In MobiSys 2004 Workshop on Context Awareness, Boston,MA, USA, 6 June 2004.

Context Provisioning and SIP Events

Dana Pavel, Dirk Trossen
Nokia Research
5 Wayside Road, Burlington, MA 01803, USA

Abstract-- There is a general consensus that future services in mobile networks will be user-tailored and adaptive to the user's needs. Since context awareness can provide the required means for creating such services, it has become an important topic in research and also in industry. Apart from aspects on application level, such as adaptation concepts but also reasoning, learning and modeling issues, the provisioning of context information locally but also on an Internet scale is an important issue in context awareness. In this, the event delivery framework provided by the Session Initiation Protocol (SIP) nowadays provides means for event delivery throughout the Internet. It therefore seems to be a promising candidate for a standard-based solution for this part of a context-aware infrastructure. In this paper, we will present issues around context provisioning in general. Based on this, we will outline what is missing in the current SIP event framework with respect to those issues. The discussion of these issues can be used as a basis for future efforts to evolve SIP events into a context provisioning framework.

Index Terms—context provisioning, SIP events, semantics

I. INTRODUCTION

Ubiquitous computing is seen as the third wave in computing, when most of the things around us will incorporate computers, but also when the technology will move to the background and adapt to people [1]. One of the main characteristics of ubiquitous computing is the change from a machine-centric to a user-centric paradigm. In this new paradigm, smart environments should adapt to people rather than the other way around, as it is mostly prevalent nowadays. Adapting to user's needs involves making decisions or suggestions that satisfy the user. As it happens in real world, the better informed a person the better suggestions (or decisions) he can make. Therefore, in order for a machine to make suggestions or decisions that meet user's needs, it has to have enough information about the user and everything related to him that could influence the reasoning process.

An application that has the ability to collect information about user's current context and exploit this information for adapting the application, communication medium, or user interface, is called *context-aware application* [2]. Context includes any information that can be used to characterize the situation of an entity, where an entity can be a person, place, or an object [3]. Given this definition, the amount of information that can be included in context is very vast. Nowadays, for most of the applications, the usual context parameters are location and time. More sophisticated context information, however, can contain, for example, information about people and resources around, environmental data (e.g., weather, light, sound), affective information about users [4], and much more. In this, certain information can be *interpreted* or *aggregated* [3], which then might serve again as input for other context information.

An important aspect in realizing context-aware applications is how to provision context information to the application logic. In this, we have to keep in mind that context information does not necessarily reside on a single device. Instead, context information can be widely dispersed throughout the Internet. For instance, calendar information of relevant persons for my context will reside on a corporate server. Weather information can easily be retrieved from different sources in the Internet or from a sensor network. In addition, the actual application logic that uses the provisioned context information for performing a particular context-aware operation can be distributed as well, in particular when we consider aggregation of context information. Hence, it is desirable to have a solution for context provisioning that would integrate into the Internet architecture in order to facilitate wide deployment and therefore interoperability between applications and services.

This paper will present requirements for context provisioning that will set the stage for more specific considerations of a context provisioning solution that is based on the Session Initiation Protocol (SIP) [5] and its event delivery extensions. SIP has been standardized by the Internet Engineering Task Force (IETF) as a fundamental piece of the Internet architecture, hence it bears the hope of being a natural candidate for context provisioning throughout the Internet. Beyond an introduction to SIP and a discussion of why to consider SIP events for context provisioning, we will particularly address issues that still need deeper consideration in future work before SIP events can indeed be used as an Internet-wide context provisioning solution. Within our research on context awareness, we are currently in the process of developing solutions for some of the presented issues. Due to the work in progress character of this work however, this paper will focus on the issues as such rather than presenting dedicated solutions for them.

The remainder of the paper is as follows. Section II gives a brief overview of the underlying SIP technology, including the SIP event framework that is used in this paper. We will outline the problem space in Section III, before presenting an analysis of the SIP event framework with respect to the outlined problem space in Section IV. We will conclude the paper in Section V and give an outlook of our future work.

