# SLSR: A Flexible Middleware Localization Service Architecture

Filip Lemic*†, Vlado Handziski*, Ivan Azcarate*, John Wawrzynek†, Jan Rabaey†, Adam Wolisz*†
*Telecommunication Networks Group (TKN), Technische Universität Berlin
†Qualcomm Ubiquitous Swarm Lab, University of California at Berkeley, USA
Email: {lemic,handziski,azcarate,wolisz}@tkn.tu-berlin.de, {johnw,jan}@eecs.berkeley.edu

***Abstract*—Location information of mobile devices is a foundational input to location-based services and a valuable source of context information in wireless networks. To maximize the value, we need location information that is accurate, robust, and promptly and seamlessly available. Unfortunately, individual localization services seldom satisfy all these requirements. For achieving that vision, a set of challenges has to be addressed, pertaining to handover, fusion, and integration of different sources of location information. Current approaches for integration of individual localization services are either not specific enough or are limited in scope and lack flexibility. In the following, we provide a detailed design and a prototypical implementation of the Standardized Localization Service (SLSR), a middleware architecture for achieving those goals. We instantiate the service in an office environment and perform exhaustive performance benchmarking in a testbed specifically designed for supporting such experimentation. Our results characterize the effects of different functional components envisioned in the SLSR on its performance. Our results also quantify the accuracy benefits of fusion of representative sources of location information.**

## I. Introduction

We are witnessing a rapid growth in demand for location information of mobile devices. Location information is an enabler of location-based services [1] and a decisive context information for improving and enhancing wireless networks [2]. These use-case scenarios require seamless, accurate, and robust location information [2]. However, there is no single prominent solution that currently satisfies all specified requirements. On the contrary, some solutions are able to provide satisfactory performance in some environments, while some others provide good performance in other environments [3]. Their deployments can overlap, which is pronounced for indoor environments due to the absence of the Global Positioning Service (GPS), resulting in a plethora of alternative approaches for such environments.

For promptly, seamlessly, and robustly achieving the desired accuracy, there is a need for an entity for fusion, handover, and integration of different sources of location information. This can be done either on the level of fusion of raw data, i.e. resources used for generating location information, or on the level of fusion of location estimates provided by different elementary sources of such information. The former introduces organizational issues, an example being that many sources of raw data for localization are part of other systems to which a localization service does not have full access. Furthermore, this approach raises ownership and privacy issues, e.g. it is hard to put a price on the access to resources for generating location information and it is difficult to prevent their leakage.

We envision a future where localization services are not going to leak out raw data, but will rather be black-box services with a price tag for providing location estimates. Hence, to improve localization performance, we are facilitating the handover between and the fusion of black-box services that can provide location estimates in a given space at a certain time. Standardization of such an architecture is given in our previous work [4]. While such architectures have been abstractly specified, examples being [5], [6], a full specification and implementation of such a service is still lacking.

To fill this gap, we provide the Standardized Localization Service (SLSR), a fully specified and implemented localization service architecture[1]. The architecture enables provisioning of location information to the applications irregardless of the services that are providing this information. This functionality is supported by a middleware entity of the SLSR, i.e. the Integrated Location Service (ILS). Assuming that the user is mobile and serviced by different Services for Location Information Provisioning (SLIPs), the ILS becomes a proxy with respect to the requirements for location information. We contribute by specifying functional properties of the SLSR, e.g. long-term interpretation of requests for location information, usage of cached location information, mapping of location information types, selection of SLIPs that can provide location information, and fusion of their results. Our contributions also include an implementation of the SLSR and its rigorous evaluation in an office-like testbed environment designed for performance benchmarking of localization solutions [7].

Our results characterize the influence of different functional components to the overall performance of the SLSR. Example-wise, our results show that the usage of cached location information benefits both accuracy and latency of provisioning, while different strategies for selection of SLIPs change the dynamics of the performance of the SLSR, e.g. improve its power consumption, while trading-off accuracy. We further demonstrate the benefits of this middleware approach compared to a single localization solution in terms of localization accuracy enhancements.

## II. Design of the SLSR

The Location-Based Application (LBA) is a component of the SLSR, defined as the entity that requires location

978-1-5090-6299-7/17/$31.00 © 2017 IEEE

[1] https://github.com/terraswarm/standardized_localization_service

information of a mobile device. The central entity of the SLSR is the ILS, serving as a setting for fusion, integration, and handover of different SLIPs. A SLIP is an entirety capable of providing location information of a mobile device.

Previously described main components of the SLSR could be stand-alone applications on a mobile device to be localized, but could as well be deployed in an environment surrounding the device. The latter is tightly related to the Internet of Things (IoT) vision of smart environments providing services to the users and devices in them [8]. The first design decision pertains to positioning each of the specified components in these classes.

The majority of LBAs are stand-alone entities on a mobile device, primarily used for providing some location-based service to the users. However, it has been suggested that location information could be used for improving various aspects of wireless networks [2]. Such scenarios often require location information of a mobile device to be shared with other entities surrounding the device, and therefore we also envision LBAs to be deployed in the environment surrounding the device.

