Enabling Internet-of-Things Services in the MobilityFirst Future Internet Architecture[1]

Jun Li, Yan Shvartzshnaider*, John-Austen Francisco, Richard P. Martin and Dipankar Raychaudhuri
WINLAB, Rutgers University, New Brunswick, NJ 08901; *NICTA, University of Sydney, Australia

Abstract — **In the emerging paradigm of pervasive computing, applications change their behaviors in response to their environmental context, which is provided by the smart objects in the Internet of Things (IoT). Due to the inherent heterogeneity of physical world objects, realizing the IoT requires service layers to fill the gap between the low level interfaces of networked objects and the applications which use them. In this paper, we show that the MobilityFirst Future Internet Architecture is an ideal platform for realizing pervasive computing in an IoT framework. In particular, MobilityFirst's identity based routing, overloaded identities, content caching and in-network compute plane are excellent building blocks for IoT applications. We then present a detailed example of a location based service built using MobilityFirst.**

Keywords-* *Future Internet architecture, Internet of things, Middleware, Semantic web, Context awareness

I. Introduction

While the current Internet does an excellent job of connecting devices across networks, the types of devices it connects and the applications it supports have changed markedly. The vision of pervasive computing is coming to pass with the plethora of powerful, highly mobile Internetworked devices in use that run context dependent applications. The current Internet is designed to link machines across the networks they are directly connected to, and as such is strongly network-focused. A device's identity is its network address. This morphism does not support the current usage pattern of devices quickly moving between networks, re-associating with public access points, connecting to multiple networks at once using multiple technologies, while using stream-based applications. Any change in network membership or association requires a device to obtain a new network address, a new identity, disrupting stream-based communication. The current Internet's strong network focus is also hostile to context- and location-based services. Since a device's identity is its network address, any context application needs to maintain some type of translation table from address to identity, which no mean feat is given the many types of network organizations and infrastructures.

What is required is now an Internet of Things (IoT) [1], in which a device's identity is separate from its network address. This single change would allow communications from identity to identity, stepping over the problems of mobility management by pushing the identity/address table into the network itself. Separating identity from network address also allows any logical ontology to be built on top of identities; making context services much easier and direct to implement. While conceptually simple, this change requires the network to operate on and translate between a flat identity space and a hierarchical address space on the scale of the Internet, requiring a clean-slate redesign of the current infrastructure.

While providing for mobility, identity-based routing makes networked entities directly accessible even without IP interfaces or dedicated hosts, such as sensors or services. This provides a generic functional basis to implement IoT middleware services as a part of Internet functions. The IoT network presumes both connectivity and semantic (generalized M-to-M) data delivery, requiring middleware services to bridge Things and applications use them. The ultimate goal of IoT is the Internet itself can replace the middleware platform, enabled by identity based routing, and ever increasing on-router storage and computing capabilities.

Designed to elegantly handle mobile devices and context services, the MobilityFirst project [2] proposes an alternative "clean slate" Internet infrastructure engineered from that bottom-up to support identity-based routing, multi-homing, self-certification and delay-tolerant network transport as core network functions. We argue that a MobilityFirst Future Internet Architecture is an ideal platform for implementing the IoT framework in a pervasive computing environment.

The MobilityFirst architecture provides both identity-based routing and support for context services and general delivery ontology at a network level - providing the necessary functions to implement the IoT inherently, without a middleware service layer.

The remainder of the paper is organized as follows: First, we give an overview of MobilityFirst architecture, focusing on the key elements that we see as beneficial to the IoT vision. Next, we give a briefly description to highlight the related work. Third, we discuss how MobilityFirst can support IoT data and service distribution at core network layer. And then, via a location context service example, we describe the protocol design primitives to enable IoT services on MobilityFirst platform. We also present the work progress on proof-of-concept prototyping system. Conclusions and future work are given finally.

[1] Research Supported by NSF Future Internet Architecture (FIA) program under grant #CNS- 1040735

II. MobilityFirst Future Internet Architecture

The design goal of MobilityFirst is to support the increasing needs of mobile and pervasive computing as native functions in the future Internet architecture, reducing the cost and inefficiency of the overlay solutions over current IP based Internet architecture.

MobilityFirst introduces a global unique identifier (GUID) for every networked object, separated from its dynamic access network locations represented by network addresses. Although the idea is not new, supporting it at core network layer requires a clean-slate future Internet architecture design.