{dana.pavel,dirk.trossen}@nokia.com

II. A PRIMER TO SIP

Before we tackle the problem of using SIP events for context provisioning, we considered that a brief introduction to SIP would be useful. This section covers the main architectural principles as well as SIP events as an extension to the core SIP standard.

A. General SIP Architecture

Multimedia communication in the Internet nowadays is mostly based on the *session model*. This model defines a communication relation between two or more endpoints in the Internet as a *session*, described by its specific semantic, e.g., resources and protocols to be used for the actual data transfer. The session-specific semantic information is distributed among the session members prior to starting the data transfer in order to setup a common initial state of the session. After exchanging this information, the actual data transfer can be started. With this approach, session initiation and data transfer protocols can be separated from each other, to be developed independently.

This session model described above is the foundation for the Session Initiation Protocol [5], which has been standardized by the IETF. The architecture to realize the functionality of SIP is illustrated in Figure 1.

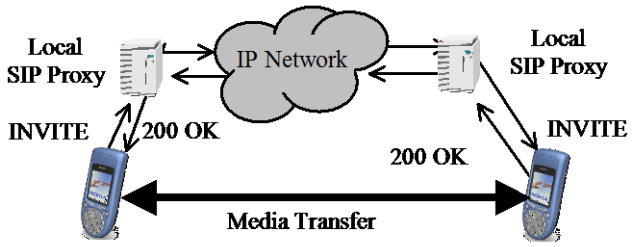

Fig. 1. SIP Architecture

Each endpoint registers with its local SIP proxy before engaging in any session relation. This registration ensures that invited endpoints can be located by SIP. In this, the term *local* simply indicates a logical relationship between endpoint and SIP proxy and does not imply any proximity on network or geographical level. In other words, a local SIP proxy might be far apart from an endpoint with respect to network, i.e., IP hops, or physical distance.

To initiate a session, an endpoint sends an invitation request (INVITE in Figure 1) to its local SIP proxy server, which forwards the request to the SIP proxy backbone infrastructure. This infrastructure is very similar to the Domain Naming Service (DNS), using functionality like registrars and location servers for locating endpoints in the Internet. The invitation request carries the description of the session, i.e., its semantic, as payload information. After the INVITE message eventually reaches the corresponding endpoint, an acknowledgement is sent back to the sender, containing a return code, e.g., *200 OK*, similar to HTTP requests. Upon reception of the return code, the two endpoints will engage in the actual media transfer.

Although not shown in Figure 1, SIP is not restricted to point-to-point scenarios. Invitations can be sent to a group of endpoints, endpoints can be re-invited upon changes of the session semantic, and invitation messages can also be re-directed.

B. SIP Events

The notion of *SIP events* extends the usage of SIP towards an event delivery framework in the Internet.

As defined in [6], SIP events generally represent state information for a particular resource that is associated to the particular SIP event. The specific semantic of SIP events is not specified in [6]. Instead, SIP events with specific semantics are supposed to be defined in separate standardization documents, following the template description that is given in [6], specifying for instance behavior of the network entities, the format of the state information and rate limitations for notifications on state changes. An example for such specific event description is the *presence event*, as defined in [7], describing the current presence state of a user in the Internet. Other proprietary examples are described, e.g., in [8] for location events. Such enforcement of specifying each and every possible event within a standardization body, in this case the Internet Engineering Task Force, places a heavy burden when it comes to applicability of SIP events in application-specific usage scenarios, in particular for context provisioning as we will discuss in Section III.

For realizing the actual event delivery functionality, [6] introduces two new SIP methods, namely SUBSCRIBE and NOTIFY. A subscriber sends the former message for initial subscription to an event and receives the latter for the initial notification and all subsequent ones that are related to this subscription. For that, the SIP infrastructure is used to route the subscription and notification requests from the subscribers to so-called *SIP event servers*, hosting the state information of the particular event. It is also possible to create so-called *one-shot subscriptions*, implementing a fetch operation. With this, the SIP event server merely sends back the current state of the event, without creating a subscription to future changes in this state.

