Supporting Automatic Media Resource Selection Using Context-Awareness

Johan Kristiansson, Josef Hallberg, Sara Svensson, Kåre Synnes, Peter Parnes*

Abstract. *In a world of ubiquitous media resources, such as cameras, displays, microphones, provided by a wide variety of multimedia systems, there is a need for automatic resource selection which seamlessly utilizes the best available communication tool from a user perspective. This paper therefore presents an algorithm which uses context-awareness to support automatic media resource selection. The algorithm takes advantage of an abstract classification of privacy, quality, and cost in order to compare media resources to user's preferences. As a proof of concept implementation, the algorithm has been incorporated into an e-meeting application called Marratech.*

1 Introduction

As users' requirements on mobility grow and distributed collaboration becomes increasingly pervasive, the need grows for people to communicate and retain group awareness in mobile settings. Current research trends in networking and multimedia envision ubiquitous multimedia communication where users can seamlessly meet and communicate anytime, anywhere, and from any device. Imagine a world full of communication equipment and softwares, such as cameras, microphones, instant messengers, and e-meeting clients[1], which are scattered throughout the environment and are automatically configured and utilized whenever the user wants to communicate. This scenario raises new challenges, which must be addressed before the vision can become a reality.

Today, users are required to manage a wide variety of communication tools in order to communicate with other users. These tools typically force the user to manually set up communication links, for example entering phone numbers, logging in to e-meeting servers, and configuring cameras. In addition, many tools are inherently incompatible, which requires all communication parties to have access to the same set of tools in order to be reachable by each other. The configuration task becomes even more complicated and time-consuming as the need for ubiquitous communication spreads and the diversity of available communication tools increases.

Ultimately, the best available communication tool should automatically be utilized transparently to other users and independently of which tools they are currently using. Research has been conducted

*Department of Computer Science and Electrical Engineering, Luleå University of Technology, Luleå, Sweden, email: {johan.kristiansson, josef.hallberg, sara.svensson, kare.synnes, peter.parnes}@ltu.se

[1]The term "e-meeting" denotes a group teleconferencing session that can include video, audio and chat among other media. Rather than requiring a dedicated meeting room, e-meetings can take place from the user's desktop and be used for either formal or informal communication.

to provide interoperability [7, 11] and support for mobility [5, 10, 13] to allow use of different tools. However, how to enable a system to automatically choose the most beneficial tool for the user is a problem which so far has received less attention because of its complexity. This paper addresses this problem by decomposing it into more manageable parts, which can be solved independently, in order to provide a deeper understanding of the underlying problems. The paper proposes an algorithm which connects these parts together and uses context-awareness to decide which communication tool that is best for the user.

As a proof of concept, the paper demonstrates how the proposed algorithm can be used to automatically switch between communication tools when using a commercially available e-meeting software. By using the prototype, users can participate in an e-meeting session and the system *automatically* invites an IP-telephony service, which includes a telephone or a mobile phone to the session, as a complement when a desktop e-meeting client does not provide good enough privacy. Similarly, the system can automatically invite an IP-telephony service if the currently used network connection does not provide good enough performance to support audio.

The rest of the paper is organized as follows. Section 2 describes related work and the contribution of this paper. Section 3 introduces a conceptual communication model and discusses issues that must be considered when selecting resource. Section 4 outlines the proposed algorithm and discusses two issues related to its usage. Finally, in section 5 the proof of concept prototype is described followed by a discussion and future work in section 6.

2 Related Work

Context-awareness is a widespread approach to enable systems to provide users with an increased service value. The area of interest to this paper is context-aware communication as defined by [8]. Early work in this field include the ActiveBadge system [15], which tried to improve channel selection through autonomous routing of phone calls to the phone nearest to the user. In contrast, this paper focuses on autonomous routing of media streams to the most appropriate communication resource from a user's perspective.

