uBeeMe – a Platform to Enable Mobile Collaborative Applications

Jan Gäbler, Ronny Klauck, Mario Pink, and Hartmut König
Computer Networks and Communication Systems Group
Brandenburg University of Technology Cottbus - Senftenberg, Germany
eMail: {jgaebler, rklauck, pink, koenig} @ informatik.tu-cottbus.de

***Abstract*—The vision of ubiquitous communication is driven by increasing user mobility and the need to collaborate. Mobile collaborative applications, such as mobile audio/video conferences, collaborative writing, social networks, and mobile gaming, are already important parts of human interaction. In this paper, we present the *uBeeMe* platform that supports the development of cooperative applications for ubiquitous and mobile scenarios. It provides reusable basis components to relieve the application developer in solving recurring problems and to enable him/her to focus on the real problem. Based on a possible collaborative application scenario on ground exploration we identify components which should be contained in such a platform. Thereafter, we give an overview of the platform structure and implementation and describe some its main components in more detail. These are mobile group communication, localization support, and mobility management.**

***Keywords*—Mobile Collaboration, Collaborative Applications, Mobile Group Communication, User-Localization, XMPP, Handover, Mobility Management**

I. Introduction

Mobility and cooperation will be characterizing features of the future Internet. Already today they play an important role in the design of many Internet applications in the form of collaborative tools, such as whiteboards, application and desktop sharing, collaborative editors, etc., which are often integrated in audio or video conference software. The growing mobility of Internet users requires to collaborate not only in wired networks, but also with partners who are only accessible via mobile devices to include them, e.g., via a mobile conference, in decision making processes and to provide them the needed information over information sharing platforms. The increasing power of mobile devices, especially tablets and smartphones, allows this. The challenging aspect in designing such applications is the need to support heterogeneous application scenarios, devices, resources, and communication schemes. In addition, they should simultaneously support both mobile or nomadic users, and stationary ones.

A wide range of mobile collaborative applications are conceivable, such as mobile conferences, collaborative writing, mobile games, mobile e-health applications, remote diagnostics, monitoring/tracking, mobile Business-to-Business (B2B) applications, and many more. Machine-to-machine communication in which mobile devices and embedded systems ad hoc connect and cooperate, or explore their environment is another important area of these applications. The development of each of such services is costly, not only with respect to the technical software development process, but also because many adaptations to different runtime environments (hardware interfaces, operating system software) are required. Often the interface requirements change with the appearance of new operating systems. The requirements for the design of services depending on user acceptance and new demands vary constantly as well. In addition, the integration of additional features, such as seamless network changes, group support, and security functions, may be required.

In order to enable efficient application developments despite this diversity of design demands the development of a platform that provides important functions of collaborative applications (apps) is an appropriate approach. It allows the application developer to focus on the problem to be solved and relieves him/her of the repeated development of similar components. This can be done in two ways: (1) centralized by using a dedicated server infrastructure to handle all application-specific tasks or (2) distributed by integrating the application in a peer-to-peer (P2P) like manner into the mobile devices. In this paper, we follow the second way and present with *uBeeMe* a platform which provides basic functions to set up collaborative applications. Unlike already existing mobile platforms [1], [2], [3], it not only supports the development of mobile apps, but also provides basic functions for supporting collaboration, such as the simultaneous use of different network technologies, mobile group management, localization, handover support, and security features. The *uBeeMe* platform can be used as basis for developing a wide range of cooperative applications. In the following we describe a possible collaborative application scenario in which we identify important components such a platform should contain. Thereafter in Section III, we describe the design objectives and the architecture of the platform. In the following chapters we introduce the main components more in detail. In Section IV we describe how a stable group communication in mobile settings can be ensured. Section V discusses how cooperating devices and objects can be located. Section VI deals with the change of mobile devices between networks. In Section VII we describe how the collaborative scenario can be implemented with *uBeeMe*. Section VIII provides related work. Some final remarks conclude the paper.

COLLABORATECOM 2013, October 20-23, Austin, United States

DOI 10.4108/icst.collaboratecom.2013.254049

II. Application Scenario: Collaborative Ground Exploration

The damage caused by natural or man-made disasters is becoming increasingly larger. Fast aid in disaster areas is therefore of utmost importance. Due to the growing population densities in endangered areas and the expansion of urban, densely populated residential areas, the number of people potentially affected is increasing. In addition to financial losses and a substantial environmental degradation, a large number of lives are often lost. Disasters in urban areas go often hand in hand with the destruction of the urban infrastructure, such as communication networks, roads, waste supply and disposal facilities. Of particular severity are disasters in which toxic or harmful substances are released. Such events may be the result of chemical accidents, terrorist attacks, or reactor disasters, as recently demonstrated in Fukushima. Often a large part of the local infrastructure is destroyed, roads are usually impassable, and a large number of people is buried or missed. Without additional means, search and rescue of victims is only possible with great danger for the rescue teams, e.g., by poisoning or receiving significantly increased radiation doses. Robots can effectively support the rescue workers in this situation.