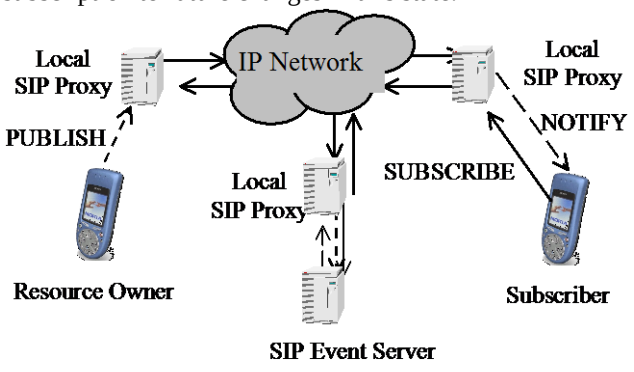

Fig. 2. SIP Event Architecture

State information can be changed at the SIP event server in many ways, including proprietary and non-SIP methods. The method in [9] envisions the usage of a new SIP method, called PUBLISH, that allows SIP-compliant devices to publish event state information to the SIP event server, subject to their authorization to do so.

Figure 2 shows the overall architecture for SIP events, depicting a mobile subscriber and resource owner with the SIP event hosted at a fixed SIP event server. It can be seen that the devices apply the SIP infrastructure, i.e., SIP proxies, to have the requests routed appropriately.
In conclusion, the SIP event framework in [6] defines an Internet-wide event delivery mechanism, in which the actual event semantics are defined separately.

III. Context Provisioning and SIP Events: The Problem Space

In the following, we will outline requirements for context provisioning in the Internet and will further discuss why we believe that SIP events could play a role in providing a solution for such problem.

A. Requirements for Context Provisioning

The requirements for a solution that provides context information throughout the Internet are heavily influenced by the following pre-requisites.
We believe that the sheer mass of possible application-specific context information cannot be captured by defining appropriate semantics within any standardization body. In this, highly aggregated and even (through some reasoning) derived context information is particularly hard to capture in some kind of standardization-based schema. As a solution for this dilemma, we believe that ontologies representing common knowledge bases for context will have to happen. These common knowledge bases can then be used by applications that will simply point at the appropriate semantic descriptions that are used for the particular case. Further, we believe that there will exist different means of transport in order to provide context information (e.g., through web services). Hence, a synergy is desirable re-using ontologies for *any* means of context provisioning.
These fundamental pre-requisites, together with widely accepted criteria for context provisioning solutions [2][3], lead to the following requirements:

1. *Modeling of Information*: It is required to define an information representation model that properly reflects the aggregating and deriving nature of context information. Such a model should enable reasoning capabilities and proper provisioning in the same hierarchical manner, allow for easy extensions and even support application- or category-specific semantics, such as for certain industries (e.g., health care).
2. *Support of Aggregation*: A provisioning solution must support the concept of aggregating a multiplicity of context sources on transport level in order to determine a specific piece of context information. Such context aggregation, also reflected in the underlying model, is likely to be application or service-specific.
3. *Mode of provisioning*: Besides the simple fetching of information, a solution should support a subscription model, which allows for provisioning of the particular context information upon changes in the information. This subscription would eliminate constant polling for the information.
4. *Security & Privacy*: Context information is usually of highly private nature, such as a person's location or even affective state. Hence, measures have to be in place that allow for a secure transfer of the information. Further, the information access has to be controlled properly by allowing for the definition of proper access policies. In [11], an extensive overview can be found on possible threats for location information. This threat analysis can easily be mapped onto general context information.
5. *Support of Legacy Technologies*: Context sources that are provided through legacy technologies, such as deployed sensor networks, must be supported. Hence, a solution for context provisioning shall enable to bridge or unify such different technologies into an Internet-wide technique.
6. *Deployment:* In order to be deployed in the Internet, a solution for context provisioning must rely on technologies that either are or will be available throughout the Internet in order to ensure deployment but also interoperability between different sources. Also important is the scalability of such solution in order to support a large number of sources, which could possibly be dispersed throughout the Internet. Further, a solution should also work in rather localized environments, such as home or enterprise.
7. *Discovery of Information*: Available context sources need to be properly discovered throughout the Internet. In this, the changed availability of context sources should be supported as well. Further, discovery requests as such need to take into account context information, such as location, time, but also user's interest or affective state.

