a hard problem, and is heavily dependent on the country and point of view of the people. There is not a unique definition of privacy. Privacy protection is not only technical, but is also a society and legal problem. In our case, we would like that data stored about the user is unidentifiable, that is a context can be bound to a pseudonym or virtual ID, but not to the real identity of someone. The advantage is twofold, it improves privacy of the user, obviously, but also allows to store data without much legal trouble, since many laws follow the recommendations of the OECD about privacy[11], which prevent storing identifiable personal data without user agreement and control. Kobsa and Schreck presented a good summary of the different polls concerning privacy[18]³:

³The paper quotes many papers as source for the presented numbers. Those references can be found in the referenced paper

Table 1: User Concerns Regarding Privacy on the Internet

Respondents asserted to	% agreement (“strong”/“very” and “somewhat”)
being (very) concerned about threats to their privacy when using the Internet	81%
being willing to give out personal data when they get something valuable in return	31%, 30%, 51% according to the different papers
value being able to assume different aliases/roles on the Internet	58.8%
rate privacy as more important than convenience	75.5%
value being able to surf on Internet in an anonymous manner	81.1 %
value being able to communicate over the Internet without people being able to read the content	93.2%

In their paper they conclude that it is unlikely that users will tolerate the storage of large amounts of *identifiable* personal data, unless the benefit of it is extremely valuable for them.

Since context is something perceived as personal or private information, sharing it to an application poses problems. By combining an architecture framework with anonymising proxies, and by the use of cryptographic tools, we can ensure that user’s privacy is protected.

3.4.1 Architecture

We use a proxy between the client and the service. It protects the client from being identified by his network address. Also, the proxy provides registering and login service, and provides users with a virtual ID to connect to the AskMe service. Then any message that goes through the proxy is not read by it, because they are encrypted with the AskMe public key. Thus, the knowledge of the identity and the knowledge of the contexts is distributed between two different entities who cannot make sense of it without getting together.

3.4.2 Pseudonyms

The user uses one or more virtual IDs to identify himself to the AskMe service. So the service never knows about his real identity. The user does not lose any credits when he changes identity, since he can get anonymous credits and reputation token from the bank. In other words, users reputation is not based on their ID, but on the token that can prove it.

3.4.3 Cryptography

We used cryptography for two reasons:

- *Secure and Authenticate messages*
The client has a pair of keys for each of his virtual IDs. The user uses it to sign his messages. It then uses the public key of the AskMe application to cipher his messages. This has the advantage that the Entity Manager has no knowledge about the content of what it relays. This can be achieved by using topen protocols, and is implemented in Java by using security API or WSS4J which is an implementation of the WSS standard from OASIS.
There are other means to enforce privacy. For example, a control of the precision of the information shared by the user, which can be seen as information blurring, enhances privacy[14].
- *Passing credits and reputation tokens*
A user can possess several virtual IDs. To have several virtual IDs helps to make him untrackable, but poses the problem of persisting its credits and reputation. Since the application gives credits to a virtual ID, the user should be able to spend the sum of credits that have each of his virtual IDs. To achieve this, we put a digital cash system in place where the user can receive signed money from the application or withdraw it from the bank. To protect anonymity, we use a blinding signature scheme, as described in section 5.2.1.

3.5 Incentive

The main actor in the AskMe service is the user. Since it is a peer system of cooperation, without his collaboration, the system falls apart. We need to build an incentive system. We do this by using a credits and a reputation systems. We do not guarantee that an enquirer will get for sure an answer. In that case he should get a refund. Achieving a critical mass of users is therefore a must, so the incentive system might be flexible at the initial phase.

3.5.1 Credits

We built an economic system where users pay a definite amount of credit to make a query. To earn credits, he must help the system by issuing ratings and answers.

3.5.2 Reputation

We want to reward helpful users with some honorific points, and the more points they have, the better their reputation will be. That would give the user a valorisation of his collaborative help, but also give the enquirer more confidence in answers coming from a reputed expert. Users with a give reputation could demand answers from a more selected, high reputation set of experts.

3.6 Context Awareness

The client possesses features from the context awareness technology. Thanks to sensors, it can get its context, infer it to a higher semantical level, and possibly reason on it to provide a more elaborate context. In MobiLife, most of the reasoning is done on the server-side in order to avoid mobile terminals heavy computation. This aspect was out of the scope of this work, as we are only consuming contexts and do not care about it is not our concert how it is done. See 4.1.3.

3.7 Use Cases

In order to evaluate the success of the project, at design time, we have defined the use cases that we should be able to fulfill. We present them here since they also show the capacity of the AskMe service. We depict 4 use cases, which represent the life cycle of AskMe: “Query - Expert Finding - Answer - Rating”. The different uses cases are presented using the template from *Writing Effective Use Cases* from Cockburn[7]. A basic template can be found on the web⁴. Each use case is then represented by a UML sequence diagram.

⁴<http://alistair.cockburn.us/usecases/uctempla.htm>

3.7.1 Looking for a special place

USE CASE 1	Issuing a query	
Goal in Context	Moka, an imaginary Finnish student, is in Japan for the first time in his life. He is angered that he forgot his travel guide at home, and has no idea where to find a good typical Japanese bar. As he does not feel comfortable asking people around, he is using the AskMe service to find a restaurant. He turns on his 3G, FOMA compatible cellular phone, starts the AskMe application and asks the system for a typical, foreign-friendly restaurant	
Preconditions	Having enough credits to issue a query	
Success End Condition	Receive confirmation message from AskMe that query was sent, the enquirer has less credits	
Failed End Condition	The user should be informed that his query could not be processed	
Primary Actors	Enquirer, AskMe - all modules, Entity Manager	
Secondary Actors	Bank	
Trigger	The user will to make a query	
DESCRIPTION	Step	Action
	1	The enquirer logs in to the Entity Manager and gets a virtual ID
	1b	The Entity Manager forwards the logging in to the AskMe service
	2	The enquirer, once logged in, issues a query
	3	The AskMe service processes the query and forwards the query to a number of expert
	3b	The Entity Manager takes care of relaying the query to the experts reliably

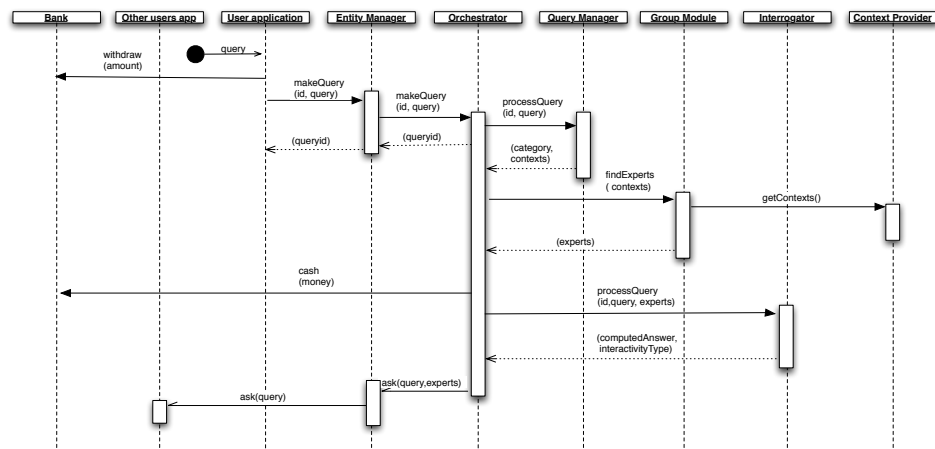


Figure 4: Sequence diagram for a query

There are different types of queries that we would like our system to find experts in. We have indeed had to limit our system to a small range of possible questions, this way keeping the system simple and easily testable and improvable. Here is a list of the different kind of queries we thought that AskMe could be efficient at:

Place of Interest Someone is looking for a recommendation for a particular place. e.g. a restaurant, a discotheque.

Local A question concerning local commodities, like if the station grocery shop is open on a Sunday.

Cultural A question about, for example, traditional habits in an unknown country.

Price Info What should a taxi ride to my destination cost?

Crowd Info Is a particular place crowded?

3.7.2 Answering a query

USE CASE 2	Expert answering a query	
Goal in Context	Hiro, a student from Tokyo, is waiting for the bus. A bit bored, he looks at his cell phone and sees a query about a non-expensive typical restaurant. He recommends his favourite one where he likes to go with his friends when they have some special event to celebrate.	
Preconditions	Running the AskMe service	
Success End Condition	Getting credits	
Failed End Condition	Receiving an error message from the AskMe service	
Primary Actors	Expert	
Secondary Actors	AskMe, Entity Manager, Bank	
Trigger	An AskMe user receives a query from the service	
DESCRIPTION	Step	Action
	1	The expert visualises the query and answers it.
	2	AskMe transmits it to the enquirer and also sends some credits to the expert
	3	The enquirer receives an answer
	3b	The expert receives some credits and cashes it to the bank

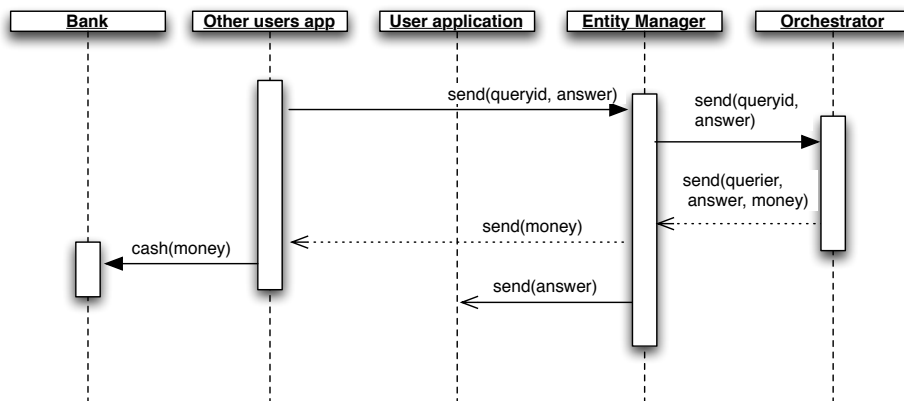


Figure 5: Sequence diagram for a answer

An expert will hopefully reply to the enquirer; we can see on the diagram that he gets credits for his collaboration, and that the user gets his answer,

asynchronously from his past request. The client has to keep sent queries in memory, so as to associate the answers to them afterwards.

3.7.3 Receiving an answer and rating it

USE CASE 3	The enquirer received an answer and rates it	
Goal in Context	After ten minutes, Moka has already received 3 replies to his request. One recommendation from HiroMonster (<i>use of pseudonyms</i> sounds particularly good. After having a delicious meal and having met several Japanese students, he decides on his way to the hotel to rate HiroMonster's recommendation with the highest mark.	
Preconditions	Running the AskMe service and having issued a query before	
Success End Condition	Receiving money for having issued a rating	
Failed End Condition	The transmission of the rating fails	
Primary Actors	Enquirer	
Secondary Actors	AskMe, Entity Manager, Bank	
Trigger	The enquirer receives an answer for a previously sent query	
DESCRIPTION	Step	Action
	1	The enquirer visualises the answer and rates it.
	2	AskMe rewards the enquirer with some credits for having rated and transmits the rating to the expert.
	3	The enquirer receives the money and cashes it
	3b	The expert gets to know how well his answer was rated

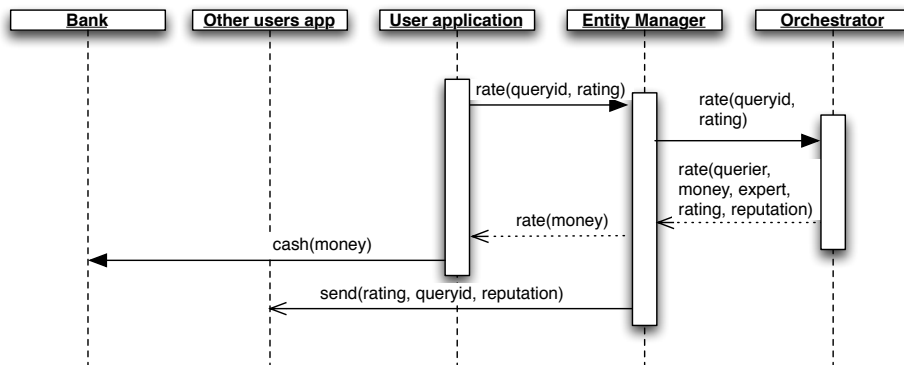


Figure 6: Sequence diagram for a rating

After viewing an answer, the user can generate a rating. This is transmitted to the Orchestrator. According to the rating, it will send a reward to the expert, but will also give some credits to the enquirer for having rated, an action that helps the system to improve itself. The expert must also keep track of sent answers, so to associate the reward message afterwards, because of the asynchronous nature of the messages.

