Abstract. Current research on context-aware systems concentrates often on the optimal use of the large amounts of context information that is available, especially on mobile devices. While the focus is typically on the process of reasoning about the context data, the architectures of these systems often do not take into account the communication capabilities and the resulting advantages, provided by mobile terminals.

Based on the Framework for Context Reasoning Systems, introduced in [PNF05], the mechanisms described in this paper are designed to improve this situation, by using the communication mechanisms in order to provide the user of a wireless information device with proactive applications that can take into account high-level context information regarding potential future situations, as well as inferred activities and goals of the user.

1 Introduction

Services and applications, especially for wireless information devices are becoming increasingly context-aware. Due to the fact that these mobile devices are almost constantly with the user, they are able to gather large amounts of context information, which can be used to learn about the users behavior and wishes in a given context. The resulting models can then be used further, in order to predict future situations and enable the development of proactive services and applications.

While major effort are undertaken to structure this reasoning process and to provide mechanisms that enable the reuse of high, as well as low-level context information, the communication abilities of most mobile devices are not necessarily taken into account.

Using software components to provide software developers a higher level of abstraction is an established field of research. Component technologies and their implementations, such as the Common Object Request Broker Architecture (CORBA), or Web Services are also already widely used in the industry. Structuring the context reasoning process in the form of components has not only the
advantage of making these components reusable, but allows the dynamic configuration of the reasoning process, based on the task at hand and on the provided data [PNF05].

Peer-to-Peer technology has recently received an increased interest, even though the underlying mechanisms, while increasingly refined, have essentially been around for decades [Sch01]. Using Peer-to-Peer mechanisms, for the different components to initiate communication makes a distributed, component-based context reasoning process still more flexible, as it enables the dynamic distribution of the involved reasoning and data management mechanisms amongst devices with different levels of available resources, as well as the distribution of available information and reasoning results amongst devices that share the same context. Especially if the location, as an often important part of a users context, is shared by different devices, reusing the results of the context-reasoning process on different devices that are in close proximity, is very beneficial, even more so, as the information can be shared using close-proximity communication mechanisms, often employed by wireless information devices.

The work, described in [PNF05] examines a framework that takes into account the commonalities between different reasoning mechanisms and proposes a structure that allows different mechanisms to share context information. Combining the components, described in this framework with Peer-to-Peer mechanisms enables not only the flexible combination of different reasoning mechanisms, but the reuse of available resources, through a flexible communication paradigm.

The remainder of this paper is organized as follows: while Section 2 discussed related work in the field of context-aware systems, Section 3 discusses basic requirements for the distribution of components over different devices and provides a short overview of Peer-to-Peer mechanisms and the Web Services component model. In Section 4 we describe in a typical scenario, how the different stages of a typical context-reasoning are executed in a distributed manner. We conclude this paper in Section 5, with a short discussion of further work on the described architecture.

2 Related Work

Different approaches have been taken to provide a common architecture for context-aware applications and the integration of different context-reasoning mechanisms. According to Moran and Dourish [Mor01], current research focuses on either different versions of a blackboard-based approach or on widget-based approaches. Usually these approaches are implemented in the form of middleware and its services, or in the form of application frameworks. Another possibility that Moran and Dourish describe is implementation using interacting agents, which are distributed over a network.

Examples of middleware approaches include, e.g., the Reconfigurable Context-Sensitive Middleware (RCSM) [YKW+02], and the CORTEX middleware, described in [DLBF+03]. For our purposes the CORTEX approach is more interest-
ing as it introduces special entities, called sentient objects, which are responsible for receiving, processing, and providing context-related information. Sentient objects are defined as autonomous objects that are able to sense their environment and act accordingly [DLBF*03]. The advantage of this approach is the possibility to re-organize them, for instance depending on their primary task.

Also some application frameworks that support context-awareness have been proposed. For example, [KMK*03] describes the implementation of a framework that supports management of context information on mobile terminals. The structure of this framework is centered on the blackboard paradigm for communication, which is handled by a context manager. Most components that use this framework, including the applications, act as clients for the context management system on the device itself. Other services can run not only on the device itself, but potentially also in a distributed environment.