B. Why Considering SIP Events?

If we consider the above outlined requirements for context provisioning, it seems that the SIP event framework, as presented in Section II.B, is a promising candidate for context provisioning in the Internet.
The technology allows for *event-based provisioning* of context information, even including simple fetch of information by defining a duration of the subscription of zero seconds (see [6]). Context information can be *aggregated* in a hierarchical manner by having SIP event servers subscribe to other SIP event servers for such aggregation, i.e., building entire hierarchies of context providers. Further, the usage of SIP URIs allows for *higher level addressing schemes* than pure IP addresses, i.e., enabling application-level addressing of context providers. But most important, SIP events rely on a technology platform, namely the SIP infrastructure, that will be deployed in future mobile devices due to the adoption in the next generation of mobile infrastructure, such as defined in 3GPP (3rd generation partnership project) standards. In

addition, SIP will also be the foundation for session initiation and presence support in desktop and server platforms. Hence, such wide adoption makes a rather ubiquitous availability of SIP events very likely and therefore drives our consideration of SIP events as a possible solution for context provisioning in the Internet.

IV. Issues Still to be Addressed

Although we argued in the previous section that it is worthwhile to consider SIP events as a mechanism for context provisioning in the Internet, there are still remaining issues to be addressed when it comes to actually using SIP events for context-aware applications. These issues relate to semantic support, privacy and access control, and access authorization for dedicated subscriptions. Currently, solutions for some of the presented issues are developed within our research on context awareness. Due to the work in progress character of these solutions, we will merely outline these issues in the following section rather than presenting dedicated solutions.

A. Semantic Support

The current process of creating SIP event packages (see Section II.B) requires the proper definition of the event package and the standardization of the semantics of the state information and processing of the information within the IETF. In other words, each and every SIP event package requires running through the process of being accepted by the appropriate IETF working groups. This means that if we want to ensure interoperability of provided context information that is based on SIP event packages, each and every piece of context information to be provided through SIP events is subject to the same (very long) standardization process.

As we outlined above, we do not believe that such process is sustainable with respect to properly capturing the sheer mass of application-specific context information possibly created within a context-aware environment. Hence, the current SIP event framework would need extensions to allow for creating event packages dynamically and dependent on the particular application's need. Hence, together with the reasoning in the requirement section III.A, we propose

- to use SIP events as a mere provisioning transport from context provider to context consumer,
- to use standardized means for context descriptions, carried within SIP event messages,
- to establish common knowledge bases for context descriptions that the application could point at within their subscriptions,
- to enable dynamically binding context information to SIP events in order to a) allow for application-specific context and b) to remove the standardization bottleneck, and
- to re-use context descriptions for other means of context provisioning, such as through Web Services.

With this in mind, the following methods for SIP events would need revision with respect to supporting such dynamic (application-specific) semantics:

- SUBSCRIBE [6]: currently the subscriptions are based on the semantics agreed during standardization. A solution is required that would allow for subscribing to SIP events, in which the subscription as such carries or points to the semantics of the particular subscription. The latter refers to the support of ontologies for semantic descriptions of context information. For that, a mechanism is required that allows for dynamically (i.e., at the point of subscription) binding the particular subscription to a particular semantic.
- NOTIFY [6]: If we see the notifications as an instantiated set of values for the particular semantic of the subscription, it is required to appropriately link the provided values in the notification to the semantics of the subscription.
- OPTIONS [6]: The OPTIONS methods allows for determining capabilities of SIP entities before engaging in any kind of dialog. Such query for the SIP user agent's capabilities needs to be extended with respect to properly signaling the support of such semantic-rich and dynamic event packages as described above.

It is important to note that any solution for abovementioned issues should properly integrate into the existing SIP event framework, i.e., no major changes to the existing SIP event infrastructure should be enforced by a proposed solution.