The ILS can be viewed as an individual per-device entity, but also as a global centralized entity for all mobile devices. The latter would allow for a global optimization of invocation of SLIPs and would simplify sharing of location information across devices and applications. However, such a global ILS would intrinsically have increased privacy and security risks, while the avoidance of such issues is of immense importance for location information [9]. Furthermore, such a global ILS would, because of the large number of devices to be supported, be hardly scalable and would therefore have to be a stateless entity. Hence, the “intelligence” of requesting location information would have to be pushed to the LBAs. This means that, for example, periodic or on an event-based provisioning of location information (i.e. when the location information substantially changes from the previous one) could not be explicitly specified, but should be done by periodically issuing requests for current location information by the LBA. However, if such advanced knowledge of future requests for location information is available at the ILS level, the ILS could leverage it for optimizing the invocation of SLIPs. This could yield reductions in power consumption and latency of location information provisioning, as indicated in [10]. We therefore believe the ILS should be fully operating on a mobile device to be localized.

Along the IoT vision of smart environments, we believe that SLIPs could be stand-alone entities on a mobile device, but could also be embedded in smart environments surrounding the device. Examples from the stand-alone category include the usage of dead-reckoning that leverages Inertial Measurement Unit (IMU) readings for generating location information [11]. An example in the latter category is WiFi-based fingerprinting, where a training database, i.e. a sampling of a WiFi environment at training locations, is needed for generating location information of a mobile device. In this case, a fingerprinting server containing the training database is deployed either in a served environment or remotely in a cloud instance, while a fingerprinting client deployed on a mobile device interacts with the server for obtaining location information [12].

A detailed functional overview of the SLSR is given in Figure 1. The procedure starts by different LBAs requesting location information of certain features from the ILS. The features of provisioning are desired accuracy and latency of location information, where location information can be requested once, or periodically or on an event for a certain duration. These requests for location information are received at a listener for requests for location information, where their arrival times are logged.

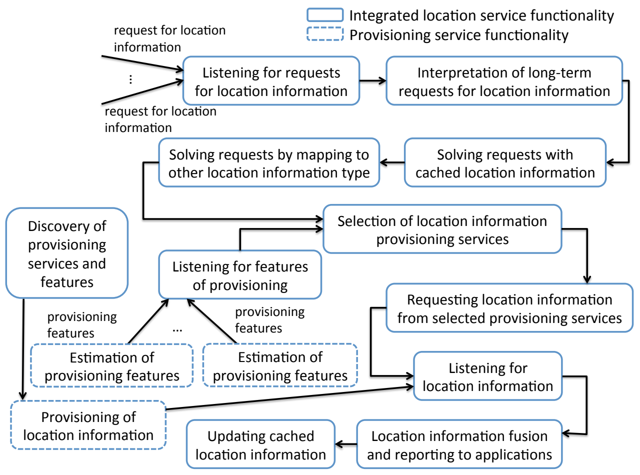
Integrated location service functionality
Provisioning service functionality
request for location information
⋮
request for location information
Listening for requests for location information
Interpretation of long-term requests for location information
Solving requests by mapping to other location information type
Solving requests with cached location information
Discovery of provisioning services and features
Selection of location information provisioning services
Listening for features of provisioning
Requesting location information from selected provisioning services
provisioning features
...
provisioning features
Estimation of provisioning features
Estimation of provisioning features
Listening for location information
Provisioning of location information
Updating cached location information
Location information fusion and reporting to applications

Figure 1: Functional overview of the SLSR

The requests are then forwarded to a long-term requests interpretation entity. The role of the long-term requests interpretation is to translate a request for periodic provisioning or a request for provisioning on events into repeated individual requests for location information. For example, if a request specifies periodic provisioning with a period of 1 s in the duration of 30 s, the long-term requests interpretation will translate such request into 30 requests for location information with the latency of 1 s at appropriate times.

Each request is then potentially resolved with cached location information in case such information is not stale and if it has acceptable accuracy attribute for addressing the request. Similarly, each request is potentially resolved by mapping of location information types. For example, if point-level location estimate is requested, while a room-level cached information exists and is not stale, the mapping of one type to another can be performed, hence the request can be resolved. All requests resolved by either cached or mapped location information are immediately reported back to the LBAs.

All unresolved requests are used as an input in an algorithm for selection of SLIPs. In addition, provisioning features of available SLIPs are also used as an input to the algorithm. These features of provisioning are obtained through the discovery of SLIPs and their features, which is initiated by the ILS. Upon request, all available SLIPs estimate and report their provisioning features to the ILS. Selection of SLIPs to be invoked is therefore based on the requirements from LBAs and provisioning features of available SLIPs. The selection can be subject to various optimization in relation to relevant metrics, i.e. accuracy of location information, and latency and power consumption of provisioning. The selected SLIPs are then invoked for generating location information.

Location information from different SLIPs are received at a listener for location information. Upon receiving each location information, the ILS decides, based on the requirements from an LBAs, if this information should be reported back to the LBA. Adversely, the location information can be fused with location information from other SLIPs before its reporting. This fusion process enhances the accuracy of reported location information, while trading-off latency of provisioning, since a certain delay is introduced by waiting for multiple SLIPs (usually with different latency features) to report their location information. The reported information is additionally used for updating cached location information.

## III. Implementation of the SLSR

### A. GDP-based Implementation of Distributed Messaging

For the prototype implementation of the distributed messaging in the SLSR we have selected the Global Data Plane (GDP) [13]. The GDP is designed as a universal middleware platform, offering prominent types of data access. The GDP natively supports a single-writer log-based messaging, while also providing a Representational State Transfer (REST) and publish/subscribe interfaces for accessing data. Logs are encrypted and only applications with proper encryption keys can access their data, which ensures data privacy and security.