Also Mayrhofer [May04] describes an architecture for context-aware systems, which is however focused on devices with limited resources, but which does not take into account the distribution of the reasoning mechanisms.

Another framework approach is The Context Toolkit [DA00]. The framework separates acquisition and presentation of context information from the application that requires it, by using so-called widgets. The focus of this work lays in the automatic inference of higher level context information from lower-level sensory data.

Another system, developed by the Hydrogen project [HSPL02] describes a three-layered architecture, designed to overcome existing problems, found in context-aware mobile systems. The framework consists of adaptor, management and application layer. For the communication between the different layers, the described framework utilizes an XML-based protocol. The aim of the framework is the provision of an architecture that is lightweight, extensible, robust, and which also enables the possibility of adding further meta-information to the system.

Described in [PNF05] is a framework for context reasoning mechanisms which is based on the commonalities that were found in different inference and learning mechanisms, and which is used as a structure for a reasoning engine that may be part of context-aware systems. While the framework provides also a structure for the components that different reasoning mechanisms have in common, certain aspects need to be studied further. Amongst those are the suitability of different distribution mechanisms and software-component models.

3 Distribution

The context reasoning process, as part of context-aware applications, especially when used on mobile devices, faces many challenges. Most are related to the limited resources (e.g. energy, processing power, access to data, storage, etc.), available on a wireless information device. Such devices typically do not possess the same amount of resources, available to for instance personal computers. Due to their communication capabilities however, mobile devices have the potential
to access resources available on other computers and servers, as well as the data and information available on other mobile devices.

While distributing the reasoning process over different devices, with different capabilities, available resources and data has thus a high potential, several factors need to be taken into account, such as

- the rapidly changing availability of other devices and the fact that they may at any time become unavailable
- the availability of data, required for the reasoning process
- the communication overhead incurred in the distribution of the context-reasoning
- the communication-related monetary costs to user
- the indirect costs, such as power consumption, intermediate storage, etc.
- timing issues, such as communication delays, which may influence the correctness of reasoning results, or their usability for specific applications

### 3.1 Devices

Winograd [Win01] also compares the different criteria that are important when choosing an architecture for a context-aware system [Win01]. Since we are considering mobile devices rather important for our research in context-aware systems, certainly the efficiency and low overhead of the architecture receive a very high priority. Due to the requirements that reasoning systems have, it may however not always be possible to process all information on the mobile device, which makes it necessary to also consider the possibility of distributing this functionality to other devices, or networked server mechanisms.

In our research we are also considering historical information for the learning process. The amount of historical information often taken into account, makes the existence of a central logical information entity essential, which in turn needs to be distributed in order to allow the different involved devices to participate the reasoning process.

The use of a widget or services-based approach, where the information is not centrally accessible is thus hardly feasible. However, due to the potential amount of historical data, which cannot be stored on the mobile device, it must be possible to store parts of the information (e.g. after a certain time) on a remote location, but still keep it available for the learning mechanisms.

While wireless information devices will always lack resources, compared to for instance personal computers, high-end servers can not only provide computing power, necessary for most reasoning mechanisms, but the distribution would also enable the more efficient storage of relevant data that is required for the reasoning mechanisms to generate more useful results.

### 3.2 Peer-to-Peer

One problem that currently most component technologies and their implementations are faced with is the reliance on a central brokering mechanism, which
manages the discovery amongst the different components and often also the communication between them. The different components require either the explicit knowledge of the location of other components, which must either be provided directly to the single component, or must be obtained via a brokering mechanism. In the later case it is however necessary, that each component knows the location of the brokering mechanism itself. Usually these brokering mechanisms allow components to submit a description of their offered services, together with a signature of their usage. Other components, requiring these services may then obtain the necessary information from the broker and then communicate with the other components directly. However, even though these brokering mechanisms can be distributed, highly redundant and reliable, they can be considered a single point of failure, since in the case of their absence, a new component is usually not able to find other services that it may require, unless it is provided with the required information in another way. This can jeopardize the functionality of a complete system, where components rely on the services provided by other components.