The above proposal is substantially different from approaches such as presented in [17] that simply enrich presence notifications with context information. Our proposal calls for the development of an independent context description language, which can be re-used by any means of context provisioning, and the appropriate extensions to SIP events in order to support such context language. Further, the support for context ontologies and therefore semantic knowledge bases is key in our approach.

B. Privacy and Access Control

Even though written with respect to location information, [11] gives an excellent overview of possible threats to context information in general when it comes to preserving the user's privacy. The threats described in [11] are meant to lay ground for the IETF's work on location but also on presence privacy.

The findings in [11] let us conclude that a system is desired that allows for control of access, distribution, and retention of context information in general, in addition to protection from denial of service, spoofing, and eavesdropping attacks on the system.

Many efforts are currently ongoing to establish systems that would prevent threats as described in [11]. However, these systems lack of a certain unification character when it comes to supporting arbitrary context information, i.e., most of the systems, e.g., [12], tackle well-defined context information, such as presence or location. What is required is a unified privacy framework that will eventually evolve

towards a rule-based system as envisioned in [13]. Such kind of system has to be properly integrated in a SIP event system to the extent that subscriptions are only granted based on proper access policies. For instance, as for the proposed solution in [12], this can be achieved by having the SIP event server subscribe to the access policies of particular context information. In that case, any future change of access policies will be propagated to the SIP event server, to be used in future or existing subscriptions.

Other drawbacks of current approaches for privacy and access control in SIP event environments are the lack of conditional access policies, such as "allow access to item X, if and only if the requested confidence is less than 95%", the lack of application-specific policies that are dependent on application-specific semantics of the context information, and the problem of finding appropriate access policy servers. The latter refers to the problem as to how a particular SIP event server, hosting a particular piece of context information, would find the appropriate policy server that hosts the access policy for this particular piece of context information.

C. *Support for "Dedicated Access"*

In order to illustrate what we mean with "dedicated access", consider the following example. A user desires to obtain location-based data from a service provider, such as map information indicating the current location of the user. For that, the service provider needs one-time access to the user's location. Let us further assume that the transaction between user and service provider takes place via HTTP (e.g., through the web site of the service provider). What is needed in this case is a "dedicated access" to the particular location information by the subscriber, granted by the user as an owner of the location resource information.

The approach in [12] describes the current IETF approach in presence to tackle the access authorization problem, i.e., whether or not to grant an incoming subscription. However, this approach demands that the appropriate policy exists at the time of subscription. It further assumes that the URI of the potential subscriber is known to the policy maker (usually the resource owner) at the time of the policy definition. With such solutions, the following problems occur when it comes to such "dedicated access":

- *Timing*: The timing of access policy insertion (assuming that there did not exist a policy for the service provider beforehand) with the actual access is crucial, since one would like to avoid situations in which subscriptions are rejected because the proper policy has not yet been inserted at the policy server.
- *Complexity*: The user in our example would need to insert the policy, signal the existence of the policy towards the service provider in order to start the subscription process, and remove the policy again after the (location) information has been retrieved by the service provider. Although temporal access policies could minimize the transactions, they have not yet found entry into the currently discussed solution [12]. In cases of mobile devices, on the user but also service provider side, such rather complex temporal transactions are not desired due to the delay of the overall transaction.
- *Identity*: In such "dedicated access" scenarios, the problem arises that certain service transactions between user and subscriber (the service provider in the example) do not allow for a determination of a proper (permanent) identity of the subscriber to be used in the policy insertion. To illustrate this problem, consider the following "visitor" scenario. A user is entering visitor promises, such as a museum or enterprise premises, and the SLP (Service Location Protocol [16]) discovery agent is obtained through methods such as DNS-SRV [15]. We further assume that the discovery agent allows for service selection based on user information such as current location (based on an indoor system), activity, currently used communication device, and so on. Hence, the discovery agent requires access to this information of the user. If we consider the use of SIP event subscriptions for this retrieval, a policy-based approach as proposed in [12] would have the problem to identify the proper identity to be placed in the policy for the access. Since the user simply obtained the discovery agent's IP address (which is probably even a private IP address), no valid identity can be placed in the policy, and therefore no valid policy could be defined in this case, prohibiting proper access to the context information.