As shown in Figure 1., the proposed GUID is a string in a format of owner's publicKey + hash(S), where *S* is a value chosen by the owner of the networked object, which can be anything from a serial number, a MAC address or a content URI etc. The owner can opt to certify the publicKey through a certificate authority when a third credential party is required. The GUID is unique as long as the publicKey in it is unique. And one publicKey may be used for multiple networked objects differentiated by Hash(S).

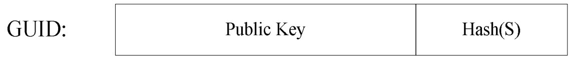

Figure 1. GUID Structure

The separation of naming and addressing allows MobilityFirst to support identity based routing, which allows dynamic address binding (mobility), multiple address binding (multi-homing) and late binding (temporarily disconnection). Multicast and anycast can then become a native support since a GUID can be bound to multiple addresses.

Figure 2 depicts the MobilityFirst (MF) core network architecture. The assumption is that MF routers have a sizeable storage and a computing plane in addition to the basic routing function. The MobilityFirst network includes following three basic services:

1) Global Name Resolution Service (GNRS): A core MF function that maintains mappings between GUID to network address (NA). This is a logically centralized service distributed on all MF routers. The idea is to apply *k* pre-determined hash functions to a given GUID and produce *k* IP addresses. MF routers with any of these *k* IP addresses are selected to serve the mapping of the GUID to its network addresses (NAs). Recent work shows it should be possible to perform this translation much faster (50-100ms) than DNS name resolution by leveraging existing routing tables [3].

2) Hybrid GUID/network address routing: An MF router makes routing decisions for GUID identified data blocks. It uses GNRS to find the network address(s) for a GUID. In case an explicit network address cannot be obtained from GNRS, it may route based on intermediate network address and use "late-biding" from GUID to its actual network address.

3) Delay-tolerant network (DTN) transport: An MF router can deliver GUID-only identified data blocks from/to a networked object. The transport performs a cache and forward (CNF) style, hop-by-hop delivery. The data block is cached in router storage and forward to next hop based on MF routing decisions.

In addition to baseline services, MobilityFirst assumes the existence of a name assignment service (NAS) chosen by the owner of a networked object. Through the NAS, the owner assigns and publishes a GUID with its semantic descriptions for his networked object to a searchable space, such as google or a semantic Linked-data space [16].

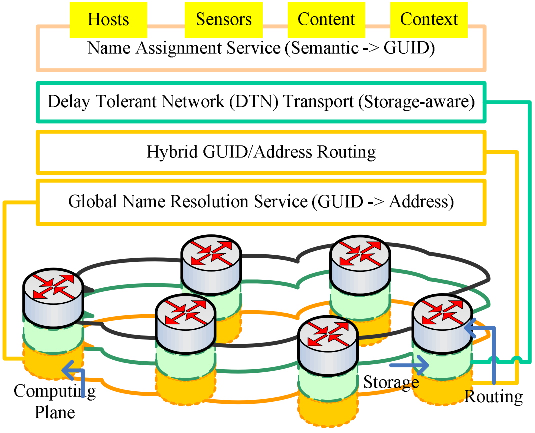
Hosts
Sensors
Content
Context
Name Assignment Service (Semantic -> GUID)
Delay Tolerant Network (DTN) Transport (Storage-aware)
Hybrid GUID/Address Routing
Global Name Resolution Service (GUID -> Address)
Computing Plane
Storage
Routing

Figure 2. MobilityFirst Core Network Architecture

MobilityFirst assumes a GUID is overloaded by an identifying any kinds of networked objects, such as mobile terminals, fixed hosts, sensors, content and (context) services. The identity based routing makes MobilityFirst an ideal IoT platform. For example, sensor data, regardless of its access network address, is identified by its GUID and accessible to IoT applications directly at the networking layer. More importantly, IoT middleware services can also be identified at the networking layer through its GUID; and be cached and executed on routers equipped with caching and computing capacities.

III. Related Works

IoT concept is evolved from machine-to-machine (M2M) applications [4,5], in which, physical objects of concern are connected to the network through a dedicated radio frequency identifier (RFID) and/or wireless sensor networks. The networked objects have application specific identities and middleware services. As IoT is evolving to target a wide area network platform, generic identity and middleware services are expected to share resources at the networking layer.

