Online estimation of context dynamics and its impact on context sensitive applications

Lomholt Jakobsen, Morten; Olsen, Rasmus Løvenstein; Rasmussen, Jakob Gulddahl

Published in:
1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology, 2009. Wireless VITAE 2009

DOI (link to publication from Publisher):
10.1109/WIRELESSVITAE.2009.5172573

Publication date:
2009

Document Version
Publisher's PDF, also known as Version of record

Link to publication from Aalborg University

Citation for published version (APA):
Lomholt Jakobsen, M., Olsen, R. L., & Rasmussen, J. G. (2009). Online estimation of context dynamics and its impact on context sensitive applications. In *1st International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology, 2009. Wireless VITAE 2009* (pp. 921-925). IEEE (Institute of Electrical and Electronics Engineers). https://doi.org/10.1109/WIRELESSVITAE.2009.5172573

Online estimation of context dynamics and its impact on context sensitive applications

Morten Lomholt Jakobsen[1], Jakob G. Rasmussen[1], Rasmus L. Olsen[2]
[1]Department of Mathematical Sciences, Aalborg University, Frederik Bajers Vej 7G, 9220 Aalborg, Denmark
{lomholt,jgr}@math.aau.dk
[2]Center for TeleInFrastruktur, Aalborg University, Niels Jernes vej 12, 9220 Aalborg, Denmark
rlo@kom.aau.dk

***Abstract*—Context management systems allow services and application easy access to remote dynamic context elements, in turns allow these to adapt to the users context. The fact that most information is dynamic and changes value over time, leads to potential risk of obtaining information mismatching the true value due to the communication delay. The probability for this we call the mismatch probability. Obtaining the mismatch probability is not trivial, as it involves knowledge of both the delay process and description of the dynamics of the information element. This paper addresses online estimation of the parameters related to the information element needed to derive the mismatch probability, and the effect that this estimation has on a context sensitive service discovery for Personal Networks.**

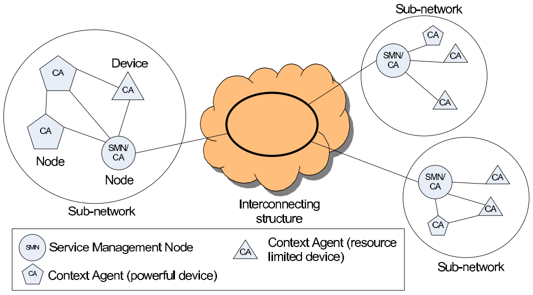
Sub-network
CA
SMN/ CA
CA
CA
Device
CA
CA
CA
Node
SMN/ CA
Node
Sub-network
Interconnecting structure
SMN/ CA
CA
CA
CA
Sub-network
SMN Service Management Node
CA Context Agent (resource limited device)
CA Context Agent (powerful device)

Fig. 1. Example configuration of a PN with involved entities in Context Aware Service Discovery

I. Introduction and background

Personal Networks (PN), [2], is a new network paradigm introduced recently in the MAGNET and MAGNET Beyond project, as a response to the requirements of new generation networks and its aim to support the user in personal manner without being obtrusive, [1] and [5]. An example of such a network is shown in Figure 1. PN's offer the user to connect personal devices on a local and global scale, e.g. connecting devices in his PAN, car network to his home and/or office network. The connectivity is being managed automatically and securely by a large set of components, [1], and allow the user access to a large range of services. Having access to a large set of services, geographically dispersed may not be optimal for the user, as he or she may not know which is optimal in a given situation. Hence some intelligent system that can assist the user in selecting the most optimal service under the given circumstances is needed.

Context aware service discovery for Personal Networks allows services to be ranked according to their relevance to the user in a given situation. Existing solutions such as Jini, [6], and UPnP, [7], do not sufficiently take into account context information when discovering services considering potential size and dynamics expected in PN's. Based on the network structure shown in Figure 1 we make use of a **Context Management Framework** (CMF) consisting of **Context Agent**s (CA), [10] that are capable of gathering and distributing context information in the network. Context Information is provided by Context Sources (CS) that are locally managed by the local Context Agent and made available to all CA's via an CMF specific protocol. In addition to the context management framework, there is also a dedicated **Service Management Platform** (SMP) available, consisting one **Service Management Nodes** (SMN) per cluster, which is responsible for maintaining service information of a cluster. Both SMNs and CMNs from different clusters are connected in an overlay network, that allows them to efficiently distribute service and context information available in the PN. However, context information is often dynamic and when accessed remotely over a network they are exposed to delays which in turns affects the reliability of information being used. In this paper we look at ***the reliability of context aware service discovery***, which includes techniques to estimate reliability parameters and how it affects the end users experience.

