Supporting Network Access and Service Location in Dynamic Environments

Dirk Kutscher
Technologiezentrum Informatik (TZI)
Universität Bremen
dku@tzi.org

Jörg Ott
Helsinki University of Technology
Networking Laboratory
jo@netlab.tkk.fi

Steffen Bartsch
Technologiezentrum Informatik (TZI)
Universität Bremen
sbartsch@tzi.org

April 23, 2007

Keywords: Mobility, Roaming, 4G, WLAN, Service Discovery

Abstract

The ubiquity of WLAN services, ranging from campus networks to commercial hotspots and community WLAN services, requires a new approach for automatically locating and using services. We present a service selection infrastructure independent of the current network attachment and location of users in a way that users can obtain information of a specific network (or any other service) without being required to be connected to a particular one. Information about networks and services are distributed in a way that allows for using this information offline, e.g., when still looking for appropriate network access.

1 Introduction

Wireless networks and services have emerged into a diverse landscape of competing, over-lapping offerings that users can select from depending on their device capabilities, intended applications, intended mobility patterns, and acceptable tariff schemes. City-wide WLAN networks and community-based WLANs are challenging commercial public 3G networks in terms of performance and costs for network access. WiMAX may further add to this variety soon.

The increasing diversity of network access alternatives enables mobile users to frequently roam between private, public, and corporate networks. Multi-interface mobile devices[1] have entered the consumer market and allow users to switch between 3G and WLAN networks.

Based on network connectivity through one or multiple currently selected access networks, a mobile user may have access to a set of network-based applications such as web-based information systems, VoIP infrastructure services, and network-provider-specific multimedia services. For example, in a university campus setting, the campus WLAN access network may provide access to information services about the network and building facilities as well as a local telephony service. For commercial networks, the network service provider may couple the network access to additional services, e.g., entertainment services such as video-on-demand. FON, a commercial WLAN community network operator, has recently announced to develop a telephony service based on the community-operated FON WLAN hotspot infrastructure.

[1] such as various Nokia N and E series phones or the Fujitsu Siemens LOOX T Series

Navigating within this expanding set of diverse network access and service alternatives has become increasingly challenging. For **network access**, service selection is typically performed *on-demand* (e.g., by scanning for available networks on the radio channel) or *in-advance* from the web while connected, e.g., by reverting to web-based "hot-spot locators". Figure 1 depicts the web-based hotspot locator application for the FON network. The *on-demand* network selection has the disadvantage that it is not very reliable, may be time-consuming, and, foremost, is only workable when the user is currently within the network's coverage. The *in-advance* network selection has the disadvantage that it requires an existing network connection to be useful, i.e., it cannot be used when the user is currently offline and still looking for network access—a scenario that is quite typical when considering mobile users looking for usable WLAN hotspots in a city. As depicted in figure 1 the distribution of hotspot location can be quite sparse, so that being connected may be the exception rather than the rule, especially for mobile users.

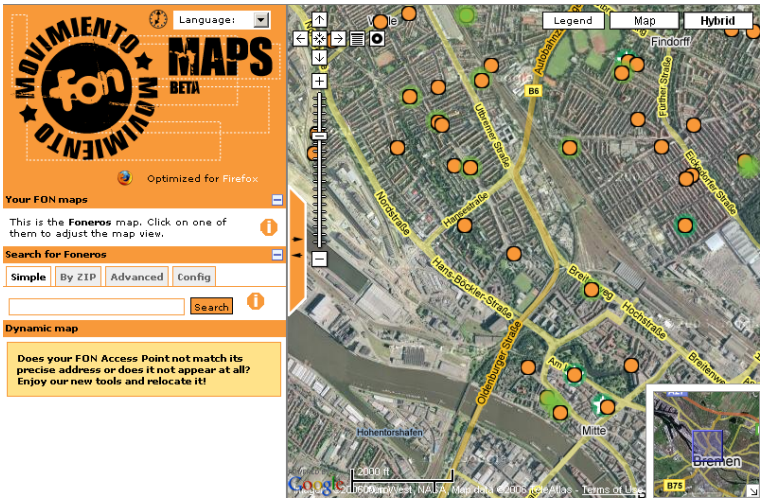

Figure 1: FON Hotspot Locator

For locating and selecting **applications**, there is an existing set of protocols for service location such as the Service Location Protocol (SLP, [GPVD99]) and UPnP [Cor00]. However, these are targeted at users who are already connected to local network and cannot be used in advance, i.e., in order to base a network selection decision on the availability of certain services. Other forms of service location/selection are web-based portals, e.g., the Lufthansa FlyNet portal, however these also require a connection to the respective network.

What is needed, is a service selection infrastructure that is independent of the current network attachment of users seeking service (and may also be independent of their present location) in a way that users can obtain information of a specific network (or any other service) without being required to be connected to a particular one. Information about networks and services must be distributed in a way that allows for using this information offline, e.g., when still looking for an appropriate network access. It also must deal with the dynamics of service offerings, e.g., when dealing with community networks, where new access points may be installed or go out of service frequently. In order to allow users to select services on the basis of proximity (e.g., locating the closest WLAN hotspot) the service selection should leverage location information (geographic positions and civic locations) of both service and user locations.

In this paper, we present the design and an implementation of *Network Service Maps* as a means to capture and convey encompassing network coverage and service descriptions to mobile users. We start by reviewing related work on service discovery specifically for wireless networks and services in section 2. In section 3, we introduce network service maps and their operation in detail and describe our implementation in section 4. Section 5 addresses two specific usage scenarios

based upon which we evaluate our approach through measurements and simulations. Section 6 concludes this paper with a brief review of our findings and hints at future work.

2 Related Work

Currently, the most common solution to finding WLAN network access are hotspot finders. These are applications for finding locations of WLAN services by address or geographic position, either web-based for online usage, or offline applications for mobile devices such as laptops and mobile phones. Web-based hotspot finders, such as *FON Maps*[2], usually feature a map view with available hotspots in addition to search and listing functions. Offline finders typically exhibit a different behavior: The *wiPod*[3] is an example of a simple off-line hotspot finder for *Apple iPod* MP3 players. The *iPass Offline Hotspot Finder*[4] for the Microsoft Windows XP platform has a more elaborate user interface including displaying pre-downloaded maps. While online hotspot finders are limited with respect to usability, e.g., they cannot be used when the user is offline and looking for service, offline finders usually lack an update mechanism, i.e., in dynamic environments they are likely to present outdated information. Implementations for both variants are typically proprietary, i.e., they only provide access to provider specific databases, thus presenting a very limited view of network resources. As for web-based online hotspot finders, interesting aspects may be found in community-based locators, such as *hotspotr*[5], which enables users to contribute and rate WLAN hotspots, focused on free WLAN Internet access in cafés.

To overcome the limited flexibility with respect to provider-independence, roaming WISPs, such as *iPass*[6], entertain roaming agreements with a large number of WISPs, so that users have a greater basis of ISPs and hotspots to choose from. The idea is that only one tool, provided by the roaming WISP, is needed for finding hotspots so that the whole connectivity establishment process in simplified. However, even with WLAN aggregation and roaming, it is currently not likely that a single, unified solution will be available that is applicable to all commercial hotspots.

Using wireless Internet does not only involve finding the location of services, but also requires configuration and authentication processes. While operating systems support automated access to WEP- and WPA-protected WLANs by storing pairs of ESSIDs and associated user credentials, WLAN hotspots do not make use of these but use open access. For accessing such WLANs, the configuration includes setting the appropriate SSID and performing web-based authentication, which is currently common with public WLAN hotspots. Web-based authentication (also known as UAM—Universal Access Method [ABS03]) relies on the concept of captive portals, i.e., users are requires to enter credentials into a web form, before Internet access is granted. Especially in mobile scenarios, when moving from one access point to another, this may be a frequent hassle to the user. Smart clients, such as the FON WiFi Connection Manager[7], can facilitate this procedure by automatically associating to the provider's access point when in reach. Credentials are sent to an authentication server after additional probing messages for verifying the hotspot's identity.