3.7.4 Earning reputation

USE CASE 4	The expert earns reputation	
Goal in Context	Hiro, before turning his cell phone off for the night, looks at the notification icon from AskMe. He earned a few credits thanks to his last recommendation and even more stepped up a level up in the AskMe reputation ladder. He feels happy to have helped someone, logs out of AskMe and falls quickly asleep.	
Preconditions	Running the AskMe service	
Success End Condition	The expert gains reputation points	
Failed End Condition	No change	
Primary Actors	Expert	
Secondary Actors	AskMe, Entity Manager, Bank	
Trigger	AskMe service received a good rating about an expert's answer	
DESCRIPTION	Step	Action
	1	Upon receiving a good rating message, the system sends to the expert some reputation points.
	2	The expert receives a message about which answer got a good rating and how many points he gained

We have seen in these four scenarios the basic operations of AskMe and its potential to share personalised recommendations between users.

4 Design and Architecture

From a high-level view, the system is made up of several modules, each of them presenting a Web service interface. We used Apache Axis 1.3 as implementation of the SOAP (Simple Object Access Protocol) and deployed the WSs on a servlet container using Apache Tomcat. The choice was easy, both are open source and free, and have all the required functionality. For persistent storage, we used files and an SQL database, MySQL. This also helps in sharing data between the different modules of the AskMe system. Also, there is a centralised WS registry where all the modules publish their location. Neither the database nor the registry are shown below. The whole project was developed with the Eclipse IDE using the Web Tools Project.

The main interfaces are available in the appendix C.

4.1 Overall Architecture

Here is an overview of the AskMe service and an example of the data flow for a query.

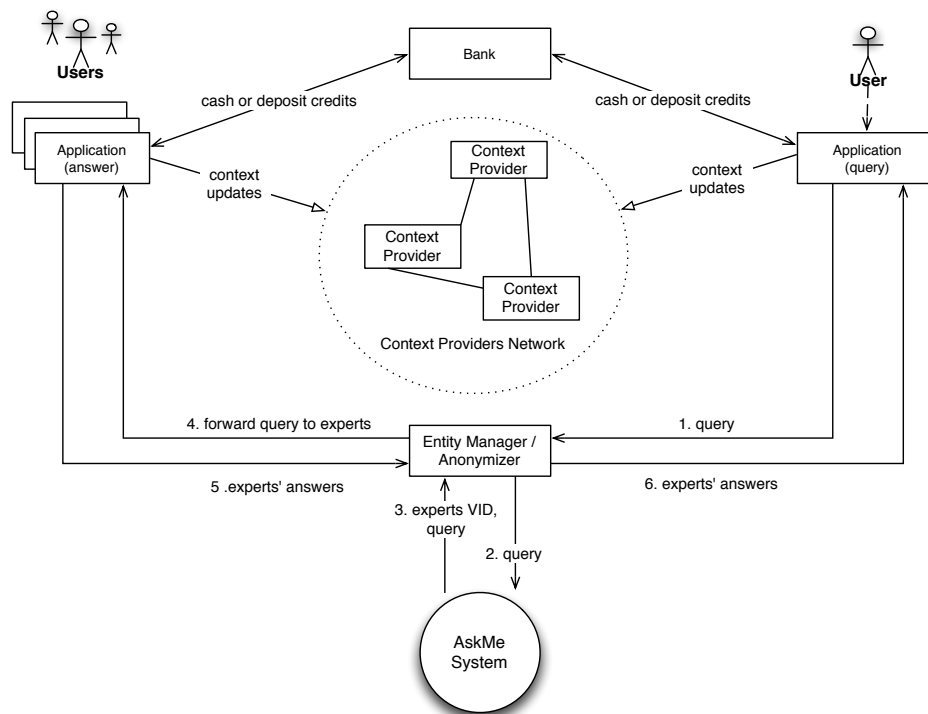


Figure 7: AskMe from the client point-of-view

First the user issues a query. It is transmitted to the Entity Manager, which works as a proxy, it is then sent to the AskMe system, which will find the experts and post the question. Afterwards the query is relayed by the Entity Manager, which is the only one that knows the IP addresses of the users. Experts then send back their answers to the enquirer. The schema shows that users communicate their context to the context providers network from time to time. We assume that clients have a public IP. In real life, there should be NAT and proxies on the way and this assumption would no more hold. However this a generic problem in actual networks since IP addresses are getting scarce and will not be discussed any further.

4.1.1 Client Software

The GUI was targeted to run on a PDA, as the AskMe service is expected to be deployed in, let us say, five years. It was not necessary to constraint ourselves on a slow and very limited connectivity of an actual mobile phone. It is said that with every click, half of the potential users get lost, so the GUI has to be as simple and proactive as possible. We implemented a demo running on a HP iPAQ hx4700 running on Windows Mobile 2003 2nd edition.

4.1.2 Entity Manager

We use the module Entity Manager as a proxy to the application. Its main role is to hide the identity of the user by changing his “real ID” into a pseudonym or virtual ID, and also to hide his network address, which is an IP address in our context. Furthermore, it can provide extra services, like anonymising the context of the user, before it is published on the context provider network.

Then the user, when communicating with the AskMe application, can use a public key so that the entity manager cannot know the nature of the message. We clearly see that knowledge about the user has been split in two. His identity is known by the entity manager and the information about his context and queries by the application. Thus, the cooperation of the two modules is needed if we want to associate the context to the user. So the user has to trust that at least one of the two parties will not cooperate with each other.

However, the possibility of collaboration between the two modules is entirely negative. Imagine that one of the expert emits a racist opinion or anything against the law. First, the AskMe application could ban that ID and ask the Entity Manager to do the same, and secondly, it could ask for identities for law enforcement purposes. Whether the Entity Manager will collaborate

or not is a policy decision.

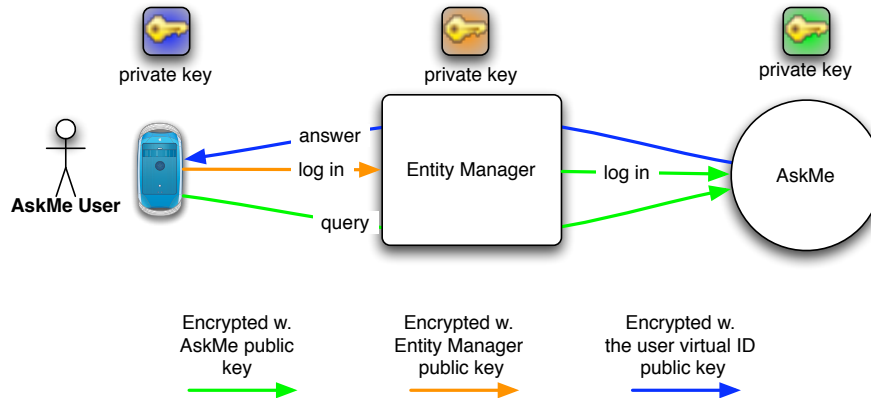


Figure 8: Use of public cryptography in AskMe

To do this, we have to encrypt the content. A nice thing would be to use WS security, which encrypts and signs the whole SOAP envelope or just some of its elements. However, the SOAP implementation of the client is quite light, and using WSS4J, an implementation of WSS could be quite heavy on the terminal. So far, we only send the content using the JCE⁵ Bouncy Castle. Indeed, Sun's extension is part of JSSE since Java 1.4 only and also less complete than the Bouncy Castle one's. SSL encryption was not a candidate, because sometimes we want some part of the message to be understood by the Entity Manager. For example, when AskMe sends some messages back to the client, it gives the virtual ID to the Entity Manager, who then gets the IP of the client to relay the message. So to conclude, with plain old encryption, we do meet our requirements, but using WSS would be a cleaner solution. Adapting WSS4J for the terminal or some other XML security would be a good investment, but was outside the focus of this thesis.

4.1.3 Context Providers Network

The AskMe service needs the context of the user, which is stored and distributed via a network of context providers. Part of the MobiLife, the framework provides high level context to our information system. The client regularly updates its context to the network and can be seen as a context provider too. The network then offers its high level contexts to numerous different services. In that framework, we can see AskMe as a *Context Consumer*.

⁵Java Cryptography Extension

4.1.4 Context Provider Anonymizer

In MobiLife, the client pushes its context to the Context Provider network occasionally with his real ID. There is therefore little concern about privacy⁶. To protect it, we have to use a middle agent that hides the real identity to the application. A context provider anonymiser has to be put in the middle and be informed by the Entity Manager of the mapping between the real ID and the virtual IDs. The AskMe service then gathers contexts only through that service. This can be seen as a client side extra assurance to the ongoing work of protection in the MobiLife framework.

4.1.5 WS broker

We recall the architecture of a service-oriented architecture where a Web service is published in a service broker, which is in effect a UDDI registry server. There is an open source UDDI server for Java called jUDDI. It is

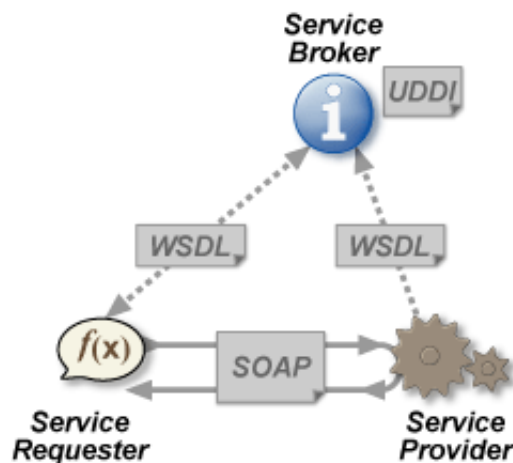


Figure 9: Web Services Architecture

easy to install and operate, basically it is a Web service coupled with a persistent data storage, which is in this case a relational database in the case of jUDDI. The Web service provider can publish to jUDDI, using a user account. The drawback is that doing so is not as easy as sending the WSDL file to the registry. One must first map a WSDL to a UDDI object. IBM has developed a wrapper around UDDI4J, which is an API for UDDI and uses the recommendations of the W3C for the WSDL-UDDI mapping. Unfortunately, documentation and samples are scarce and oversimplified.

⁶Privacy in MobiLife is not handled with pseudonyms, but with a trust engine that hands out permission tokens to read the data

As jUDDI was offering too many functionalities that were not yet useful in the case of our project, we decided to build our own lightweight registry, which just registers the names and the endpoints of Web services.