Distributed single-writer log is a messaging pattern based on a partitioned and replicated commit log service. The GDP also supports the discovery of such logs and granting reading access to a particular log. A log is named by an opaque 256-bit number and consists of a series of records, each record consisting of a record number, a timestamp, and data. The mapping of the distributed messaging between the components of the SLSR and the GDP is performed by modeling each interaction primitive by a log in which one entity is allowed to write new data. Other entities are allowed to read and receive notification of the data being written in the log. A favorable feature of the GDP-based distributed messaging is the support for caching and historical information storage and retrieval. This feature is used for the implementation of the module for resolving requests for location information by leveraging cached location information. The GDP provides the possibility of addressing services of the same type at once, which is leveraged in the implementation of the module for the discovery of SLIPs and their features.

Interaction between the ILS and each LBA is implemented as a set of GDP logs. Distributed messaging supported by the GDP allows the LBA to access location information provided by the ILS even in case such LBA is not deployed on a mobile device. The LBA is envisioned to be the only writer to its log for requesting location, with the entry data being the parameters defining the desired features of location information (location type (e.g. point or room-level location estimate), accuracy, latency, once/periodically/on events, duration). The ILS is subscribed to that log and upon request reports location information by writing it to a log for reporting location information to which the LBA is subscribed. The registration of an LBA with the ILS is therefore a matter of creating a set of GDP logs and registering them with the ILS.

The registration of a SLIP is performed by the SLIP granting the ILS a read permission to its "reporting location log", where the first entry to that log is the address of the SLIP (256 bit identifier) and the location information type(s) that can be provided. The ILS is a writer to a discovery log to which all SLIPs are subscribed. SLIPs subscribed to that log report their availability and provisioning features by writing into a log for reporting provisioning features, where an entry defines an offering (i.e. accuracy, latency, and power consumption). The procedure continues in the same way as for the interaction between the LBA and the ILS.

As mentioned, the GDP provides support for REST and publish/subscribe-based messaging. Due to the lack of space, in the following we only discuss how the distributed messaging in the SLSR can be supported by the REST messaging paradigm. The REST messaging pattern's key abstraction is a resource, where the resource is any named information. The REST pattern bases messaging on four operations: creating, reading, updating, and deleting a resource. An LBA can request location information from the ILS using a request for location information, with the payload being the desired features of this information. The ILS reports this information as the response to the request, where the response consists of the requested location information. In case the consuming LBA requires periodic or on an event reporting of location information, an observe flag should be set in the request. When the observe flag is set by the LBA, the ILS will continue to reply after the initial resource has been reported, therefore the asynchronous nature of responses can be achieved. Registration of a SLIP with the ILS is done by issuing a register service request to the ILS, with the payload being the address of the SLIP and location information type that can be provided by the SLIP. During the discovery of SLIPs, the ILS issues a service discovery request to each suitable registered SLIP. As the response, the invoked SLIPs report their provisioning features. The ILS requests location information provisioning from a SLIP in the same way as an LBA requests location information from the ILS.

### B. Selection of SLIPs

A time-bucketed operation of the ILS allows for better utilization of the SLIPs by resolving multiple requests with the same invocation of SLIPs, hence it offers benefits in terms of reduced power consumption of the SLSR [10]. The SLSR is for that reason implemented to operate in a periodic time-bucketed fashion. The duration of each time-bucket is hard-coded to 0.5 s, corresponding to half of the usual update period of roughly 1-2 s needed for users' tracking scenarios [14].

Selection of SLIPs can be subject to various optimizations in relation to accuracy, latency, and power consumption of location information provisioning. In the current implementation of the ILS, the selection of SLIPs can be based on two selection algorithms in details described in [10], while here we provide just their rough overview. The primary aim of both algorithms is latency requirements satisfaction and the secondary aim is the satisfaction of accuracy requirements of all requests received in the duration of a time-bucket. The Per-Request Satisfaction Algorithm (PRSA) is subject to per-request power minimization, while the Per-Time-Bucket Satisfaction Algorithm (PTSA) is subject to a more global per-time-bucket power minimization. The decision on which algorithm to use is left to the user, since one algorithm achieves better performance in terms of power consumption, while trading-off accuracy satisfaction, as analytically indicated in [10] and experimentally evaluated and confirmed in this work.

A step preceding the execution of both algorithms is the generation of "virtual SLIP". The currently used algorithm for the generation of virtual SLIPs is based on fusion of SLIPs with similar accuracy features. Such a fusion yields a more accurate virtual SLIP in comparison to each SLIP in the combination. The similarity in the accuracy is based on the accuracy difference threshold $acc_{th}$ between the accuracy features of different SLIPs. If the similarity between SLIPs exists, the virtual SLIPs are created as all possible combinations of such SLIPs. The latency of each combination then equals the maximum latency among all SLIPs, while the power consumption feature equals the sum of power consumption features of all SLIPs in the combination. The accuracy $acc_{virtual}$ of the generated virtual SLIP can, assuming uncorrelated location information from the SLIPs, be approximated as follows, with $acc_{pr,i}$ being the accuracy of the $i^{th}$ SLIP:

$$acc_{virtual} = \frac{\frac{1}{N}\sum_{i=1}^{N} acc_{pr,i}}{\sqrt{N}} \tag{1}$$

### C. Implementation of the ILS

The ILS is realized as a multi-threaded service. One thread supports listening for requests for location information from LBAs and for provisioning features from SLIPs, and notifies other threads if a request or a provisioning feature was received. Moreover, the same thread performs the discovery of available SLIPs by reading from a register service log and discovery of their provisioning features by writing a discovery request in a proper discovery log. The discovery duration for the SLIPs that are "in vicinity" to a mobile device is fairly small, while this duration increases with the increase in the distance between the mobile device and a particular SLIP. By proper timestamping of the requests and responses for discovery and provisioning features, we leverage this property for distinguishing SLIP surrounding a mobile device from the ones that cannot provide location information of the device in a given environment.