We have observed [OK05a] that many variants of UAM exists, which additionally complicates automated authentication. Providing automated hotspot association today requires a significant amount of probing and guessing on the client side. Some first proposals for simplifying these processes on the client side have been developed. E.g., *Devicescape*[8] offers connectivity management software for automatically associating with hotspots of a comparably large number of WISPs. It exploits the fact that access to global DNS services is unrestricted in most hotspot configuration, even without prior user authentication. The *Devicescape* approach uses the DNS to deliver provider-specific information (how to log-on) and user credentials (for the corresponding WISP)

[2] http://maps.fon.com
[3] http://anchorfree.com/wipod
[4] http://ipass.com/misc/offlinefinder.html
[5] http://hotspotr.com
[6] http://ipass.com
[7] http://www.fon.com/images/media/common/QIG_symbian.pdf
[8] http://www.devicescape.com

to mobile devices. *Devicescape* is planning to offer Internet access on a re-sale basis, enabling wireless Internet usage at any WISP's hotspot that *Devicescape* has agreements with. It should be noted that this solution only addresses the issue of facilitating the log-on process – users still have to locate hotspots themselves (or use available hotspots opportunistically). Also the approach is limited to those (larger) WISPs that are supported by *Devicescape*.

Of course, users may want to use other networks besides WLAN, e.g., 3G networks. Mobile operators extend their network coverage and performance by adding public WLANs as access networks for data and voice services using SIM card-based authentication similar to the cellular networks. This is known as *Generic Network Access (GAN)*, formally referred to as *Unlicensed Mobile Access (UMA)*. Client-based solutions such as *Birdstep's* smart client products[9] support multiple access network technologies, including 3G and WLAN. The client application offers to automatically choose the best option at a time, according to configured preferences, and connect to available services. Additionally, the smart client may establish higher-level services, such as VPN connections, according to user configuration. Still, this application only takes a defined set of sources into account, depending on the services offered by a specific service provider that has distributed the client software.

In research, most approaches for finding wireless networks are motivated by handover aspects. [DFHX05] defines requirements on a handover information service. These are based, in parts, on [80205], which describes link-layer information necessary for handover independent from its transportation media. These efforts focus on handover scenarios, though, limiting its universal applicability as IS. For adding information about roaming providers of a WLAN hotspot, [LM04] even proposes transporting this information in the IEEE 802.11 ESSID or a to-be-standardized 802.11 Information Element. Here, flexibility is limited by focusing on WLAN technology.

3 Network Service Maps

In [KO06b] and [KO06a], we have described the *Network Service Maps* approach and its application to distributing network service information for campus WLAN installations.[10] The *Network Service Maps* approach can be characterized as provider- and network-independent, representing an extensible information service that is built on the notion that receivers obtain service descriptions from arbitrary sources over different networks and compose *individual service maps* based on filtering with respect to current location, sought-after service, personal preferences, etc.

It differs from existing network information systems in its generality and network- and topology-independence, and it differs from existing service-location approaches in its applicability to wide-area, location-based service description distribution. The information service is based on a general service description information framework that provides different transport mechanisms for supporting heterogeneous network environments and on a data model for service description that enables receivers and transceivers to flexibly (re-) composing received service information with respect to different criteria.

We have defined a model for describing network services in a way suitable for distribution in heterogeneous, multi-operator networks, focusing on the following requirements: 1) use of a generalized network service description language that is not limited or tied to specific link-layer technologies and architectures; 2) scalability at both infrastructure and receiver side at least in terms of number of mobile users as well as services; 3) support for a wide range of service descriptions not limited to plain Internet connectivity; and 4) extensibility (for describing new services and configuration parameters).

The transport and data container concepts of Network Service Maps are based on the Internet Media Guides (IMG) [NWL+06] framework – which has originally been developed for delivering guides describing the availability of multimedia contents and programming from many originators to an arbitrary number of receivers. IMGs offer transport operations for one-to-many broadcast,

[9] http://www.birdstep.com

[10] http://www.service-maps.net/

request/response and subscribe/notify delivery. As shown in figure 6, these transports enable delivery of Service Maps directly from the original source to the user as well as by way of intermediate transceivers.

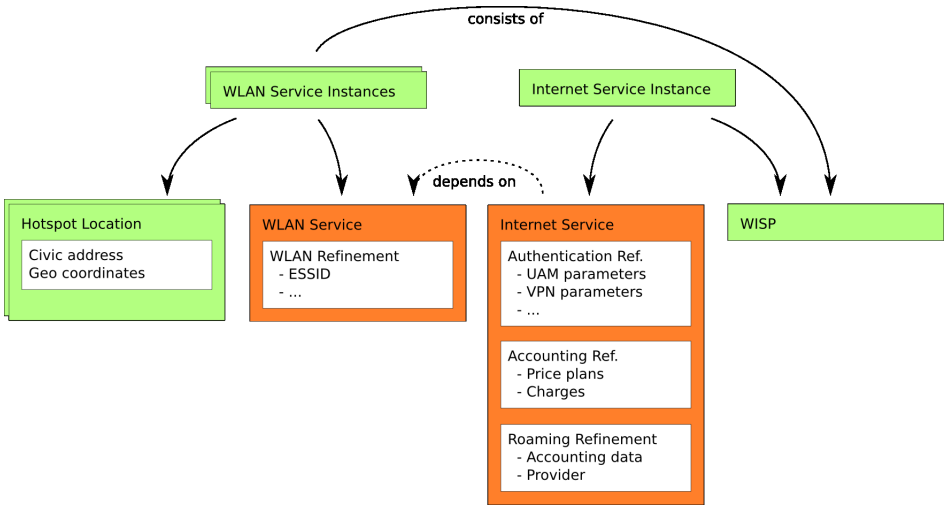

Figure 2: Network Service Maps Data Model

In the following, we will present different aspects of the Service Map architectures: the Network Service Maps data model (section 3.1), filtering and aggregation of service maps (section 3.2), Service Map URNs (section 3.3), caching (section 3.4), broadcast distribution (section 3.5), bootstrapping (section 3.6), contributing (section 3.7), and security (section 3.8).

3.1 Network Service Map Data Model

The Service Maps data model for service descriptions is a general application-independent format, which has been described in [KO06a]. It provides the notion of *refinements*, detailed descriptions of certain, typically application-specific aspects of a service description. Refinements can be provided as external fragments of a service description and can be made available on-demand or in a separate multicast distribution channel.

We have developed a set of Service Map refinements for describing specific aspects of network services, particularly for WLAN hotspot services. As depicted in the conceptual diagram in figure 2 and the sample Service Map in figure 3, Network Service Maps are structured into two layers of services. On the lower layer, unprotected access networks are provided, represented here by the service named `#wlan-local-access-fon`. For direct identification of these services by users or software, the tags "wlan" and "access network" are used. These services are bound to locations through instances, as shown to the bottom of the service map.

Based on fundamental services such as *local network access*, additional services may be available such as *Internet access*. This relationship is also expressed in the service map as a *service dependency*, e.g., as depicted for service `#wlan-internet-fon`. For this kind of hotspot-based Internet access, we are using the tags "public," "hotspot,", and "Internet access.". In contrast to access network services, higher level services are not associated to any location, but are available everywhere, given that the dependencies are met. For the higher layer services, details of authentication methods are needed. Also, as usually charges apply for their usage, additional tariff information is included, too. Moreover, as many providers offer roaming agreements, valid roaming offers may be described additionally.