Similar to the architecture proposed in this paper, the Aura architecture [12, 14] tries to utilize resources in the user's environment to give the user access to desired tools, while simplifying the configuration tasks needed by the user. In contrast to the algorithm proposed in this paper, Aura does not consider the possibility of several resources with the same capabilities. Hence, Aura does not provide any methods for comparing resources in order select the resource that best satisfy a user's needs. Neither does Aura take other aspects, such as cost, into account when selecting a resource.

The Aura architecture identifies simple tasks performed by the user, such as editing text, and then migrate the whole task to a different resource as the user change location, for instance moving a text from one device to another while changing text editor if needed. This means that Aura does not take advantage of the possibility to split up certain tasks and then distribute these tasks to different resources that better meet the user needs. One such task is the communication task which can be split up into sending and receiving of different media. The algorithm proposed in this paper treats these parts as media resources that are selected separately which makes it possible to distribute the tasks onto different devices.

Architectures which focus on network mobility [5, 10, 13] aim at providing support for location up-

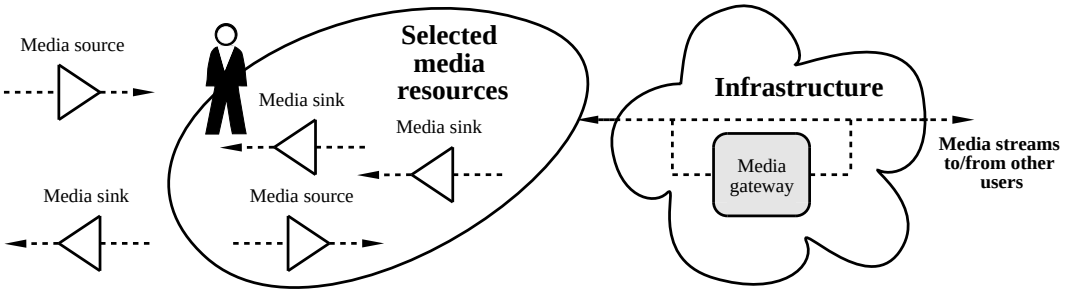

Figure 1. A conceptual communication model.

dates and for preserving established connections when changing terminals or moving across networks. This paper does not look into network mobility in this sense, but rather on how to provide support for automatic selection of appropriate terminals to switch to. The main contribution of this paper is an algorithm which decomposes this selection problem into more manageable parts, thus making it possible for mobility architectures to become more autonomous.

3 Media Resource Management

This section starts with presenting a conceptual communication model which is used throughout this paper. It then goes on with describing the different issues which needs to be considered in order to enable a system to automatically make a choice which suits the user. In particular, the section considers the problems of valuing and comparing resources and proposes possible solutions to them.

3.1 Communication Model

Figure 1 illustrates the used conceptual model showing two communication abstractions, *media source* and *media sink*. A media source is an abstraction over a media-capturing hardware device, such as a camera or microphone, and the software that generates the media stream. After being generated, the media stream is distributed to the recipients via a network infrastructure. When the media stream has been received and processed by the recipients, a media sink presents the media stream to the user. A media sink is an abstraction of a media-rendering hardware device such a loudspeaker or a display and its accompanying software, which is used by the device to send or receive the media stream. The word *media resource* is used to denote either a media sink or a media source in the rest of this paper.

For the sake of simplicity, this paper introduces the term *media resource type* to be able to differentiate between sinks and sources of a particular media. For example, all media sinks which are associated with microphones are of the type audio sink, and all media sources associated with cameras are of the type video source. This makes it possible to conceptually handle situations when a user only wants to either send or receive a particular media. Media resources which are contained within the same unit are defined as *interdependent media resources*. For example, a phone has two interdependent media resources, one of the type audio sink and one of the type audio source, and an e-meeting client might have even more interdependent media resources.