A. Context aware service discovery in Personal Networks

In the following we briefly sketch the important details of how context aware service discovery is realized in the settings of Personal Networks.

In Figure 2 the interaction between the context management framework (CA's) and the service platform (SMN's) is shown at a conceptual level.

When a context sensitive service discovery is triggered, the SMN executes a normal service discovery, followed by an evaluation of the found services and how they context wise relate to the user. Subsequently, the SMN needs to request for context information from the CA in order to evaluate services that have already been discovered. To determine what is relevant of information, which depends on the specific service type, we assume here this can be directly fetched from the CMF, e.g. via access to a CA. In practice this is a

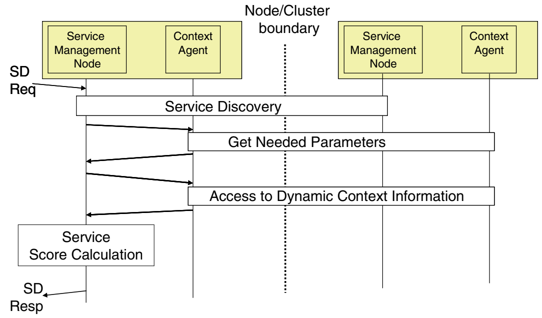

Fig. 2. Component interaction for achieving context sensitive service discovery as described in [1].

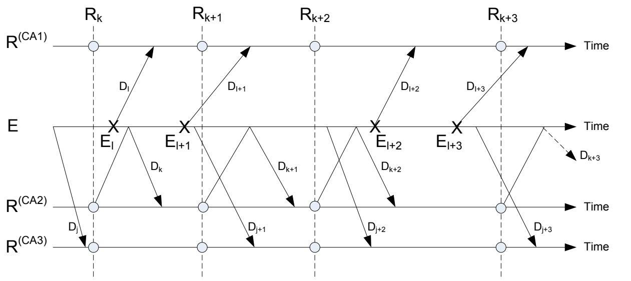

Fig. 3. Three different access strategies, with requests occurring on exact same time on different nodes. From $R^{(CA_11)}$ information is accessed with an event driven strategy, from $R^{(CA_2)}$ this is obtained with an reactive strategy and from $R^{(CA_3)}$ information is updated periodically, and accessed from a local cache.

rather complex problem to address. Knowing the information to be obtained, the SMN requests the local context agent for this information, which provides the requested information by one of the means described in details in the following subsection. Following this step, the SMN can now calculate a numerical service score associated to a service, which indicates the level of relevance to the user. The higher the score, the more relevant the service is to the user. Several research papers and projects addresses context aware service discovery, e.g. [8], [9], but none of them addresses the fact that context information is dynamic, which affects the reliability of the service scores. We address this issue in this paper and propose to use knowledge of mismatch probabilities to increase the reliability of context aware service discovery as an example application, but the algorithm proposed may be easily mapped to other alike context aware decision processes. The algorithm we evaluate in Section III, taking into account estimation techniques investigated in Section II. In Section IV we conclude the paper.

B. Remotely accessed context information

For the CA to access remote information from another CA which provides the desired context information, there are three principal approaches;

- a *reactive approach*: The requesting CA sends a request to the providing CA, which returns a response with the given value.
- a *proactive event driven*: The requesting CA has made a subscription to the providing CA to send updates when some external event has lead the context value to violate some subscription condition.
- a *proactive periodic*: The requesting CA has made a subscription to the providing CA to send updates with specified time intervals to the requestor.

These strategies are exemplified in Figure 3, with a given context providing CA (E) and a set of requesting CA's ($R^{(CA_{1-3})}$) for each of the possible strategies. Whatever approach is taken, there is always some probability that the information has been changed by the underlying context process by the time the SMN gets this information. For example in Figure 3 request R_k will lead to a correct value for all CA's, i.e. $R^{(CA_{1-3})}$, R_{k+1} only on $R^{(CA_2)}$ using the reactive strategy, R_{k+2} only on $R^{(CA_{1,3})}$ using proactive strategies, and finally on R_{k+3} only correct response on $R^{(CA_{2,3})}$ using a reactive and periodic update strategy. It is the probability for not getting a correct value that we call the mismatch probability. In [4], analytic expression for the mmPr based on the different update strategies were derived. In short, the mmPr depends on a set of factors, namely:

- The times between events changing the context value, modelled by a stochastic Event process
- The delay between the nodes, modelled as a stochastic Delay process
- The times between requests are arriving, a process which turns out to not impact the mismatch probability at all, see [4]
- The update strategy taken, e.g. reactive or proactive.