The WLAN refinement describes an IEEE 802.11a/b/g wireless access point. Thus, it specifies typical parameters, such as ESSID and authentication methods. In most WLAN hotspots and in many campus WLAN installations, authentication is done above the WLAN layer. To facilitate

```
<service-map ...>
  <location id="main-station-wlan"><!-- ... --></location>
  <provider id="fon" name="FON"/>

  <service id="wlan-local-access-fon">
    <tag>access network</tag>
    <refinement
        xmlns:wlan="urn:uni-bremen:params:xml:ns:service-maps:wlan">
      <wlan:wlan>
        <wlan:essid>FON_AP</wlan:essid>
      </wlan:wlan>
    </refinement>
  </service>

  <service id="wlan-internet-fon">
    <tag>internet access</tag><tag>public</tag><tag>hotspot</tag>
    <dependencies type="all">
      <service-reference ref="wlan-local-access-tcom"/>
    </dependencies>
    <refinement
        xmlns:auth="urn:uni-bremen:params:xml:ns:service-maps:authentication">
      <auth:authentication
          xmlns:uam="urn:uni-bremen:params:xml:ns:service-maps:authentication:uam">
        <uam:uam>
          <uam:provider-id>Credentials-for-Fon</uam:provider-id>
          <uam:url>https://www.fon.com/login/gateway/processLogin</uam:url>
        </uam:uam>
      </auth:authentication>
    </refinement>
    <refinement
        xmlns:acc="urn:uni-bremen:params:xml:ns:service-maps:accounting">
      <acc:accounting><!-- FON tariff -->
        <acc:details-web-link>http://www.fon.com</acc:details-web-link>
        <acc:tariff>
          <acc:required-plan>Linus</acc:required-plan>
        </acc:tariff>
        <acc:tariff base="time">
          <acc:required-plan>Bill</acc:required-plan>
          <acc:required-plan>Alien</acc:required-plan>
          <acc:charge-unit unit="h">24</acc:charge-unit>
          <acc:cost-per-unit currency="EUR">3</acc:cost-per-unit>
        </acc:tariff>
      </acc:accounting>
    </refinement>
  </service>

  <instance>
    <location-reference ref="main-station-wlan"/>
    <service-reference ref="wlan-local-access-fon"/>
    <provider-reference ref="fon"/>
  </instance>
  <instance>
    <service-reference ref="wlan-internet-fon"/>
    <provider-reference ref="fon"/>
  </instance>
</service-map>
```

Figure 3: Service Map Example

the network association for clients in such networks, the authentication procedures are described in authentication refinements. The authentication refinement itself only serves as a container for listing available authentication methods, each in its own namespace. Provided are descriptions for the web browser-based *Universal Access Method* (UAM) and *Virtual Private Network* (VPN) [ABS03, OKK05]. UAM parameters consist of a provider ID for specifying the provider of the authentication web page, a protocol and the verification host to send credentials to. For describing the associated cost, we have developed a corresponding refinement. It is not meant to cover all possible price plans, but should handle most the current models reasonably well, as it was tested on a variety of WISPs.

3.2 Filtering and Aggregation

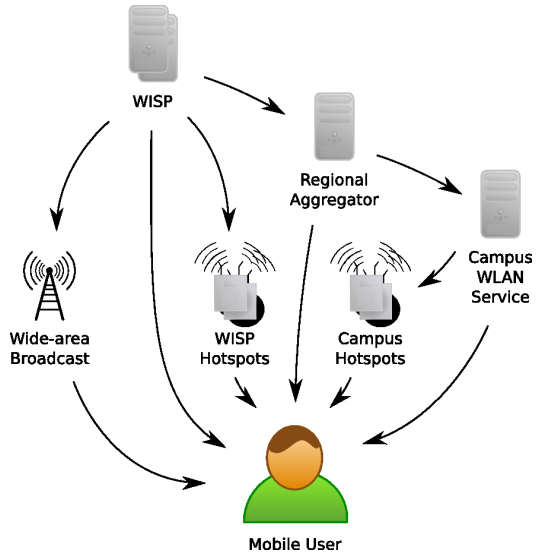

Figure 4: Service Maps Distribution Paths

Filtering and aggregation is a core concept in the Service Map architecture. It allows Service Map senders and transceivers to offer specific parts of original Service Maps as well as combinations thereof, which is important, e.g., when aggregating Service Maps from different providers. A common use case for filtering is to limit the distributed Service Map set to service descriptions for a certain geographic area. Additionally, users should be able to specifically query for needed service descriptions, formulating the range in form of filter expressions. Filter expression semantics are tightly bound to the Service Map data format, however they must be flexible enough to support application-specific queries.

Different types of filter expressions are supported. Tags are a universal means of describing data items in Service Maps and can also be used in filter expressions, using so-called *tag filters*. A tag filter provides the tags that have to be present for inclusion in the resulting Service Map. *Location filters* are a different filter type. Locations in Service Maps may be specified as coordinates in a geographic coordinate system. If coordinates are given as a filter expression, the filter only matches service instances that are reachable from the given location (taking the service coverage area into account). With an additional range parameter (see below), only service instance locations inside a circle with the specified radius are considered. Service Maps also provide civic location specifications. A *civic location filter* is a string that needs to be present in a location's civic address for the associated instances to be matched. Finally, filter expressions can be used to select refinements of a specific type. In Service Maps, refinement types are specified by the XML namespace URI. For filtering of Service Maps according to refinement internal information, *XPath expressions* may be

given. XPath expressions are evaluated on single data items, matching those that return non-empty node sets or a boolean *true* value.

3.3 Service Map URN

Identifying Service Maps uniquely is an important property of the architecture since Service Maps may be distributed over multiple different channels and receiving multiple copies at a receiver cannot be excluded. Service Maps provide a Service Map uniform resource name (URN) that is employed as an identification mechanism. Based on the notion of IMG URN schemes as suggested in [Gre06], globally unique identifiers are created using domain names and dates in addition to locally crafted names. Also, fragments and specific versions of Service Maps may be named. The namespace id as defined in [Moa97] of the Service Map URN is "svcmap". Examples are presented in the following:

```
urn:svcmap:example.org:20061028:campus-wlan#coord=53.10663,8.852487;range=100
urn:svcmap:example.org:20061028:campus-wlan?6453#refinement-2343
urn:svcmap:example.org:20061128:wlan#xpath=//tariff[@type='volume']
```

We have also defined rules for comparing URNs, including URNs with filter expressions, which semantically represent subsets of the result information set.

There are several scenarios where a resolution mechanism is needed for clients to find a location for retrieving Service Maps from. For Service Map URN resolution, the DNS-based Dynamic Delegation Discovery System (DDDS) is used. DDDS is described in [Mea02b, Mea02d, Mea02c, Mea02a]. In this approach, similar to the DDDS URI application, a domain name's authoritative DNS server offers rewriting rules for its Service Map URNs to available distribution operations and protocols. The final result is a URI for retrieving the Service Map.

3.4 Caching

Caching is a common architectural paradigm for increasing data access performance and it applied for Service Maps for two different purposes: for increasing scalability of infrastructure components with large numbers of mobile clients and for enabling clients to store regularly required service description for fast access. In Service Map infrastructure systems, requests from large numbers of clients for specific Service Maps may stress the original sending server. Therefore, intermediates may take load from strained senders by temporarily storing heavily requested items for clients that choose to use them as intermediate source.

Important issues in these infrastructure caches are to hold the most requested items in store and to preserve data consistency between source and cache. Access statistics are employed to approximate which items will be requested last from the current set when storage capacity constraints of the cache are reached.

For keeping cached items up-to-date, publish/subscribe transport mechanisms may be employed, letting the cache subscribe to changes at the sender. Then, any updates at the source are signaled to the cache which will consequently retrieve the changes. If no subscription service is available at the source, the cache has to regularly check for changes when clients request the item. On every occurring request, the cache will check if it may be directly served from the local store. If requests for filtered, non-local Service Maps are handled, the store may be checked for supersets of the filter expression, requiring only additional application of filters and preventing extra request at the source.

Caches in client applications may follow a similar approach. Still, in contrast to caches of infrastructure components, client-side caches not only reduce the source's load, but increase performance for the end user significantly. When browsing available services, caching may enable smooth interaction with the user interface as no additional requests for Service Maps have to be sent to sources. In broadcast environments, items may be repeated only at long intervals, increasing

waiting times if the receiver does not cache sent items. In many situations, Service Maps available at one point in time may be completely unavailable later, e.g., in the presence of changing or intermittent connectivity. Thus, the cache replacement algorithm has a higher impact in client components as cache misses may not only lead to decreased performance, but to unacceptable interactivity or even missing services.

Client cache replacement algorithms may take access statistics for each Service Map into account, with access statistics generated by results of specific searches, including types of services and location constraints. Additionally, the algorithm should consider the physical distance of the user's previous positions from the service scope of a specific Service Map entry.