The Host identity protocol (HIP) [6] proposes to have identity and address split for networked devices. However, it is designed as an end host protocol, rather than be used for routing.

IPv6 is considered as the default for the evolving future Internet architecture. The solutions for mobile and pervasive computing are proposed in mobile IPv6 [7] and 6LoWPAN [8], respectively. 6LoWPAN intends to assign an IP address to every physical world object as its identity. For mobile and pervasive devices, the identities in IP address may have nothing to do with their network access locations. Most of sensors and tags attached to physical world objects may not be able to run IP stack anyway. Thus, it is questionable to use IP as identity when it doesn't offer routing information.

Named Data Network (NDN) [9], another NSF future Internet architecture proposal, goes to another extreme. It uses uniform resource identity (URI, the name of content) as the address of the content (a networked object). The intention is similar to MobilityFirst that it shifts the focus from network centric naming to content centric naming, but combining the identity and addressing causes similar problems as of IPv6.

A fundamental problem is the security of the identity. Since the identity of IPv6 or NDN must be routable, it must be in a text string aggregatable in a routing table, which makes such identities vulnerable to spoofing. Another layer of protocol must then be implemented, such as IPsec, to verify the integrity of the content from an identity. The GUID proposed by MobilityFirst is self-verifying when a private key signature applied to the content from an identified source.

Identity is a key for smart objects but there is still a huge gap between the way objects present themselves and the way IoT applications use them. Middleware services are another key to make objects smart by bridging things and applications in IoT space.

The IoT middleware services have been developed in three directions in recent years. The first is ontology specific systems for a category of applications. Mobiiscape [10] is a typical example based on mobile ontology that provides a core service to determine the location of pedestrians and taxis based on certain mobility patterns. A second direction is the general purpose system for a wide range of applications; HYDRA [11], UBIWARE [12] are typical examples. They try to offer a semantic based platform accept user specified ontology for a wider area of IoT applications, such as, home healthcare, smart grid, fleet management, connected consumers and smart space (home, neighborhood or city) etc.. The effort in these two directions may have good merits for research communities but are difficult to converge to a uniformed approach widely accepted by IoT community.

The third direction is driven by the W3C standard semantic web technologies [13]. The semantic web is originally specified for building the semantic correlation of data inside a webpage. It uses a resource description framework (RDF) [14] and a web ontology language (OWL) [15] to define the relationship between data sources in web pages. Recently, the technology is widely adopted for data owners to publish their data to the public space. Based on the semantic web technology, a concept of building a Linked-data space [16] is being promoted for IoT. Linked-data spaces include billions of data sources belonging to various kinds of information categories. Internet applications can access these data sources, which include static data from various database systems as well as dynamic data representing the status of physical world objects [17]. Under the Link-data space, Things, which may be represented by sensor data, can be accessed by just using database queries through RDF query languages, such as SPARQL [18].

IV. Enabling IoT Services in MobilityFirst

This section discusses how MobilityFirst (MF) core network can directly support the delivery and access of IoT data and service.

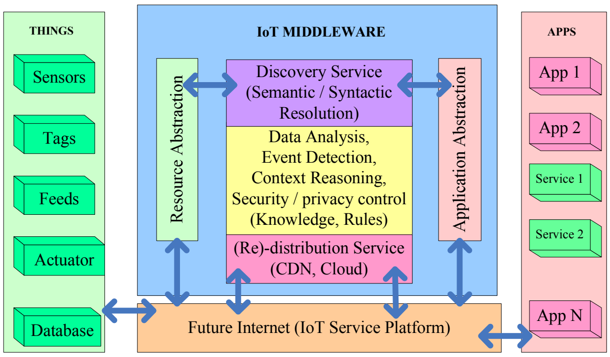
THINGS
Sensors
Tags
Feeds
Actuator
Database
IoT MIDDLEWARE
Resource Abstraction
Discovery Service (Semantic / Syntactic Resolution)
Data Analysis, Event Detection, Context Reasoning, Security / privacy control (Knowledge, Rules)
(Re)-distribution Service (CDN, Cloud)
Application Abstraction
APPS
App 1
App 2
Service 1
Service 2
App N
Future Internet (IoT Service Platform)

Figure 3. IoT Middleware and Future Internet

An IoT survey [19] defines key features a generic middleware must have, which includes discovery, security, interoperability, data management and context detection. In Figure 3., we summarize them into three service categories, 1) discovery, 2) data processing and 3) data (re)distribution. Discovery service allows applications finding Things (sensors) through certain semantic / syntactic resolution. The data processing service generates new data through applying knowledge and/or rules on existing data. And the (re)distribution service delivers data from their sources to different locations for better accessibility. In addition to the cost of sensor network deployment for Things, middleware development and deployment are expensive. Therefore, our goal is to have the architecture of the future Internet providing native support on open access of the IoT smart objects and services to eliminate the cost of overlay deployment.