Thus each of the modules tries to publish their endpoints at startup, using the initialize method of a servlet that Tomcat calls at startup. Each WS module and the client can then inquire about the endpoints of other modules from the registry. Hence the only endpoint that has to be stored in the modules memory is the registry address.

We can imagine that a module is replicated for robustness and load balancing reasons. Through the registry, the module will have access to all these modules. Thus having a registry allows much flexibility in our system.

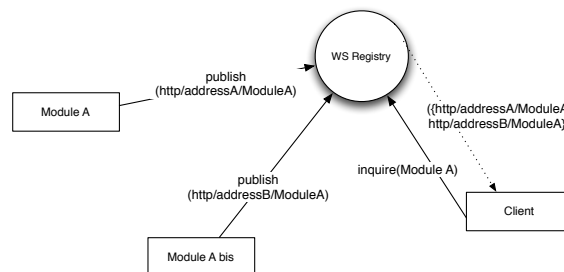


Figure 10: Registry operations

4.1.6 Bank

We need a module to hold the accounts of the clients, and issue digital cash, the so called AskMe credits. It is more convenient to organise it outside of the AskMe service, since this way it can issue and cash money for the AskMe users and keep a persistent status of their account in a database. This account is linked to the real ID of the user, there is no need for a pseudonym since the bank cannot know from whom the credits come. On the other hand, the AskMe application gives the money to a virtual ID. That is the great advantage of digital cash: the protection of anonymity. Moreover the credits could also be used in another service.

Also, to speed up calls and avoid waiting for the bank in case of an operation, the client can withdraw money in advance and then keep it in the client memory until the next query.

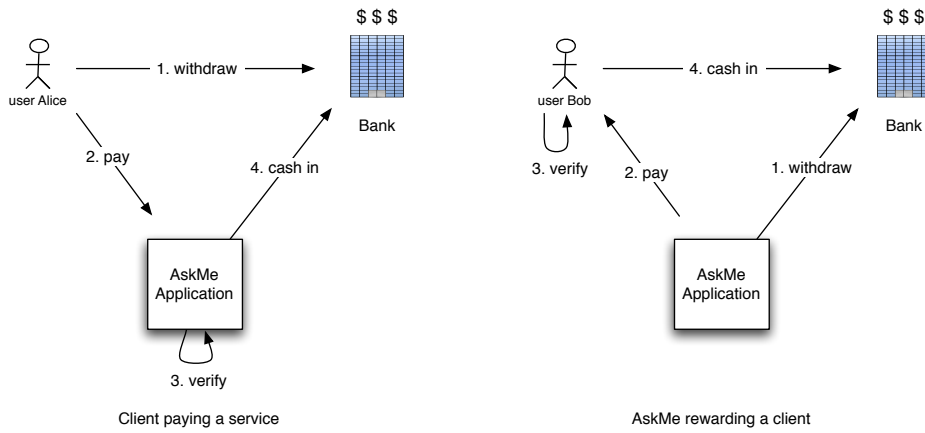


Figure 11: Credits transfers in AskMe

4.2 The AskMe Modules

Let us now zoom into the AskMe service and see its different parts.

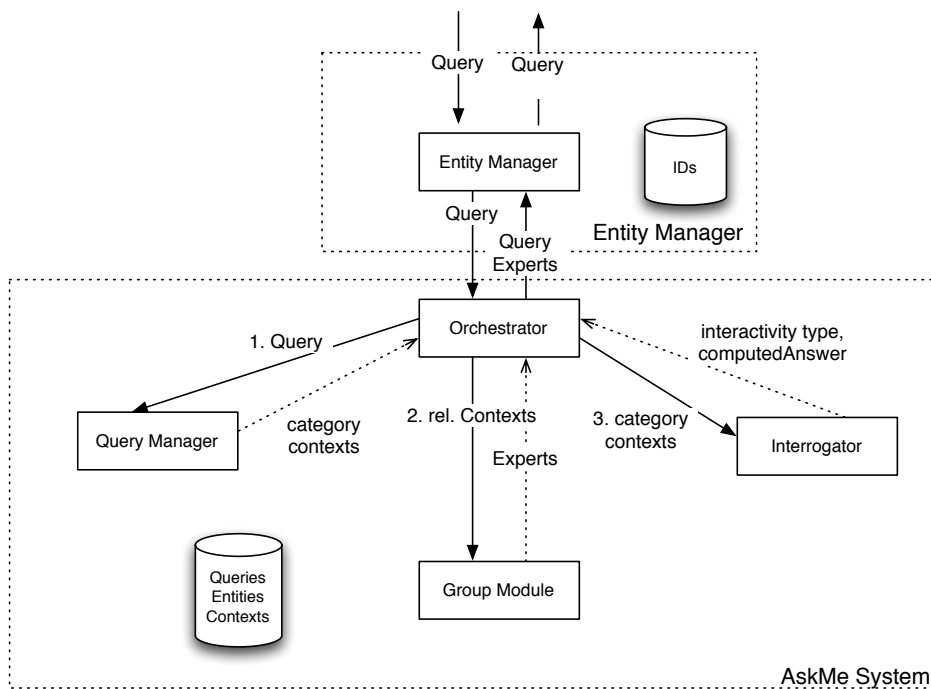


Figure 12: The AskMe system with data flow in details

AskMe is made of four modules, which share a unique database schema. This

allows the Group Module, for example, to know who is on-line when creating groups. The Entity Manager acts like a facade for the client and is the only one that can access the Orchestrator, the portal module of AskMe. One can notice that the Entity Manager has two interfaces, one for AskMe so that it can send messages back to the client and one for the client. AskMe has a star architecture, in spite of the fact that the query needs to be processed successively by the Query Manager, the Group Module and the Interrogator. A star topology is better than a ring, since we get error report and point of error at the central node. If the central node falls, the system fails, but if any other is disconnected, the Orchestrator can look for a replacement, running service.

4.2.1 Orchestrator

The Orchestrator is the gateway and heart of the service. It handles all the requests coming in or going out of our system. As is, it is in charge of various tasks. It manages the logging in and out of the users, communicates with the bank and manages credits and reputation payments.

4.2.2 Query Manager

The Query Manager is the first module that processes the query from the client. Its purpose is to get the category of the question and find the context variables that best define the experts according to the latter. It is aided by the enquirer who selects the category of his question when issuing it. Afterwards the Query Manager can do some refinement by using natural language processes, and by finding keywords like “restaurant” or “eat”, can deduce that the category is “looking for a place to eat”.

Once the category is determined, the Query Manager has to find which contexts should represent an expert in that category. This is executed by a machine learning component and will be explained later. See 5.1



Figure 13: Query Manager Black Box

4.2.3 Group Module

Part of the MobiLife project, this module can return groups consisting of people having similar profiles, similar tasks or forming a social group. It is also where we plugged our group simulator in order to validate our system. The Group Module receives a list of context variables-values plus an operator on it and returns a list of users, which should match the received criteria at best.

For example, when the Group Module receives “temperature between 20 and 30”, it will give one point to all users whose context satisfies this predicate. It can be a little trickier, like the location variable: “position of expert must be less than 300 meters than the office of enquirer HiroMonster”. Hence for each satisfied predicate, users get one point, and the k users having the most points are the experts returned by this module. k is a value defined by the Orchestrator, and will depend on how many users respond to queries. If only a small of experts effectively answer, then k should be set high. The point system is a bit weak and should be improved, for example, by including a weight for each predicate according to the score that the context got in the Bayesian network (See 5.1.3).



Figure 14: Group Module Black Box

4.2.4 Interrogator

The purpose of this module is to decide if the enquirer should interact with the experts or not. We could imagine initiating a push-to-talk session, a chat or a simple message exchange or even not contacting the experts at all. For example, if the enquirer asked if a certain discotheque is crowded, we can imagine that according to the number of experts found in that particular discotheque, we could extrapolate that the discotheque is crowded and directly return that information to the enquirer.

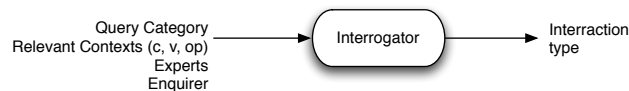


Figure 15: Interrogator Black Box

5 Implementation Key Points

In this chapter we present the most relevant part of the project and discuss it in detail. In part 5.1, we will discuss the machine learning algorithm to improve the expert finder. Part 5.2 shows how we implemented the incentive system. Finally, part 5.3 will describe the implementation of the client software.

5.1 Machine Learning

We would like our system to work for various requests and with a non-definite list of context variables with minimal manual work. As these are changing over time, we want it to adapt itself with some machine learning algorithm. We have chosen Bayesian networks for that purpose, considering them the simplest way to begin with. Bayesian networks are easy to understand, since their graphical representations are explicitly understandable by anyone. Furthermore it is a well established field and many libraries are available to handle them. In addition, they offer some flexibility, for example about the structure learning, that allows us to dynamically adapt our network, if new contexts arrive or if we want to change the “complexity-computation effort” trade-off.

The problem is the following: we have a query and its category and would like to determine which contexts might generate the best group of experts to answer the query. For example, if the enquirer is looking for a discotheque, consulting people of his age group or younger people sounds relevant, but their cuisine taste seems a priori less helpful. Another example is that if a woman is looking for a good make up salon, asking women appears to be better than asking men. So we would like the network to find out these relevancies by itself. To achieve this, we need to give the network some feedback. Therefore, we demand the AskMe enquirer to rate all the answers he receives. Then the Bayesian network has feedback to work on and might reinforce or diminish the belief about a context being relevant to a particular category.

5.1.1 Bayesian Network

A Bayesian network (BN) is a directed acyclic graph, where nodes represent variables and arcs dependencies among the latter. The Bayesian network represents the joint distribution of all the variables represented in the graph, which can be written:

$$P(X_1, X_2, \dots, X_n) = \prod_{i=1}^n P(X_i | \text{Parents}(X_i))$$

In classification problems the network has a class variable, which represents the different values that the output of the classifier might have.

Given a training set, one can build the most probable Bayesian network corresponding to this training set, using a maximum likelihood algorithm. As the numbers of nodes and possible values grow, it can become quite unfeasible to build the optimal network, since it is a difficult problem. There are some approximation techniques that shortcut building a BN in an affordable amount of time by sacrificing precision. In our case, we will always have a tree form for the network, since we want the success of a query to depend on each context variable. The success is a classification of a query answer as good or bad according to the user's rating of an answer. Therefore the attribute class "success" is the root and connects all the context attributes.

For our purposes, we decided to have multinomial discrete variables, meaning that infinite values like a temperature are discretised building the BN. The discretisation process was done arbitrarily in regular intervals. However discretisation that takes in account the value of the class variable is indeed a better solution. Refining this is left for next steps.

We built a naïve Bayesian network whose class variable is the category of a query and all the other nodes represent context. A context might have absolute value like cold, temperate, warm or relative value like close to, or far from. Those variables need to refer to something and in general they refer to the enquirer's context, for simplification.