Peer-to-Peer technologies, as described for instance by Milojicic et. al [MKL+02], provide a potential solution for this kind of problem. Even though current middleware implementations could be considered hybrid Peer-to-Peer, due to redundant brokering mechanisms, the advantages of pure Peer-to-Peer, or Super-Peer technology would make these component models much more reliable.

Using decentralized search mechanisms, as described by Lv et. al [LCC+02] provides a very effective means for different components of a system to find each other. Using a Peer-to-Peer approach removes the necessity of a brokering mechanism and allows reasoning or data management components to find and directly connect to each other, in order to form a functioning context reasoning mechanism.

Such mechanisms furthermore inherently take into account the current situation of the user’s device, with regard to availability of other devices and their provided components, enabling them to change their behavior, based on the surrounding context (available peers, services, etc.).

This availability can explicitly take into account other devices that are in the immediate vicinity, using for instance short-range radio and Personal Area Networks (PAN), as well as devices with a higher level of available resources, which can be accessed remotely. Peer-to-Peer mechanisms furthermore provide the benefit of higher reliability since in the case of failure, or sudden disappearance of one context-reasoning or data management component, another component can often be found, without the necessity of contacting a central entity.

3.3 The Web Service Component Model

The currently perhaps most important application of XML messaging comes with the Web Service component model.

XML-based Web services are the fundamental building block in the move to distributed computing on the Internet. Open standards and the focus on communication and collaboration among people and applications have created
an environment where Web Services are becoming the platform for application integration. Applications are often constructed using multiple XML Web Services from various sources that work together regardless of where they reside or how they were implemented.

In order to facilitate this work further, Web Services use standardized mechanisms, such as the Simple Object Access Protocol (SOAP) [Wora], the Web Services Description Language (WSDL) and Universal Description, Discovery and Integration (UDDI) [Worb]. While the SOAP protocol specifies how the exchanged messages are structured, WSDL is used to describe the interfaces of a particular component. The UDDI standard enables the publication and lookup of specific components, in order to allow for the runtime cooperation between the different components.

As described earlier, the use of Peer-to-Peer technology makes the use of a central UDDI-based brokering mechanism unnecessary. Instead it allows the different components that form the reasoning framework, to find each other directly and dynamically.

The overhead that comes with the component and Peer-to-Peer mechanisms is especially for a mobile device with limited resources, not negligible. While on a server or personal computer each of the different reasoning or data representation mechanisms can be encapsulated into a single component, this approach is not feasible for the mobile terminal.

For this reason it is necessary that the different mechanisms that reside on the same terminal form one component, together with the mechanisms required for the dynamic discovery of other, remote components. This is possible, due to the fact that the different mechanisms that are available on the mobile terminal remain almost static and do not easily change over time.

Due to the use of WSDL as the description language for the different components, it is however still possible to apply the same component model, while for the implementation it is necessary to integrate these platform dependent requirements into the code-generation phase.

4 Scenario

The following scenario describes a typical distribution of reasoning activities amongst two mobile devices and one server (e.g. a device with a magnitudes higher level of available resources).

4.1 Service Discovery

In the initial stage, a service discovery through Peer-to-Peer mechanisms, as described above, takes place. In order to simplify the scenario, it is assumed that the structure of available services and data management components does not change during the described reasoning process. In such a case, a dynamic re-discovery, during the reasoning process will be necessary, using the available communication mechanisms.
4.2 Initial Context Reasoning

Based on the architecture described in [PNF05], Figure 1 depicts the initial gathering of context information on mobile device A, as well as basic context reasoning tasks, which do not require major resources, such as processing power. Due to the localization of this task to only one device, also the amount of data is manageable locally.

![Diagram](image_url)

**Fig. 1.** Initial Context Gathering and Reasoning on Device A (adapted from [PNF05])

Raw *Context Data*, or context atoms, are provided by sensing mechanisms, or obtained from the application. The device stores the data in the *Data History* and starts with a first step of the context reasoning, for example the *Pre-Processing* and data cleaning. The resulting higher-level context is provided as *Derived Context*, from where it is obtained by the next step of the reasoning mechanism.