Figure 4. shows middleware services in above three categories are performed in MobilityFirst in following ways:

MobilityFirst uses a *Name Assignment Service (NAS)* to perform *discovery* service: owners of IoT objects (e.g. sensors) and/or IoT middleware services publish GUIDs with certain semantic attributes into a searchable space. As shown in Figure 4. they are *GUID*_s and *GUID*_c, respectively. Applications can lookup the middleware service identified by *GUID_c* later through NAS.

MobilityFirst uses its on-router storage and computing capability to cache and execute *data processing* service, such as a context reasoning service, traditionally implemented on overlay servers. In Figure 4. the service, identified by *GUID_c,* is cached and executed on a MF router.

MobilityFirst performs *distribution or re-distribution* of IoT data and service through its native support of multicast and anycast based on identity based routing.

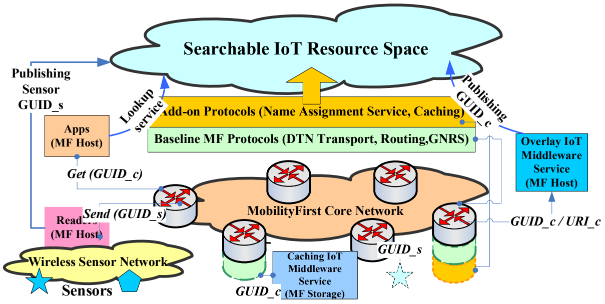
Searchable IoT Resource Space
Publishing Sensor GUID_s
Lookup service
Publishing GUID_c
Add-on Protocols (Name Assignment Service, Caching)
Baseline MF Protocols (DTN Transport, Routing,GNRS)
Apps (MF Host)
Overlay IoT Middleware Service (MF Host)
Get (GUID_c)
Send (GUID_s)
Readers (MF Host)
MobilityFirst Core Network
GUID_c / URI_c
GUID_s
Wireless Sensor Network
Sensors
GUID_c
Caching IoT Middleware Service (MF Storage)

Figure 4. IoT Data and Services in MobilityFirst FIA

MobilityFirst is effectively performing CDN or cloud computing services without overlay servers. The public key comprised GUID makes data/service access and integrity control possible without adding a security protocol layer.

In next section, we will show how IoT service caching is implemented through a specific context aware service example.

V. Implementing IoT Service Caching

We will use a context-aware IoT application as an example from UbiCab [10] to demonstrate how IoT middleware service is enabled inside the MobilityFirst core network.

The example is stated as: "*James walks on NYC streets and wants to find an empty cab closest to his location*". In this example, we assume that James and taxi drivers have sensors on their phones, providing GPS location information as data. The context of this application is the relative locations of James and taxi drivers. A context service, as an IoT middleware, is needed to bridge sensors (providing GPS location) and the application (phone call program) automatically.

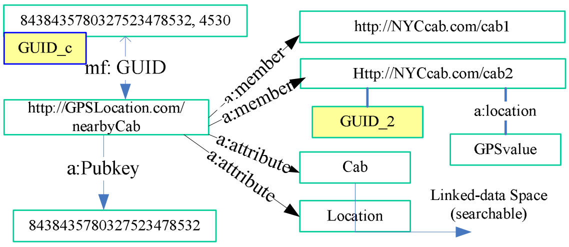

Figure 5. Location Context Service in RDF

We use a linked-data inspired approach to implement the middleware service for this location aware context. First, a company GPSlocation.com offers this service at *http://GPSlocation.com/nearbyCab*, implemented in an RDF graph shown in Figure 5. This context service has two attributes to describe itself, tagged as *cab* and *location.* A taxi driver can query this RDF graph and joins as a member of the service by adding a link to its own URI *http://NYCcab.com/cab1* in the RDF graph with a predicate "*a:member*". The sensor data, taxi driver's GPS location, is linked to taxi driver's URI with a predicate "*a:location"* and the value is updated by his phone periodically. The database containing this RDF graph must be searchable by users in a searchable space, which we refer as IoT resource space in Figure 4.