We have previously studied different scenarios and the impact on metrics on the mmPr in e.g. [4]. However, for all previous work we assumed that the CA was already informed about the relevant information needed to perform the mismatch probability calculations. This, however, needs to be available in order for the CA to derive the mismatch probability, and how it is done will impact the functionality of the system. We consider two different approaches on how to obtain information about required information dynamics:

1) The Context Source (CS) provides additional information about the process itself, i.e. type and parameters when being registered at the CA. This requires the CS to include additional information.
2) The CA monitors the data coming from the CS, and estimates both type and parameters.
3) The CA monitors the data coming from the CS, and estimates parameters related to the event process. The CS has provided information of the type of event process.

In this paper, we investigate how the CA can estimate the information dynamics remotely, which is needed for calculating the mismatch probability. We consider in the following case 3), i.e. that the CA is aware of which process type that drives the dynamics of the information element, that is, which event process type is known. But the parameter values describing the process is not known. The delay needs to be known as well, but to keep focus on the event process, we limit our selves from estimation of the delay as well, i.e. we assume the delay to be known a priori.

II. Online estimation of information dynamics

In the following we focus now on the reactive access strategy, and leave other strategies for future study. The reactive strategy performs well in terms of reliability and only produces network overhead when information is actually needed, [4]. If we assume the event process is known to be a homogeneous Poisson process with an unknown rate λ. For Poisson processes, an unbiased and consistent estimate of the rate is given by

$$\hat{\lambda_k} = \frac{E(t_k)}{t_k}$$

From [4] we know for the reactive case, that the mismatch probability, and the following estimated mmPr, can be calculated as

$$\begin{aligned} mmPr &= 1 - e^{-\lambda/\nu} \\ mm\hat{P}r_k &= 1 - e^{-\hat{\lambda}_k D_k} \end{aligned}$$

where $\nu = 1/D$ is the downstream delay rate assuming exponentially distributed delays, the k is an index referring to the k-th request, e.g. D_k is the k-th delay associated to the k-th request (see Figure 3). Using the estimated value of the event rate, $\hat{\lambda_k}$, we find that the mismatch probability is biased as a function of the observation time, i.e.

$$\begin{aligned} \mathbb{E}\left[mm\hat{P}r\right] &= 1 - \mathbb{E}\left[\exp(-\hat{\lambda}_k D_k)\right] \\ &= 1 - \exp(-\lambda t_k(1 - \exp(-D_k/t_k))) \quad (1) \end{aligned}$$

where t_k is the observation time, and D_k is the observed delay for the k-th request. It is easily shown that $f(t;D) = t(1 - \exp(-D_k/t)) \rightarrow D_k$ as $t \rightarrow \infty$, i.e. the bias disappears when the observation time goes to infinity. A plot of the bias level is seen in Figure 4 for various levels of delays.

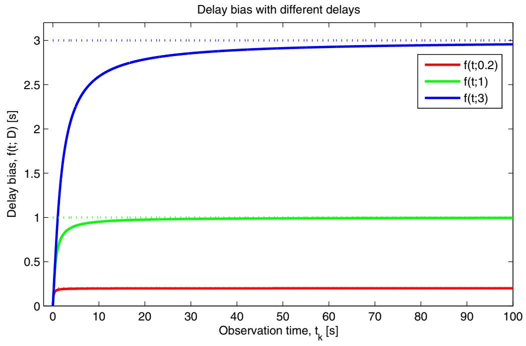

Fig. 4. Plot of the bias function $f(t;D) = t_k(1 - \exp(-D_k/t_k))$ for $f(t;0.2)$, $f(t;1)$, $f(t;3)$

