# Trust and Mobility aware Service Provision for Pervasive Computing

Liam McNamara, Cecilia Mascolo and Licia Capra

Department of Computer Science, University College London
Gower Street, London, WC1E 6BT, United Kingdom
{l.mcnamara | c.mascolo | l.capra}@cs.ucl.ac.uk

**Abstract.** Successful provision of services in pervasive networks represents a major challenge. The requester of a service or resource should be able to decide whom to rely on, not simply based on the apparent QoS that various providers offer, but also based on the *trustworthiness* of the providers, as well as their *mobility* patterns. Existing approaches offer a rather simplistic solution to the problem: either resources and services are discovered and selected purely based on their name/category, or at most by looking at the QoS promised by the various providers. However, there is no guarantee for the client that the provider will indeed deliver the advertised QoS; moreover, given the dynamicity of the pervasive setting, there is no guarantee that the selected provider will be available to the requester for the duration of the service. In this paper we present a service provision framework that reasons about mobility and trust to enable effective service selection in mobile ad-hoc networks. We illustrate our ideas through a scenario of service sharing on public transport where passengers are able to select other passenger's devices for information transfer, based on a novel combination of quality attributes (e.g., network connectivity), trust and mobility.

## 1 Introduction

Mobile devices and embedded computers are gradually becoming pervasive in modern life. An extremely large percentage of people already possess devices such as portable music players and mobile phones. These devices are quickly growing both in terms of supported functionalities and computational capabilities. Crucially, an increasing number of these devices are being networked, so that ad-hoc networks will rapidly emerge, thus enabling services such as file sharing, interactive games and information exchange, among these mobile peers. To truly realise Mark Wieser's ubiquitous computing vision, the management and operation of these devices cannot be explicitly human controlled. Rather, spontaneous communication and co-operation must be enabled, with the burden of configuring devices being abstracted away from the user, thus fostering a simpler and more satisfactory user experience.

The concept of *service orientated* computing is commonly applied to Web applications in order to highlight the fact that the unity of exchange on the network is in fact a service. This concept is probably even more useful in wireless environments, where there is little fore-knowledge of which services and peers will be available at any given time and location. Therefore the ability to dynamically locate and compare available services is crucial. While the vision to offer pervasive services is very appealing, it also comes with a number of challenges. Most importantly, the ability to seamlessly search for the desired service, and to select the one which has specific *qualities* that will maximise the user's utility. Two major issues underlie this challenge: first, the ability to assess whether a provider will indeed deliver a service at the promised QoS. Service providers may inflate their service advertisements, in order to secure a client's request. We thus need to provide the device with the ability to reason about the *trustworthiness* of the various providers, so as to make accurate predictions about actual QoS (as opposed to advertised/promised QoS), and consequently maximise the chances of a satisfying service delivery. Second, the service selection mechanism must account for an orthogonal parameter, that is, *mobility*. For example, we must maximise the chances that the service requester and provider will be co-located for a sufficiently long period of time, in order to complete the service delivery.

Some solutions in this space have been presented, especially on intelligent QoS-aware routing in multi-hop networks [8], though less attention has been given to higher level tasks such as *service selection*. There is currently a lot of focus on security and robustness in mobile networks. One approach is maintaining reputations about the performance of hosts in a system to discourage misbehaviour. When applied to service selection, these reputations allow decisions to be made about the expected future performance of a peer providing (or consuming) a service. A single centralised repository is often employed to keep track of global reputations, such as in CONFIDANT [9]. Unfortunately, this leads to a single point of failure and load in the system, thus also limiting scalability. Distributed global trust methodologies have also been proposed [7], but connectivity to a sufficient number of non-malicious peers is required. This is useful in overlay networks on the Internet where any host in the network can feasibly be communicated with, but less so in mobile networks. Some local reputation systems have been suggested, as used by Dewan et al [5], where each peer maintains its own opinion of other peers, which we believe offer greater flexibility and robustness. We consider the requirement of having a tamper-proof security module in each device [1] a key weakness. This causes the whole reliability of the framework to depend on the security of the module, not to mention the lack of widespread deployment of these modules. One approach which seems quite close to ours is [4]. In that paper, a model for trust based service interaction in decentralised systems is presented. However the selection of the provider there happens in quite a quantitative way and quality is hardly considered. Furthermore, we consider mobility as an essential qualitative measure on which the service provider must be chosen; we have not seen any approaches making use of this information so far. The trust model used in [4] is also quite limited in the sense that the long-term storage of reputations is centralised.