In MobilityFirst, we also add an additional triple for GUID in the RDF graph with a predicate "mf: GUID". *GUID_c* is assigned to the context service and *GUID_2* is assigned to cab2.

If taxi drivers and James want to use such location context service, they will query to the searchable IoT resource space to find this context service *GUID_c*. For example, they can use an RDF query with "cab" and "location" as attributes to be matched.

This context service can run as an overlay to the Internet. Taxi drivers and passengers can directly call the URI of the context service to join the membership and get a nearby cab, respectively. As an Internet overlay, the location context service is subject to longer delay due to demands are overloaded or unbalanced.

In this example, since the context service is implemented using an RDF graph, it can be cached on MobilityFirst routers in a similar way to caching content. This RDF graph can be updated by taxi drivers by joining membership, current GPS location etc. Since the context service is GUID identified it can be routed efficiently by the MobilityFirst routers.

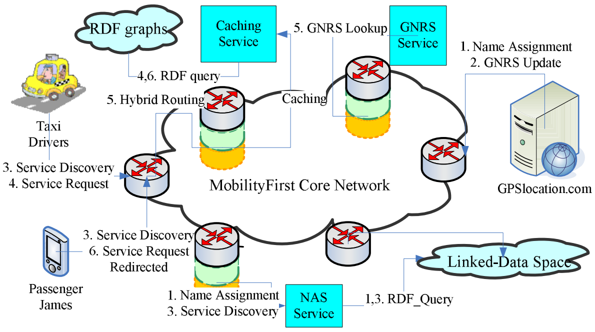

Figure 6. Location context application procedure

The application scenario is comprised of the following steps, illustrated by Figure 6.

Name Assignment. The owner of the context service publishes its *GUID_c* and the service description (RDF graph of Figure 5) through MobilityFirst (MF) *NAS* function, running on both MF host and routers, to a searchable Link-data space.

GNRS Update. When the context service connects to MobilityFirst network, the access router calls *GNRS* update to provide a GUID to network address mapping.

Service Discovery. Taxi driver or James can find *GUID_c* through an RDF query requesting services with tags "location" and "cab".

Service Request. Taxi driver makes a service request to *GUID_c* with an RDF query to join the service membership.

Routing and GNRS Lookup. The service request toward *GUID_c* is routed to the *NA_c,* obtained from a *GNRS* lookup.

Service Request Redirection. James makes a service request to *GUID_c* with an RDF query to match GPS location to a nearby taxi. James' request to *GUID_c* is redirected to *GUID_2*, the driver of cab2.

Note that in this example, there is no CDN overlay is needed to support *multi-homing*, *multicasting* for a location context service because these are embedded in the MobilityFirst core network.

As usual, the benefit of caching is more significant for location dependent services. In our example, it is best to have the middleware service being offered at a server close to taxi drivers and passengers so that the traffic can be localized with lower latency. Traditionally, a service provider has to buy edge computing service from content distribution network (CDN); recently, cloud computing offers another alternative to deploy a distributed service. The native support of caching in MobilityFirst provides with the ability to distribute a service directly through core network routers. This option may not be suitable for heavyweight service requiring a high end web server. Nevertheless, it could be particularly useful for lightweight IoT middleware service. As shown in our example, when the context service is implemented in an RDF graph, caching a service is as simple as caching a data and service processing is as simple as a database query. The algorithms behind cache replacement can also be similar to what for data caching.

Once, for example, the service *GUID_c* is cached, any request to *GUID_c* is intercepted by the MF router. In our example, when it is a request from taxi driver, it must contain an RDF query requesting either join membership or an updated GPS location. The computing layer of MF router runs a SPARQL engine to interpret the RDF query and makes a corresponding database operation. When it is a request from James, a passenger, it must contain an RDF query containing its own GPS location and looking for the ID of a taxi with a GPS location nearby.

Another benefit of MobilityFirst caching is that there is no need for end-to-end connectivity between the requester and the location context server, if the service is currently cached in an accessible MF router. This is especially useful for ad-hoc or DTN use cases, where the application network can be dynamically formed and isolated for a period of time.