In our application scenario we consider a group of autonomous robots which cooperatively explore disaster areas or unknown grounds, which are difficult or dangerous to access for human beings because they are difficult to reach or contaminated with chemicals or radiation. The robots are equipped with an ad-hoc communication capability and connected with a stationary control center. The latter is installed outside the hazardous area at a location that protects the rescue workers from radiation and other hazards, and serves the operational coordination of the autonomous units. The task of the robots is to search for victims and to map the area regarding selected parameters (e.g., radioactivity, poison concentration, etc.) to prepare a hazard map. The autonomous units should exchange all state, position, and measurement data among each other and the control center, including positions of victims and found obstacles to create a global knowledge. Thus, each autonomous unit knows the position of the other units and can adapt its search strategy accordingly. If some units fail the global group knowledge ensures the individual measurement results, even if the corresponding autonomous unit is getting lost during operation. When losing an autonomous unit, the entire group accordingly adapts its search strategy. The provision of a global knowledge among all group members requires a group communication protocol that ensures the consistency of the knowledge at each node. Moreover, a network abstraction is needed that performs the routing and the message forwarding between different networks to enable communication between stationary and ad-hoc networks. A localization module is ultimately required in each autonomous unit for locating the cooperating units and to determine their IP address as presumption for setting up the group or for finding find lost units to reintegrate them into the group.

The search in unknown or disaster areas is dangerous. For example, falling debris may damage or destroy individual autonomous units, or they may fail otherwise due to the local conditions. Regardless of individual problems, the primary objective of the autonomous units is to completely explore the area, to map it, and to search for victims. The autonomous units usually apply a w-formation when moving through an exploration area. In this formation each autonomous unit stays in communication range to at least two other units, e.g., its neighbors on the left- and the right-hand side. Each peer can communicate with other peers, since intermediate peers act as relaying peers. Depending on the environment of the exploration area (w.r.t. its surface, possible debris or barriers, etc.), the formation has to be adapted continuously. In addition, it is necessary that each peer has a stable communication link to the remaining group, otherwise a different formation has to be applied.

Due to environmental conditions, it may happen that individual autonomous units temporarily lose their connection with the communication group. Reasons for these disconnections may be areas with a partial spark shade, sinks, or other unfavorable conditions for radio transmission. Connection outages and resource exhaustions are generally a problem mobile collaborative applications have to cope with. They are frequent reasons for node failures. Many protocols exclude the respective nodes when they do not acknowledge a message or react appropriately. This helps to preserve virtual synchrony [4], but application-, state-, and resource-related information is lost for the respective peer. It has to explicitly be re-invited, what requires additional resynchronization efforts. Therefore a rejoin function in group communication protocol that allows one to re-integrate into the group after a limited connection loss of members is desireable.

In unknown areas there is usually no infrastructure-based communication network available or only to limited extend, e.g., after a disaster. Therefore, the autonomous units are primarily equipped with an ad-hoc communication capability. However, if infrastructure-based networks are available, which provide a higher bandwidth and transmission capacity compared to the ad-hoc communication, these should be used, i.e., the autonomous units are supposed to change into a more performing network. This serves to optimize the communication connections of the individual, autonomous units. A handover module is required in each autonomous unit to perform the network change and to select the optimal network. The selection of the network should depend on the applied parameter setting, e.g., signal strength, cost, energy consumption, and others. Usually each mobile device independently chooses the optimal network for it. In a group-oriented application like the exploring robots this may lead to an uncoordinated network selection that can cause increased energy consumption. Therefore, a coordinated handover decision should be applied that ensures an optimal quality of service for the collaborative application and a low energy consumption for the involved mobile devices.

III. *uBeeMe* Platform

A. Design Aspects

The *uBeeMe* platform (see Fig. 1) provides reusable basis components to support the development of mobile collaborative applications. It aims at relieving applications developers from solving recurring problems, such as network handovers, group communication, device localization, and firewall/NAT[1] traversal. There have been four main requirements when designing the platform:

- **Collaboration support:** The platform should support a spontaneous communication and cooperation across different networks with various partners without having to resort to a complex server infrastructure. Therefore, a peer-to-peer (P2P) approach has been chosen with the aim of building a lean overlay structure that can be configured according to the application requirements.
- **Mobility Support:** The applications should be network-independent and can be used simultaneously both in wired and in mobile networks. When a user changes the network the application should not interrupt the application as possible.
- **Abstraction, encapsulation, and extensibility:** To enable cooperation with users who use other devices or are active in other settings, the platform must support multiple operating systems, system devices and network interfaces. New components and environments should be easy to integrate.
- **Secure communication:** A confidential and secure group communication support in a mobile environment should be provided for applications that require it, such as meetings or collaborative editing of documents. Since no central security authority is used, a distributed key management for secure deployment and distribution of keys is required.

The *uBeeMe* platform has a modular structure allowing application developers to select the required functions for the particular application. This significantly reduces development efforts and saves resources, especially on mobile devices. Using a plug-in mechanism, it is further possible to dynamically reload modules at runtime and to remove them after use. Note that *uBeeMe* is not a middleware because it also permits communication with non-*uBeeMe* applications via standardized interfaces, e.g., XMPP [5]. *uBeeMe* envisages that third-party developers can integrate their own components into the platform to gradually expand its functionality and to flexibly adapt it to new requirements.

Application developments for mobile end systems still highly depend on the given operating system environment. The *uBeeMe* platform wants to support an application development independently from a certain operating system. It will gradually be made available for all major mobile and wired network operating system environments. Currently it is available for Android, Windows, and Windows Mobile.