In this paper, we propose a framework where each peer can independently reason about the QoS and reliability of other peers to select the 'best' service provider available. Every peer maintains a local repository of trust information on recent/frequent hosts it has interacted with. These trust values will be modified based on the outcome of previous service provisions and, more precisely, based on how accurately the provider represented itself. No infrastructure is required by this approach. The lack of a global opinion is in accordance with the intuition that trust and utility are subjective concepts, so they will vary between users and social groups. This paper builds on previous work [3], where we outlined a QoS based service discovery and selection framework, now with the addition of trust reasoning linked to quality of service. Furthermore, we have introduced mobility as a qualitative parameter to reason about when choosing a service provider, in a way which we have not seen done before. In this paper, we are only considering single-hop communication, so routing concerns are not being addressed. In order to illustrate the functioning of our service selection framework, we present a case study related to service provision on public transport, where heterogeneous bandwidth and network provision is present, and where different users have different mobility patterns (e.g., get off at different stops). This scenario represents a useful application in real world situations and can be deployed with today's technology.

The paper is organised as follows: Section 2 contains a description of the public transport scenario and its requirements. Section 3 describes the key characteristics of our approach and its operation. Finally Section 4 concludes the paper and outlines future work.

## 2 Scenario

In this section we introduce a public transport scenario that highlights the pervasive service provisioning issues we intend to tackle with our approach. The scenario also gives an idea of the possible applications that would benefit from our approach.

### 2.1 Service Provision on Public transport

In big cities, such as London, a very large number of people commute between suburbs and the centre on public transport (e.g., buses and trains). Commuters on these vehicles are usually in quite close proximity, most carry handheld devices with one or more network interfaces (e.g., 802.11, Bluetooth, GSM), their patterns of mobility are quite "seasonal" (in the sense that they travel usually at the same time, repeating the same path day after day), and tend to stay on the vehicle for quite a prolonged period of time[1]. In addition, devices are often diversely equipped: some have GPS receivers, others have embedded cameras, sensing abilities (e.g., temperature, light) etc. The development of smaller and more accurate measuring devices, such as the Galileo project [6], will only increase the possible range of uses. As a result, a wide variety of services could be offered or shared among people, through their devices. To mention a few:

[1] This is also true for people in coffee shops as indicated by [4].

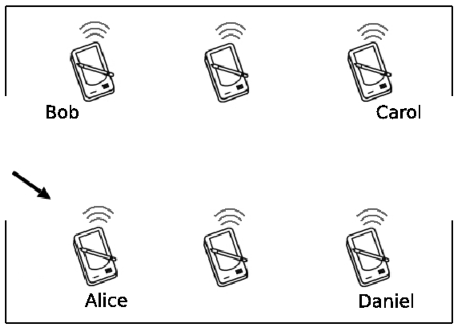

**Fig. 1.** Train scenario diagram

- Location information sharing: a device with a GPS receiver could be serving location information to others.
- Exact time information: a GSM device could offer this.
- News headlines, stock market levels: someone able to access the Internet through a GPRS phone could forward fresh information to others.
- Gaming: devices could participate in a shared game for the duration of their trip.
- Software components: new applications/functionalities could be shared and downloaded from a peer.
- Information about traffic and delays: commuters traveling in different directions could inform each others.

Although there may not be any apparent gain in offering a GPS position to other peers (waste of resources), it may actually increase the device's reputation, which should prove useful in future.

To illustrate a possible scenario, let us assume that Alice gets on a busy train with her PDA (see Figure 1). Many other passengers have mobile computing devices, including Bob, Carol and Daniel. As soon as she gets in, Alice starts her 'Travel Planner' program, causing her PDA to broadcast a request for location information. The request will include quantifiers about the accuracy and age of the information required. The location will then be frequently requested at regular intervals, to enable Alice to plot her movement on her screen. Any device that receives this request, and that is providing a matching service, can potentially respond to her. Let us assume that both Bob, Carol and Daniel have a GPS receiver (i.e., they are able to provide the service), and that they send back a response, containing all pertinent attributes of their service (i.e., accuracy and aging). Moreover, each device maintains information about the user's mobility pattern, based on his/her past journeys; this information is also sent back to Alice, so that her device may estimate for how long she will be co-located with each service provider, thus being able to make a selection that tries to minimise resource expensive switches of service providers. After receiving responses from co-located devices, Alice can then decide who to rely on. This will involve pruning all unacceptable responses that do not satisfy her requirements (e.g., accuracy lower than a minimum threshold). The remaining services are all feasible to use, but Alice should select the 'best', according to her own utility. To this end, let us assume that Alice is mainly concerned with accuracy of the location data. A first ranking of the short-listed service providers will then be done based on the advertised accuracy, calibrated by how much Alice trusts these claims. Let us also imagine that, besides accuracy, Alice is interested in minimising switches among service providers. The ranking will thus be influenced by the probability of each service provider remaining co-located with Alice for as long as she is expected to remain on the train. As a result, it may be that Carol promises better accuracy than everyone else; however, if Alice's trust in her is low as a result of past interactions, Carol will actually go down in the ranking. Similarly, it may be that Daniel promises higher accuracy than Bob, but he is expected to get off at the next station, while Bob is predicted to leave after Alice. In such a situation, our framework would put Bob at the top of the ranking.