This UbiCab example has low data rate. In case a context service requires high data rate, for example, a person wants to see snapshots of the street nearby. And these snapshots are uploaded by the people on the street. Assuming every access network can get enough number of people uploading pictures, it is best for the network operator to cache the pictures as close to edge as possible.

Through this location context service example, we can see MobilityFirst offers significant native supports for IoT data and services to be visible, routable, cacheable and executable in the core network of future Internet architecture.

VI. Current Progress

We have implemented a baseline prototype of the MobilityFirst network architecture that can enable network presence and reachability for middleware services using GUIDs. Our prototype network consists of Click-based MobilityFirst routers that route GUID-addressed packets to network entities using a reliable hop-by-hop DTN data transport. Apart from running a storage-aware control protocol to determine routes, each router also interacts with a GNRS server (optionally running on the router itself) to enable dynamic GUID-to-location binding for mobile entities. Network hosts run a corresponding protocol stack and API to allow applications to interact with the network. Currently, the host protocol stack and API have been implemented for both Linux and Android platforms and support access to Ethernet, WiFi and 4G-WiMAX networks. We have tested and validated our prototype implementation on the ORBIT experimental platform that supports evaluation on a variety of network types up to a scale of 400 nodes [21].

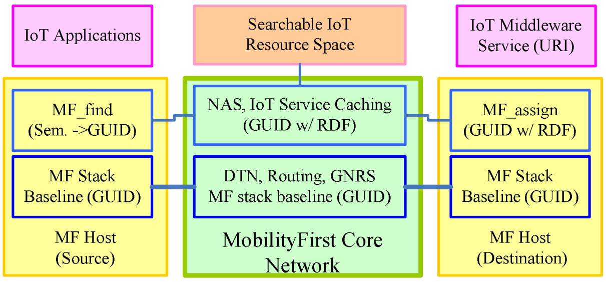
IoT Applications
Searchable IoT Resource Space
IoT Middleware Service (URI)
MF_find (Sem. ->GUID)
NAS, IoT Service Caching (GUID w/ RDF)
MF_assign (GUID w/ RDF)
MF Stack Baseline (GUID)
DTN, Routing, GNRS MF stack baseline (GUID)
MF Stack Baseline (GUID)
MF Host (Source)
MobilityFirst Core Network
MF Host (Destination)

Figure 6. MobilityFirst Protocol Stacks

Figure 6. shows the protocol stacks of MobilityFirst system, consisting of:

1) Baseline protocol stacks:

On MF hosts: DTN transport is implemented that delivers GUID identified data blocks.

On MF routers: DTN transport, intra-network routing and the distributed GNRS service are implemented. They route GUID identified data blocks hop-by-hop based on routing decisions.

2) Name assignment service (NAS) function:

On MF hosts: we plan to implement a NAS library that provide an API to 1) IoT middleware services for GUID publish and 2) IoT application for GUID lookup based on semantic information.

On MF routers: we plan to implement SPARQL engine queries to retrieve information from the service provider. We assume that the service provider leverages the Linked Open Data space [16] to describe and manage information in a triple store.

3) Service caching function:

Content caching is a default MF computing layer function. Caching function for content requires a sequence of decision making computation on content insertion, cache replacement, availability detection mechanisms. We plan to extend the caching function with semantic web technologies. We see, service as content equivalent is represented in an RDF graph and service execution is with RDF-queries.

For MF hosts and routers, we used a typical Linux box on the ORBIT platform, which has an Intel i7 2.93GHz processor, 3GB RAM, an Atheros WiFi (a/b/g) card and an Intel 6050 WiMAX/WiFi card. While large scale evaluations are pending, we believe such router set up can run quite number of cached services at few hundred requests per second load.

We plan to demonstrate aforementioned UbiCab scenario by implementing the MF_assign() and MF_find() functions on MF hosts (Linux and Android), RDF graph caching and execution on MF routers (click router on Linux box on ORBIT). The location context service is run on an MF host as the overlay server, which publish its GUID to IoT service database. Linked-Data space is to be implemented simply in a local SQLite database that can make RDF triple-store. A few android phones with MF host stacks will run as the phone app using the context service for either taxi drivers or passengers. In order to emulate the location information in indoor environment, we will bundle each phone with an active RFID tag and use our wireless sensor network to determine the indoor location of the tags, as the substitutes for GPS values. Passenger's phone is also an MF host. It runs a specially developed app that requests and returns "near by" cab. The "near by" metric is based on RFID tag locations.