[1]NAT – Network Address Translation

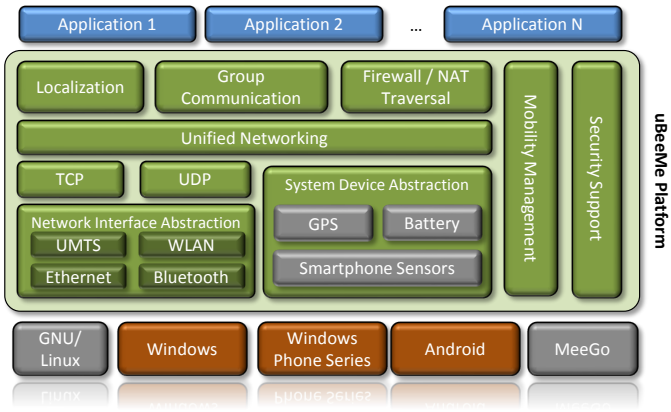

Figure 1. *uBeeMe* architecture

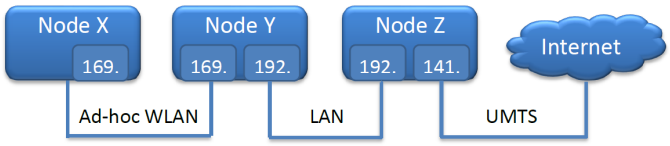

Figure 2. Communication in the Unified Network

B. Platform Architecture

The architecture of the *uBeeMe* platform is depicted in Fig. 1. It divides in core components and modules to support the application. Core components are those ones that provide basic functions. These are the event system for the distribution of platform-internal events, the network abstraction for the system-wide use of different network technologies, the module management for loading and unloading of modules, and a poll mechanism for providing system and platform information.

The *uBeeMe* platform aims at supporting applications running in different network environments. The network interfaces of the respective communication devices are encapsulated through the *network interface* abstraction component. It provides a uniform interface for accessing UMTS, WLAN, Ethernet, and Bluetooth and allows it to set up/release connections or to query of connection parameters, respectively. Below the unified network a TCP and UDP interface is provided. TCP is required for such applications like the localization, UDP for chat, media transmissions, etc. Analogously, device components, such as GPS or position and acceleration sensors, can be accessed via an uniform interface - the *device abstraction*.

The module *unified network* unites the available networks to a virtual network in which all nodes can directly communicate with each other. It is also responsible for the routing of data to be sent. This enables the application to set up point-to-point connections with participants in different networks (with potentially different network technologies) in a unified way. Fig. 2 gives an example of a network configuration consisting of three mobile devices. Here, the node X has no direct connection with node Z or the Internet, only an ad-hoc WiFi connection to node Y.

The *unified network* module ensures that all nodes can

communicate with each other by performing a routing of the data across network boundaries. For ad hoc networks, the *unified network* completely handles the routing, while in infrastructure networks, the communication software embedded in the operating system performs this task. Nodes that are active in several networks provide other nodes access to their networks by acting as a relaying peer. Thus, the *unified network* is transparent to the application.

IV. Mobile Group Communication

Group communication is an important feature in mobile environments to support cooperation among partners in wired and in mobile networks. In practice, collaborative applications are usually provided using a fixed infrastructure, e.g., a dedicated group server, to handle and decide all group-related tasks. A growing number of mobile collaborative applications, which are characterized by a limited number of nodes which often spontaneously connect and closely interact, as described above. Other examples are mobile multiplayer online games and group monitoring systems. Established group communication systems, such as ISIS [6], JGroups [7], or GCP [8], are based on LAN- or WAN-centered approaches and inadequately support mobility. Often fluctuating numbers of participants (churn), unexpected disconnections, and an unstable quality of service (QoS) are insufficiently or not at all addressed in these protocols.

For the *uBeeMe* platform, a flexible distributed group communication protocol, called *Moversight*, has been developed that addresses the mobility-related issues through a lightweight peer-to-peer (P2P) approach [9]. *Moversight* provides a virtual synchronous group communication service for closed groups [4] to support a reliable data transport and to ensure a consistent view on the system state for locally performed group decisions. Unlike other protocols, it allows the temporary absence of group participants caused by churn effects without affecting neither the virtual synchrony in the group nor delaying their communication. Peers can only join by explicit invitation. The protocol does not require central infrastructure elements, such as a bootstrap or group access server, sequencer, or rights broker. It provides a flexible, scalable, and self-organizing structure which together with the unified network (cp. Section II) is also suitable for ad-hoc network environments.

Moversight supports closed groups up to 100 members. The group is divided into clusters to reduce the communication cost and to limit the link management effects caused by moving peers. Each cluster has a master which manages the communication among the clusters. The other members of the cluster are called slaves. Each peer is assigned to exactly one cluster. It has the property to change the assigned role, if required, because each peer has the same knowledge of the group. The assignment to clusters is optimized using a dedicated placement strategy. The principle of the cluster-based communication scheme is depicted in Fig. 3 a). When a peer (slave) wants to disseminate a message to the group, it first sends it to its master which forwards it to the masters of the other clusters, i.e., only the masters communicate among each other. The masters disseminate then the message to their slaves. Since *Moversight* ensures virtual synchrony, it has to ensure a total ordered data delivery. To ensure the latter each message contains the peer ID of the sender, the peer ID of the last hop, and the ID of the current view of the group composition. The delivery decision depends on the global reception time which is based on the logical time concept of [10] (see Fig. 3 b). When a peer receives a message, it adds it to the message queue and marks it as *undeliverable* first. The queue orders the messages according to their logical arrival time. A peer acknowledges the reception to its master with the logical *local reception time* (LT). The master collects the LTs of its cluster. To reduce the number of LTs processed in the whole procedure the master selects the maximum value of the collected LTs (including its local one) and sends it to the originator master, which in turn collects the logical local reception times of all clusters and calculates the logical *global reception time* (GT) of the message. The originator broadcasts the global reception time to all group members. These update the value of the respective message in their message queues, mark them as *deliverable*, and reorder the queues according to the arrival times. The deliverable messages at the top of the queues are handed over to the application until the first undeliverable message appears.