The interpretation of this result is that for a given deterministic delay, D, it will take a certain amount of time before the bias has become low enough to be negligible. Where exactly the border is to be located is a tradeoff between the observation time and the estimation bias, which in turn is application dependent. A short observation time is desirable from the applications point of view, as knowing the mismatch probability can help the application reducing the probability of making wrong decision (shown in Section III). A longer observation time leads to lower mmPr bias which is also desirable. This is a balance which the application could be set via a specific Quality of Information interface to the context manager. However, even though a low biased estimator has been achieved, it is not very useful if it has a high variance. Thus, elaborating on the variance of the estimator, the following expression can be derived

$$\begin{aligned} \mathbf{Var}\left[mm\hat{P}r_k\right] &= \mathbb{E}\left[\exp(-2\hat{\lambda}_k D_k)\right] - \mathbf{E}\left[\exp(-\hat{\lambda}_k D_k)\right]^2 \\ &= \exp(-\lambda t_k(1 - \exp(-2D_k/t_k))) \\ &\quad - \exp(-2\lambda t_k(1 - \exp(-D_k/t_k))) \qquad (2) \end{aligned}$$

meaning that for $t \rightarrow \infty$ then $\mathbf{Var}\left[mm\hat{P}r_k\right] \rightarrow 0$, hence the estimator is consistent. This also implies that a longer observation time is beneficial for a given application, and once again we see the tradeoff between keeping a short observation time, and providing a reliable mismatch probability indication. Figure 5 shows a plot of the standard error (i.e. $g(t,\lambda,D) = \sqrt{\mathbf{Var}\left[mm\hat{P}r_k\right]}$) versus the observation time for different delays and event arrival times.

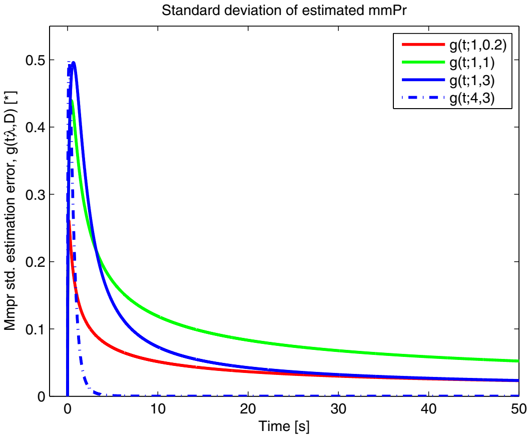

Fig. 5. Standard error as a function of observation time.

From Figure 5 it is also clearly seen that a longer observation time is beneficial as the standard deviation is reduced with increased observation. It can also be seen that the decay depends not only on the delay, but also on the event time interval. A higher event rate leads to a faster convergence towards zero than a slower event rate. An example of how the estimation affects the mmPr over time is shown in Figure

6, showing a scenario where the true mmPr is 0.45, and using the estimated value of the event rate $\hat{\lambda}_k$, the mmPr estimate has increased variance in the beginning, and over time becomes more certain.

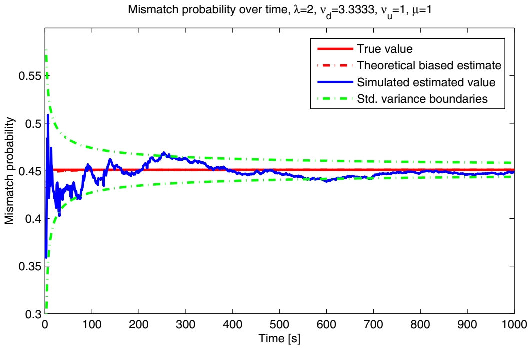

Fig. 6. Example simulation run, where each time a response has been obtained, the estimation of the event rate is being updated.

With these results in mind we now propose to set a threshold time for when mismatch probability can be considered stable. As shown in Equation (1) this is a matter of determine when the bias is acceptable, and does only depend on the delay and observation time. This means we have to find a minimum observation time, t, from the following equation

$$t(1 - e^{-D/t}) - D \leq \epsilon \tag{3}$$

where ϵ can for example be expressed as a percentage of deviation between the true delay and biased delay, e.g. $\epsilon = D\alpha$ ($0 < \alpha \leq 1$), which can only be solved numerically due to the exponential term. In practice, the CA would have only to check if the bias is below the dedicated threshold each time it receives a response with the given delay. This decision scheme will be used in the following section.

III. Impact of Remote Access to Dynamic Context Elements

Correct selection of services based on the ranking functionality of context sensitive service discovery depends heavily on whether there is a mismatch or not. In [3] we investigated different approaches of how mismatch probabilities can be used to reduce score errors in context sensitive service discovery. The provided examples are not only limited to context sensitive service discovery, but may apply to all applications depending on remote dynamic accessed information, but with different impact on using mismatching information ranging from harmless to serious consequences.