The second thread aims at addressing newly arrived requests for location information by leveraging cached location information or by performing mapping between different location information types. For leveraging cached location information the second thread requires the parameter specifying cached location information, which is generated in the third thread discussed below. In the current implementation, we use a very simple strategy for determining if cached location information is considered stale, i.e. it is stale if it is older than 0.5 s. With the same intuition as for the previously discussed time-bucket duration, the staleness metric of 0.5 s has been selected because it is two times smaller than the update period of 1 s required for users' tracking scenarios. The second thread also handles the selection of SLIPs to be invoked for provisioning.

The third thread handles listening for location information from SLIPs. The same thread also handles fusion of location information and reports location information to LBAs based on their requirements. Finally, this thread deals with updating the cached information with the newly arrived location information and reporting the updated cached information to the second thread.

## IV. Instantiation of the SLSR

For the instantiation of the prototype implementation of the SLSR we selected TWIST testbed, which is an office environment in its usual usage. We used a Lenovo ThinkPad laptop as a mobile device in the instantiation. The components of the SLSR were realized as Python 2.7-based daemon processes. For the instantiation of SLIPs we selected a set of Wireless Fidelity (WiFi) Received Signal Strength Indicator (RSSI) fingerprinting-based localization solutions.

WiFi RSSI fingerprinting-based localization solutions require a training phase in which a survey of a WiFi environment is performed at predefined locations, i.e. at each location a set of scans for WiFi beacon packets' RSSI measurements is taken and stored in the training database [12]. A mobile device whose location is to be determined then generates its own scan of the WiFi environment and sends it to a fingerprinting server. At the server, this scan is compared with the scans from the database and the most similar one is reported as the estimated location of the mobile device. Additionally, fingerprinting can also include a post-processing procedure, e.g. k-Nearest Neighbors (kNN), where instead of reporting just one training location as the estimated location, a set of closest matches is merged and reported as the location estimate, which benefits the accuracy and decreases the variability of errors [12].

An increase in the density of training locations benefits the accuracy of fingerprinting. Similarly, different procedures for calculating similarities between training fingerprints and the user's generated one yield different accuracies and latencies of provisioning [12]. For generating multiple instances of SLIPs with different accuracy and latency features, we therefore used different densities of training locations and different algorithms for calculating similarities between fingerprints.

Different densities of training locations are depicted in Figure 2. The first set of SLIPs uses a "small" number of training points, namely one training point per office, with training points as indicated with blue dots in the figure. The second set of SLIPs uses a "medium" number of training points, i.e. two or four training points per office, depending on an office size, as indicated with red dots in the figure. The

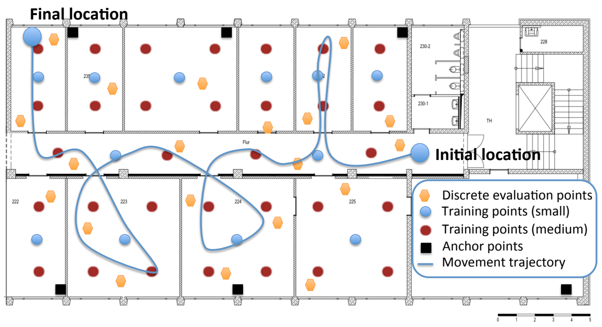

Figure 2: Locations of training points, WiFi APs, discrete evaluation points, and movement trajectory

third set of SLIPs uses a “large” number of training points, i.e. the combination of the small and medium sets of training points. Black squares indicate the locations of WiFi APs used as transmitters of beacon packets for all instantiated SLIPs. We used two algorithms for calculating similarities between fingerprints, namely Euclidean distance of averaged RSSIs and Pompeiu-Hausdorff (PH) distance of RSSI quantiles, with their detailed descriptions given in [12].

As the result, we instantiated six SLIPs out of the combination of training dataset densities and algorithms for calculating similarities between fingerprints. These SLIPs were able to generate and provide point or room-level estimate of location information of a mobile device. In addition, we instantiated two SLIPs that were able to provide only room-level location estimates. This mimics a usual positioning service based on iBeacons that is able to provide only coarse-grained room-level accuracy of localization with lower power consumption in comparison to WiFi-based fingerprinting (e.g. [15]). Due to the shortage of iBeacons, we mimicked their operation by leveraging fingerprinting, more precisely by using a small training dataset and the two algorithms for calculating similarities between fingerprints, and by limiting the reporting of location information to only a room-level type of provisioning.