Existing group communication protocols that apply the virtual synchrony paradigm do not support mobility [6], [7], [8]. They usually immediately exclude peers when they have network connection problems. This is not acceptable in mobile environments due to the high probability of network- and mobility-related communication failures. An excluded peer loses its synchronization with the application. For this reason, *Moversight* allows a limited connection loss. It provides a rejoin operation which resynchronizes the rejoining peer with the group preserving virtual synchrony. The principle is called

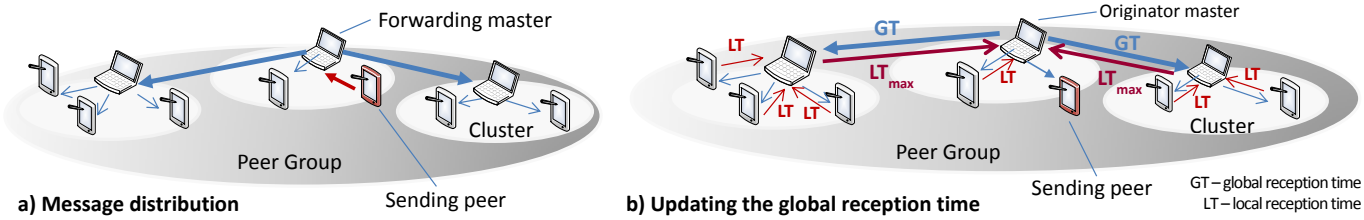

Figure 3. Principle of message distribution and delivery in *Moversight*

the *mobile optimistic virtual synchrony* (MOVS). When a peer loses the connection with the group *Moversight* continues to deliver messages to the group as before, but it informs the group members which peers have been lost. When one of the peers rejoins the group within a certain time interval, the current view remains unchanged. The state of the rejoined peer is resynchronized with that of the group during the rejoin operation. If the lost peers are unable to reconnect a new view for the reduced group will be installed. Thus, the application or the respective services are able to determine whether they have to discard the messages or to process them (depending on the application- and service-related requirements). This allows the application to progress in situations in which it would otherwise be blocked by member exclusion and re-invitation, and subsequent state transfer operations. A detailed description of this principle and other aspects of the Moversight protocol is given in [9].

We have evaluated *Moversight* in various scenarios whether it is able to create a group and disseminate group data, and to ensure mobile opportunistic virtual synchrony in a reasonable time. We implemented the protocol using the *OMNeT++*[2] simulation framework on top of the *INET*[3]-UDP/IP stack to simulate a realistic Internet protocol stack. We considered various group size scenarios up to 96 members which were connected via a backbone network with 10, 100, or 1000 INET routers to exclude network-related effects in the measurements. We stepwise enlarged the group size to 12, 24, 48, and 96 members with a cluster size of six each. Fig. 4 shows as example the measured delay for disseminating messages to the groups. Since the messages are delivered in a total order based on the global virtual reception time, the message delivery time depends on the number of messages sent in parallel. The more peers simultaneously send messages the greater is the delay of individual messages due to the total message ordering. Depending on the queue length of each peer and its position in the overlay, the message delay varies for each peer in the group. To mitigate the impact of the delay distribution we calculated the average message delay for the data exchanged in different simulation settings based on the average arrival time. Further measurements are presented in [9]. The results show that *Moversight* performs efficiently enough to provide a stable basis for mobile collaborative applications with up to 100 members.

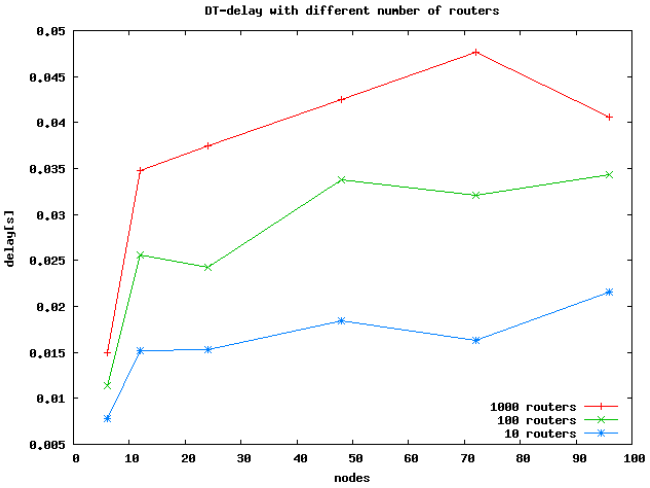

Figure 4. Average message delay for various group sizes

V. Localization Support

The localization of participants and devices is of crucial importance in mobile cooperation to set up communication relations and to provide group awarness. This is the task of the *localization* module which closely cooperates with the *firewall/ NAT traversal* module. The *localization* module determines the transport addresses of the communication partners. Each user is assigned to a fixed user ID that is mapped on a valid transport address.

A design goal of the localization support has been to create a possibly decentralized and adaptable solution for locating users in ad-hoc and infrastructure environments using a standardized signaling protocol. For this, the *Extensible Messaging and Presence Protocol* (XMPP) [5] has been chosen, which has its origins in the Jabber protocol. XMPP provides a collection of open technologies for the implementation of instant messaging (IM) services, voice over IP (VoIP) applications, and collaborations. The architecture of XMPP is based on a decentralized client/server structure. Several centralized networks are connected together to a network by means of servers thus preventing a single point of failure or an overload situation. Participants with Internet access directly register and authenticate at an XMPP server, whereas participants in ad-hoc environments find each other via the XMPP Extension Protocol (XEP) *Serverless Messaging*. *uBeeMe* ensures that all participants always use the same user identification JID (for details we refer to [11]). So it is possible to explicitly search for participants regardless of the environment they are in. To couple ad-hoc networks with the conventional XMPP infrastructure, the routing functionality of the *unified network* module is used. For this, the nodes with Internet access act as relaying peers (i.e., agents). An agent has in addition to the WLAN modules for UMTS or Ethernet to provide Internet sharing. This has the advantage that each node directly authenticates at the server and it ensures the compatibility with the XMPP standard. The existing server infrastructure can be used. This has the further advantage that the users of the *uBeeMe* platform can use any JID. They are not confined to a platform-specific user ID. For example, users of the instant messengers *Google Talk* can continue to use their account and the associated contacts on the *uBeeMe* platform. Fig. 5 gives an example of a network structure in which the mobile users form an ad-hoc network and join a multicast group.