Since such scenarios of using "surrounding" services are typical ones in ubiquitous environments [1][2], a solution is required that properly accommodates such cases. The approach in [14] provides such solution by creating the subscription data on the user's (i.e., resource owner's) side, signing it with the user's credentials, and handing the data to the subscriber. The subscriber in turn uses the signed data in its subscription, which can be verified by the SIP event server (i.e., if the signature is valid, the submission is granted). Such approach would not involve any policy transactions (and it would also solve other problems mentioned in IV.B). As a disadvantage, the approach requires extending the SIP event framework by allowing such "signed" subscriptions.

V. CONCLUSIONS AND FUTURE WORK

Context awareness is going to become a major trend in future wireless services as it promises adapting and tailoring services to user's needs and current situation. In this, the required information, i.e., the context, needs to be provisioned to the appropriate service entities, possibly throughout the Internet.

This paper presented requirements for context provisioning in general and considerations for using the SIP event framework for such provisioning specifically. Although SIP events seem to be a promising candidate for context provisioning in the Internet, the paper presented key issues

still to be addressed in order to use SIP events in context-aware environments.

In part, solutions for these issues are currently under development. This includes solutions currently investigated within our own research, in particular in the area of semantic support and dedicated access. Our future work will focus on concretizing these solutions and providing proof-of-concept prototypes, being embedded into a context provisioning solution within a distributed environment.

REFERENCES

[1] M. Weiser, "The Computer for the 21st Century", Scientific American (1991)

[2] B. N. Schilit, N.I. Adams and R. Want, "Context-aware computing applications", Proceedings of the 1st International Workshop on Mobile Computing Systems and Applications, pp. 85-90, Santa Cruz, CA, IEEE (1994)

[3] A. Dey, "Understanding and Using Context", Personal and Ubiquitous Computing, Vol. 5, No. 1 (2001)

[4] R. W. Picard, "Affective Computing", The MIT Press, Cambridge, MA (1997)

[5] M. Handley, H. Schulzrinne, E. Schooler, J. Rosenberg, "SIP: Session Initiation Protocol", The Internet Society, RFC 3261 (2002)

[6] A. Roach, "SIP-Specific Event Notification", The Internet Society, RFC 3265 (2002)

[7] J. Rosenberg et al., "SIP Extensions for Presence", Work In Progress, Internet Draft (2003)

[8] D. Trossen, "Providing a Location Service based on the SIP Presence Extensions", Proceedings of IEEE Wireless (2002)

[9] A. Niemi, "Session Initiation Protocol (SIP) Extension for Event State Publication", Work In Progress, Internet Draft (2003)

[10] T. Bray, J. Paoli, C. M. Sperberg-McQueen, "Extensible markup language (XML) 1.0 (second edition)", W3C Recommendation REC-xml-20001006, World Wide Web Consortium (2000)

[11] M. Danley et al., "Threat Analysis of the Geopriv Protocol", The Internet Society, RFC 3694 (2004)

[12] J. Rosenberg, "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)", Work In Progress, Internet Draft (2004)

[13] J. Cuellar, "Geopriv Requirements", The Internet Society, RFC 3693 (2004)

[14] D. Trossen, H. Schulzrinne, "On-Demand Access Authorization for SIP Event Subscriptions", Work In Progress, Internet Draft (2003)

[15] W. Zhao et al., "Remote Service Discovery in the Service Location Protocol via DNS SRV", Work In Progress, Internet Draft (2004)

[16] E. Guttman et al., "Service Location Protocol Version 2", The Internet Society, RFC 2165 (1999)

[17] Wei Li, "A Service Oriented SIP Infrastructure for Adaptive and Context-Aware Wireless Services", Proceedings of the 2nd International Conference on Mobile and Ubiquitous Multimedia, Norrköping, Sweden (2003)