Considering the scenario of context sensitive service discovery, the reliability of the service relevance evaluation can be done according to the following scheme: For each service discovery request, context information is accessed, and thereafter used to do the service evaluation, see [3]. Based on one or more context information related to the service itself (x_s), context related to the user (x_u), an expression for the service score can be expressed as

$$S_{SSF}(\overline{\mathbf{x}}_s, \overline{\mathbf{x}}_u) = \frac{\sum_{n=1}^{M} w^{(n)} f_{CSF}^{(n)}(x_s^{(n)}, x_u^{(n)})}{\sum_{n=1}^{M} w^{(n)}}$$

with the context information organized in vectors of elements $\overline{\mathbf{x}}_s$ and $\overline{\mathbf{x}}_u$, respectively. Since any information accessed remotely, e.g. the service context, there is also a possibility for the score value to be erroneous. Therefore, a green or red indicative flag is raised and shown graphically to the user next to the score value, showing whether the obtained score value is reliable or not. Using the notion of x for the value of a single information element (the true value), $\tilde{x}$ for the value being accessed remotely, and the score function $S()$, the decision of setting the indicator is as follows:

- A green flag is raised if $S(\tilde{x}) = S(x)$
- A red flag is raised if $S(\tilde{x}) \neq S(x)$

The decision of whether a green or red flag is shown to the user, can more generally be mapped into a binary value, where a 1 indicates a green, and 0 a red. The algorithm considered is thus as follows:

1) If mmPr is not known: always use the information, i.e. decision is always 'show green'.
2) If mmPr is known: The application compares the known mmPr to a threshold, $\theta \in [0; 1]$, and if
   - mmPr$\leq \theta$: The decision is 'show green'.
   - mmPr$> \theta$: The decision is 'show red'.

In some cases the system will display a green flag when, in fact it was supposed to show a red flag. This case is considered an error, and it is of course desirable to reduce the probability of this happening. Thus, the probability of making the wrong decision (of which flag to show) can be estimated as

$$\widehat{CDM} = \frac{\#\text{correct decisions made}}{\#\text{decisions made}}$$

Now we wish to investigate to what extend it helps to use mismatch probability to indicate also the red flags, and thereby assist the user in identifying potential erroneous service score values. Figure 7 shows the resulting probability of making a correct decision with an increasing delay between the Context Source and requesting SMN/Context Agent.
Selecting θ has implications in the results shown in Figure 7. If the value is chosen too low, e.g. $\theta = 0.3$, then as the mmPr goes beyond 0.3 decisions are more likely to be wrong, i.e. more red flags are being shown to the user than really needed, and vice versa for a higher value of θ, e.g. 0.8. The latter part simply indicates that more green flags are shown than is really needed. Based on these arguments the value, $\theta = 0.5$, has been chosen and shown to be a good choice. However, for cases where information may return to previous state (i.e. a state change may cancel out a mismatch), or in case there are more than two options for the decision algorithm to take, e.g. if there are three colors to indicate the reliability of the service score, e.g. green, yellow and red, this value may need to differ. However, this is left for future study.
Choosing ϵ in Equation (3) is a matter of weighing between

the level of bias/stability acceptable to the application (the effect of using mismatch probability too early is clearly seen in Figure 6), and the time needed to wait for the mismatch probability to become acceptable. The decision point when the bias level is considered reasonable, has for illustrative purpose been chosen to 10% of the delay value, i.e. $\epsilon = 0.1D$.

We show two graphs of the CDM; 1) where samples are taken when bias level is higher than ϵ, and 2) when it is below. Exactly the same amount of samples are used for all graphs.

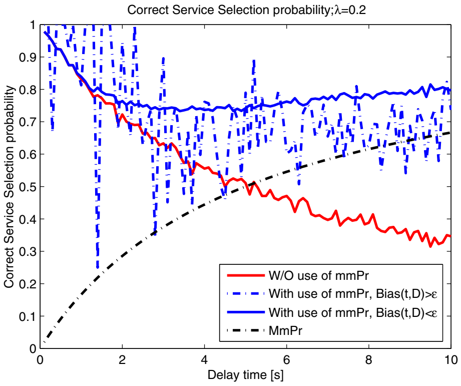

Fig. 7. Evaluation of how knowledge of the mmPr can be used to improve the probability of making correct decisions as a function of the delay.