[2]http://www.omnetpp.org/
[3]http://inet.omnetpp.org/

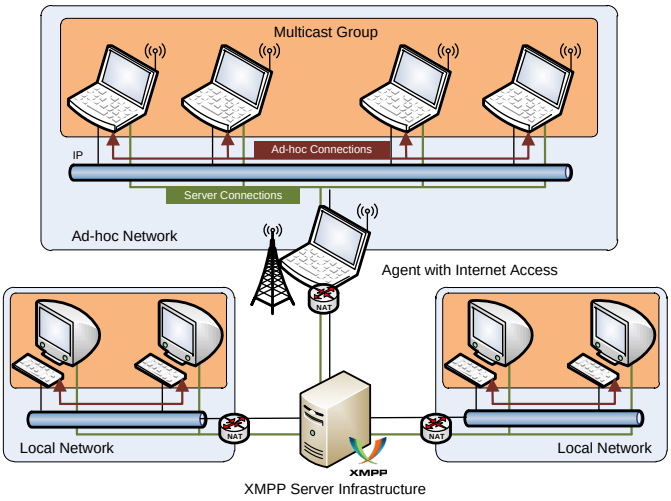

Figure 5. Connecting an ad hoc network with an XMPP infrastructure

In the ad-hoc network each node evaluates the published messages of the multicast group and creates based on this a list of participants. The relaying peer (i.e., agent) resides in both environments and connects the ad-hoc and infrastructure networks. Thus, a dedicated node that exclusively acts as gateway is not required. Additional localization services are provided based on the XMPP basic functionality and the XMPP protocol extensions. This applies in particular to XEP-0166 - *Jingle*, XEP-0174 - *Serverless Messaging*, and XEP-0250 - *C2C Authentication using TLS*.

In addition, an encryption of the XMPP communication for both environments has been introduced. The encryption and authentication of client/server connections is an integral part of the XMPP core specification. The authentication of the participants in the ad-hoc environments, in contrast, is defined by XEP-0250 using the *Secure Remote Password* (SRP) protocol [12]. The SRP protocol was originally designed for client/server authentication. In order to use the protocol for a pairwise authentication in ad-hoc environments adjustments were necessary. So the password that the partners, for example, orally agree is entered on both nodes during the authentication process. Then *Jingle* performs the localization by exchanging the transport addresses via the authenticated and encrypted communication channels. Bypassing NAT systems using techniques, such as *Interactive Connectivity Establishment* (ICE) [13], is necessary nowadays because Internet providers have started to regulate and protect the data traffic in mobile environments with the help of NATs.

Jingle has been chosen because it enables the localization of participants behind NAT systems. In addition, the necessary XML messages for the address exchange have already been defined, so that there is no need to define a separate message format for this. If the participant is located behind a NAT system, the necessary information for NAT traversal is transmitted. For this purpose, the *Jingle* ICE UDP transport method specifies the attributes using the ICE protocol. A NAT traversal is only relevant for the infrastructure environment and not for connections within the ad-hoc environment. In the ad-hoc environment *Jingle* is solely used to exchange the transport addresses over the authenticated and encrypted XMPP communication channel to exchange. Both environments consistently use the *Jingle* extension to exchange addresses.

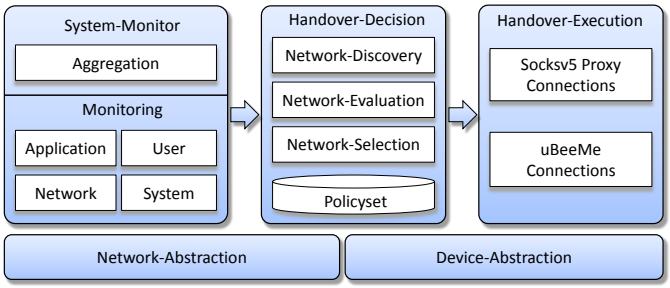

Figure 6. *uBeeMe* mobility management

VI. Mobility Management

The *uBeeMe* platform supports an always-on usage. The *mobility management* ensures in close interaction with the *device* and *network abstraction* and the *unified network* a seamless network exchange which is either initiated when losing the connection to the network or when falling below a defined quality of service level. Changes of the application requirements may also trigger network changes. The mobility management supports both intra-technology (horizontal) and inter-technology (vertical) handovers. Regarding the latter, the mobility management aims at supporting soft handovers when the mobile device is connected with two networks simultaneously, so that the connection is moved without interruption. As heterogeneous networks differ in their requirements and characteristics, reconfigurable handover mechanisms and protocols are used for this. These mechanisms allow users and applications to optimally adapt their handover strategies at runtime. A reconfigurable mobility management is used which interacts with the mobile device, the network interface and the different protocols (see Fig. 6).

A challenge for every network change represents the search for available networks for which an authorization exists. An inherent problem of the network search is that it requires that the target network interfaces are activated. This increases the energy consumption depending on the search frequency. Configurable software defined radio interfaces that monitor all frequencies are a possible solution, but they rarely deployed today. Usually centralized localization services are used, such as OpenBMAP [14] or Skyhook [15], which provide information on available networks based on GPS coordinates or cell identifiers without activating the network interfaces.