3.5 Broadcast Distribution

Broadcast/multicast is one of the Service Maps distribution mechanisms (ANNOUNCE), enabling the distribution on unidirectional broadcast links, such as DVB-H links. Service Map distribution systems may provide multiple links and transport alternatives, broadcast being only one alternative. The key issue in broadcast distribution is the decision which items to broadcast and at which priority, also often referred to as *broadcast scheduling* [SRB97, BCPL06]. In simple scenarios, broadcast service operators may configure statically which data to broadcast from the local store. For other scenarios, such static configurations may not be feasible. Instead, dynamic scheduling may be employed, reacting on users' demand. Scheduling is then based on access statistics which need to be carefully evaluated as broadcast items will not generate as many requests as others due to client-side cache hits and request suppression. If no access statistics are available, as with broadcast senders using unidirectional media, representative statistics need to be acquired from infrastructure components in the same region.

3.6 Bootstrapping

Bootstrapping refers to the process of automatically finding Service Map resources by employing standardized receiver configurations and lookup procedures — an important feature for efficient and automated operation of Service Maps clients. When a Service Maps-enabled end device enters a previously unknown network environment, there may be active Service Map services. In order to detect and use these services descriptions, the device has to acquire basic information, *bootstrapping* information. As Service Maps support different transport services, the specific bootstrapping process depends on the network type. We have defined three different procedures, that have to be tried in order:

For broadcast/multicast enabled environments, the client should try to join a specific FLUTE [PLL+04] session on a standardized multicast address for receiving bootstrapping Service Maps. If the device does not support FLUTE for reasons such as limited computation power, a second bootstrapping scheme should be tried. The scheme is also multicast-based, but delivers Service Maps simply inside of single UDP datagrams. A simple text-based protocol, similar to HTTP, is employed here.

In unicast-only networks, we are relying on existing IP auto-configuration schemes and use local DNS servers for retrieving the local bootstrapping Service Map. The client device resolves a well-known local bootstrapping Service Map URN[11], as described in section 3.3, with the request being authoritatively handled by the local DNS server. By applying URN resolution, the client can obtain a, possibly local, URI for retrieving a bootstrapping Service Map.

Bootstrapping Service Maps provide information on available Service Maps and delivery or retrieval alternatives. This information may be represented in a service refinement of a Service Map. The bootstrapping refinement contains a descriptive name of the service description service

[11] E.g. `urn:svcmap:bootstrapping.local:200701:bootstrapping` which employs a commonly used, but non-standard `.local` top level domain. It is not listed with reserved top level domains in [EP99], but is used in Multicast DNS [Che06] for similar purposes.

and additional information such as details to available Service Maps, including a URI and a human-readable description. A Service Map provider may choose to give an alternative location in form of a URL, which might only be locally available. Finally, broadcast session configurations for receiving Service Maps can be provided.

3.7 Contributing

The previous subsections have addressed how service providers can disseminate service maps to users and how the latter can define their current area and depth of interest. While, in theory, this may provide the mobile users with current service maps of her surroundings, practice has shown that the purely provider-driven approach is insufficient for two major reasons:

1. The service providers or aggregators typically offer only static information about hotspot locations and services (derived, e.g., from some internal database) since they have no way of maintaining this data truly up to date. For example, when a newly established hotspot location is entered in the database there is usually no precise knowledge when this location will actually become operational. Similarly, temporary unavailability of hotspots is usually not reported. As a result, some part of the information in service provider hotspot databases will likely be incorrect.

2. There may be additional access opportunities not covered by any of the service providers which will thus be missing from the service maps. As a consequence, less access opportunities will be known to mobile users than actually exist.

One way to address these two shortcomings is to involve those mobile users who also benefit from encompassing and up-to-date information. As users move around, their mobile devices capture beacons from available WLANs (as is done by tools such as *network stumbler*[12]) and record the available information about the WLAN together with their respective location. In addition to information available from passively observing WLANs, mobile devices can also actively attempt to access the hotspot and to additionally determine whether UAM is applied and which service providers are available, e.g., by parsing the web pages comprising the captive portal as we have discussed for automating authentication [OKK05].

Three prerequisites need to be fulfilled: Firstly, the mobile devices need to be turned on to fulfill this function. This is quite likely for WLAN-enabled mobile phones and may hold for PDAs. Even laptop users may collect such information when they stop and power on their computers, e.g., in a café. Secondly, positioning information needs to be available. This can be derived from internal or external GPS devices[13], from location information derived from cell towers (which is accessible from mobile phones), and from location information supplied from the network, e.g., via DHCP [PSL04]. Finally, the additional activity must not consume so much battery power that it would impact the usability of the mobile devices. With today's mobile phones or PDAs, however, permanent WLAN operation is not an issue (i.e., it does not reduce the battery lifetime to below 24 hours) and even scanning for WLANs in intervals every few minutes seems workable.

We have defined a *contribution interface* for network service maps which enables mobile users to "upload" their observations about available WLANs. The contribution operation is depicted in figure 5: mobile users may upload their observed data via HTTPS towards one or more configured *upload servers* whenever the user is connected anyway and the upload does not increase the cost (significantly). The upload is password-protected and uploads are logged to prevent malicious uploads aiming at confusing the database. The upload server ensures that reports do not overwrite one another, anonymizes the incoming data (detailed location records constitute sensitive information after all), and stores the results in an *incoming database*.

[12] http://www.stumbler.net

[13] For example, the Nokia N95 features WLAN and built-in GPS, so do other recent mobile phones. Mobile devices can also connect to external GPS receivers via Bluetooth or USB.

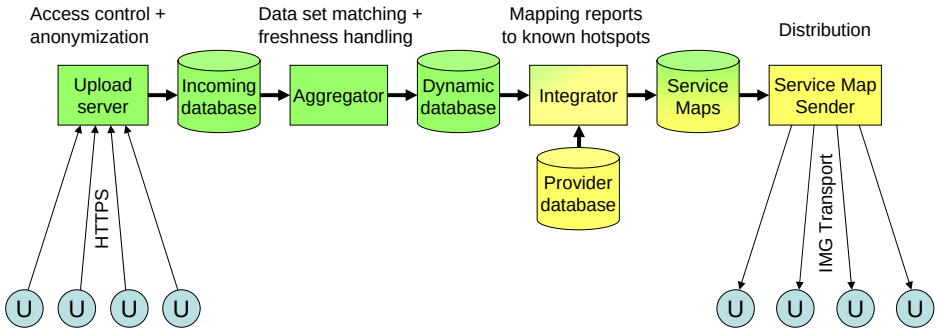

Figure 5: Overview of the contribution process

The data structures are similar to those presented in figure 3 but they lack administrative context information such as who owns which access point, which access points belong together and form a single ESS, etc. Logging is solely done based upon access point identifiers (typically their MAC addresses) and all other attributes are reported per access point. During the subsequent processing, an *aggregator* function gathers all reports per access point and ensures that only the most recent records are used if contradicting information has been reported. The output is the *dynamic database* containing the reconciled reports of all users. In the next step, the *integrator* function peers these reports with administratively configured information about hotspots received from the service providers (e.g., via service maps). The integration is done based upon location information, access point identifiers, and ESSIDs, but further information may be considered in the future. The service maps resulting from this process include a) hotspots known from the service providers augmented by information about current availability and b) hotspots not known from the service providers before. The resulting service maps are then fed into the service map distribution platform for dissemination.

To keep the collection and aggregation process scalable, multiple upload servers may be used that follow common naming conventions when storing incoming reports. The aggregator function may be split across many instances according to geographic regions of a service map, ESSIDs, etc. Finally, the entire chain may be invoked at different intervals: from instant updating whenever a new report comes in for small scales to regularly scheduled updates, e.g., once per hour or per day.