## V. Evaluation Setup, Scenarios, and Procedure

As mentioned, TWIST testbed is an office environment in its usual operation, while also offering a fully automated testbed infrastructure for the evaluation of indoor localization solutions [7]. The testbed features a mobility platform capable of autonomously carrying a device to be localized to different evaluation locations. The same mobility platform can be used for moving a device to be localized over a predefined trajectory, which is suitable for the evaluation in tracking solutions, in contrast to evaluation at discrete points only. For its positioning in space, the autonomous mobility platform matches a highly accurate floor-plan of the testbed environment with depth information provided by a Kinect camera, achieving less than 10 cm in average localization errors in the testbed space. This is an order of magnitude lower localization error than what the current state of the art indoor off-the-shelf indoor localization solutions can achieve [16], hence the location information from the mobility platform can be used as the ground-truth location of a mobile device for the evaluation purposes. The mobility platform with its speed of 0.5 m/s is somewhat slower than a usual walking speed of a person and the trajectory followed by the platform is to a certain level unnatural for a person. However, the platform reduced the disturbances due to a person’s body, such as shadowing and subtle shaking, thus increases the objectiveness of the evaluation results. Furthermore, the mobility platform enables extraction of reference locations of a mobile device that are unavoidable for the evaluation. This cannot be achieved by a test-person, especially in tracking scenarios, and we therefore leveraged location information provided by the mobility platform as the reference in our evaluation.

The testbed infrastructure also supports monitoring of Radio Frequency (RF) interference context that can considerably degrade the performance of RF-based localization solutions [17], thus reduce the objectiveness of evaluation. The evaluation experiments have therefore been performed on weekends with minimized and controlled influence of RF interferences. The raw data from the experiments, i.e. the WiFi beacon packets’ RSSI scans at different locations in the testbed, were collected and stored in a web-platform for streamlined experimental evaluation of localization solutions [18]. This enabled streamlined evaluation of the SLSR by inputing the collected raw data, calculating location information, comparing them with ground truths, and calculating performance metrics. The web-platform enhanced the objectiveness and comparability of our results, for example when typifying the benefits of different functional components of the SLSR. Two repetitions of all experiments were performed to make an argument about the stability of our performance results.

The evaluation was performed in a localization and two tracking scenarios. For the localization scenario a set of 20 discrete evaluation points was defined, as with orange dots indicated in Figure 2. As the performance metrics we specified point and room-level accuracies. The point accuracy is defined as the Euclidean distance between a ground-truth and an estimated location, while the room-level accuracy characterizes the correctness of estimated room. Clearly, increasing the number of SLIPs also benefits the availability and robustness of location information provisioning by increasing the redundancy of information, but due to the lack of space these influences have not been explicitly characterized.

For the evaluation in the tracking scenarios, we specified the trajectory of movement of a mobile device, as shown in Figure 2. During the course of movement along the trajectory, we constantly collected and stored the raw RSSI scans. We then leveraged the collected scans as input for generating location estimates, while changing the structure and parametrization of the SLSR. The aim of the evaluation in the first tracking scenario was to demonstrate that an increase in the number of SLIPs benefits localization accuracy and to characterize the nature of this benefit. In this scenario, we used each scan of WiFi environment taken along the trajectory for estimating the location of the mobile device.

In the second tracking scenario, we aimed at showing the benefits of different functional components of the SLSR: using the long-term requests interpretation in contrast to “pushing the intelligence" of requesting location information to the LBAs; using cached location information and mapping between location information types for potentially resolving requests for location information; using different algorithms for selection of SLIPs; using different estimators of provisioning features. Metrics used for performance evaluation in the second tracking scenario include satisfaction of accuracy, latency, and both requirements, and the total power consumption of the SLSR. Requirement satisfaction is a binary metrics defined as the capability of the SLSR to provide location information of with a better characteristic than the requirement specified in a given request.

Requests for location information have been generated according to, what we believe is, a realistic scenario for location information requirements. The first request represents a usual person' tracking scenario with conventional requirements, i.e. 1 m in average localization error provided periodically with the period of 1 s during the course of the trajectory and with the location information type being point-level location information [19]. The second requirement resembles a light switching scenario, in which based on the presence of a person in a room the light is either turned on or off [20]. The location information type is semantic (i.e. room-based) provisioning required periodically and frequently with the period of 1 s. The third request mimics a smart thermostat scenario, in which the heating in a room in turned on based on the presence of a person in the room [21]. Clearly, this request requires a periodic semantic provisioning of location information with a high requirement for accuracy and low requirement for latency of provisioning (5 s). The fourth scenario mimics a self-configuring printer scenario in which, if the mobile device is in the same room as the printer, a fine-grained periodic provisioning is needed with the accuracy of 1.5 m and latency of 3 s, while in all other rooms only room-level information is needed upon a change of location information. The fifth scenario resembles a High-Definition (HD) uncompressed video streaming using 60 GHz narrow beams [22], in which these narrow beams have to "track" the user as it moves in the environment. In this scenario, fast (1 s) and accurate (1 m) periodic provisioning of location information has to be available for successful tracking of the user [23]. The sixth request mimics a location-aided horizontal WiFi handover mechanism, in which a fine-grained tracking is required in handover areas (1.5 m/1 s), while a course-grained on an event provisioning of location information is needed in non-handover areas [24]. The seventh request represents the Device to Device (D2D) link establishment scenario, in which if two devices are in vicinity a D2D communicational link is established between them [25]. This scenario requires a fairly fast provisioning of location information (1.5 s) when it changes from the previous location, with low requirement on the accuracy of location information (2 m).