One of the advantages of using the proposed abstractions is that each media resource can be viewed and handled as an isolated unit, thereby enabling computer-mediated communication to be enriched

and improved by letting users freely combine different media resources. To provide such functionality, the system needs to be able to automatically detect and select the best available media resources, and ensure compatibility between senders and receivers. For example to allow a mobile phone to be used as an audio source in an e-meeting session, a SIP-gateway can be used to convert Public Switched Telephone Network traffic to IP traffic. The problem of ensuring compatibility is not addressed in this paper.

3.2 Media Resource Selection Problems

One problem with automatic media resource selection is what level of autonomy should be used. As pointed out in [8], autonomy needs to be carefully balanced in order to provide the desired effect. It is crucial not to remove too much of the user's sense of control as well as to avoid switching to a media resource which the user does not want to use. These types of system failures will most likely not be tolerated. However, at the same time, involving the user in too many of the decisions will burden the user and counteract the requirement of minimal effort. When it comes to autonomously deciding which media resource to switch to, there are several problems which need to be considered; when to look for a better resource, when to switch, and when a specific media should be used. Solving these problems in a good way requires determination of complex causalities and access to a wide variety of contexts, e.g. network performance of current resource, improvement achieved if changing, a user's preferences, and so on. User studies need to be conducted in order to determine these causalities and which contexts to use.

To clarify what information that is relevant to base the decision upon, two concepts need to be more clearly defined, namely *best resource* and *available resource*. This can be done by discussing the following two design issues which are related to the two concepts respectively:

- How can media resources be compared?
- Which media resources should be considered available in a given situation?

The rest of this section discusses these two issues and proposes possible solutions.

3.2.1 Comparing Media Resources

In order for the system to be able to decide between media resources, they have to be comparable in some way. Through this comparison, it must be possible to value media resources in a way that concur with the user's preferences, thus enabling the system to act according to those preferences. One dimension which is interesting from a user point of view is of course quality. The user will most likely want to use the better one of two microphones in a room. When considering media resources, privacy will also play a significant role in what the user prefers. Parameters such as the number of people in the room, what part of the room that is captured by the camera and how publicly the display is located will most likely affect the user preference.

In a future with an abundance of media resources, it is quite likely that all resources can not be used for free. This introduces a third interesting dimension, namely cost which can involve both monetary and non-monetary values. For example, the cost involved could be to pay a given amount of money per byte or allowing a newsletter or advertisement to be sent to the user. These types of

costs are difficult to determine, although for different reasons. The non-monetary costs will have different values to different users, making valuing of resources in a way which concurs with user preferences complicated. Monetary costs are often difficult to decide since the actual cost in many cases is connected to how much data is transferred or how long time the resource is used.

A classical solution when valuing an object is to use some form of classification. To use one quality, one cost, and one privacy classification level to prioritize media resources is thus one way to approach the problem of comparing media resources. Besides giving a foundation for comparison, the use of classification levels will also make it easier for users to convey their preferences in different situations, e.g. ”best possible quality, least possible cost, and at least privacy level seven”. To use classification also provides researchers and developers with an abstraction which separates the issue of valuing media resources from the rest of the selection process.

In order for classification levels to function correctly, the system should be able to reflect a user's opinion of what each level should entail. This implies that users need to have the possibility to influence how the classification levels are calculated. For example, one user might consider it unacceptable (low privacy level) to use a microphone when there are other people in the room while another user might tolerate it (higher privacy level). Another issue is how the different classifications should be weighed in order to take importance into account. This calculation also needs to take the user preferences into account. For example a user might consider bandwidth more important than cost in a certain situation and the opposite in another situation. A different user might always prioritize cost before both privacy and quality.

As many environments are dynamic, a media resource's classification values will change over time. When considering the quality level, network specific context such as packet-loss rate and available bandwidth makes dynamic quality classification a necessity. Also in the case of privacy, the level need to be dynamically recalculated since a resource could be considered to have a lower level of privacy if there are other people in the room, than if the user is alone.