3.8 Security

As information delivered through Service Maps is of public nature, encryption is typically not required. However, the *authenticity* and *integrity* of Service Map information is a crucial property, as, e.g., maliciously modified Service Map documents may lead to non-functional client software. For example, an attacker might add a large number of inexistent services, preventing the client from utilizing real services, which would be a client-side denial of service attack. Because multiple layers of transceivers may be involved in the transportation of Service Maps to a receiver, there is a high potential for such modification at different levels. In addition, for scalability reasons, it is impossible for the client to build a trust relationship with every level of transportation. Also, transceivers would need to authenticate every upstream sender in order to establish trust.

Thus, the goal is to provide for a scalable authentication means of Service Map sources that works in spite of filtering or aggregation of Service Maps along the distribution path. For scalability reasons, asymmetric cryptography is employed, offering large-scale distribution of public keys to receivers for validation of Service Maps. This validation is achieved by public key-based digital signatures, choosing from a list of popular algorithms. Every Service Map implementation with authenticity capabilities needs to support at least RSA/PKCS1 and DSA [RSA77, Nat00].

As Service Maps are using an XML-based format, the *XML Digital Signatures* (XMLDSig, [ERS+02]), is used. For fulfilling the requirement of authenticity validation after filtering, the

concept of *authenticated data structures*, based on Merkle hash trees should be utilized . In this approach, digests of atomic parts of a document are signed as described in [Tam03, MND+01], so that no expensive asymmetric signature operation is needed per atomic part. Still, missing elements may be ignored without losing the ability of validating the remaining parts. In case of Service Maps, the atomic parts are composed of the basic data types, i.e. location, provider, service and instance, as well as refinements. Thus, any of these parts may be filtered out, keeping the remainder verifiable.

4 Implementation

We have implemented a service maps distribution infrastructure that allows operator-independent aggregation and filtering of service information as well as distribution of service information over the different service maps transport mechanisms (ANNOUNCE, QUERY/RESOLVE, SUBSCRIBE/NOTIFY). In addition, we have developed corresponding client implementations: a web-based service maps browser and a stand-alone implementation for mobile devices that is intended to interact with corresponding functions for network selection on mobile devices, e.g., for automating network access based on information obtained from service maps.

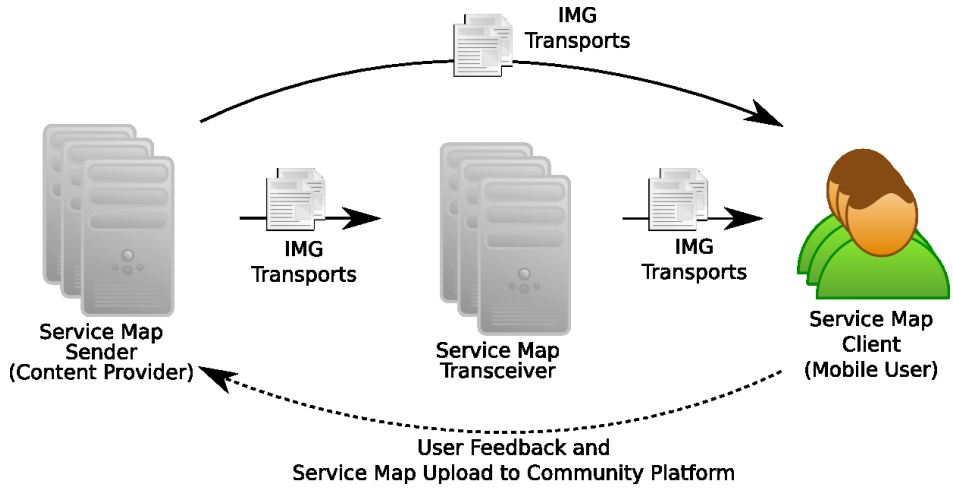

Figure 6: Implemented Components

Figure 6 depicts the three major Service Map components. Service Map Senders offer original Service Map content. This content may be aggregated and/or filtered by Service Map Transceivers, tailored for specific regions or applications. Service Map Clients on user devices acquire and receive Service Map data from Transceiver or Sender. For the Service Map Client, the main objective is to provide a Service Map GUI which enables a user to easily find needed services.

Our Service Maps infrastructure components implement the concepts described in section 3. They have been realized in C++ and are available as Open Source Software. In the following section 4.1, we provide an overview of our server components, Service Map Sender and Transceiver, and the Service Map Client component. In addition, we outline the Service Map client GUI in section 4.2, and describe a minimal GUI variant in section 4.3, a large screen GUI variant in section 4.4 and a GUI for mobile devices in section 4.5. We are operating a Service Maps distribution platform that is described at `http://service-maps.net/`, which also links to the web-based service maps browser providing access to a dynamic database of network (and other) services.

4.1 Service Map Components

Service Maps Senders are the original source for Service Maps and provide for multiple ways of distributing these to receivers and transceivers. Therefore, they implements the broadcasting, publish/subscribe and request/response transports and provide data management services such as authentication, broadcast scheduling and filtering and aggregation. In addition to the sending capabilities of Service Maps Senders, *Transceivers* offer receiving of Service Maps for later re-distributing them in unchanged or filtered and/or aggregated form. Therefore, transceivers implements the majority of the Service Map architecture concepts. Broadcast, publish/subscribe and request/response is included for acquiring and distributing Service Maps. The data management services Service Map URI resolution, filtering and aggregation, caching, broadcast scheduling and authentication are covered. *Service Map Clients* are responsible for receiving and retrieving Service Maps, managing connectivity and providing context information to GUIs as described below. Retrieval is implemented for broadcast, publish/subscribe and request/response. Also, Service Map URI resolution and authentication for validating Service Maps is included. For the transport mechanisms, the Service Map components are based on the *Pamina distribution platform*[14] which uses the *Papageno* FLUTE implementation.

4.2 Client GUI Overview

Although the Service Maps approach is intended to facilitate automated network association, GUI-based client software is still important, e.g., when a user requires a quick overview of available services in her proximity. Context awareness plays a major role in providing adequate user-experience for mobile GUIs. Context awareness does not only include the knowledge of the current position, but also all other factors influencing the user in his decision-making. In scenarios where a user is looking for specific services at a travel destination, a map view for orientation is helpful. Also, it helps the user to learn of existing and available services at once, as the graphical presentation is typically easier to interpret than an XML-based list of services. Parts of a large-screen GUI with a map view on the right side is shown in figure 7. Icons represent service instances on the map, with specific images associated with certain services and providers. For each service instance, a pop-up window presenting detailed information regarding the instance may be opened.

Figure 7: Service Map Large-screen GUI Detail

While map views are useful for gaining an overview of the surrounding services, users sometimes do not need as much information, if their only concern is to quickly use a specific service. For

[14] https://prj.tzi.org/cgi-bin/trac.cgi/wiki/Pamina

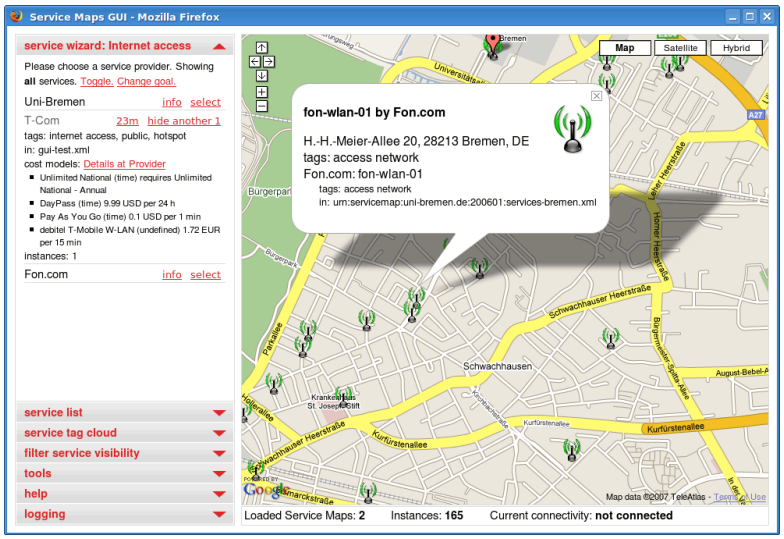

Figure 8: Service Map Large-screen client screenshots

these scenarios, we have developed a tool called *service wizard* (depicted in figure 7) that enables the user to browse services by refining search criteria in a stepwise fashion. At first, all available service provider for the specific task, such as Internet access, are shown. Favorite service providers are highlighted and sorted to top, while providers with unreachable services are only shown on demand. After selecting a provider, depending on the choice, any necessary dependencies are handled in the same way. Given that there are any dependencies and there is more than one provider for it, a choice of further options is offered. At one point, which may already be after selecting the first service provider, all necessary dependencies are met and the services on the dependency stack selected. If no, or no suitable, service instances are available at the current location during one of the steps, currently unreachable ones are added to the list. In this case, the service wizard shows the nearest service instance's distance to the current position and allows jumping to that instance on the map view.