## VI. Evaluation Results

In the first step of the evaluation, we aimed at evaluating the static performance of each SLIP instantiated under the SLSR. The summary results averaged over two (highly comparable) repetitions of each experiment are given in Table I. As visible from the table, with an increase in the density of used training dataset (from "small" to "medium" and "large"), the average accuracy of location information provisioning increases as well. Furthermore, the algorithm PH Distance of RSSI Quantiles shows better accuracy of provisioning. This is due to higher complexity of its pattern matching procedure and due to the fact that RSSI quantiles were used as fingerprints, which provides more information than the usage of average RSSI values in the other algorithm [12]. Both conclusions are consistent for both repetitions of experiments, making an argument about the reliability of the results. Furthermore, with an increase in the number of training points the latency of reporting location estimates also increases, which is consistent with the well-known trade-offs of WiFi fingerprinting [17]. Also, a more complex algorithm has higher latency of location information provisioning. These summarized performance results are used as static provisioning features of each SLIP in the second tracking scenario. Due to the inability to directly measure the power consumption of location information provisioning, we leverage a simple characterization of a power profile of each SLIP, i.e. all WiFi-based SLIPs are given a constant power profile of 2. The SLIPs that mimic iBeacon-based location information provisioning were given a two times smaller power profile. This is consistent with reported comparisons of the WiFi vs. Bluetooth power consumptions, e.g. [26].

TABLE I: Summary performance results of the instantiated SLIPs

| SLIP | Accuracy | Room accuracy | Latency | Power consumption |
|---|---|---|---|---|
| Euclidean small | 3.14 m | 52.5 % | 0.66 s | 2.0 |
| Quantile small | 3.05 m | 55.0 % | 0.72 s | 2.0 |
| Euclidean medium | 2.41 m | 62.5 % | 0.86 s | 2.0 |
| Quantile medium | 2.22 m | 65.0 % | 0.91 s | 2.0 |
| Euclidean large | 1.96 m | 75.0 % | 1.05 s | 2.0 |
| Euclidean semantic | / | 52.5 % | 0.66 s | 1.0 |
| Quantile semantic | / | 55.0 % | 0.72 s | 1.0 |

The results for the first tracking scenario are given in Figure 3. Each group of box-plots depicts localization errors for two repetitions of an experiment. The results demonstrate a high level of stability across the two repetitions, which excludes potential unforeseen outside influences.

The first six groups of box-plots depict the results achieved by the two algorithms with different densities of a training dataset, namely small (S), medium (M), and large (L). As visible from the figure, there is a substantial increase in average localization errors in comparison to the static scenario. The reason for that is the movement of the device to be localized in this scenario, which influences the quality of the obtained WiFi RSSI measurements. Furthermore, in this scenario the less complex algorithm, i.e. the Euclidean Distance of Averaged RSSIs, achieved better performance. This is due to the fact that in moving scenarios the usage of multiple RSSI scans for estimating location does not benefit the localization accuracy because the older scans are essentially taken at some previous locations along the movement trajectory.

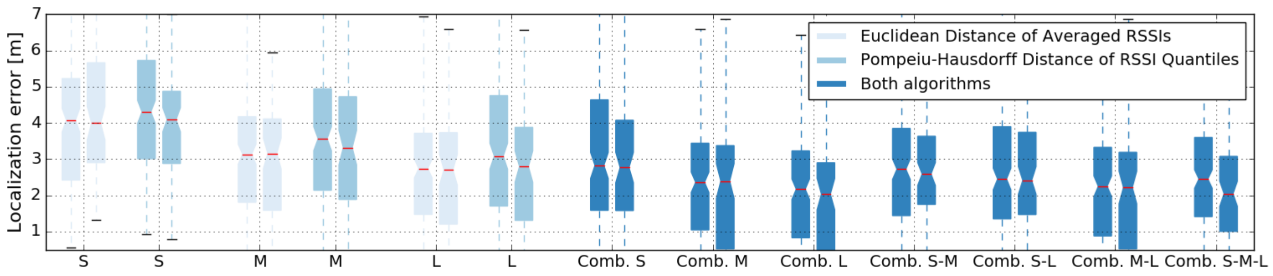

Figure 3: Influence of fusion of location estimates on the final accuracy of location information provisioning

The following box-plots depict the localization accuracies in case of fusion of estimates provided by the two algorithms. For example, the "Comb. S" group of box-plots presents the results of fusion of location estimates provided by the two algorithms, both of them leveraging the small training dataset. Similarly, the "Comb. S-M" depicts the localization accuracies in case the estimates provided by four instances of SLIPs, i.e. the two algorithms leveraging small and medium datasets. The figure shows that the fusion of location estimates is beneficial, but only if the fused instances have comparable accuracy features of provisioning. For example, the "Comb. S" localization errors are generally smaller than the localization errors of any instance of SLIPs that leverages the small training dataset. The same conclusion holds for instances leveraging medium and large training datasets, for both algorithms, and for both repetitions of the experiment. However, in case instances with different localization accuracies are fused, the resulting localization errors are generally comparable or worse than the ones obtained by fusion of only instances with the highest accuracies. For example, the instances labeled with "Comb. S-M" have larger localization errors than the ones labeled with "Comb. M". This means that the fusion should be performed only if the fused instances have comparable accuracies, which is consistent with Equation 1.