Based on this information and the analysis of other network connection, system, and application level parameters, the need for a handover is decided. The decision for or against a handover usually requires some calculations, which influence the energy consumption of the mobile device depending on the required accuracy [16]. Therefore, we apply a multi-step procedure starting from handover initiation, network search,

and final decision-making. The initiation step monitors the state of the mobile during rest and move using a lightweight threshold algorithm. When a critical state is reached the search for appropriate networks is triggered and the network change is performed. In group-oriented applications, as described in Section II, such singular handover decisions may lead to an uncoordinated network selection. Mobile systems are anxious to select networks that optimally meet their own requirements. Therefore, they prefer networks with the highest data rate, the highest signal strength, or the best energy efficiency combined with the lowest cost. Because of this "egoistical" behavior and the lack of coordination among the mobile systems, the network selection of the group may be inefficient. The local decisions may reduce the energy consumption of individual mobile systems, but the energy consumption may increase for the entire collaborative application. For example, when a mobile system of the group changes in a network that has a lower data rate, then the other systems need more time to transfer their data to this system due to the reduced data rate. To mitigate this problem the decision algorithm supports a coordinated handover decision. It calculates a group benefit for each available networks based one which it proposes which one to choose.

For the handover decision, the *uBeeMe* platform applies a fuzzy-based decision algorithm which supports vertical handovers on off-the-shelf devices [17]. The algorithm uses an adaptive parameter set which is determined by a continuous system monitoring, e.g., by querying the network layer and higher protocol layers. It estimates the remaining time to evaluate the quality of the discovered networks, to select the best of them, and to handover the connection to this network. Often the discovered networks though do not allow to setting up a connection because firewalls and NAT routers prevent the communication. To take such constraints into account, the handover algorithms is optimized regarding NAT traversal. This allows it to include the time needed for NAT traversal into the target network assessment to estimate the feasibility of the handover decision.

We evaluated the applicability of the algorithm in various real-life experiments on several mobile devices regarding connection optimization and assurance. Fig. 7 and Fig. 8 show an experiment in which the network load is analyzed when transmitting a large file using a relayed TCP connection. In Fig. 7) the transmission starts first in a UMTS network, but it appears that it works at 100%, so that the algorithm decides after 40 seconds to search for alternative networks and after 40 seconds to perform a handover into a WLAN which provides a higher data rate. Fig. 8) shows the opposite process of a connection assurance. Here the file download starts in the WLAN, but due to user movement to the network edge it switches to the UMTS network after 90 seconds because the latter now provides a better connection quality.

A fundamental problem in evaluating available networks is that the mobile devices are usually equipped only with one WLAN interface. If the network interface is in use WLANs in the immediate environment can be detected, but not which

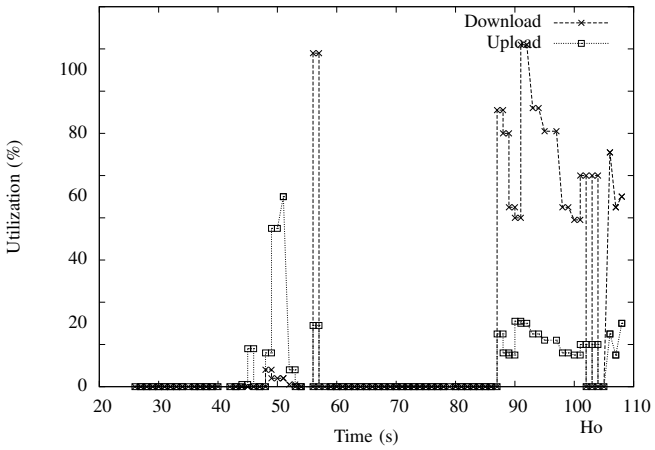

Figure 7. File download handover to WLAN

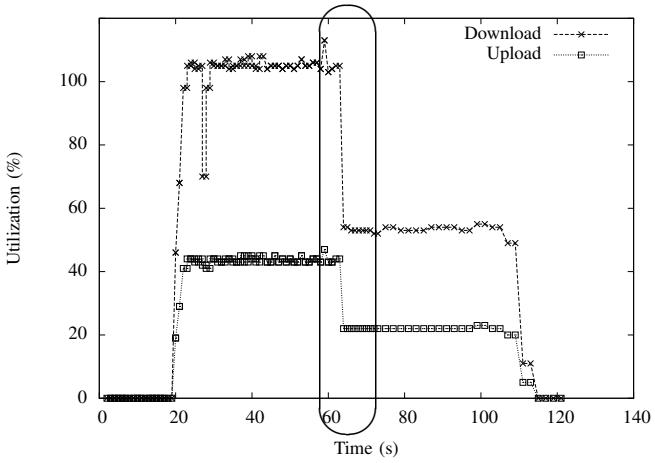

Figure 8. File download handover to UMTS

ports they use. Approaches, such as IEEE 802.21, that can obtain information about alternative networks and open ports from a central information server are usually not available both with the network providers and on private base. Therefore, there is a high probability that an alternative network does not allow the service use. In particular, if the network uses only MAC authentication, i.e., a connection is established to the network, but one cannot communicate about it. The measurement of alternative WLANs also interrupts the communication of the application (see Fig. 9 a). To nevertheless ensure service availability over the destination network and to select the best one from the set of available alternative networks we use virtual network IEEE 802.11 interfaces [18], which use the physical network interface in a time-division multiplexing manner (see Fig. 9 b,c). Each of the logical network interfaces can be connected to a wireless network. Thus, several networks can be analyzed almost in parallel regarding the provided quality.