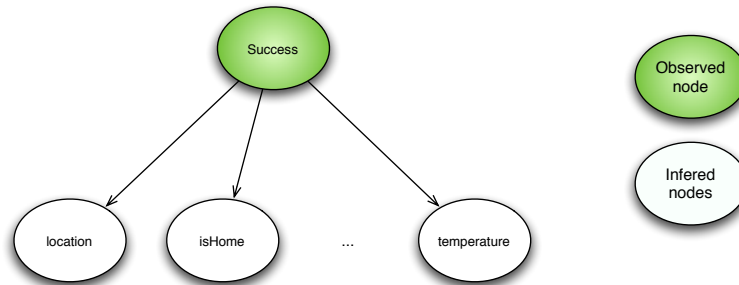


Figure 16: A naïve Bayesian Network

5.1.2 Training

The probabilities of each variable are kept up-to-date thanks to counters. There is a counter for each value. Normally, when training a BN with a data set, each time the value of a variable is encountered, it is incremented by one. After training, one has the marginal probability for each variable. Since we want to train our BN with feedback, we associated each data set

entry, which contains the relevant context variable-values of the expert, with a rating, which is the feedback from the enquirer. The rating is an integer value between -2 and 2. If the score is positive, we assign the rating as a success, otherwise as a failure. If the score is equal to 2 for example, we train it twice as a success. Thereby we expect to influence the inference of our Bayesian network and thus get the best relevant contexts after as little training as possible.

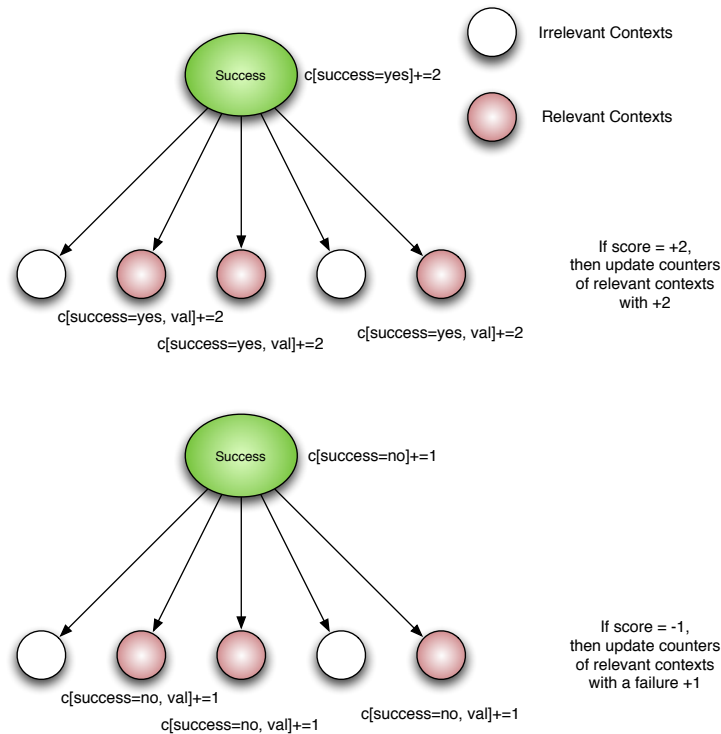


Figure 17: Reinforced training of the Bayesian network

There is however an inconvenience in the training, which we called the Mastermind effect. Indeed, we can see the relevant contexts to find an expert as the secret colours to find and the feedback rating as the white stones shown by the opponent in the game as response. The two main differences are that we do not know how many contexts are the good ones and ordering does not matter. Our problem is of course more complex. What we wanted to depict with this example is that when receiving a good feedback, the system does not know which specific context did improve the score. That allows neutral or even "bad" contexts to be trained as success even if they are not. Hence it will be no surprise to see neutral contexts being dragged up with the good ones for example. This can be quite surprising when looking at the output of the BN for the first time to see expected good contexts at the top and one

or two neutral contexts with a good score among them. Further training though, will consistently increase the counter for the good contexts, while the others will pop in and out of the query result, eventually falling behind.

Another limitation so far is that we do not support multi-valued training, because the data is represented in the Weka tool in a comma separated value (CSV) kind of format, the ARFF format, where multi-values are not allowed. A workaround is to binarise multinomial contexts, but this is not satisfactory. As we keep the problem simple for the moment and experts' contexts are mono-valued, this is not a problem. But it could be a good improvement to consider.

This leads to another remark about our variables being multinomial with different cardinalities. A variable with many possible values will have its marginals having more variance than a binomial one for example. This can influence a lesser performance when the data-set training of the BN is small.

5.1.3 Inference & Context Scoring

Inference is the process observing some variables of the network, eg. the success variable and get the marginals for each other attribute. The inference, naïvely done, requires a lot of computation because of numerous multiplications[23] Better algorithms, as used in many code libraries, are quite complex and will not be discussed here. To calculate which contexts are the more relevant, we reverse the classification problem by first observing the class variable and then query the context variables; i.e. one can calculate for each value v of a context c_j a score, using the marginals of the variables for success and failure:

$$score(v_i) = \frac{P[c_j = v_i | success = yes, c \neq c_j]}{P[c_j = v_i | success = yes, c \neq c_j] + P[v = v_i | success = no, c \neq c_j]}$$

$$score(v_i) \in [0, 1]$$

Then, the best value is chosen according to the maximum of the score. The score of the context variable itself is determined by it:

$$score(c_j) = \max(score(v_i)) = v_{max}$$

The Query Manager outputs the k-best pair of contexts variable-value according to $score(c)$, with an operator deduced from the context (e.g. a position context is automatically bound with the inArea operator, etc.), plus some random contexts variable-value in order to introduce some noise and gather feedback on variables we have not contemplated yet. It helps exploring new context variables-values since the training can lead to a stubborn system. We see also that potentially it can also output twice the same

context variable with two different values, if the score for both values are good. This is a quite positive feature, specially that in reality we can expect multi-valued variables (an expert can like Chinese food *and* Italian food. However, since the training does not support multi-valued training, the Query Manager only outputs one value per context variable at the moment.

5.1.4 Improving the Structure

A naïve structure surely does not represent the true nature of our variables. Indeed, claiming that all contexts are pair-wise independent cannot hold (e.g there is a clear relation between acquisitive level and age. Considering both independent gives too much weight to each of the events). Thus, one can go a bit further with the complexity and describe our network as Tree Augmented Network Bayes (TAN), thus expressing partial dependencies among the contexts themselves, for example being in a restaurant is highly correlated with the activity context eating. In a TAN, a node has at most one parent attribute other than the class attribute. For a concrete example, please refer to Appendix D. Additionally, in a more complex tree, getting the best contexts is different. One sequentially observes nodes as long as we make decision about the best contexts in the tree. We indeed observe the best scored node to v_{max} , then infer again the dependant nodes, get the best scored node, etc.. The result is in which order we have chosen the contexts, the first one being the best.

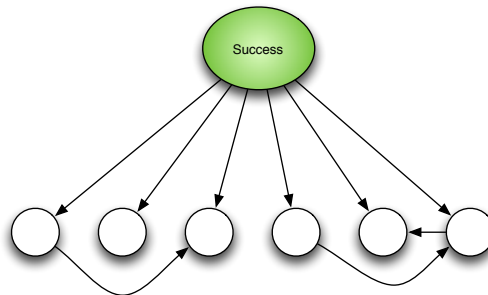


Figure 18: A Tree Augmented Naïve Bayesian Network

5.1.5 Implementation

There are mainly two open libraries for Bayesian networks: JavaBayes and Weka[34]. JavaBayes is only for Bayesian networks. It has a nice interface (both API and GUI) and is very easy to use. It is good for visualisation and

inferring. When it comes to training, however, it has very few algorithms implemented. Its development seems to be stalled and its source code is fairly hard to understand with many uncommented sections and unimplemented methods.

On the other hand Weka is a complete machine learning library and is actively developed. It has many algorithms and can be extended easily. It is also very good support for Bayesian Networks, but has no possibility to observe a variable for then inferring other ones. Being otherwise very complete, it has been a clear choice for our implementation. We finally used Weka for training and learning the BN, and JavaBayes for the inference. Weka can pass the data defining the BN to JavaBayes thanks to a BIF file (BayesNet Interchange Format), which is an XML document that fully describes the network.

For further information, Sun has also emitted a JSR (Java Specification Request), the Java Data Mining API (JDMAPI⁷). It is a promising step to having a unique interface when it comes to data mining, but the implementation for the Bayesian networks is fairly poor, as there is only support for naïve networks. The implementation itself can be summed up to a flowchart:

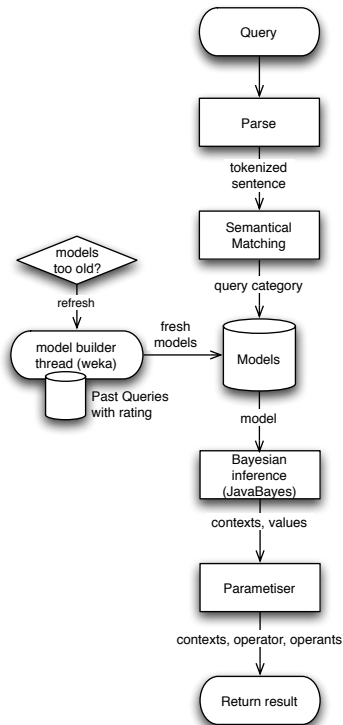


Figure 19: Query Manager Processes

⁷<http://www.jcp.org/en/jsr/detail?id=73>

The flowchart describes the different processing steps inside the Query Manager. One can see that Bayesian networks can be trained, and then uses several times before being trained again, thereby saving computation.

5.1.6 Performance Considerations

The Query Manager module needs to perform big amounts of computations. If we do not consider the Natural Language Processing, the Bayesian network process can be highly optimized. Once the Bayesian networks are generated, it is possible to keep them in memory and already run the inference process and memorise all the results. Thus for choosing the context variables, the program only has to get from the memory the result of the inference for a specific question category and directly take the k-best (or threshold based) contexts. Hence the heavy task of learning the structure and inference can be done “offline” by another processor or when the load on the machine is low (during nights for example).

5.1.7 Scalability Considerations

Using naïve Bayes networks, scalability is hardly an issue. Time and memory complexity grow linearly with the number of query categories, context variables and available training data. Also, in a naïve BN, the inference is trivial, since there is only one dependency. That is not the case anymore for other types of Bayesian networks, which present higher degrees of complexity. Indeed, structure learning and inference take more time and are often exponential (e.g. for the inference, it is exponential with the size of the junction trees, derived by the Bucket tree algorithm used in the JavaBayes inference algorithm[8]). On the other hand, as explained above, the Bayesian networks computations can be done offline. Therefore the complexity of the module can be adapted to how often we want our Bayesian networks to be refreshed. The design makes it very flexible to change from a type of BN to another, and so, it is a decision that can be made later.

5.1.8 Personalisation

So far, we have one Bayesian structure per query category. The idea is to use the context in its first application: personalisation. The idea would be to maintain for each user personal Bayesian networks. This is of course unfeasible, both because of the time it would take each user to train it, and because of the amount of processing and storage power that would be required. On the other hand, if we group the people into stereotypes, we can reduce the number of instances enormously, thus enabling high volume

rollouts. We then expect to have better results, that is more accurate experts. The requirement on these stereotype groups is that they must be large enough, to ensure that the networks are sufficiently trained. At the very least, the members of a group must present some common, meaningful characteristic, such as students or retired people.

5.1.9 Natural Language Processing

We also investigated the possibility of using natural language processing in order to find the category of a query automatically, without any help of the user, but by founding some keywords and processing it semantically. We used an English parser in order to eliminate small words like “the” or “a” and also to have already a disambiguation for some words, by just knowing if it is a verb, an adjective or a noun. Then we look for a semantical meaning of each word, hoping to match one of the category types. This is far from being automatic and also from having a good success rate i.e. a category is not always determined But for some simple queries, like trying to find out a restaurant, it works.

5.2 Incentive system

Why should people answers to queries? It takes time, battery and bandwidth. There are a priori no reasons for it except a community sense and a fuzzy warm feeling of doing the right thing. That is why we have to build an incentive system, so that people get somehow rewarded for their cooperation. We have devised it so that people need to answer questions in order to be able to make them themselves.