In the second tracking scenario, we generated a realistic load for the ILS by using the previously specified requests for location information. We captured the satisfaction of requirements using the following procedure. If the localization error of location information reported to an LBA is smaller than the localization error (i.e. accuracy) specified in the corresponding request for location information, then the requirement is considered satisfied. Similarly, if the latency of provisioning of location information is smaller than the one specified by the request, the latency requirement is considered satisfied. Both requirements of a request are satisfied if both accuracy and latency exceed the ones specified in the request. The consumed power metric specifies the total power consumed by all SLIPs during the course of a movement along the trajectory, with the power profile of each SLIP being as specified in Table I.

Table II summarizes the obtained results for the two algorithms for selection of SLIPs (i.e. PRSA and PTSA). The total number of requirements differs slightly for the two repetitions of the experiment because of small differences in movement of the mobility platform in different repetitions. The "Basic" type of the SLSR includes only basic functionalities of the ILS, i.e. reception of requests, discovery and invocation of SLIPs, reception of location estimates, their fusion and reporting to the LBAs. The "Caching & mapping" type additionally includes caching and mapping functionalities. Caching functionality enables addressing requests with cached location information if the features of such information exceed the requirements of the request and if cached location information is not stale. Mapping functionality supports mapping of point to room-level location information type and vice-versa, if appropriate type of location information exists in the cache. The "Long-term interpretation" type of the SLSR includes the the long-term requests interpretation. In other words, previous types assumed that the LBA will issue a request for location each time this information is needed. The "Long-term interpretation" type assumes that requests for periodic or on an event provisioning for a certain duration can be specified in one request. This request is received by the the long-term requests interpretation entity, which does a request translation to multiple requests at appropriate time instances. The listed types of SLSR leverage static features of provisioning of different SLIPs, as specified in Table I. The "Dynamic" type uses a dynamic specification of the latency and accuracy features of provision of different SLIPs. This is possible because we know the accuracy and latency features of provisioning of all SLIPs and for all WiFi scans collected along a trajectory, as discussed in the previous tracking scenario.

The results in Table II show that the PTSA, which does a more global over a time-bucket optimization of selection of SLIPs, consumes roughly 30 % less power in total. However, for the "Basic" type of SLSR the number of satisfied accuracy and both requirements of the PTSA is roughly 30 % smaller that the same satisfactions of the PRSA. Both algorithms utilize the same strategy for meeting the latency requirements from LBAs, hence the latency satisfaction is the same for both of them. These experimental results confirm the ones analytically derived in [10]. Introducing the mapping and caching capabilities dramatically reduces the total consumed power, while it also highly benefits both accuracy and latency requirements satisfaction. While it is straightforward that caching and mapping functionalities benefit latency requirement satisfaction, the reason for benefiting accuracy satisfaction is due to the fact that, if an estimate of location is provided faster, due to the moving nature of the device, also the accuracy of the estimated location will be higher. In other words, latency improvements in tracking scenarios also benefit

TABLE II: Satisfaction of requirements of different instances of the SLSR

| Type | Total number of requirements | | Accuracy satisfaction | | Latency satisfaction | | Both requirements satisfaction | | Consumed power | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Per-Request Satisfaction Algorithm (PRSA)** | | | | | | | | | | |
| Basic | 659 | 665 | 141 | 154 | 388 | 391 | 113 | 121 | 2021 | 2037 |
| Caching & mapping | 659 | 665 | 352 | 361 | 504 | 511 | 303 | 312 | 1201 | 1225 |
| Long-term interpretation | 659 | 665 | 361 | 371 | 531 | 542 | 313 | 322 | 1167 | 1171 |
| Dynamic | 659 | 665 | 451 | 471 | 528 | 537 | 347 | 354 | 1387 | 1376 |
| **Per-Time-Bucket Satisfaction Algorithm (PTSA)** | | | | | | | | | | |
| Basic | 659 | 665 | 98 | 104 | 388 | 391 | 68 | 70 | 1510 | 1536 |
| Caching & mapping | 659 | 665 | 331 | 338 | 504 | 511 | 281 | 299 | 854 | 831 |
| Long-term interpretation | 659 | 665 | 341 | 351 | 531 | 542 | 294 | 308 | 833 | 812 |
| Dynamic | 659 | 665 | 433 | 452 | 528 | 537 | 339 | 341 | 1041 | 1032 |

the tracking accuracy of the SLSR. Pushing "the intelligence" of requests for location information to the ILS, as done in the "Long-term interpretation" type, further benefits the latency of location information provisioning, since it removes the latency overhead of sending a request by an LBA to the ILS. However, the benefit in terms of accuracy or/and latency satisfaction is relatively small, since the duration of issuing a request in the prototype implementation is also very small. However, presumably for the more distributed LBAs the request time will be larger, which will result in higher benefits of using the long-term requests interpretation. Finally, the usage of dynamic prediction of provisioning features benefits the latency or/and accuracy satisfaction because a more accurate selection of SLIPs can be performed. However, these dynamic features generally suggest smaller accuracy of provisioning than the accuracy features obtained by performing a static evaluation, as demonstrated by the results of the first tracking scenario and the results of static evaluation. Therefore, in many cases more SLIPs are invoked to satisfy the requirements from the LBAs. Thus, the power consumption of the "Dynamic" type is considerably increased in comparison to the "Caching & mapping" and "Long-term interpretation" types.