3.2.2 Media Resource Availability

Assigning values to media resources allows for comparison between them, but in an environment which provides a large amount of media resources, it is neither efficient nor preferable to always consider every resource as a possible choice. This brings forward the question of what characterizes an available resource in a certain situation. Naturally, only devices which are not currently busy with other users and which provides the wanted media resource type are relevant, but there are other things to consider as well. For example, when the user wants to send audio it would not be beneficial to the user if the system decided to use a microphone in a building other than where the user is located. This can be solved by using the principle of locality [4].

Locality means that the user only is allowed to use devices which are nearby the user, for example in the same room. Locality makes sense not only for reasons of usefulness but also when considering privacy. Without the principle of locality, cameras and other media sources could be used as remote surveillance devices, thereby introducing a large risk of violating other people's privacy. Locality also ensures that media sinks are not used without the receiver's permission, thereby protecting the receiver from unwanted interruptions. Two issues which need to be considered when applying the principle of locality is what is to be considered as close enough and what should happen when there is not any available resource within this area.

Algorithm 1 Media Resource Selection Algorithm

Input: A media resource type, $type$, e.g. audio sink or video source.
Output: The best media resource, $r_{selected}$, for $type$.

1: $r_{preferred} \Leftarrow null$
2: if **isNeeded(** $type$ **)** then
3: $R \Leftarrow$ all media resources r of $type$ s.t. $\forall\ r \in R$ **isAvailable(** r **)** $\bigwedge$ **Q(** r **)** $\geq Q^{MIN} \bigwedge$ **P(** r **)** $\geq P^{MIN} \bigwedge$ **C(** r **)** $\leq C^{MAX}$
4: for all $r \in R$ do
5: if **credit(** r **)** $>$ **credit(** $r_{preferred}$ **)** then
6: $r_{preferred} \Leftarrow r$
7: end if
8: end for
9: end if
10: $r_{selected} \Leftarrow$ **notify(** $r_{preferred}$ **)**

Even if the principle of locality is useful, it is not necessarily true that a user should have access to all nearby resources. This further narrows the number of relevant resources to those nearby the user which are not busy and which the user has permission to access. The presence of interdependent media resources also affects which resources are relevant. If a user uses an interdependent resource, no other user should be able to use the other resources which the used resource is interdependent with. For example, if a user is utilizing the microphone in a mobile-phone, that user should be the only user able to use the speaker in the same device. This could be seen as a special case of media resources being busy, where all resources which are interdependent with the used one are considered to be occupied.

4 Automatic Media Resource Selection

In order for a system to deal with the problems discussed in subsection 3.2 and choose a suitable media resource, it needs to access and use a situation's context, i.e. it has to be context-aware. A system is said to be context-aware *"if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task"* [1]. Context is in this case any information relevant for the system to make a suitable choice, e.g. user preferences and information about available media resources. Note that this paper does not focus on context acquisition or accompanying problems like context accuracy or confidence. In this section an algorithm for automatic media resource selection is presented. This algorithm breaks down the problem of choosing the best media resource into several abstract functions which can be implemented separately using context-awareness. This section also discusses how the algorithm should gain access to contexts and what should happen once a decision has been made.

4.1 A Media Resource Selection Algorithm

The purpose of the Media Resource Selection Algorithm (MRSA) is to select the best available media resource in accordance with user preferences. The algorithm is based on the discussion in subsection 3.2 to do this and uses classification of quality, privacy, and cost to model media resource values. The algorithm is presented in algorithm 1. The algorithm returns $r_{selected}$, which designates the selected media resource of the inputted type, $type$. Throughout the algorithm, $r_{preferred}$ is used to denote

the media resource which is the currently preferred choice. Further, the variables P^{MIN} and Q^{MIN} signify the minimum values of privacy and quality, and C^{MAX} the maximum cost value that a user favours.