5.2.1 Credits

We opted for a virtual money system which is currently only used for our application, but we can easily think that those credits to other applications, depending on the business model. New users start with a given amount of credits and can thereby make some queries from the beginning. To issue more queries, they will have to participate actively in the system.

According to the actions of the user, he earns or spends credits. The possible paying actions are described below:

By varying the values of this table, one can influence the qualities and the number of the answers per request. Finding an optimal value is out of the scope of this thesis.

To achieve the credits distribution, we put in place a digital cash system where the user can receive credits either from the application or from the

Action	Credit Amount
Issue a query	-10
Issue an answer	+2
Issue a rating	+1
Issued a good answer	+1
Issued a bad answer	-1

Table 2: The credits system

bank. To protect anonymity, we use a blinding signature scheme, as described in *Applied Cryptography* from Bruce Schneier [29].

Assume that Alice would be willing to pay 10 credits for a query, here are the steps followed by the three parties: AskMe, the bank and Alice.

1. Alice prepares n anonymous orders for 10 credits each. Sending more than one order allows the bank to verify that Alice does not cheat as it cannot verify the content of the orders, because they are blinded.
2. Alice appends a random string of 256 bits in each order, which is used as a money identifier to prevent double spending.
3. Alice blinds the n orders and sends them to the bank.
4. The bank asks Alice to reveal $n - 1$ blinding factors of its choice. It unblinds the $n - 1$ orders and verifies them and signs the remaining one blindly and returns it to Alice. It also deducts 10 credits from Alice's account.
5. Alice issues a query to the AskMe service under a virtual ID and adds her 10 digital credits document to the query, which she has previously unblinded.
6. AskMe verifies the signature.
7. AskMe sends the order to the bank.
8. The bank verifies its signature of the money and retrieves the random string and stores it in a database to prevent from double spending. It finally adds 10 credits to the AskMe account.

We used 1024 bits key pairs to implement the algorithm. Then the blinding and signatures of an order m is the following:

The bank has a public key e , a private key d , and a public modulus n . If Alice wants the bank to digitally sign an order m , the following step must be executed:

1. Alice chooses a random blinding factor, a value between one and n . Then she blinds m by computing

$$t = mk^e \pmod n$$

2. The bank signs t

$$t^d = (mk^e)^d \pmod n$$

3. Alice unblinds t^d by computing

$$s = t^d/k \pmod n$$

4. The result is a signed order from the bank

$$s = m^d \pmod n$$

This can easily be shown

$$t^d \equiv (mk^e)^d \equiv m^d k \pmod n, \text{ so } t^d/k \equiv m^d k/k \equiv m^d \pmod n$$

This digital cash system we put in place protects the anonymity of our users when paying the AskMe service. It could be updated easily to trace cheaters, by adding part of the identity of a user in the unique string. The major drawback of this algorithm is that it needs four rather big messages for withdrawals. It also requires statefulness, but it is quite easy to do that with Web service framework: this is simply done by setting a session cookie, or changing the scope of the service. There are surely lightweight protocols existing, as an example there is a proposal from Ham, Choi and Xie[15]. Unfortunately, their payment system is not in accordance with digital cash principles since the bank knows where the user spends the money.

5.2.2 Economic System

The behaviour of the money flow has not been validated. At the first look, it appears like a closed economy, where users start with a fixed amount of money, and then exchange it among themselves. This could turn like a poker game nightmare (or dream) if one of the users wins all the credits of the others. But the system has leaks and sources, e.g. an expert not answering “loses” potential money and a query answered by many experts and well rated will generate some money (because the AskMe service has virtually as much credits as it wants).

5.2.3 Reputation

Another incentive is to provide each user with a reputation. Having a good reputation has no direct effect on the system, except that the user can feel valorised by a better ranking and would feel more trust for an answer issued by a high reputed user. A user would receive reputation points every time he makes a good answer, and there are several reputation steps, from rookie to guru. We could imagine that reputed user get recommendations from preferably reputed experts and experiences then a higher quality service.

The reputation would be implemented in a similar fashion as the credits. A reputation token is requested at the bank before logging in, and sent with the log-in message. The difference is that one does not lose one's reputation by withdrawing it from the bank (the bank will not decrease your reputation amount). However this system has a flaw: a user could give his token to one person, and withdraw another token and give it to another person, etc.. This problem was not existent with the credits, since the user was losing credits by doing so. To avoid that, we can put the virtual ID in the reputation token, meaning that the Bank must verify the binding between the virtual ID and the real ID, and thus need the help of the Entity Manager.

5.3 Client Software

The client implementation is an important part if one wants to bring such a project to success. Since it has to run on a PDA, constraints on screen size, input modes, battery and computation capacity must be taken into account. We implemented the client also in Java and running on the J9 virtual machine that is Personal Profile 1.0 compliant. This means there are slightly less classes than in J2SE API.

5.3.1 GUI

A thing to keep in mind is that minimising an application on a PDA is the normal operation of the upper right button, so there is no need to define a "run in background" button. But closing must be explicit.

The GUI is built around a main screen where the user can see his current status: online, credits, reputation and last event received from the AskMe server. An event is typically that he received a new answer or query from another user. We used SWT from Eclipse as graphical toolkit since it offers a native look and feel, but generally needs to be imported on the platform.

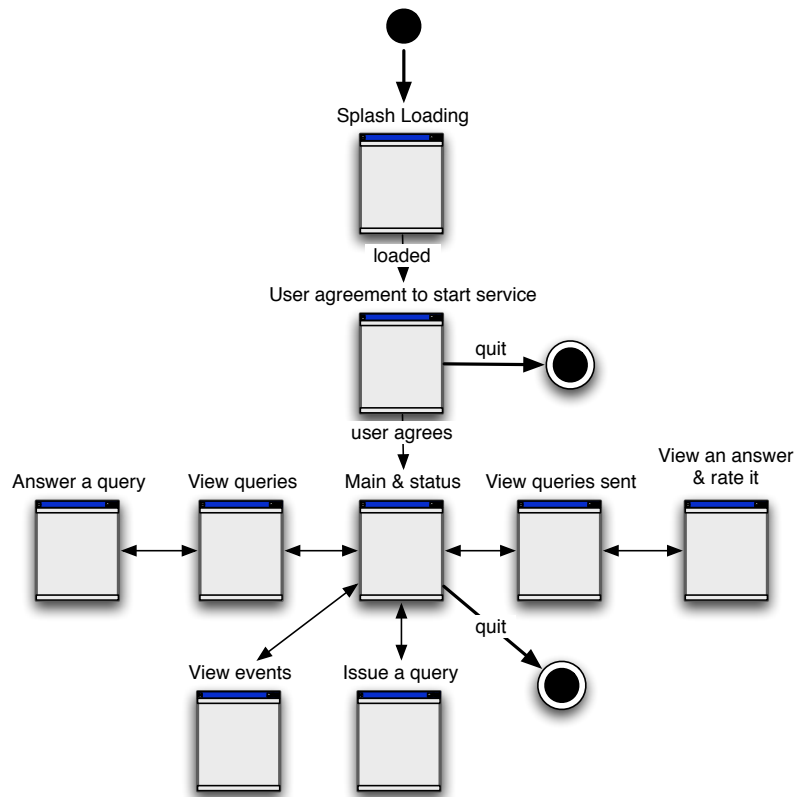


Figure 20: The GUI state machine

5.3.2 Web Service on PDA

Because of the limitation of the Personal Profile API and the little memory available, the Web Service client was generated using Sun's Wireless Toolkit (WTK) which can generate lightweight stubs for MIDP⁸ compliant JVM. It requires a small library and offers the fastest way to generate a WS client on a handheld from a WSDL. Two others solutions were discarded. Those were:

Axis Axis library is rather big (around 7 Mo) and still, it requires many classes which the Personal Profile API does not have, like `java.beans` or `java.sql` packages.

kSOAP2 This is a very viable solution, which offers a simple way to construct SOAP messages and does the marshalling/unmarshalling. But it requires much more coding than using the WTK tool.

⁸Mobile Information Device Profile

WTK was chosen because it was the easiest to use while fulfilling all our requirements. It has nevertheless only 5 primitive types against the 14 present in Axis. One must be careful, when writing the WSDL, to use only the supported types and follow the recommendations for interoperability[27] like using a document/literal wrapped style for the WSDL[17].

Queries in AskMe are asynchronous, that is when a user issue a query, many events will arrive much later than the call. In his basic form, Web service does not support that. One can couple it to JMS to offer an asynchronous framework. There is also the possibility for the enquirer to poll the Entity Manager from time to time to see if there is a message for him. But it consumes battery and also delays arrival of messages. We decided that the Entity Manager should push the messages to the clients, thanks to a Java objects stream. This is not a good solution, since it neutralises the advantage we had by using Web services, that is to be language and platform dependant. To improve this, we opt to send the message back in a XML form, which is then unmarshalled into objects at the client.

6 Test & Evaluation

One of the biggest inconveniences of this project is that we cannot perform a convincing validation of the system, since it is only a person who can measure the quality of a recommendation system. A computer could only generate dummy questions and answers. But we can validate the components of our system; we can see for example whether the Bayesian network is able to learn with some generated ratings. Hence we built a simple testing bed with a group simulator and rating agent.

6.1 Test Environment

6.1.1 Simulating Users

Miquel Martin has built a group simulator[22] in the frame of the IST-MobiLife project. It is a program running a virtual city with simulated people. They lead a simple life of a “sleeping-working-entertaining” cycle. Each simulated user has its own context and “freewill” that are customizable on purpose. For simplicity we generated users uniformly with no rules (i.e., a girl can be called John, be 10 and already have achieved a PhD). Of course, we pretend that all these people are always connected to the AskMe service and share their contexts. There is also a simple Web service, which allows us to mark the users on the map in order to mark the enquirer and the experts. A few of these context variables are listed in table 3.

In our simulations, there are about 20 context variables and 500 simulated

Context	Values
location	{{closer, close, far} to {home, location, office, destination}}
crime rate	{safe, low, medium, high, dangerous}
cuisine	{italian, thai, french, indian, german}
gender	{male, female}
activity	{asleep, working, partying, on his way}
atWork	{false, true}
hasWIFI	{false, true}
age	{teens, 20s, 30s, 40s, 50s, 60s, elderly}
etc.	{ ... }

Table 3: Contexts used for the test

users. The group simulator runs continuously and shares the contexts of the users by writing a CSV file occasionally which is then read by a Context Provider unit, which publishes the contexts through a Web service. The simulator also has a GUI, so the enquirer and his experts can be visualised

in real time.