4.3 Minimal GUI

For automated operation, users would expect that a specific previously configured service is always selected by the system automatically (when the service is available), e.g., Internet connectivity. In such cases, the minimal GUI offers using Service Maps in a optimized, efficient way. Depending on the specific platform, the GUI may only be visible through a task bar icon, indicating the current service usage status. A pop-up menu provides for the auto-usage configuration and displays a list of available service instances with summarized information.

4.4 Large-screen GUI

If the Service Map Client is employed on a large-screen device, such as a laptop, and the simple interface of the minimal GUI is not sufficient, a dedicated large-screen GUI may be used. As shown in figure 8, it offers a set of concurrently visible GUI elements. A large map view is continuously being displayed for an overview of the user's surroundings. An small status bar provides for observing the current status. On the left, a sidebar provides access to additional GUI elements such as the *service wizard*. With the sidebar showing GUI elements as selected by the user, the GUI may adapt according to context, e.g., by hiding unnecessary information.

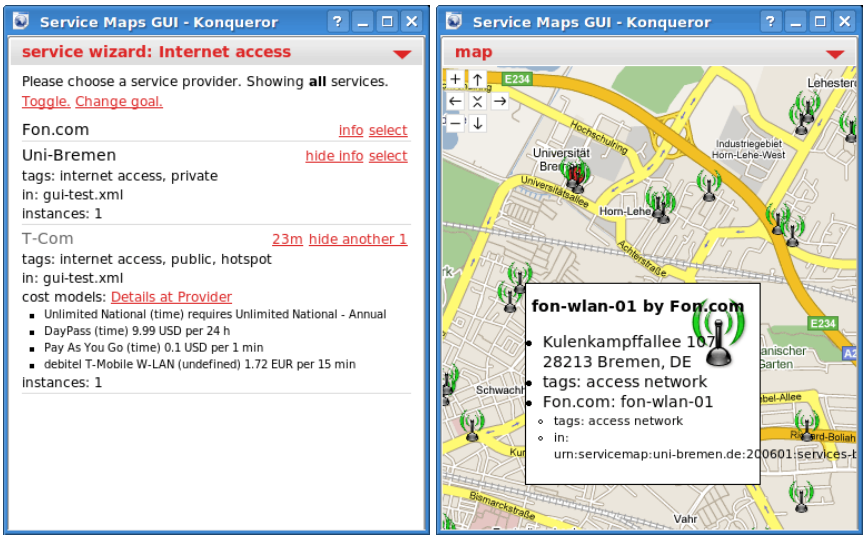

Figure 9: Service Map Mobile client GUI screenshots

4.5 Mobile GUI

The major challenge for developing a Service Map browser for mobile devices such as mobile phones is the limited display resolution. Also, mobile devices typically provide input devices that are different from those on full-size computers, such as touchscreens and joystick instead of a mouse. Instead of displaying many GUI elements concurrently on screen, the mobile-specific layout only displays one of the elements at once to ensure usability, as depicted in figure 9. The user may quickly switch between GUI elements with two clicks through a dedicated menu.

5 Usage Scenarios and Evaluations

We have setup Service Map systems for different environments and are continuously operating Service Map infrastructure systems for different usages. In this section, we describe two specific scenarios. Section 5.1 describes our permanent Service Map installation for the production campus WLAN of Universität Bremen and reports some experiences and measurement results. Section 5.2 describes a scenario for mobile usage of public WLAN services that we have set-up and validated in an emulation environment.

5.1 Campus WLAN

Universität Bremen currently operates a production campus WLAN of 400 access points. These are connected via switches to a single subnet that is integrated on layer two through VLAN tagging. The WLAN network acts as a *docking network* where users can connect without authentication, as security is provided on the network layer, i.e., VPN-based.

For supplying information about WLAN coverage and higher-level services, such as Internet connectivity and VoIP services to users, a Service Map infrastructure has been set-up. This is not only intended to provide information for helping users to manually establish connections, but also for automatic connection management (when used with appropriate client software). In particular, guest users from other academic institutions who would use the local docking network to access their home institution's VPN gateways are intended users for this service.

In order to support the maintenance and development activities for the campus WLAN, we have started to look at using Service Maps not only for distributing service access related information to end-users but also for distributing more detailed information to maintenance personnel.

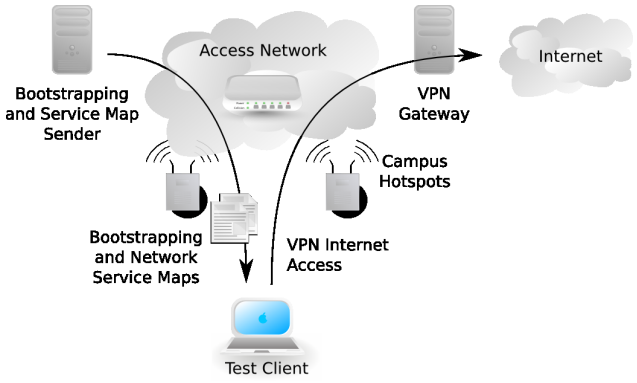

Figure 10: Campus WLAN Scenario

Different types of information are conceivable: specific configuration information, URLs for web administration tools, detailed position information ("in front of room; above intermediate ceiling") or even photos for easily identifying access points in need of maintenance, which can easily be described in Service Maps by employing the Service Map extension mechanisms (application-specific refinements). This information is certainly confidential, so it must be distributed over different channels, i.e., it cannot be part of the public information set. Web-based authentication (for QUERY/RESOLVE) can be used, as large-scale distribution of this information is currently not needed.

For evaluating Service Maps in this campus scenario, we have analyzed the Service Map receiving process with a special focus on multicast distribution. The performed tests also cover the bootstrapping process as described in section 3.6. The test setup is as follows: As depicted in figure 10, the bootstrapping and Service Map sender is connected to the campus WLAN access network. When a test client associates to an access point, the Service Map client may join the well-known bootstrapping FLUTE session, thereby joining its multicast group. Then, the client receives the bootstrapping Service Map, containing the necessary information for acquiring the campus WLAN Service Map. Depending on the bootstrapping configuration, these resource locations may be MUPPET session parameters and/or local HTTP URLs. The client uses one of these mechanisms to retrieve the Network Service Map, before employing the contained information to establish a VPN connection to the access network Internet gateway for Internet connectivity.

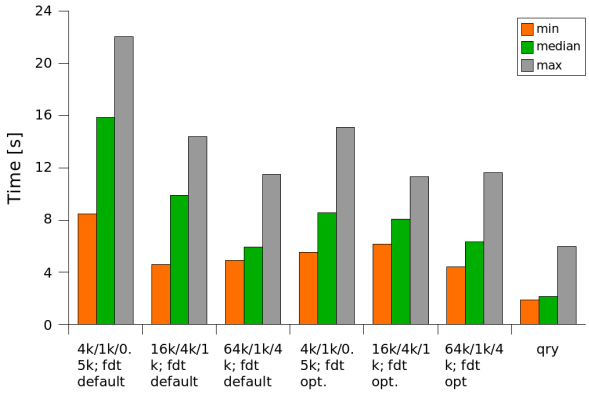

Figure 11: Campus WLAN Measurement Results

We have evaluated the Service Map bootstrapping on a FLUTE channel as well as Network Service Maps distribution via IMG ANNOUNCE (MUPPET) and IMG QUERY/RESOLVE (HTTP).