The algorithm decomposes the selection problem into seven abstract functions which need to use context-awareness in order to function properly. These are in order of appearance; $isNeeded$, $isAvailable$, Q, P, C, $credit$, and $notify$. The boolean function $isAvailable(mediaresource)$ checks if a given media resource is available for use. If the function follows the solution proposed in subsubsection 3.2.2, the function should return true if the given resource is nearby, not busy, and if the user has permission to use that particular resource.

Each of the functions $isNeeded$ and $notify$ encapsulates different aspects of the autonomy problems discussed in subsection 3.2. The boolean function $isNeeded(type)$ checks if the given media resource type is needed, i.e. if the user wants to communicate using that media resource type. For example, if a user sends audio in an e-meeting session, the media resource type *audio source* is considered needed for the user who sends the audio stream, and the media resource type *audio sink* is considered needed for all users who receive the audio stream. To decide when the user wants to start or stop using a specific media resource type, more sophisticated reasoning about contexts is needed compared to when the media resource type is continuously used.

The function $notify(mediaresource)$ is used to decide if the user wants to switch to the given media resource or not based on context. This could for example include asking the user if a switch should be performed. The argument can either be the currently used media resource, a new and better media resource, or $null$. The function then decides between continuing to use the currently used media resource if that is an option, switching to the new media resource, or not using any media resource of this type at all. The function returns the selected media resource.

The Q, P, and C functions all take a media resource and calculates the classification level of quality, privacy, and cost respectively. The function $credit(mediaresource)$ weighs the different classification dimensions together according to user preferences in order to compare them. All of the abstract functions in the algorithm need to take a large amount of context parameters into consideration and weigh them together in order to achieve user satisfaction and are therefore difficult to implement, which is illustrated through the discussion in section 3.2.

4.2 Using MRSA

When considering deploying and using the MRSA, there are two issues which need to be discussed. These are; how should the algorithm get access to necessary contexts that are needed to make decisions and how the decisions should be executed. Regarding the first issue, a context storage infrastructure with query-possibilities must exist. It must also be possible to add and update contexts to the infrastructure. One approach for developing a context storage infrastructure is to use a context-awareness platform. Schilit was one of the first to propose an architecture for this purpose [9]. Since then several context-awareness platforms have been developed, each with different characteristics, e.g. Context Toolkit [2] and Context Fabric [3]. Current research moves toward decentralized solutions for privacy and scalability reasons [3].

When using the algorithm in an existing communication system, the algorithm normally resides in an external or internal management component responsible for configuring and utilizing appropriate

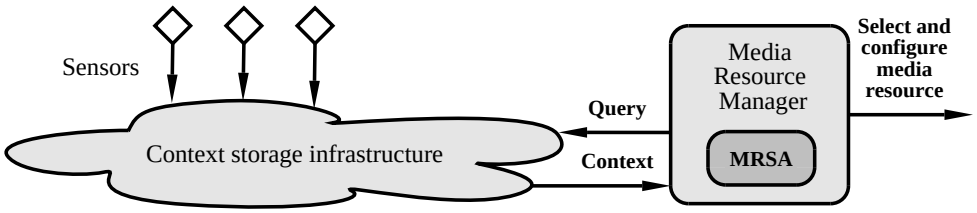

Figure 2. Algorithm Usage

media resources. The management component, the Personal Media Resource Manager, could be implemented as part of a multimedia system. An overview of how the algorithm can be used by a Personal Media Resource Manager together with a context storage infrastructure is illustrated in figure 2.

5 Evaluation

A proof of concept implementation was built by incorporating the architecture with a commercially available e-meeting system provided by Marratech [6]. Marratech is an e-meeting client which provides tools for synchronous interaction including audio, video, chat, and a shared white-board. The client connects to a media gateway called Marratech Manager in order to access and participate in e-meetings. This system was mainly chosen because of the SIP gateway which is built in to the manager, and compatibility with other prototypes developed by the authors. The rest of this section describes the implementation and two test cases in which the prototype has been used.