## VII. Conclusion

In this paper, we discussed the design, prototypical implementation, instantiation, and evaluation of the SLSR. Our results demonstrated the benefits of fusion of location information. Furthermore, our results showed a considerable decrease in the localization accuracy in tracking scenario in comparison to a static localization scenario. We further showed that even the usage of very trivial methods for caching location information and mapping of location information types highly benefits the performance of the service. Finally, having the "God view" of provisioning features, in contrast to leveraging well-established static performance benchmarks, resulted in more optimal selection of SLIPs. We envision that the "God view" could be realized by accumulating and leveraging the knowledge of historical provisioning features of different SLIPs or by dynamically predicting the performance of a SLIP by correlating the features of raw data used by that SLIP for generating location with the expected provisioning features of that SLIP. Future work will be oriented toward designing more sophisticated caching and mapping methods and investigating both aforementioned potentialities for dynamically predicting the provisioning features of various SLIPs.

## Acknowledgments

This work has been supported by the EU elastic Wireless Networking Experimentation Project (eWINE, No. 688116). This work has also been supported by the Qualcomm Ubiquitous Swarm Lab and the Berkeley Wireless Research Center (BWRC) at the University of California at Berkeley.

## References

[1] J. Schiller and A. Voisard, *Location-based services*. Elsevier, 2004.

[2] R. Di Taranto *et al.*, "Location-aware communications for 5g networks," *IEEE Signal Processing Magazine*, vol. 31, no. 6, pp. 102–112, 2014.

[3] F. Lemic *et al.*, "Toward extrapolation of wifi fingerprinting performance across environments," in *Mobile Computing Systems and Applications*, ACM, 2016.

[4] F. Lemic, V. Handziski, *et al.*, "Toward standardized localization service," in *Indoor Positioning and Indoor Navigation (IPIN)*, IEEE, 2016, pp. 1–8.

[5] S. Couronné *et al.*, "Locon-a platform for an inter-working of embedded localisation and communication systems," in *SECON Workshops*, IEEE, 2009.

[6] S. Lempert *et al.*, "Towards a reference architecture for an integration platform for diverse smart object technologies," *Proceedings of MMS*, pp. 53–66, 2011.

[7] F. Lemic *et al.*, "Infrastructure for benchmarking rf-based indoor localization under controlled interference," in *Ubiquitous Positioning, Indoor Navigation and Location Based Service (UPINLBS)*, IEEE, 2014, pp. 26–35.

[8] J. M. Rabaey, "The swarm at the edge of the cloud-a new perspective on wireless," in *2011 Symposium on VLSI Circuits (VLSIC)*, IEEE, 2011, pp. 6–8.

[9] A. R. Beresford and F. Stajano, "Location privacy in pervasive computing," *IEEE Pervasive computing*, vol. 2, no. 1, pp. 46–55, 2003.

[10] F. Lemic *et al.*, "Selection and aggregation of location information provisioning services," *IEEE International Conference on Communications (ICC'17)*, 2017.

[11] N. Moayeri *et al.*, "PerfLoc: An Extensive Data Repository for Development of Smartphone Indoor Localization Apps," in *PIMRC'16*, 2016.

[12] F. Lemic *et al.*, "Experimental decomposition of the performance of fingerprinting-based localization algorithms," in *IPIN'14*, IEEE, 2014.

[13] N. Mor *et al.*, "Toward a global data infrastructure," *IEEE Internet Computing*, vol. 20, no. 3, pp. 54–62, 2016.

[14] T. Van Haute *et al.*, "Performance Analysis of Multiple Indoor Positioning Systems in a Healthcare Environment," *Health Geographics*, 2016.

[15] M. Koühne *et al.*, "Location-based services with ibeacon technology," in *Artificial Intelligence, Modelling and Simulation (AIMS)*, IEEE, 2014, pp. 315–321.

[16] D. Lymberopoulos *et al.*, "A realistic evaluation and comparison of indoor location technologies," in *International conference on information processing in sensor networks*, ACM, 2015, pp. 178–189.

[17] F. Lemic *et al.*, "Experimental evaluation of rf-based indoor localization algorithms under rf interference," in *Localization and GNSS (ICL-GNSS)*, 2015.

[18] F. Lemic, V. Handziski, *et al.*, "Web-based platform for evaluation of rf-based indoor localization algorithms," in *ICC Workshops*, IEEE, 2015.

[19] A. Ess *et al.*, "A mobile vision system for robust multi-person tracking," in *Computer Vision and Pattern Recognition*, IEEE, 2008, pp. 1–8.

[20] R. Harper, *Inside the smart home*. Springer Science & Business Media, 2006.

[21] J. Lu *et al.*, "The smart thermostat: using occupancy sensors to save energy in homes," in *Embedded Networked Sensor Systems*, ACM, 2010, pp. 211–224.

[22] E. Perahia *et al.*, "Ieee 802.11 ad: defining the next generation multi-gbps wi-fi," in *Consumer Communications and Networking Conference*, IEEE, 2010.

[23] E. M. Mohamed *et al.*, "Millimeter wave beamforming based on wifi fingerprinting in indoor environment," in *Communication Workshops (ICCW)*, 2015.

[24] J. J. Nielsen, "Location based network optimizations for mobile wireless networks," PhD thesis, Aalborg university, 2011.

[25] M. M. Theimer *et al.*, *Specifying and establishing communication data paths between particular media devices in multiple media device computing systems based on context of a user or users*, US Patent 5,812,865, 1998.

[26] R. Friedman *et al.*, "On power and throughput tradeoffs of wifi and bluetooth in smartphones," *IEEE Transactions on Mobile Computing*, 2013.