The interaction with Bob can then be attempted, and the communication initiated using the relevant encoding and protocols. Depending on the outcome, Alice's trust in Bob will face either an increase or a reduction. The magnitude of any reduction will depend on the difference between promised and perceived service quality; in doing so, a problem that may not be the intentional fault of Bob should not be punished as much as an obvious QoS inflation on his part. The application would be able to disregard a location value that is wildly different from the previous reading (if taken recently), and signal to the middleware a failure has occurred. Conversely, if the experienced QoS is very close to the promised level, Alice's trust in Bob will increase.

### 2.2 Requirements of the Scenario

From the scenario(s) described above, we can elicit the following requirements that a pervasive service selection framework should meet:

- Ability to encode both functional attributes and QoS requirements in service requests; with the latter including the ability to express conditions on the execution context of the device (e.g.,power level, predicted co-location).
- The means to judge the outcome of a service request with respect to the relevant attributes. Moreover, the ability to distinguish between a service failure which is responsibility of the provider, and an external condition. While monitoring of service attributes can be automated by a middleware, assessing responsibility is a much harder task, and application logic must be employed to discern most problems. Still, automation of this process is important to avoid needless cognitive load upon the user.
- The mobility patterns of a peer should be made available to other hosts, to give an indication of how long both peers will be co-located (it is not the goal of this paper to discuss how these can be shared anonymously to protect user's privacy).

– The ability to bootstrap trust information in an environment where a device has never interacted before. This would be most easily achieved by asking all peers in the new environment about their opinions on (i.e., recommendations about) nearby providers. Therefore hosts should be able to share opinions, taking extra notice of the opinions from hosts they already trust.

## 3 Our Approach

We intend to create a framework that satisfies these requirements and that is also lightweight enough not to congest the network or to overly load resource-constrained devices. Internet-based resource discovery approaches (such as HTTP URIs [2]) rely on explicit naming of required resources; however, in mobile networks, this is less than ideal. Rather, the type (e.g., *printer*) and attributes of a required resource should be specified, to allow the searching and binding to a sufficiently suitable provider within communication range. This kind of search requires a shared ontology between requesters and providers. Ontologies have been investigated in OWL [10], which is predominantly an Internet technology. In this paper, we do not elaborate on the ontology in use, but rather on how the service description (QoS and mobility based) can be used for selection, how it can be monitored with respect to the actual service provision, and how it can be integrated with trust and reputation information. We will now give a description of the QoS and mobility aspects of the framework, and then focus on trust maintenance and sharing.

### 3.1 Quality of Service and Mobility Based Selection

In highly dynamic, sometimes faulty, networks, it is necessary to ascertain which peers can provide acceptable QoS levels in a given context. QoS constraints vary from application to application, and from context to context. In Q-CAD [3], we proposed using *context-aware* QoS specifications of services. For instance, a host may forgo requesting encryption of communication if their battery level is low, to conserve power. We now enrich QoS specifications with mobility considerations, so that a service description can also be evaluated based on the expected co-location time of client and server. This information is vital for services that will take a significant amount of time to complete, like file downloads.

In an open network with heterogeneous nodes, it is sensible to use an easily parseable format, such as XML, for the representation of QoS requirements. The metrics included would have to be measurable locally by the peer, but could include remote qualities, such as network congestion and co-location. Local metrics would most likely be battery power and load. Each application thus has an associated *Application Profile* that defines the attributes of importance to that specific program. When a resource/service is searched for, it is the middleware's responsibility to evaluate the profile of the requesting application (i.e., application-dependence), to monitor the operating context (i.e., context-dependence), and to then broadcast the appropriate request. In the scenario we are considering

```
[Attributes]                  [Operations]
service_type: location        -
accuracy: 50m                 <=
mobility: 2100s               >
trust: 0.1                    >
age: 360s                     <

[Rank]
accuracy: 3
mobility: 2
trust: 1
age: 1
```

**Fig. 2.** Scenario application profile

(see Section 2), mobility is specified in the profile as an estimate of the time a host will be present in the current environment. This is quite a naïve way to measure mobility; for example, it does not take into account many of the finer points of possible movement. It is, however, a useful metric to easily estimate and compare hosts' co-location, which is the mobility aspect we are currently interested in.