5.1 Implementation

The prototype is implemented in Java JDK 1.4 in order to make it easy to integrate into the Marratech source code. The implementation uses the described algorithm, with the exception that cost has not been taken into account, and is organized as figure 2 in subsection 4.2. The current implementation does not separate sinks and sources in phones, hence when either the microphone or the speaker in a phone is selected the other is automatically used. A sensor simulator was used to generate two contexts, each of which significantly changes a user's situation, namely location and network performance.

The context storage infrastructure models media resources, environments, and users as Java objects. Each environmental object models a room and keeps information regarding which media resources and which users reside within the room. The user object contains relevant contexts such as active media resources, user's location and a link to the appropriate environment object, and which media resource types the user prefers. By letting the context storage infrastructure model relationships between users, resources, and locations in this manner, e.g. media resources are linked to locations, the infrastructure provides a way to discover resources. Finally, the media resource object pertain contexts for calculating quality and privacy indexes, as well as a SIP address. A communication protocol was also designed to enable the used sensors to add context to the infrastructure.

The Marratech client reports if audio is being used to the context storage infrastructure. This context

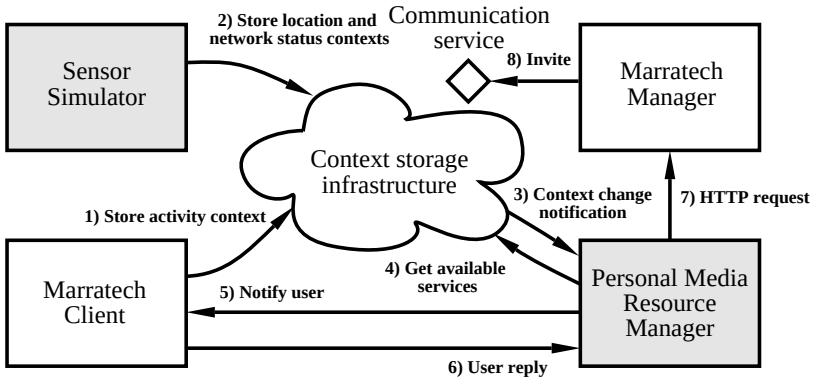

Figure 3. The interaction between the components.

is used to implement the $isNeeded$ function. When the $notify$ function is called, a dialogue opens in the Marratech client, letting the user choose whether to accept the new media resource or not. The $isAvailable$ function returns true for media resources which are nearby, here defined as located in the same room as the user, and which are not busy. The Q and P functions return the quality and privacy values for each available media resource. In this prototype these values were subjectively set by the authors.

5.2 Interaction between components

Figure 3 shows the interaction between the components in the implementation when the prototype is running and a context change triggers a need for a new media resource. When the prototype is started, Marratech starts sending information about active media resource types to the context storage infrastructure (1). The sensor simulator generates a context to significantly change the media resource need in (2). When the Personal Media Resource Manager is notified about such a context change (3) it tries to find an alternative resource from the context storage infrastructure by running MRSA (4). Once a new resource has been selected the personal manager calls the $notify$ function to ask the user to confirm the new resource (5). If the switch is approved by the user the personal manager initiates an invitation of the new resource to the session (6) by sending an HTTP request to the Marratech Manager (7) which in turn invites the resource (8).

5.3 Scenarios

The prototype was successfully tested on two different scenarios. The first scenario shows how an alternative communication service can be utilized if the currently used network connection can not provide enough quality. The second scenario demonstrates how the prototype can help the user find and select a more suitable communication service in order to better protect the user's privacy.

The test setup consisted of a desktop computer running a Marratech client located in the user's office, a mobile-phone carried by the user, and an e-meeting client which was located in a conference room. For media resources the desktop computer used a microphone and speakers and as the sound was considered to have good quality, the quality index was set to 6. The privacy index for this tool was set to 4 since the sound could be heard by others in the same room. The sound in the mobile-phone was considered to have lower quality, thus the quality was set to 3, but since the sound could only be heard by the user the privacy index was set to 6. Similarly to the stationary computer, the e-meeting client featured a microphone and speakers. However, since the microphone offered better quality as

it featured echo supression the quality index was set to 8. On the other hand, the privacy index was considered lower than in the desktop computer case. This was because other people were present in the conference room, which was not the case with the user's office. Hence, the privacy index for the e-meeting client was set to 2. As previously mentioned, these indices were subjectively set by the authors.