Measurements of the full time span from AP association to after parsing the received Service Maps data have been conducted. The Campus WLAN Service Map for the 400 campus access points and related Internet connectivity service represent a data volume of 360KB. By applying standard compression tools, we were able to compress the data to about 7KB.

For the multicast distribution tests, we have configured different parameters for each MUPPET FLUTE session, as noted in figure 11 for complete/delta/pointer bandwidth on the bottom axis. In addition, the file descriptor table's (FDT) maximum bandwidth factor was set to the default of 0.2 as well as to an optimized 0.3 and 0.5 for the bandwidth for the `complete` and `pointer` channel, respectively. Also, the MUPPET IMG ANNOUNCE transport was compared to the performance of IMG QUERY/RESOLVE of the Network Service Map data, shown in the last column.

All tests were conducted on the 802.11g radio of Cisco 1200 access points. On the client side, an Atheros AR5212 802.11abg NIC with madwifi 0.9.1 drivers on Debian Linux (etch/testing, Linux Kernel 2.6.16.20) was employed. For sending, the Service Map Transceiver, for receiving the Service Map Client was used with both making use of *Gonzo* MUPPET and *Papageno* FLUTE implementations of the *Pamina distribution plattform*[15].

As shown in the diagram in figure 11, there are significant differences between minimum and maximum data distribution durations, mostly minimum time being half of the maximum time. Irrespective of bandwidth allowed for MUPPET delivery of campus WLAN service descriptions, it is easily outperformed by direct unicast query/resolve transport. Improvements gained through larger MUPPET channel bandwidth are comparably small. This may be caused by the overhead of MUPPET mechanisms for joining multiple FLUTE channels at different rates for congestion control. If packet losses occur, the receiver implementation would leave multicast groups with higher bandwidth (in line with FLUTE's receiver-based congestion-control). Thus, even with only moderate packet losses, it is likely that the maximum available bandwidth is actually not used.

In FLUTE transport sessions the intervals of the File Delivery Table (FDT) transmission play a major role for the reception latency, especially for comparably small files. Here, it is common for the client to gather all necessary file data before a FDT arrives. Without the information of the FDT, the FLUTE reception layer may not hand the received file to upper layers, as no metadata, such as file type and URI, is known yet. Thus, with optimized FDT settings for small file delivery, significantly shorter distribution periods may be reached for lower bandwidths, as seen in column 4 and 5 in figure 11.

When we use MUPPET with a pointer channel for transmitting metadata on currently broadcast items, represented by one FLUTE session, a similar effect may be noticed. Here, the application is notified of available items when the meta data is received on the pointer channel. Then, the application may order the reception of the complete item or deltas to previous versions. Thus, transmission latency consists of the sum of pointer and complete transmission duration in this scenario. For optimizing reception rate, the MUPPET receiver has the ability to cache complete items before any pointers for them have arrived, reducing the reception delay to the maximum of pointer and complete reception delay. Still, with small WLAN Service Maps being broadcast, pointer MUPPET channels need an appropriate bandwidth for optimizing Service Map distribution duration.

With IGMP snooping enabled switches on the access network, wireless networks are generally not affected by the constant transmission of bootstrapping and Service Map distribution multicasts as long as there is no active receiver associated to a given WLAN access point. Only when a client joins the corresponding multicast group, traffic from the Service Map sender to the access point and to the actual receiver starts flowing. The same effect would apply to the different FLUTE channels for a Service Map MUPPET session. A receiver that has received the complete Service Map data over the *complete* channel, can leave that channel and listen for updates on the *pointer* channel, which would also remove load from the current AP's WLAN network.

Distribution of campus WLAN service descriptions using Network Service Maps and FLUTE-based bootstrapping has shown to work well in real-world scenarios. As seen in the tests, even with only small bandwidth of FLUTE bootstrapping channel, clients can bootstrap successfully

[15] https://prj.tzi.org/cgi-bin/trac.cgi/wiki/Pamina

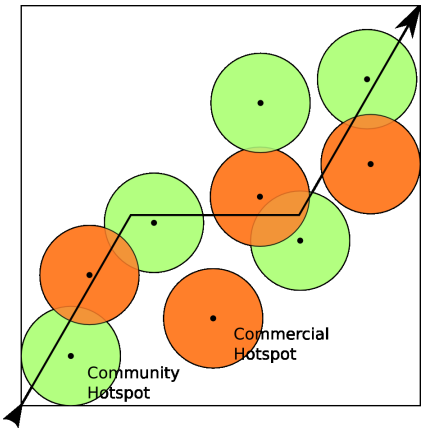

Figure 12: NS-2 Emulation Topography

and receive WLAN service descriptions in acceptable time. Given that this process needs to be done only once (by each client), a low broadcast data rate is acceptable.

5.2 Mobile WLAN

With the increased WLAN coverage in many cities that is provided by numerous wireless Internet service provider (WISP), mobile/nomadic usage of these access networks becomes feasible. Even if permanent connectivity cannot always be achieved, intermittent connectivity could still be useful as shown in [OK05b]. A user may have specific preferences concerning hotspot provider, according to purchased price plans, tariffs or memberships in community WLAN projects, offering free Internet access. However, to still enable fast switching between hotspots in mobile scenarios, manual logins are impractical — instead an automatic connectivity management tool would be desirable. Such a *smart client* should take the before-mentioned preferences into considerations when performing hotspot selection and automatic association.

For exploring the practicality of the Network Service Maps concepts in such a scenario, we have simulated a mobile node moving through a city environment, e.g., on a tram. In this scenario, the client application thereby tries to maximize connectivity through WLAN services along the track. The simulation was conducted using the *Kasuari* simulation framework[16]. It employs the *ns-2* network simulator in emulation mode which is connected to the virtual network interfaces of Xen virtual machines, one per ns-2 node. On a 2000x2000m topology, a moving node and 9 static hotspot nodes are simulated as depicted in figure 12. The path of the mobile node is shown as black line, community hotspots are colored green, commercial orange. A ns-2 “Wireless” physical layer, with a “802.11” MAC, is employed with a nominal range of 250m. The path has a total length of 3130m which is travelled by the mobile node at constant 15m/s with short pauses on the way. Total simulation time is 210s. Bootstrapping of hotspot service descriptions is implemented by Bootstrapping Service Maps, which are distributed over a FLUTE session. Network Service Maps are retrieved via HTTP IMG QUERY/RESOLVE.

Two different test cases have been simulated: In the first one, every hotspot distributes bootstrapping information and descriptions about its own local service only. This information is then used for connecting to the Internet through a web-based authentication scheme. This resembles the current approach to automated login of smart clients, i.e., requiring to associate to access points for probing for certain providers in order to authenticate for Internet usage.

In the second case, the client may collect and use service descriptions that it may received by other sources. This information is distributed in form of Network Service Maps through a wide-

[16]Kasuari is a Xen-based emulation framework that uses ns2 for simulating the characteristics of network links between virtual Linux nodes. More information: https://prj.tzi.org/cgi-bin/trac.cgi/wiki/Kasuari

area broadcast channel, such as DVB-H, for commercial hotspots. Community hotspots not only announce the local WLAN service, but also those of surrounding community hotspots, providing the necessary service information after the first association with a community hotspot. With this information at hand, the number of required bootstrapping phases can be reduced. From simulating the mobile usage scenario as described above, average connectivity times of 162.31 seconds for the first and 179.68 seconds for the second case were measured. Thus, connectivity time is increased by about 17 seconds, i.e., 10%.

These simulations were useful for assessing the usability of the Service Maps approach in mobile WLAN usage scenarios. We have observed that with the flexibility of multiple Service Map distribution paths and client side caching of received Network Service Maps, connectivity times in mobility scenarios may be significantly increased. Of course, performance could still be improved, e.g., by employing more elaborate connectivity algorithms, taking into account information such as distance from next known services and tariff schemes. It is conceivable that a user might want to configure the system with respect to optimization goals, i.e., high connectivity vs. inexpensive usage, thus enabling to decide when to change to a parallel, cheaper service if it comes into reach. Furthermore, mobile usage could be extended to other network types, e.g., 3G, WiMAX. Depending on tariff models, the latter could be used as (potentially) more expensive fallback solutions, when the system knows that no other option will be available in the near future. For a forward-looking optimization, it might also be conceivable to adapt the user's paths according to received Service Map information, e.g, by using an navigation system that is Service Maps aware.