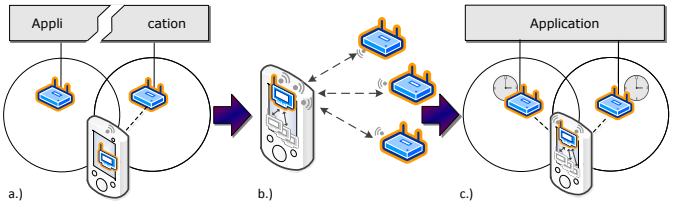

Figure 9. Virtualized 802.11 interfaces

In order to optimize the execution of several collaborative applications on a mobile device the handover functionality is not directly integrated into the application. It runs as a local operating system service. Thus, there is the option to control several applications during a network change without having to integrate the handover functionality in each application.

VII. Implementing Collaborative Ground Exploration with *uBeeMe*

With the reusable basis components of *uBeeMe* we created a robot prototype which fulfills all requirements of the collaborative scenario mentioned in Section II.

The hardware of our robot prototype consists of a LEGO NXT[4] and an Android[5]-powered smartphone. The smartphone, running *uBeeMe*, is interconnected via Bluetooth to the NXT for sending the steering commands to the robot. WLAN communication is only used between the robots to build an ad-hoc communication chain and to use the most performing network. In a group only one robot serves the Internet connection via UMTS to save bandwidth for the interaction to the group from the control center and vice versa.

The most part of our implementation consists of existing *uBeeMe* components. In *uBeeMe* a component is realized as a module which can send data to and receive data from the network. Modules can be chained up which allows a module to forward data to its successor module. Each module in this chain has access to process or extend the forwarded data. The chain of the modules always ends up with a UDP or TCP module. As an application developer we only had to take care about *uBeeMe* events from the activated modules and trigger the next steps via a module for our application. For the implementation of the collaborative ground exploration nearly all components of the *uBeeMe* platform are reused.

The *localization* and *unified networking* modules abstract the current IP address and the communication flow in the infrastructure and ad hoc network from our application (e.g., send and receive messages), while the *handover* module will broadcast an event if the connection to the control center or to the robot group gets lost during the next movement. Our application only needs to implement how to react on this event for the movement of the robot: In our case the appropriate steering command (e.g., stop, go to last position) will be sent to the NXT to prevent a connection lost.

[4]http://www.mindstorms.lego.com/
[5]http://www.android.com/

The mobile group communication is handled via the *group communication* module to ensure a synchronized view for all involved communication partners (i.e., robots, control center). If a connection loss was unavoidable for a communication partner, it will be synchronized when the communication partner rejoins the group again. Moversight uses the *localization* module to find its group members and the *handover* module to get informed about a connection loss and established connections. From an application developer point of view only the exchanged information need to be defined. The *uBeeMe* platform handles transparently the serialization of the data packets and their way through the current network structure.

VIII. Related Work

Enabling mobile collaboration can be done in different ways. Most of the proposed approaches are complex, address only specific uses cases and communication schemes while providing a limited mobility support. *SpoVNet* [19] is a virtual network architecture which provides services within heterogeneous networks. SpoVNet consists of a base-underlay, which interconnects a number of nodes, uniquely identifiable within SpoVNet, that is responsible for a base connectivity and hides the network heterogeneity. It mainly addresses wired networks for providing its virtual network services, without providing mobility related services. In contrary *uBeeMe*'s created overlay is lightweight and only application specific. The *ROMA* middleware platform [20] provides capabilities for switching seamlessly between different wireless networks. Legacy applications are bind to ROMA via dedicated access-nodes. Unfortunately, ROMA does not consider the energy consumption of its operations. *REACH* [21] oder *NetMotion Mobility XE* [22] providing server-based group communication capabilities, which equip mobile users with a static IP-address. Both approaches relay on dedicated servers, or modified software that constitute single point of failures and high maintenance costs. Distributed solutions like uBeeMe avoid these disadvantages.

The approaches *MADAM* [23] and *MUSIC* [24] investigating adaptive mobile applications. MADAM focuses on reusable adaption strategies and their representation in modeling languages. A reconfigurable middleware, which automatically adapts to a dedicated usage or environment, is provided by MUSIC. It provides an application context that remains stable within a chain of adaption operations. Both approaches focus on mobile devices and the application development process. In contrast, *uBeeMe* provides a set of reusable components, usable within a number of use cases and scenarios of mobile collaborative applications. Aspects as mobile virtual synchrony within closed groups, localization for hybrid network environments, seamless handover, or the network selection by considering its energy and capacity properties in conjunction with limited energy budgets are not considered by the alternative approaches and platforms.

IX. Conclusions

Mobility, cooperation, and ubiquitous network access will more than ever shape the development of application solutions in the near future. This requires flexible and adaptive solutions which can be optimally combined to achieve synergies. The provision of basis functions in a platform relieves the application developer from solving recurring problems. The *uBeeMe* platform presented in this paper provides such functions. Thanks to its modular structure, it can be easily extended, updated, and maintained. It further allows third parties to integrate their own components. An essential aspect of the development was the careful handling of the limited resources of mobile devices (e.g., memory, CPU, energy reserves). To support ad-hoc communication scenarios the platform does not have central infrastructure elements.

Currently we are focusing on the integration of security functions into the platform: The compliance with security requirements is essential for many collaborative applications, as mobile and wireless communication in the scenarios considered here is particularly vulnerable to malicious attacks. For this reason, the *uBeeMe* platform includes a security module that can be optionally enabled. It will provide basic services, such as authentication, authorization, encryption, and data integrity while focusing on the protection of the group communication and the reliable authentication of the partners. For the latter, a password-based authentication protocol will be provided among others [25]. Through the password-based authentication in particular the flexibility of the communication partners is to be supported. Basis of the confidential communication is the use of a common group key whose management is the core component of the *uBeeMe* security architecture. In parallel, we are working on the development of various mobile collaborative applications. Here, the focus is primarily on applications in health care, care of elderly, and environmental monitoring.