An example *Application Profile* from our target scenario is shown in Figure 2. Note that a minimum trust level may be required, to discard untrustworthy providers for which a reliable QoS prediction cannot be made. The *Operations* section defines whether the metric should be maximised or minimised to be improved. The *Rank* section defines the weighted importance that each attribute has to the application. This information is communicated to the middleware upon startup, but it remains accessible to the application for run-time modification. The same application may register different services with the middleware, to allow fine-grained customisation.

```
<REQUEST>
   <SOURCE name='aliceID' />
   <TYPE name='location' />
   <ATTRIBUTE name='accuracy' op='lessThanEqual' value='50m' />
   <ATTRIBUTE name='age' op='lessThan' value='360s' />
   <ATTRIBUTE name='mobility' op='greaterThan' value='2100s' />
</REQUEST>
```

**Fig. 3.** Service request example

A service request is then shown in Figure 3 using a simple XML format. The requester (`aliceID`) is a unique identity generated by Alice's device, for example, by hashing her public key (the corresponding private key would sign each message). The service attributes are then listed, each with their own values and operators, not including the locally stored measures (such as trust). This request will only match location providers with an accuracy of 50 meters or less, delivering information which is under 6 minutes old, and that should not move (e.g., get off the train) for at least 35 minutes.

```
<RESPONSE>
   <SOURCE name=‘bobID’ />
   <DESTINATION name=‘aliceID’ />
   <TYPE name=‘location’ />
   <ATTRIBUTE name=‘accuracy’ value=‘30m’ />
   <ATTRIBUTE name=‘age’ value=‘100s’ />
   <ATTRIBUTE name=‘mobility’ value=‘15000s’ />
</RESPONSE>
```

**Fig. 4.** Service response example

Bob’s response is shown in Figure 4. It specifies the identity of the responder (`bobID`), the identity of the receiver (`aliceID`), and a confirmation of the service type. This part is followed by Bob’s promised/advertised service attributes. As argued before, Bob’s claims may not reflect the quality that he can/will actually offer. Rather than simply believing the claims of another peer, we thus include trust reasoning about the service providers.

## 3.2 Trust based Selection

Hosts build up an opinion about every other device/service they interact with; these opinions will gradually become more authoritative with each interaction among the same pair of hosts. Initially, peers that have not been encountered before will have a neutral reputation, neither positive nor negative. This value would be increased after successful interactions, while appropriately decreased following unsatisfactory service deliveries. We use a trust range of values between $[-1, 1]$, with the following meaning: -1=Highly Distrusted, -0.5=Distrusted, 0=Neutral (usual default), +0.5=Quite Trusted, and +1=Highly Trusted.

Rather than just change in a linear fashion, trust should increase with exponentially decaying increments, and decrease multiplicatively. A reasonably low default trust value would combat susceptibility to Sybil attacks, where malicious hosts can simply generate more identities to avoid being punished for past misbehaviours. An application will use the trust in a service provider to estimate the accuracy of the promised QoS; moreover, it may specify that only providers possessing a trust value higher than a certain (application-specific) threshold should be considered. This would obviously be high in sensitive operations, such as monetary transfers, and relaxed for minor tasks, such as location information gathering.

The formulae we use to update trust are the following:

On success : $\mathbf{T_{new}} = \mathbf{T_{old}} + e^{-\alpha \mathbf{T_{old}} + 1.5}$

On failure : $\mathbf{T_{new}} = ((\mathbf{T_{old}} + 1) * (\beta * \gamma)) - 1$

with $\alpha$ being a user-tunable parameter, $\beta$ representing the error's severity and $\gamma$ representing user's patience (all to vary in $[0, 1]$).

Trust values are not only used by clients to rate services, they are also used by the providers to prevent their misuse. Attempting to provide service to all peers who ask, can leave the providers open to DOS attacks, such as with half-open TCP connections taking up resources. Ignoring peers with a poor reputation will provide an incentive for all peers to behave correctly. Old or unimportant opinions should be discarded according to certain host specific constraints. This could be as simple as least-recently-used as soon as a storage limit is reached.