6 Conclusions

This paper has described architectural characteristics of the Network Service Maps with a focus on real-world, potentially large-scale applications. The fundamental motivation for the Service Maps approach is that, in the presence of today's ever-increasing WLAN services, a better support for service selection and automated service configuration is an urgent need to enable people to efficiently access these new services and thus increase their value and utilization further. This is clearly reflected in different solutions being developed by service providers, roaming operators and device manufacturers that aim at facilitating WLAN usage. Although the number of different solutions is impressive, their overall usability is limited, since most of the solutions are limited to specific providers only. Even though some approaches exist that are in principle operator-independent, e.g., the Devicescape approach, they have other shortcomings, e.g., that auto-configuration is only possible for specific WLAN access network types or that auto-configuration is possible, but service search is not.

To overcome these shortcomings, Network Service Maps are independent of the current network attachment of users seeking service (and may also be independent of their present location) so that users can obtain information of a specific network (or any other service) without being required to be connected to a particular one. Our architecture provides application-independent data models as well as aggregation and filtering functions which support operator independence. Using the complementary IMG transports in our distribution platform allows addressing a wide range of application scenarios with different configurations and network architectures. While most network information services rely on static information that is made available to clients, the Service Maps approach can also accommodate dynamically changing service information, e.g., service information that originates from other users and is made available via the contribution facilities as described in section 3.7. This feature is especially important with respect to the proliferation of WLAN community services, i.e., hotspots that are operated by "normal" users, not by companies. The availability of these hotspots typically changes very fast—too fast for static hotspot position databases.

The Service Map system has been implemented and corresponding infrastructure components are in permanent operation. We have been able to gain many interesting insights and performance figures from deployment in real-world production networks such as a large-scale campus WLAN

network. These experiences have shown that the general approach works very well and that the different transport mechanisms, especially the ANNOUNCE variant, is both a resource-friendly and robust transmission technology that is able to adapt to a wide range of network conditions.

The mobile usage of WLAN hotspot service is a promising application that we have been working on earlier [OK05b], however so far only on an opportunistic basis. Our emulation measurements with Network Service Maps have shown that there is significant potential that this technology can help to perform mobile WLAN usage in a more predictive and hence more robust and efficient fashion. We will continue validating these observations by extended tests and measurements in our future work. One key to gaining broader experience will be the large-scale operation of a contribution-based service maps infrastructure, that will also become available at http://www.service-maps.net/.

References

[80205] IEEE 802.21, 2005. http://www.ieee802.org/21/.

[ABS03] B. Anton, B. Bullock, and J. Short. Best Current Practices for Wireless Internet Service Provider (WISP) Roaming, Version 1.0. Wi-Fi Alliance, February 2003.

[BCPL06] Jonathan Beaver, Panos K. Chrysanthis, Kirk Pruhs, and Vincenzo Liberatore. To broadcast push or not and what? In *MDM*, page 40, 2006.

[Che06] S. Chesire. Multicast DNS. Internet Draft, draft-cheshire-dnsext-multicastdns-06.txt, Work in Progress, 2006.

[Cor00] Microsoft Corporation. Universal Plug and Play Device Architecture. available online at http://www.upnp.org/, June 2000.

[DFHX05] Greg Daley, Stefano Faccin, Eleanor Hepworth, and Qiaobing Xie. Some Requirements for a Handover Information Service. Internet Draft draft-faccin-mih-infoserv-01.txt, Work in progress, October 2005.

[EP99] D. Eastlake and A. Panitz. Reserved Top Level DNS Names. RFC 2606, 1999.

[ERS+02] Donald Eastlake, Joseph Reagle, David Solo, Mark Bartel, John Boyer, Barb Fox, Brian LaMacchia, and Ed Simon. XML-Signature Syntax and Processing. W3C Recommendation, February 2002. http://www.w3.org/TR/xmldsig-core/.

[GPVD99] Erik Guttmann, Charles Perkins, John Veizades, and Michael Day. Service Location Protocol, Version 2. RFC 2608, June 1999.

[Gre06] J. Greifenberg. Identifiers for Internet Media Guides (IMG). Internet Draft, draft-greifenberg-mmusic-img-urn-01.txt, Work in Progress, 2006.

[KO06a] Dirk Kutscher and Jörg Ott. Enhancing User Mobility with Network Service Maps. In *Proceedings of TERENA Networking Conference 2006*, May 2006.

[KO06b] Dirk Kutscher and Jörg Ott. Service Maps for Heterogeneous Network Environments. Mobile Data Management Conference 2006, May 2006.

[LM04] Yui-Wah Lee and Scott C. Miller. Network Selection and Discovery of Service Information in Pubilc WLAN Hotspots. pages 81–92, October 2004.

[Mea02a] M. Mealling. Dynamic Delegation Discovery System (DDDS) Part Four: The Uniform Resource Identifiers (URI) Resolution Application. RFC 3404, 2002.

[Mea02b] M. Mealling. Dynamic Delegation Discovery System (DDDS) Part One: The Comprehensive DDDS. RFC 3401, 2002.

[Mea02c] M. Mealling. Dynamic Delegation Discovery System (DDDS) Part Three: The Domain Name System (DNS) Database. RFC 3403, 2002.

[Mea02d] M. Mealling. Dynamic Delegation Discovery System (DDDS) Part Two: The Algorithm. RFC 3402, 2002.

[MND+01] C. Martel, G. Nuckolls, P. Devanbu, M. Gertz, A. Kwong, and S. Stubblebine. A general model for authenticated data structures. Technical Report CSE-2001, 2001.

[Moa97] R. Moats. URN Syntax. RFC 2141, May 1997.

[Nat00] National Institute of Standards and Technology. *FIPS PUB 186-2: Digital Signature Standard (DSS)*. National Institute for Standards and Technology, Gaithersburg, MD, USA, 2000.

[NWL+06] Yuji Nomura, Rod Walsh, Juha-Pekka Luoma, Hitoshi Asaeda, and Henning Schulzrinne. A Framework for the Usage of Internet Media Guides (IMGs). Informational RFC 4435, April 2006.

[OK05a] Jörg Ott and Dirk Kutscher. A Mobile Access Gateway for Managing Intermittent Connectivity. June 2005. Proceedings of the IST Mobile and Wireless Communication Summit 2005.

[OK05b] Jörg Ott and Dirk Kutscher. Exploiting Regular Hot-Spots for Drive-thru Internet. In *Proceedings of KiVS 2005, Kaiserslautern, Germany*, March 2005.

[OKK05] Jörg Ott, Dirk Kutscher, and Mark Koch. Towards Automated Authentication for Mobile Users in WLAN Hot-Spots. In *Proceedings of VTC Fall 2005*, September 2005.

[PLL+04] T. Paila, M. Luby, R. Lehtonen, V. Roca, and R. Walsh. FLUTE – File Delivery over Unidirectional Transport. RFC 3926, 2004.

[PSL04] James M. Polk, John Schnizlein, and Marc Linsner. Dynamic Host Configuration Protocol Option for Coordinate-based Location Configuration Information. RFC 3825, July 2004.

[RSA77] R. L. Rivest, A. Shamir, and L. M. Adelman. A method for obtaining digital signatures and public-key cryptosystems. Technical Report MIT/LCS/TM-82, 1977.

[SRB97] Konstantinos Stathatos, Nick Roussopoulos, and John S. Baras. Adaptive data broadcast in hybrid networks. In Matthias Jarke, Michael J. Carey, Klaus R. Dittrich, Frederick H. Lochovsky, Pericles Loucopoulos, and Manfred A. Jeusfeld, editors, *VLDB'97, Proceedings of 23rd International Conference on Very Large Data Bases, August 25-29, 1997, Athens, Greece*, pages 326–335. Morgan Kaufmann, 1997.

[Tam03] R. Tamassia. Authenticated data structures. In *Algorithms - ESA 2003, 11th Annual European Symposium*, 2003.