This second step, responsible for the *Feature Extraction*, resides on the same device, as it does not require large amounts of resources. An alternative step at this stage would be the aggregation of data from different sources, as they also have been provided to the Derived Context component by the Pre-Processing step. Once the results of this step are available, they can be used by another reasoning mechanism (e.g. *Feature Selection*), also residing on the device, to provide an again higher-level of context.

The data is being shared between the different reasoning mechanisms, via the Derived Context component. Later on, as the specific information, stored in the Derived Context is of no further relevance to the active reasoning process, it is stored in the *Repository*. 
4.3 Advanced Context Reasoning

In case more elaborate context reasoning mechanisms, such as specific Classification mechanisms are required, this task can be taken over by a remote server mechanism, which has more resources available (see Figure 2).

The data required for this part of the reasoning process is provided through the Derived Context component, which is shared between the different devices and which automatically transfers required data from one device to another. The same applies to the Data History and the Repository components, which are also shared and which distribute required data to the different reasoning components on the different devices.

The high-level context information, which results from the classification process, is then again provided to the Derived Context component and thus available also on the mobile device. At the same time, the resulting model is also stored in the repository, in order to be available for further use.

4.4 Action Inference

In the final stage, as shown in Figure 3, the resulting high-level context information can be used by a reasoning mechanism on the device, which combines it with the preferences and application-provided actions. The result of this process is then an action, or a sequence thereof, which will be executed by the application. This step is executed, again, on the mobile device, where the application itself resides.
4.5 Data Sharing

The ability of different components to communicate with each other, also enables other devices to subscribe to a device’s Data History, Derived Context, or Repository. This enables another part of the scenario, in which a second mobile device uses the results of first device’s distributed reasoning process.

As shown in Figure 4, the second device is still required to perform initial stages of the reasoning process itself, such as the pre-processing stage. Since it has access to the results of the reasoning process of the first device and of the server, it is however able to reuse the provided model, once it has established that the same context is applicable.

The device is then able to execute only the last stages of the reasoning process, the adaptation to the user's preferences and the application. Other stages of the reasoning process, especially those that require a higher level of resources can be omitted. A further advantage is the fact that only local communication is required, as it is not necessary to contact the server.

Especially with devices that are in close proximity to each other, this sharing and the re-use of high-level context information is very beneficial and time saving. Often these devices do share the same, or at least a similar context, and are furthermore able to make use of close-proximity communication mechanisms, such as Bluetooth [Gro], in order to exchange the calculated models. In addition, devices are able to share context information, or low level sensor data. The described communication mechanisms also enable the storage of large
amounts of data in the Context History, and in the Repository on a remote storage mechanism, instead of deleting the information as further storage resources are required on the mobile terminal itself.

5 Conclusions and Further Work

In this paper we presented how Peer-to-Peer techniques, combined with a software-component approach can benefit the context-reasoning process of context-aware applications, especially taking into account the communication capabilities available on many current mobile devices.

While the component-based context-reasoning framework, which uses Peer-to-Peer discovery mechanisms, is currently under development, several issues still require further investigation.

One of these is issues is the development and integration of a suitable event model, which is required for many context reasoning mechanisms to function properly and for the usability of applications and services based there-upon. Due to the dynamics involved in the distribution over different types of devices, primarily wireless information devices, and the uncertainty of network connections between devices, this poses a major challenge. This problem has also been addressed in other works and several potential solutions have been proposed. However, their applicability within the scope of context-aware systems still needs to be evaluated.

Another issue that is partially discussed in [NPLF05] is that of security and privacy. Further work is required to include also mechanisms for authentication, confidentiality, integrity and non-repudiation, which enable the sharing of data between the different components, without compromising the user’s privacy. At the same time, the correctness of the results of the distributed reasoning process needs to be ensured. For further material, regarding security in Peer-to-Peer networks, we refer for instance to [SM02].

Of course it is also necessary to motivate the different devices to share their services and information. Especially for mobile terminals this is critical, due to the very limited resources available. Thus, there is a high potential for selfishness, as discussed for instance in [AH00], and which should be countermanded by different cooperation stimulation mechanisms. Also this aspect needs to be studied further.

References