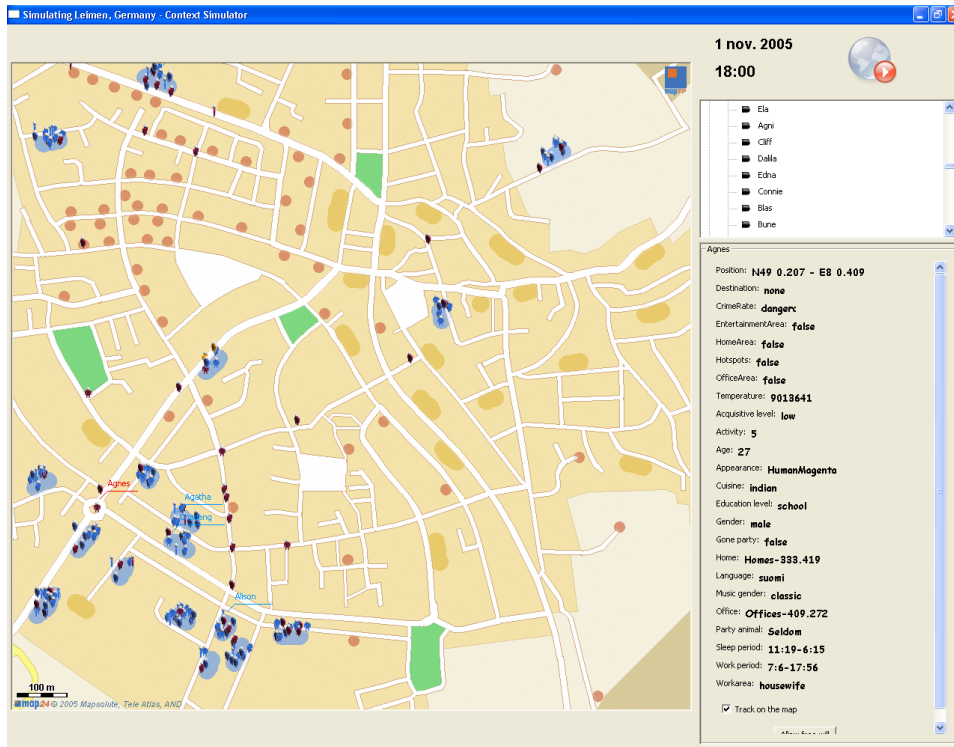


Figure 21: Simulator showing the enquirer (red) and the experts (blue)

Then the Group Module of the AskMe service can ask at any time for the contexts of the users. The Group Module then finds people matching the relevant contexts found by the Query Manager module. On the other side, there is a client simulator that generates queries using the virtual IDs used by the group simulator. We are not interested by what the client received from the system in return.

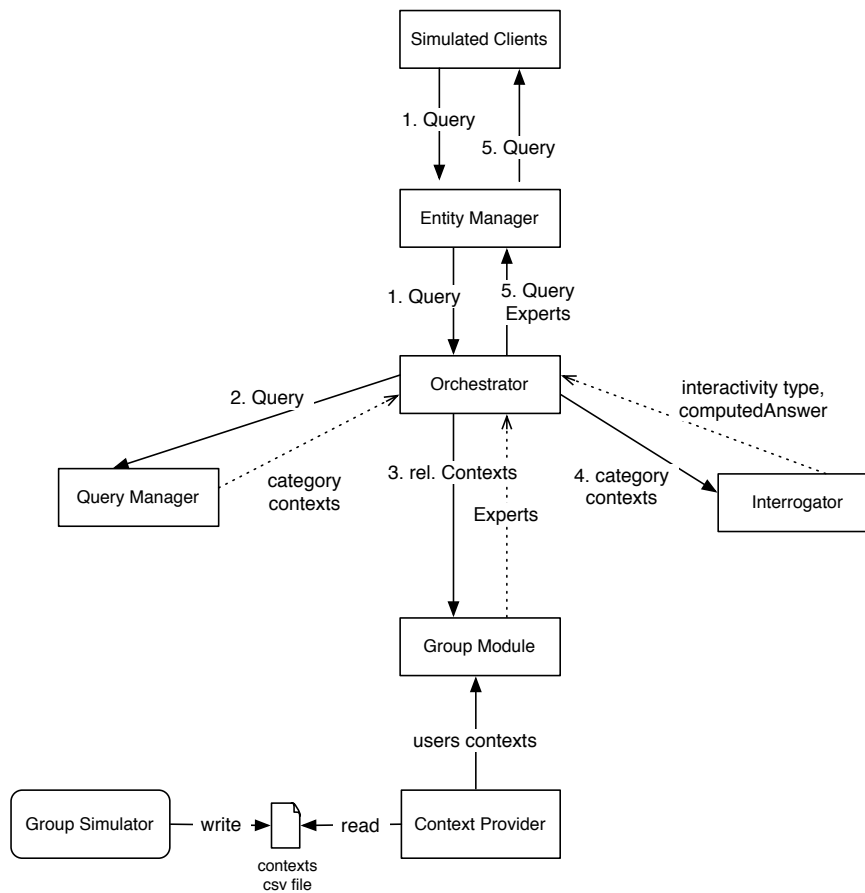


Figure 22: Simulation Testbed

6.1.2 Metrics

To test our performance, we have chosen to observe the rating evolution as metric, because it is our main preoccupation to satisfy the user. We assume that if we choose the right contexts, we will find the right experts. And if we find the right experts, the users will get good answers for their queries, which results in good ratings. That is far from reality, but it is already a start.

6.1.3 Rating Generation

We have to generate the feedback for the Bayesian network. In real life it would be the appreciation of the enquirer for each answer it receives, which is

not possible in simulation. Thus, we implemented a rating generator, which is found in the Orchestrator (that is the simplest way since the Orchestrator has full information).

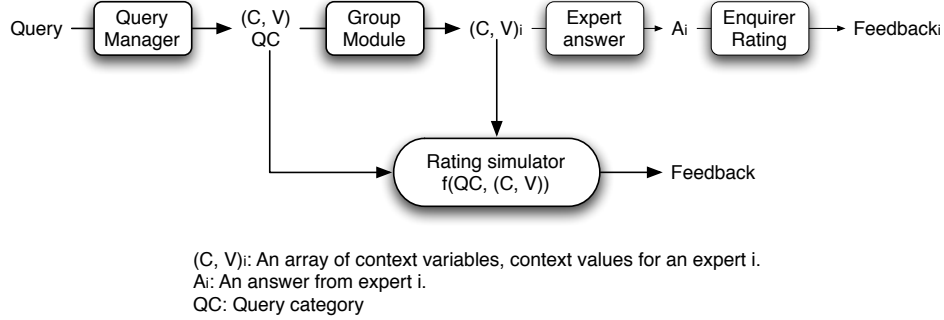


Figure 23: Generating the Feedback

For rating the queries, we define a set of predicates on the contexts. If a context satisfies one of these predicate, the rating is either incremented or decremented. The rating has its value in a range from 0 to 4, with an initial value starting at 1. Thereby the neutral value for the network training is 2. The scoring is deterministic and quite simple. We then expect that the

Context	Value	Score
age	between 18 and 30	+1
age	>60s	-1
position	in a radius of 500m of enquirer's	+1
at Entertainment	true	+1
party animal	> "going out sometimes"	+1
party animal	< "going out sometimes"	-1
music style	dance	+1
music style	electronic	+1
music style	classic	-1
education level	university	+1

Table 4: Scoring rules for the query category discotheque search

context variables-values that are adding a positive point to be output by the Query Manager.

6.2 Results

First of all, we could not make so many measurements as putting in place the system was long and the group simulator was only available short before

the end of this thesis. However, there is already some basic confirmation that the concept works. We expect to have more measurements and results for the oral presentation of this thesis.

As seen before, we can generate the feedback of the Bayesian network by either base the rating on the output of the Bayesian network or on the actual contexts of the experts. The network always starts with no particular belief; before the first training, it chooses context variables randomly. For the following measurements, we always have the same question category: “looking for a discotheque”.

6.2.1 Short-circuit training

Here we show the evolution of the rating measured based on the output of the Query Manager and use this rating as feedback for the network learning. There is no need for a group simulator for this simulation. The Query Manager is trained with its own output, which makes this simulation as simple as possible.

- *Configuration*

Population: None

Training: Based on rating generated from the Query Manager output

Training rate: Network trained every 20 queries

Curve: Smoothen over 5 measures

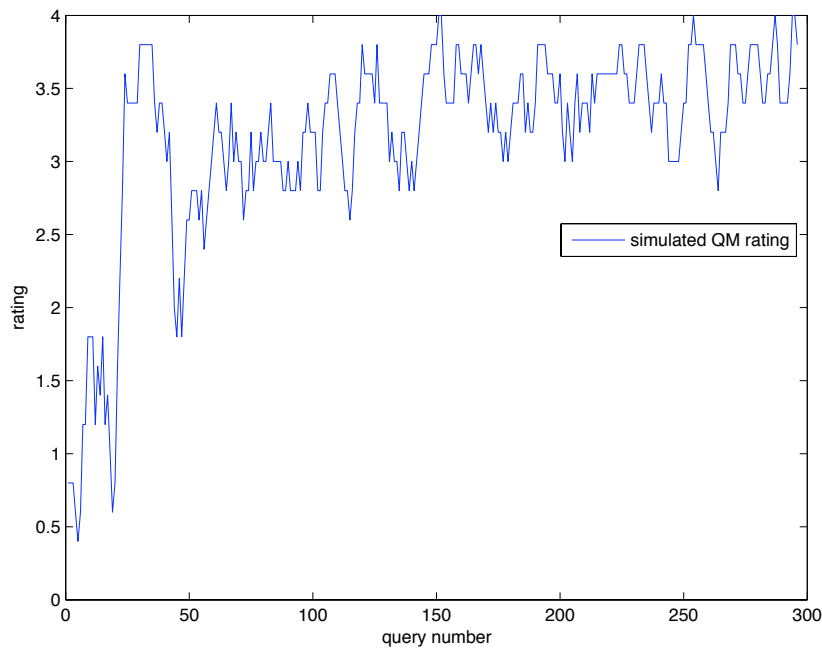


Figure 24: Simple scenario evaluation

We recall that the score is a value between 0 and 4. We can see that the score starts badly because the BN has no idea which contexts to choose as it has a uniform belief at start-up. One notices that the BN is refreshed every 20 queries. This can be particularly seen after the first refresh, where there is a huge performance increase. We finally remark that the rating stabilises itself around 3 and 4 after a while, which is exactly what we wanted to observe. It

cannot achieve a steady 4 because of the randomly chosen context variables. Furthermore, we can see that after only 20 queries, the network already outputs good results.

6.2.2 TAN performance

As a second measurement, we propose to compare the output of a TAN with a naïve Bayesian network. The training is yet based on the ratings of the experts.

- *Configuration*

Population: 500 users

Number of experts: 5 per query

Training: Based on the simulated experts ratings

Training rate: Network trained every query

Curve: Averaged over the experts ratings and smoothen over 4 queries

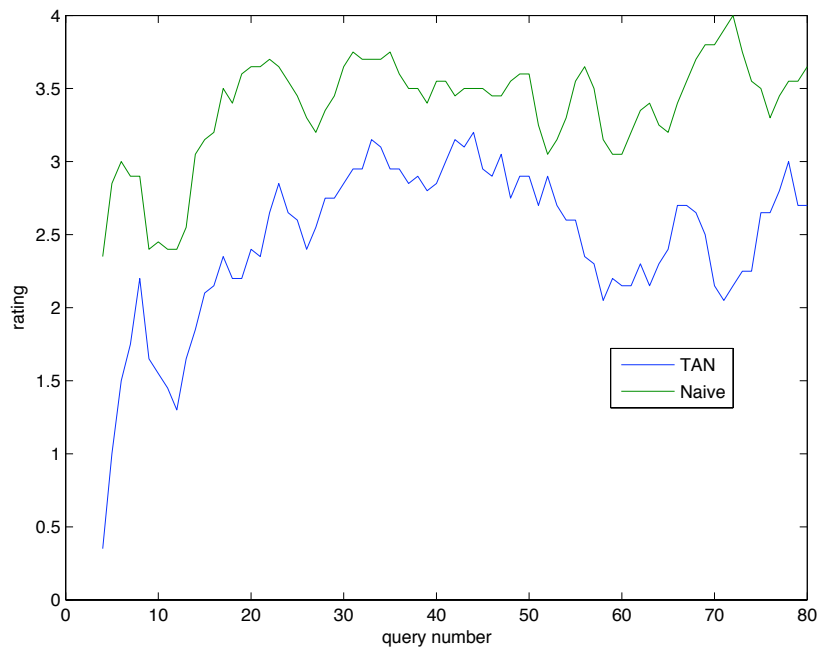


Figure 25: TAN and naïve structures performance

The results can be quite counter-intuitive. However we have to recall that the contexts of the users were generated uniformly, thus contexts should

have a small correlation between them. Although the network is trained through reinforcement, one should observe the dependencies between the variables that influence the score. However, in such a simple environment, the dependencies learning seems to be an handicap and the naïve network converges faster to a higher success rate. The TAN learning algorithm has also a threshold parameter that influences how many additional dependency arcs are generated. It might be interesting to test this parameter.