References

[1] (2013) Appcelerator platform. [Online]. Available: http://www.appcelerator.com/platform/appcelerator-platform/

[2] (2013) Brewmp platform. [Online]. Available: https://developer.brewmp.com/home

[3] (2013) Phonegap platform. [Online]. Available: http://phonegap.com/

[4] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," *ACM SIGCOMM - Computer Communication Review*, vol. 32, pp. 205–217, August 2002.

[5] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Core," IETF, Request for Comment 6120, Mar. 2011. [Online]. Available: http://www.ietf.org/rfc/rfc6120.txt

[6] K. P. Birman and R. V. Renesse, *Reliable Distributed Computing with the Isis Toolkit*, K. P. Birman and R. V. Renesse, Eds. IEEE Computer Society Press, 1994.

[7] A. Montresor and M. A. Zamboni, "The jgroup reliable distributed object model," in *In Second IFIP WG 6.1 International Working Conference on Distributed Applications and Interoperable Systems (DAIS99)*, 1999, pp. 389–402. [Online]. Available: http://www.jgroups.org/papers.html

[8] M. Zühlke and H. König, "Gcp - a group communication protocol for supporting closed groups in the internet," in *SMARTNET 2002. Proceedings of IFIP TC6 WG 6.7 7th International Conference on Smart Networks 2002*, 2002, pp. 211 – 227.

[9] J. Gäbler and H. König, "Moversight: A group communication protocol for mobile collaborative applications," in *Proceedings of the 6th Joint IFIP Wireless and Mobile Networking Conference (WMNC 2013)*. Dubai, United Arab Emirates: IEEE, April 2013.

[10] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Commun. ACM*, vol. 21, pp. 558–565, July 1978.

[11] R. Klauck and M. Kirsche, "Combining Mobile XMPP Entities and Cloud Services for Collaborative Post-Disaster Management in Hybrid Network Environments," *Mobile Networks and Applications*, vol. 18, pp. 253–270, April 2013. [Online]. Available: http://dx.doi.org/10.1007/s11036-012-0391-1

[12] T. Wu, "The SRP Authentication and Key Exchange System," IETF, http://www.ietf.org/rfc/rfc2945.txt, Request for Comment 2945, Sep. 2000.

[13] J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols," IETF, http://www.ietf.org/rfc/rfc5245.txt, Request for Comment 5245, April 2010.

[14] (2013) Openbmap. [Online]. Available: http://openbmap.org

[15] (2013) Skyhook. [Online]. Available: http://www.skyhookwireless.com

[16] J. Marquez-Barja, C. T. Calafate, J.-C. Cano, and P. Manzoni, "An overview of vertical handover techniques: Algorithms, protocols and tools," *Computer Communications*, vol. 34, no. 8, pp. 985–997, Jun. 2011. [Online]. Available: http://dx.doi.org/10.1016/j.comcom.2010.11.010

[17] M. Pink, T. Pietsch, and H. König, "Towards a seamless mobility solution for the real world: Handover decision," in *Wireless Communication Systems (ISWCS), 2012 International Symposium on*, 2012, pp. 651–655.

[18] A. J. Nicholson, S. Wolchok, and B. D. Noble, "Juggler: Virtual networks for fun and profit," *IEEE Transactions on Mobile Computing*, vol. 9, no. 1, pp. 31–43, 2010.

[19] R. Bless, C. Hübsch, S. Mies, and O. Waldhorst, "The Underlay Abstraction in the Spontaneous Virtual Networks (SpoVNet) Architecture," in *Proc. 4th EuroNGI Conf. on Next Generation Internet Networks (NGI 2008)*, Apr. 2008, pp. 115–122.

[20] A. Popescu, D. Erman, K. de Vogeleer, A. Popescu, and M. Fiedler, "Roma: a middleware framework for seamless handover," in *Network performance engineering*, D. D. Kouvatsos, Ed. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 784–794. [Online]. Available: http://dl.acm.org/citation.cfm?id=1996629.1996669

[21] F. Evers and J. Seitz, "Reach: A roaming-enabled architecture for multi-layer capturing," in *Wireless Communications and Networking Conference, 2008. WCNC 2008. IEEE*, 2008, pp. 2699–2704.

[22] (2011) Netmotion mobility xe. [Online]. Available: http://www.netmotionwireless.com/products

[23] M. Alia, G. Horn, F. Eliassen, M. Khan, R. Fricke, and R. Reichle, "A component-based planning framework for adaptive systems," in *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE*, ser. Lecture Notes in Computer Science, R. Meersman and Z. Tari, Eds. Springer Berlin Heidelberg, 2006, vol. 4276, pp. 1686–1704. [Online]. Available: http://dx.doi.org/10.1007/11914952_45

[24] K. Geihs, R. Reichle, M. Wagner, and M. Khan, "Modeling of context-aware self-adaptive applications in ubiquitous and service-oriented environments," in *Software Engineering for Self-Adaptive Systems*, ser. Lecture Notes in Computer Science, B. Cheng, R. Lemos, H. Giese, P. Inverardi, and J. Magee, Eds. Springer Berlin Heidelberg, 2009, vol. 5525, pp. 146–163. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02161-9_8

[25] F. Liu and H. König, "A simple balanced password-authenticated key agreement protocol," in *Proceedings of the IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. Changsha, P.R. China: IEEE, November 2011, pp. 403–408.