As the results show, the probability of making a correct decision is reduced as the delay increases, i.e. the reliability of context sensitive decision makings. For Algorithm 1) which does not consider the mmPr, this continues decreasing as the delay increases, and in fact, due to the nature of the simple *accept all* this approach leads to a probability equal to $1 - mmPr$ of the given scenario. For Algorithm 2), using the mmPr actively after the bias of the estimate has dropped significantly, the curve breaks at some point, and as the delay increases the probability of making a correct decision is again increased. This happens because, as the delay increases, the need for showing the red flag to the user is also increased at the same time (i.e. an increasingly amount of the information accessed, becomes unreliable). This leads to an increased amount of correct decisions being made. Without the mmPr this is not possible. The dash-dotted line in Figure 7 shows the case of Algorithm 2) using the samples when there is a significant bias on the mmPr estimate, i.e. illustrating the effect of only using the mmPr when it is reasonably stable. This clearly shows the necessity of waiting for the bias to disappear, and that mmPr has to be reasonable reliable before using it!

IV. Conclusion and Outlook

This paper has addressed access to remote dynamic context information, and how parameters needed to calculate the so-called mismatch probability can be estimated remotely. First we suggested a simple estimator for the information dynamics of remotely accessed information element, and found that mismatch probability estimates based on this estimator, are biased. However, the bias is reduced as the observation time is increased. Thus, for mismatch probability to be useful when the information dynamic is being estimated remotely, there need to be a minimum period of observation time. This time can be found by solving Equation (3) with respect to the observation time, however, it can only be solved numerically. From the result section, we proposed a strategy to use mismatch probability actively to make decisions on whether a calculation based on remote information is valid or not, and showed by simulation the benefit of this algorithm. In this respect, the results also shows that the most benefit can be seen when the ratio between the delay and event rate is high. This is natural, since as long the event rate is relatively low compared to the delay, the probability of making wrong decisions are also relatively low. As the event rate gets high compared to the delay, the probability of making wrong decisions also increases. Using the mismatch probability in the proposed way effectively reduces and keeps the probability of making wrong decisions low compared if not doing so. At the same time we showed the effect of not considering the bias level of the estimator. Two result graphs were shown, namely one using samples from when the bias was large, and one using samples from when bias was low. The results clearly showed the minimum benefit of determining the observation time needed, and base the usage of mismatch probability on the actual level of the estimation bias.

References

[1] M.Jacobsson, et al., Specification of PN Networking and Security Components, MAGNET Beyond Deliverable 2.3.1, December 2006, IST-027396, http://www.ist-magnet.org

[2] I.G. Niemegeers and S.M. Heemstra de Groot: From Personal Area Networks to Personal Networks: A User Oriented Approach, Personal Wireless Communications, Kluiwer Journal, May 2002, P.??-??

[3] R. L. Olsen, H.-P. Schwefel and M. Bauer, Influence of Unreliable Information on Context Aware Service Discovery, Third workshop on Context Awareness for Proactive Systems, CAPS'07, Guildford, United Kingdom, 18-19 June, 2007

[4] R. L. Olsen, H.-P. Schwefel, M. B. Hansen: Quantitative analysis of access strategies to remote information in network services, Global Telecommunications Conference, GLOBECOM 2006 - IEEE. IEEE (Institute of Electrical and Electronics Engineers, Inc.), 2006.

[5] My personal Adaptive Global NET (MAGNET Beyond), IST-FP6-IP-027396, http://www.ist-magnet.org/

[6] Sun Microsystems Inc., Jini Network Technology, http://wwws.sun.com/software/jini/

[7] Microsoft Corporation: Understanding Universal Plug and Play, White paper, UPnP forum, http://www.upnp.org/, June 2000

[8] M. Khedr, A. Karmouch, R. Liscano and T. Gray, Agent-Based Context-Aware Ad hoc Communication, Proceedings of Mobile Agents for Telecommunication Applications : 4th International Workshop, MATA 2002, Barcelona, Spain, October 23-24, 2002. Page(s) 105-118.

[9] S. Cuddy, M. Katchabaw; H. Lutfiyya, Context-aware service selection based on dynamic and static service attributes Wireless And Mobile Computing, Networking And Communications, 2005. (WiMob'2005), IEEE International Conference on Volume 4, 22-24 Aug. 2005 Page(s):13 - 20 Vol. 4

[10] L. Sanchez, R. L. Olsen, M. Bauer, M. G. Genet: A Generic Context Management Framework for Personal Networking Environments, First International Workshop on Personalized Networks, 2006