5.3.1 Scenario 1: Quality degradation

In the first scenario, the user is connected to an e-meeting using the stationary computer when the audio quality suddenly drops because of bad network performance. When this context change is perceived by the Personal Media Resource Manager, it causes the personal manager to run the algorithm in order to find new media resources and the quality index for the Marratech client's audio sink and audio source is recalculated to zero. A mobile-phone is found which satisfies the desired quality and privacy indices. The new media resources are approved by the user, the phone is invited to the session, and the user is able to communicate again.

5.3.2 Scenario 2: Privacy considerations

In the second scenario, the user is leaving the user's office to temporarily visit a public conference room when another user wants to talk to the user. Consequently, as the user is away from the office, it is not possible to communicate using the Marratech client run on the desktop computer located in the office. To allow the users to talk to each other, the Personal Media Resource Manager starts looking for an alternative communication tool. An available e-meeting client and a mobile-phone are found. However, since the user has P^{MIN} set to 4 the MRSA selects the mobile-phone even though the e-meeting client provides better quality.

6 Discussion

This paper considers the problem of how to automatically choose the most beneficial media resources for the user. The paper proposes to decompose the problem into manageable parts which can be solved individually. The parts which are identified include finding available media resources, creating an abstraction level for comparing the resources, and providing the necessary information needed to select appropriate media resources. Based on the information provided by these parts, the proposed algorithm, MRSA, performs media resource selection.

This paper proposes that only media resources which are nearby, not busy, and which the user has permission to access are considered as available. In the implementation, resources which reside in the same room as the user are considered to be nearby and can be discovered using the context storage infrastructure. However, depending on the size of the room there may still be occasions when a user does not benefit from using two available media resources together, e.g. when a microphone is too far away from the screen that the user is currently using. Other users' privacy preferences have not been taken into account when determining if a nearby resource is available or not.

An abstraction level for comparing media resources is proposed by classifying each resource with quality, privacy, and cost indices. This solution provides a useful abstraction from both users and developers. For users it supplies a coarse-grained control interface, and for developers the issues of how to value media resources and how to choose between them are separated. What information is needed to form the indices of different media resources is not covered in this paper.

In order to provide necessary information needed to select appropriate media resources, context-awareness was used. The contexts used in the proof-of-concept implementation when selecting a media resource is the user's location, the user's communication needs, the available media resources, and the user's privacy and quality preferences. Since users' location is used for access control it is important that the location can be verified. However, access control and other security issues have not been considered in this paper.

In the proof-of-concept, the algorithm is used to select a new media resource every time the user changes location and when a change in network performance occurs. This works well when only a limited number of contexts can trigger the algorithm. However, in a more advanced system, which uses more contexts, it might be required to execute the algorithm in intervals instead to avoid constantly triggering it.

The implementation and the two experiments performed in this paper were done to provide insight in the functionality of the algorithm. Although limited, these experiments showed that it was possible to integrate media resource selection into Marratech, the e-meeting software used in this paper. As a whole, the paper provides useful knowledge and deeper understanding of the issues involved in developing a system which automatically chooses the most beneficial media resources for the user.

6.1 Future work

Although a working prototype was developed there are many issues that remain to be studied. Only brief tests have been performed and the authors would like to conduct user studies to further explore how to increase the benefit of automatic media resource selection. User studies could also give valuable input on how to balance autonomy with user control.

In order to increase the benefit of a user study, a calculation model for privacy-, quality-, and cost-indices and their prioritization should be developed. When developing this model it is also important to study which contexts should be considered when determining the privacy-, quality-, and cost-indices. It is also important to evaluate this calculation model with users in order to determine its comprehensibility.