VII. Conclusions And Future Work

MobilityFirst Future Internet Architecture posses the qualities necessary to fully realize the Internet of Things. It provides identity-based naming at a network level which separates addressing from delivery and allows the network layer to support arbitrary ontology, enabling; support for mobility through delay-tolerance, seamless handoff, and multipath delivery, support for context-awareness with a network-level context engine.

In the paper, we argue MobilityFirst can provide network layer support of above functions. Through a location context aware application, we present the detailed procedure of integrating IoT service into MobilityFirst core network.

Besides the ongoing work to complete the prove-of-concept prototype to demonstrate this location context service application, we plan further study the issues of service caching, which include the authority of service update in cache, the service integrity and consistency maintenance, load balancing, cache replacement algorithms etc. Furthermore, we need to compare MobilityFirst on-router service with traditional CDN edge computing and cloud computing. The financial question of who is to pay the cost of on-router caching services is still an open and one of the challenging issues we face.

REFERENCES

[1] Luigi Atzori, Antonio Iera, Giacomo Morabito, "The Internet of Things: A survey", Computer Networks, May 2010.

[2] MobilityFirst: NSF FIA project. http://www.nets-fia.net/

[3] T. Vu, A. Baid, Y. Zhang, T.D. Nguyen, J. Fukuyama, R.P. Martin, D. Raychaudhuri, "DMap: A Shared Hosting Scheme for Dynamic Identifier to Locator Mappings in the Global Internet", to appear on The 32nd Int'l Conference on Distributed Computing Systems, ICDCS 2012.

[4] "Initial M2M API analysis", EU FP7 project IoT-A (Internet of Things architecture". 2011

[5] Geng Wu et al. "M2M: From mobile to embedded internet", Communications Magazine, IEEE, April 2011; Volume: 49 Issue:4

[6] Moskowitz, R. and Nikander, P., "Host Identity Protocol Architecture," RFC 4423, May 2006.

[7] C. Perkins, D. Johnson and J. Arkko, "Mobility Support in IPv6", Internet Engineering Task Force, RFC6275, Jully 2011

[8] "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) : Overview, Assumptions, Problem Statement , and Goals" (RFC4919), August 2007

[9] Named Data Network (NDN) Project http://www.named-data.net/ndn-proj.pdf

[10] Byoungjip, Kim , SangJeong, Lee , Youngki, Lee , Insoek, Hwang , Yunseok, Rhee "Mobiiscape: Middleware Support for Scalable Mobility Pattern Monitoring of Moving Objects in a Large-Scale City", Journal of System and Software, July, 2011

[11] Martin SARNOVSKY at el. "HYDRA: Network embedded system middleware for Ambient Intelligent Devices." HYDRA, EU FP6 project, ended December 2010.

[12] Michal Nagy, Artem Katasonov, Oleksiy Khriyenko, Sergiy Nikitin, Michal Szydlowski and Vagan Terziyan, "Challenges of Middleware for the Internet of Things", Automation Control – Theory and Practice, Chapter 14, pp 247 – 270. , Intech, December 2009

[13] W3C Semantic Web: http://www.w3.org/standards/semanticweb/

[14] RDF: Resource Description Framework: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

[15] OWL: Web Ontology Language: http://www.w3.org/standards/techs/owl#w3c_all

[16] Christian Bizer, "Evolving the Web into a Global Data Space", 28th British National Conf on Databases, July 12th, 2011, Manchester, UK

[17] Juan F. Sequeda, Oscar Corcho, "Linked Stream Data: A position paper", 8th International Semantic Web Conference (ISWC-2009), Washington DC, USA, October 26, 2009.

[18] SPARQL: Query Language for RDF http://www.w3.org/TR/2008/REC-rdf-sparql-query-20080115/

[19] Soma Bandyopadhyay, Munmun Sengupta, Souvik Maiti and Subhajit Dutta, "Role of Middleware for Internet of Things: A Study", International Journal of Computer Science & Engineering Survey (IJCSES) Vol.2, No.3, August 2011]

[20] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramach, H. Kremo, R. Siracusa, H. Liu, and M. Singh. Overview of the ORBIT radio grid testbed for evaluation of next-generation wireless network protocols. In Proceedings of the WCNC, pages 1664–1669, 2005.