In a real life scenario, where there would be more noise, the TAN network might be better. With normal classifier problems, it depends on the data whether it is better to have a TAN or a naïve network. In our case, this is slightly more complicated since we influence the data with the reinforced learning.

6.2.3 Group Module quality

At last we compare the rating generated at the output of the Query Manager and the rating generated from the experts. The Bayesian network is naïve and its training is based on the experts ratings.

- *Configuration*

Population: 500 users

Number of experts: 5 per query

Training: Based on the simulated experts ratings

Training rate: Network trained every query

Curve: Averaged over the experts ratings and smoothen over 4 queries

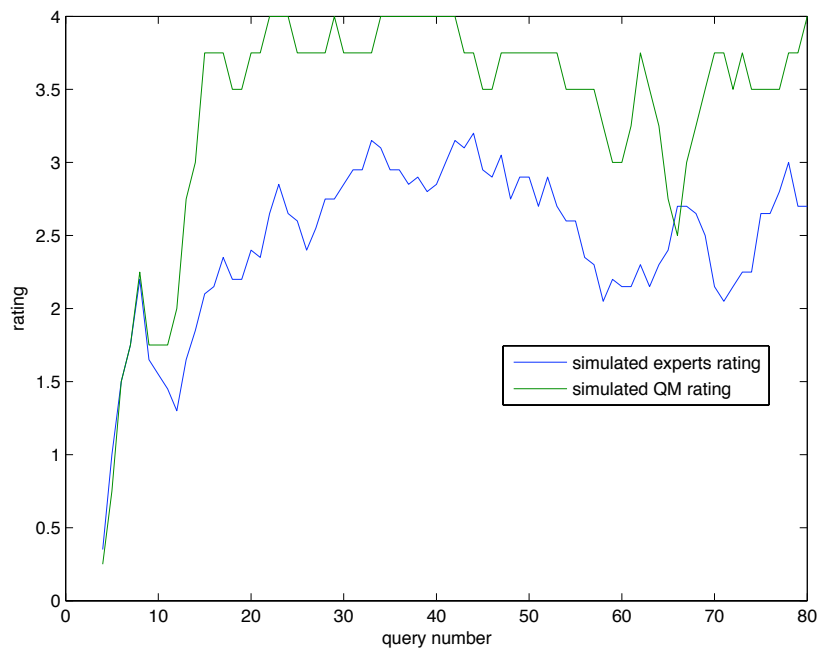


Figure 26: Experts finder quality

One can observe that the Group Module cannot always find the people satisfying the context predicates output by the Query Manager. In that case, the experts' contexts differ from what has been requested by the Query Manager and that is why we observe that the experts' ratings are generally below the Query Manager generated ratings. This is not surprising at all and it will be the same case most of the time in real life that there are not enough people matching the relevant contexts. The counter peak at query

65 is due to the Group Module that outputs an expert with a good context that the Query Manager did not request.

6.3 Discussion

We do not propose an optimal solution yet, for we are undecided if naïve Bayes is better than TAN for example. We do not know under which conditions one is better than the other. This needs further investigations.

We compared the performance according to the training rate of the network. Without surprises, the more it is trained, the quicker the rating improves. We are quite satisfied with the results, which show that the network can output efficient context variables after only 20 queries and then slowly improve to a rating around 3 - 3.5. We hope by specialising our networks with stereotypes, we can improve this long-term performance.

There are other measurements we could do, like comparing the performance of the rating with the time of the day. Thus one can possibly observe repeatable fluctuations since particular contexts are time-dependant, like the “atEntertainment” context. These fluctuations of performance would be perfectly normal. Another possible evaluation would be to observe the performance in relation with the number of users and possibly deduce a minimal amount of users in order to have a good rating. This number would be the minimal amount of users needed for a quality of service. Finally, other scenarios (i.e.. question categories) should be developed and also the rating generator system could be a bit less deterministic in order to have some natural noise, because so far it is the Query Manager that adds some random contexts, but that should not be its role to add noise.

7 Conclusion

We have presented what we believe to be an innovative information system where context is not only used for personalisation, as it is often the case for ubiquitous computing applications, but also to find experts to build a novel information retrieval system.

AskMe integrates successfully a myriad of different areas. We have designed and implemented eight Web services, which five of them are specific to AskMe. We integrated the application with a group simulator in order to test our system, which is in ubiquitous computing hardly done. The results show that the learning is effective and give us a lot of promises for future work. Furthermore privacy has been one of our main concerns and is duly protected in our service through architecture design, pseudonyms and cryptographic mechanisms. Last but not least, we have put in place an effective incentive based system that should ensure a quality of service for our application. AskMe was a vast project and we had much pleasure having our hands on such many areas.

We have seen through the examples given in the related work section that collaborative knowledge sharing is a current trend, and thus our approach seems viable for a mobile community. To deploy such a service, there are still some challenges to face. Still, we are confident that such a service can exist in the near future. If the people are ready to accept being judged by an automatic system whether they are experts regarding their knowledge about specific questions remains an open issue.

7.1 Future Work

AskMe covers a broad range of computer disciplines. Possibly each part of the system could be improved to a state of the art. Some improvements were already suggested in this thesis. The project just offered a glimpse of what could possibly be done. Below, there is a list of possible improvements according to the field.

The study on the virtual IDs management was very light. We think that we can deepen into a robust framework for managing IDs, but virtual IDs management can be a thesis on its own. Concretely, the role of the Entity Manager as an anonymizer *and* a virtual IDs repository is discussable. This should be split into two entities. Also, virtual ID management could be done in a broader way, among several services, since enhancing privacy does not only concern AskMe but any context-aware service. A paper offering a framework for providing a virtual ID for several services[19] and also to protect the IP through a couple of proxy, la Onion routing[12], has been proposed by Hauser[16].

Furthermore we did not use much of the capacity of ontologies, which represents a weakness. It could have helped defining context variables and then link efficiently to a natural language processing algorithm, in order to extract from the question string, whether some contexts are needed. For example, knowing that “position”, “home” and “destination” are location variables can help.

The query categorisation is one of the Achilles’ heels of our system, since it is not yet automatised and is a major factor of success in finding the right experts. We think that having the user choose the category of its question, and then using some NLP on the query could help finding sub-categories or variations.

The system should be designed for the user. It is extremely important to have an efficient GUI and a good mechanism to remunerate the efficient contributors. One drawback of the service is that to get three answers from the system, it will also require the people to answer three questions to issue themselves only one query. Due to the limitation of handhelds devices that usually do not have a hardware keyboard, answering three queries can take a long time. So far, we think that the service will be used by only a few people that would agree to spend enough time with the application. We should optimise queries and answers input the most as we can, by implementing macros that help the user quickly constructing his query. For example, we could integrate location directories, so a user can quickly indicate the location. Moreover integration of chatting and PoC to connect an enquirer with his expert could be an improvement to have more details about a particular answer.

The gathering of the contexts of all the user hardly scales, e.g. to know if there is an user at one particular place, one must retrieve the location context of all the users! It still works with a few thousand users, but what if millions of users join. Also, the algorithm of the Group Module would need improvements in order to prune quickly ineligible users, since so far it would give a score to each user for each query, which does not scale.

7.2 Acknowledgements

These last lines not only conclude this thesis but also my five years of study at EPFL. I would like to thank all the professors that have taught me during this pleasant time of my life. I do not regret at all to have chosen to study Communication Systems and feel confident about my future.

More personally, I would like to thank my parents for the never ending support they gave me, even though I left them to go abroad several times. Nadine, to have supported me at every instant and for her unconditional love. I am thankful to all my friends from EPFL for all the good time we

had together, from the pain of revising exams to the entertaining time we had at Satellite, and I wish a lot of success to every one of them.

The last six months in Heidelberg have been wonderful, though the weather was colder than I expected. The staff at NEC is open-hearted and competent, and that has made the work there much easier. I had a wonderful time with the students from many different horizons and will miss this good atmosphere. I would like to thank Julia, my roommate, who kindly cooked for me from time to time and helped me integrate to the German life. I thank Miquel Martin for his 6 months of supervising, I hope I have not disappointed him with my work done. I am also grateful to the last corrections brought by Oana Jurca.

I hope the few readers of this report got it why I have chosen Moka as a first name for the Finnish student. Otherwise you should start reading this thesis again, from the very first page!

A Installation and user guide of the software

By the end of the project, all modules should be available from CVS, compilable and possible to be run from an Ant file. Web services are packed into war files and the client software into a jar file. Also, there are scripts available to generate the database tables.

PDA Client The Java programmed was intended to run for Personal Profile 1.0 configuration. A compliant virtual machine is J9 from IBM, which can be run on many platforms. The client has to be compiled compliant to Java SDK 1.3 and has to follow the restrictions of the PP 1.0. The installation of the virtual machine should be done prior to running the program, obviously.

There are two things that prevent portability. The virtual machine must have the native library of the SWT. So for each platform, a different SWT library must be delivered. Furthermore, the root path-name within the Java client program must be explicitly given, meaning that pathnames to files are hard-coded. These two problems can be easily tackled for any installation, but needs user's manipulation.

Web Services Each module can be installed on a different computer. Two things are to be configured. One is the address of the WS registry (since we do not use a public UDDI registry). And secondly, the address of the database and the datasource configuration must be set according to the environment.

Database When installing the database, one must first create all the tables and accounts using the SQL scripts. Note that the scripts will only work with MySQL (since accounts privileges are DBMS dependant). The table containing the contexts are generated from a Java program in the Query Manager, since it is generated dynamically from the contexts listed in an enumeration class.