In any system which deals with personal information, it is important to ensure users' privacy and provide appropriate security. These issues are not considered in this paper and needs to be further researched. This means ensuring that it is possible to verify a user's location through the deployed positioning system, as well as attending to appropriate authentication and access control to the system in a ubiquitous environment.

Finally, the authors would also like to develop and utilize a media gateway which supports conversion of media between a number of different formats. With better conversion support it becomes possible to utilize a wider range of media resources together in the system. The idea is to make conversion between different media resource format in an automatic fashion in order to further relieve the users.

6.2 Acknowledgments

This work was funded by the Centre for Distance spanning Health-care (CDH), the Centre for Distance spanning Technology (CDT), the PhD Polis project, and the C4 project which is supported by Objective 1 Norra Norrland - EU structural fund programme for Norra Norrland.

References

[1] Anind K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5(1):4–7, 2001.

[2] Anind K. Dey, Daniel Salber, and Gregory D.Abowd. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Anchor article of a special issue on context-aware computing in the Human-Computer Interaction (HCI) Journal*, 16:97–166, 2001.

[3] Jason I. Hong and James A. Landay. An architecture for privacy-sensitive ubiquitous computing. In *MobiSYS '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*, pages 177–189, New York, NY, USA, 2004. ACM Press.

[4] Marc Langheinrich. Privacy by design - principles of privacy-aware ubiquitous systems. In *UbiComp '01: Proceedings of the 3rd International Conference on Ubiquitous Computing*, pages 273–291, London, UK, 2001. Springer-Verlag.

[5] Petros Maniatis, Mema Roussopoulos, Edward Swierk, Kevin Lai, Guido Appenzeller, Xinhua Zhao, and Mary Baker. The mobile people architecture. *ACM Mobile Computing and Communications Review*, 3:36–42, July 1999.

[6] Marratech. http://www.marratech.com, June 2005.

[7] Roland Parviainen and Peter Parnes. The mim web gateway to ip multicast e-meetings. In *Proceedings of SPIE Conference on Multimedia Computing and Networking 2004*, January 2004.

[8] B.N. Schilit, D.M. Hilbert, and J. Trevor. Context-aware communication. *IEEE Wireless Communications*, 9(5):46–54, October 2002.

[9] William Noah Schilit. *A system architecture for context-aware mobile computing*. PhD thesis, Columbia University, 1995.

[10] H. Schulzrinne and E. Wedlund. Application layer mobility using sip. *ACM Mobile Computing and Communications Review*, 4:47–57, July 2000.

[11] SIP: Session initiation protocol, rfc 2543. http://www.faqs.org/rfcs/rfc2543.html, 1999.

[12] João Pedro Sousa and David Garlan. Aura: an architectural framework for user mobility in ubiquitous computing environments. In *WICAS3: Proceedings of the IFIP 17th World Computer Congress - TC2 Stream / 3rd IEEE/IFIP Conference on Software Architecture*, pages 29–43, Deventer, The Netherlands, The Netherlands, 2002. Kluwer, B.V.

[13] H.J. Wang, B. Rama, C. Chuah, R. Biswas, R. Gummadi, B. Hohlt, X. Hong, E. Kiciman, Z. Mao, J.S. Shih, L. Subramanian, B.Y. Zhao, A.D. Joseph, and R.H. Katz. ICEBERG: An internet-core network architecture for integrated communications. *IEEE Personal Communications*, 7:10–19, Aug 2000. Special Issue on IP-based Mobile Telecommunication Networks.

[14] Zhenyu Wang and David Garlan. Task-driven computing. Technical Report CMU-CS-00-154, School of Computer Science, Carnegie Mellon University, May 2000.

[15] Roy Want, Andy Hopper, Veronica Falcão, and Jonathan Gibbons. The active badge location system. *ACM Transactions of Information Systems*, 10(1):91–102, 1992.