Devices should be able to share their opinions about other hosts. This is particularly important to retrieve information about providers we have never interacted with before. Shared opinions (i.e., recommendations) should only be trusted as much as the host announcing them. A peer that provides a good opinion of another, only for that peer to act maliciously, should suffer a reduction in trustworthiness. This should discourage groups of malicious peers from colluding and giving false-positive opinions. Additionally, opinions from hosts that are distrusted should be ignored, to avoid malicious peers spreading lies. The importance assigned to recommendations should be much lower than a peer's own experience; particularly suspicious hosts may even decide to ignore others opinions. If used, the following formula is computed to make a prediction of a host's trustworthiness:

$$\mathbf{T_{new}} = \mathbf{T_{old}} * (1 - \delta) + ((\mathbf{T_{op}} * \mathbf{T_{sr}}) * \delta),$$

where:

| | |
|---|---|
| $\mathbf{T_{sr}}$ | - Trust value of opinion source, within $[0, 1]$ |
| $\mathbf{T_{op}}$ | - Trust opinion given by peer, within $[-1, 1]$ |
| $\delta$ | - Faith in the opinion system (user-tunable), within $[0, 1]$ |

The predicted trust in a service provider will be used by our service provision framework to estimate its actual service quality, thus providing a more application-tailored and robust service selection.

Rather than providing a single numeric indicator of a host's trustworthiness, which will be a subjective measure and perhaps based on different factors than the announcing peer uses, it should be possible to request a brief transaction history. This would contain the stated level of service promised by the provider and outcome from the interaction, allowing a host to make judgments based on their own criteria. This significantly increases the amount of state a peer has to maintain and would be beyond the capabilities of an embedded device. However, it could be implemented as a separate service and optionally run on top of the core middleware by more resource rich devices. Transaction repositories could also be run by trusted parties in the social group, such as the train operator, to increase the passenger's satisfaction, or by a completely separate third-party for profit. These would allow some permanence of the past data and would allow wider dissemination of behaviour information.

## 4 Conclusion

This paper suggests the combination of trust, mobility and QoS estimations to provide a more reliable and rewarding pervasive service experience in mobile ad-hoc networks. The decentralised trust management model allows the dynamic calibration of the service selection, based on a history of service provisions; this should in turn promote co-operative behaviours among the various peers. The combination of mobility patterns with QoS information gives insightful details on the foreseen service availability. Estimating a host's mobility is an extremely hard task without knowledge of position, time and historical movement patterns. An effective lightweight metric needs to be devised to allow communication of expected future movements, a subject of further work.

We now intend to formalise the syntax and semantics of application profiles and service requests. We will then realise our framework in a component-based architecture and implement it in a lightweight middleware. It is our plan to then evaluate our ideas in terms of increased reliability in P2P service provisioning, by means of simulation of realistic scenarios, which include real mobility traces.

## References

1. A. Pfitzmann, B. Pfitzmann, M. Schunter, M. Waidner. Trusting mobile user devices and security modules. *IEEE Computer*, Feb 1997.
2. T. Berners-Lee. Uniform Resource Identifier (URI): Generic Syntax, Jan 2005.
3. L. Capra, S. Zachariadis, and C. Mascolo. Q-CAD: QoS and Context Aware Discovery Framework for Mobile Systems. *ICPS*, July 2005.
4. R. Chakravorty, S. Agarwal, S. Banerjee, and I. Pratt. MoB: A mobile bazaar for wide-area wireless services. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 228–242. ACM Press, 2005.
5. P. Dewan, P. Dasgupta, and A. Bhattacharya. On Using Reputations in Ad hoc Networks to Counter Malicious Nodes. In *Proc. of the 10th International Conference on Parallel and Distributed Systems*, pages 665–672, 2004.
6. ESA Galileo Team. Galileo Telecommunications paper.
7. S. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The EigenTrust Algorithm for Reputation Management in P2P Networks. In *Proc. of the 12th International World Wide Web Conference*, 2002.
8. S. R. Medidi and K. H. Vik. QuaSAR: Quality of Service-Aware Source Initiated Ad-hoc Routing. In *Proc. of 1st IEEE International Conference on Sensor and Ad hoc Communications and Networks*, Oct. 2004.
9. Sonja Buchegger and Jean-Yves Le Boudec. Performance analysis of the CONFIDANT protocol: Cooperation of nodes. In *Proceedings of IEEE/ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC)*. IEEE, June 2002.
10. W3C: Web Ontology Working Group. OWL Web Ontology Language Reference, 2004.