B Glossary

Ant	A Java build tool
ARFF	Attribute-Relation File Format
Axis[2]	A Java platform for creating and deploying web services applications
BN	Bayesian Network
BouncyCastle[25]	A cryptographic API for Java
CSV	Comma Separated Values
DBMS	Database Management System
FOMA	Freedom of Mobile Multimedia Access
IR	Information Retrieval
J9	Java Virtual Machine that is supported by many platforms
JavaBayes[9]	A library for Bayesian networks in Java
JCA	Java Cryptography Architecture
JCE	Java Cryptography Extension
JDMAPI	Java Data Mining API
jUDDI[3]	Java open source implementation of UDDI
JVM	Java Virtual Machine
JWNL[30]	Java WordNet Library: API to access WordNet relational dictionary
LSIR	Laboratoire de Systèmes d'Information Répartis
OASIS	Organisation for the Advancement of Structured Information Standards
NEC	Nippon Electrical Company
NLP	Natural Language Processing
PP	Personal Profile
SDK	Software Development Kit
SOAP	Simple Object Access Protocol
Stanford Parser[33]	Java implementation of probabilistic natural language parsers
SWT	Standard Widget Toolkit
TAN	Tree Augmented Naïves Bayes
UDDI	Universal Description, Discovery and Integration
W3C	World Wild Web Consortium
Weka[26]	An external library for machine learning
WordNet[32]	A lexical reference where English nouns, verbs, adjectives and adverbs are organised into synonym sets, each representing one underlying lexical concept
WS	Web Service
WSDD	Web Services Deployment Descriptor
WSDL	Web Services Definition Language
WSS	Web Service Security
WSS4J[4]	AN API for WSS for Java
WST[31]	Web Service Tools
WTK	Wireless Toolkit, Sun's toolkit for handheld applications
WTP	Web Tools Platform

C Modules Interfaces

Below, the interfaces of the main modules involved in AskMe are presented.

java.mi.Remote
«interface» EntityManagerExternal
<ul style="list-style-type: none"> + makeQuery(String, Query) : void + logOut(askme.entitymanager.LogOutRequest) : void + heartBeat(askme.entitymanager.HeartBeatRequest) : void + initiateChat(askme.entitymanager.InitiateChatRequest) : void + getAdvertisement() : java.lang.String + sendRating(askme.entitymanager.SendRatingRequest) : askme.entitymanager.SendRatingResponse + sendResponse(askme.entitymanager.SendResponseRequest) : askme.entitymanager.SendResponseResponse + debug(askme.entitymanager.DebugRequest) : void + log(askme.entitymanager.LogRequest) : askme.entitymanager.LogResponse

java.mi.Remote
«interface» OrchestratorExternal
<ul style="list-style-type: none"> + makeQuery(askme.orchestrator.MakeQueryRequest) : void + login(askme.orchestrator.LoginRequest) : void + logOut(askme.orchestrator.LogOutRequest) : void + initiateChat(askme.orchestrator.InitiateChatRequest) : void + getAdvertisement() : java.lang.String + sendResponse(askme.orchestrator.SendResponseRequest) : askme.orchestrator.SendResponseResponse + sendRating(askme.orchestrator.SendRatingRequest) : askme.orchestrator.SendRatingResponse

java.mi.Remote
«interface» QueryManager
<ul style="list-style-type: none"> + queryProcess(askme.querymanager.ProcessQueryRequest) : askme.querymanager.ProcessQueryResponse + getAdvertisement() : java.lang.String

java.mi.Remote
«interface» GroupModule
<ul style="list-style-type: none"> + findCandidates(Context) : askme.types.CandidateReport[] + getAdvertisement() : askme.groupmodule.GetAdvertisementResponse

java.mi.Remote
«interface» Interrogator
<ul style="list-style-type: none"> + processQuery(askme.interrogator.ProcessQueryRequest) : askme.interrogator.ProcessQueryResponse + getAdvertisement() : java.lang.String

java.mi.Remote
«interface» EntityManagerInternal
<ul style="list-style-type: none"> + transmitAnswer(askme.entitymanager.TransmitAnswerRequest) : void + transmitAnswer1(askme.entitymanager.TransmitAnswer1Request) : void + sendParameter(askme.entitymanager.SendParameterRequest) : void + getAdvertisement() : java.lang.String + transmitCredits(askme.entitymanager.TransmitCreditsRequest) : java.lang.String

java.mi.Remote
«interface» Bank_PortType
<ul style="list-style-type: none"> + signOrder(askme.bank.SignOrderRequest) : askme.bank.SignOrderResponse + revealOrders(askme.bank.RevealOrdersRequest) : askme.bank.RevealOrdersResponse + deposit(askme.bank.DepositRequest) : askme.bank.DepositResponse + getBalance(askme.bank.GetBalanceRequest) : askme.bank.GetBalanceResponse + getAdvertisement() : java.lang.String

java.mi.Remote
«interface» WSRegistry_PortType
<ul style="list-style-type: none"> + publish(askme.wsregistry.PublishRequest) : void + inquire(askme.wsregistry.InquireRequest) : askme.wsregistry.InquireResponse + getAdvertisement() : java.lang.String

java.mi.Remote
«interface» ContextProviderSoap
<ul style="list-style-type: none"> + getAdvertisement() : org.ist_mobilife.www.v0_3.context.ContextProviderAdvertisement + getContext(org.ist_mobilife.www.v0_3.context.ContextQuery) : org.ist_mobilife.www.v0_3.context.ContextElement[] + getLastContext(java.lang.String, java.lang.String) : org.ist_mobilife.www.v0_3.context.ContextElement + pushContext(org.ist_mobilife.www.v0_3.context.ContextElement) : void + pushLastContext(java.lang.String, java.lang.String) : void + addEntity(org.ist_mobilife.www.v0_3.context.Entity) : void + subscribeToContext(org.ist_mobilife.www.v0_3.context.ContextSubscription) : java.lang.String + updateSubscription(java.lang.String, org.ist_mobilife.www.v0_3.context.ContextSubscription) : java.lang.String + removeSubscription(java.lang.String) : boolean + sendCommand(org.ist_mobilife.www.v0_3.context.ContextProviderCommand) : boolean + isFeaturingEntity(java.lang.String) : boolean + isAvailableContext(java.lang.String) : boolean

D Tree Augmented Naïve Bayes sample

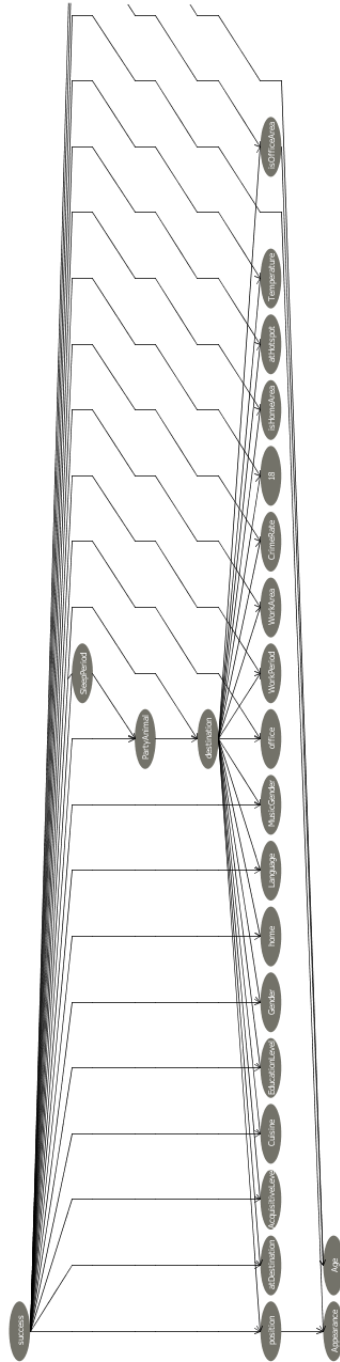


Figure 27: TAN of a sample dataset

References

- [1] Gregory D. Abowd, Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggle. Towards a better understanding of context and context-awareness. In *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pages 304–307, London, UK, 1999. Springer-Verlag.
- [2] Apache. Axis. Available from: <http://ws.apache.org/axis/>.
- [3] Apache. jUDDI. Available from: <http://ws.apache.org/juddi/>.
- [4] Apache. WSS4J. Available from: <http://ws.apache.org/wss4j/>.
- [5] P. J. Brown. The stick-e document: a framework for creating context-aware applications. *Electronic Publishing*, 8:259–272, June-September 1995.
- [6] Mauro Brunato, Roberto Battiti, Alessandro Villani, and Andrea Delai. A location-dependent recommender system for the web, November 2002. Available from: citeseer.ifi.unizh.ch/article/brunato02locationdependent.html.
- [7] Alistair Cockburn. *Writing Effective Use Cases*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000.
- [8] F. Cozman. Generalizing variable elimination in bayesian networks, 2000. Available from: citeseer.ifi.unizh.ch/cozman00generalizing.html.
- [9] F. Cozman and André Hideaki Saheki. Javabayes 0.41. Available from: <http://www.pmr.poli.usp.br/ltd/People/asaheki/>.
- [10] P. Floréen, M. Przybilski, P. Nurmi, J. Koolwaaij, A. Tarlano, M. Wagner, M. Luther, F. Bataille, M. Boussard, B. Mrohs, and S. Lau. Towards a context management framework for Mobilife. In *IST Mobile & Communications Summit*, Dresden, 19-23 June 2005. Available from: <http://www.cs.helsinki.fi/u/ptnurmi/papers/285.pdf>.
- [11] Organisation for Economic Cooperation and Development (OECD). Guidelines on the protection of privacy and transborder flows of personal data, 2001. Available from: www.oecd.org/dsti/it/secur/PRIV-EN.
- [12] David Goldschlag, Michael Reed, and Paul Syverson. Onion routing. *Commun. ACM*, 42(2):39–41, 1999.
- [13] Google. Google Answer. Available from: <http://answer.google.com>.

-
- [14] Marco Gruteser and Dirk Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *MobiSys*, 2003.
- [15] Wooseok Ham, HyungChoi, Yan Xin, Misung Lee, and Kwangjo Kim. Secure one-way mobile payment system keeping low computation in mobile devices. In *Proc. of WISA2002*, pages 287–301, 2002.
- [16] C. Hauser. A new approach for privacy-preserving communication by combining virtual identities with mobility management. In *European Symposium on Research in Computer Security (ESORICS 2002) - Poster Session*, Zurich, 2002.
- [17] IBM. Which style of wsdl to choose? Available from: <http://www-128.ibm.com/developerworks/webservices/library/ws-whichwsdl/>.
- [18] Alfred Kobsa and Jörg Schreck. Privacy through pseudonymity in user-adaptive systems. *ACM Trans. Inter. Tech.*, 3(2):149–183, 2003.
- [19] M. Koch and W. Würndl. Community support and identity management. In *Proc. European Conf. on Computer Supported Cooperative Work (ECSCW 2001)*, Bonn, Germany, 2001.
- [20] Rajan M. Lukose, Eytan Adar, Joshua R. Tyler, and Caesar Sengupta. SHOCK: communicating with computational messages and automatic private profiles. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 291–300, New York, NY, USA, 2003. ACM Press.
- [21] Lycos. Lycos iq beta. Available from: <http://iq.lycos.de>.
- [22] Miquel Martin and Petteri Nurmi. A context simulator for data generation and application testing in mobile scenarios. To be submitted to The 3rd Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, *Mobiquitous 2006*, 2006.
- [23] Richard E. Neapolitan. *Learning Bayesian Networks*. Prentice Hall, April 2003. Available from: <http://www.amazon.com/exec/obidos/redirect?tag=citeulike-20&path=ASIN/0130125342>.
- [24] OASIS. Web Services Security (WSS). Available from: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.
- [25] Legion of the Bouncy Castle. Bouncy Castle crypto APIs. Available from: <http://www.bouncycastle.org/>.
- [26] University of Waikato. Weka 3 : Data mining software in java. Available from: <http://www.cs.waikato.ac.nz/ml/weka/>.

-
- [27] Web Services Interoperability Organization. Ws-interoperability. Available from: <http://www.ws-i.org/>.
- [28] Bill N. Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *IEEE Workshop on Mobile Computing Systems and Applications*, December 1994.
- [29] Bruce Schneier. *Applied cryptography (2nd ed.): protocols, algorithms, and source code in C*. John Wiley & Sons, Inc., New York, NY, USA, 1995.
- [30] SourceForge. JWNL Java Wordnet Library. Available from: <http://sourceforge.net/projects/jwordnet>.
- [31] Sun. Sun java wireless toolkit. Available from: <http://java.sun.com/products/sjwtoolkit/>.
- [32] Princeton University. Wordnet, lexical database for the english language.
- [33] Stanford University. Stanford lexical parser. Available from: <http://www-nlp.stanford.edu/software/lex-parser.shtml>.
- [34] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, second edition, June 2005. Available from: <http://www.amazon.fr/exec/obidos/ASIN/0120884070/citeulike04-21>.