A Demonstration of Mobility Prediction as a Service in Cloudified LTE Networks

Zhongliang Zhao*, Morteza Karimzadeh†, Torsten Braun*, Aiko Pras†, Hans van den Berg†,‡
*Institute of Computer Science and Applied Mathematics, University of Bern, Switzerland
†Department of Computer Science, University of Twente, The Netherlands
‡Netherlands Organization for Applied Scientific Research (TNO), The Netherlands
Email:zhao@iam.unibe.ch, m.karimzadeh@utwente.nl, braun@iam.unibe.ch, a.pras@utwente.nl, j.l.vandenberg@utwente.nl

***Abstract*—Location prediction has attracted a significant amount of research effort. Being able to predict users' movement benefits a wide range of communication systems, including location-based service/applications, mobile access control, mobile QoS provision, and resource management for mobile computation and storage management. In this demo, we present MOBaaS, which is a cloudified Mobility and Bandwidth prediction services that can be instantiated, deployed, and disposed on-demand. Mobility prediction of MOBaaS provides location predictions of a single/group user equipments (UEs) in a future moment. This information can be used for self-adaptation procedures and optimal network function configuration during run-time operations. We demonstrate an example of real-time mobility prediction service deployment running on OpenStack platform, and the potential benefits it bring to other invoking services.**

***Index Terms*—Mobile Cloud Networking, Mobility Prediction, Cloudification, LTE Networks.**

I. Introduction

The EU FP7 Mobile Cloud Networking (MCN) project [1]website-mcn incorporates the use of cloud computing concepts in LTE mobile networks, in order to increase LTE's efficiency and performance. In particular, the integration of cloud computing benefits in an LTE system can be realised by: (1) extending the cloud computing concept beyond the typical macro datacenter towards new smaller micro datacenter that are distributed within the 3GPP Evolved Universal Terrestrial Radio Access (E-UTRAN) and the Evolved Packet Core (EPC), and (2) deploying and running cloud-based virtualised E-UTRAN (denoted as RANaaS) and EPC (denoted as EPCaaS). The most important cloud computing principles integrated in this virtualised LTE system are the support of on-demand provisioning of LTE components and on-demand elasticity, which allow the virtualised LTE components to scale automatically.

Predicting mobile users' locations in a future moment of time has big potential in various telecom applications, including mobile access control, resource management for mobile commutation and storage, content migration, etc. For example, Information Centric Network (ICN), which provides distributed storage, caching and content relocation features could optimise the distribution of content according to the user mobility prediction results, such that the user content will be stored and available in the location she/he will visit in the future, which will reduce the content access delay [2].

A variety of mobility prediction systems have been proposed. However, most of the works focus on using cloud computing platform to provide storage and computing resources [3]. Few efforts have been done in mobility prediction for the telecom cloud or mobile cloud. In this work, we present *Mobility and Bandwidth prediction as a service (MOBaaS)* - a new service model of telecom operator cloud that enhances the telecom cloud with mobility prediction capacity. We explain MOBaaS via describing the service lifecycle of an on-demand, elastic, and pay-as-you-go mobility prediction service instantiated on top of the cloud infrastructure. MOBaaS service life-cycle management is a process of network design, deployment, run-time management, and disposal that allows to rapidly architect, instantiate, and reconfigure the network components and their associated services.

In the rest of the paper, we first describe the requirements of a cloudified mobility prediction service in section II. Section III includes the design details of the service. Description about the demonstration procedure will be given in section IV. In the end, conclusion and the future work will be provided in section V.

II. Cloudified MOBaaS Requirements

In order to make prediction, certain amount of historical data must be available beforehand such that prediction algorithm can be applied to. To predict future user location information, the historical location information of the user must be provided. Moreover, as a supporting service of the virtualised LTE networks, MOBaaS should interact with other services such that prediction requests can be received and prediction results can be delivered. Therefore, MOBaaS should also provide an interface through which the prediction request/answer message can be sent. Below we list the requirements to be fulfilled for an cloudified mobility prediction service:

- History location data of user (UE) movement.
- A web service running on the MOBaaS side, which is ready to receive any prediction requests and send back the prediction results.
- A proper design of MOBaaS, which makes it easily be instantiated, deployed, and disposed in OpenStack cloud computing platform.
- A mobility prediction algorithm that can make the prediction within limited delay.

978-1-4673-9501-4/15/$31.00 ©2015 IEEE

III. MOBaaS Overview

In this section, we describe the proposed MOBaaS architecture that is based on the MCN service architecture reference model [4], and mainly explain the service orchestration procedure of the mobility prediction service. The details about the designed mobility prediction algorithm and the implementation of MOBaaS could be found in our companion submission.

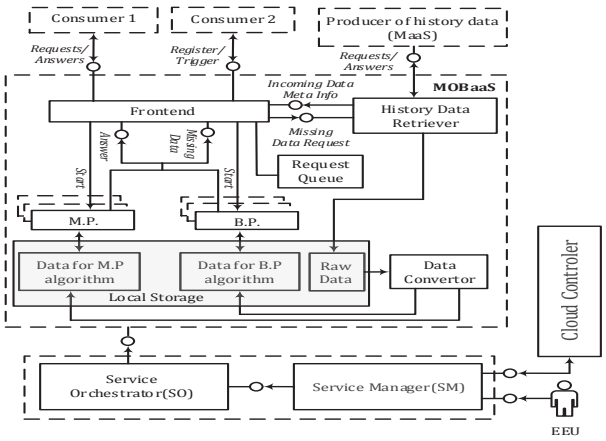

Fig. 1. MOBaaS Architecture Model.

Fig. 1 shows the architecture model of MOBaaS. Compared to many existing orchestration frameworks, MCN architecture has the feature of not only ensuring the creation and management of service-required resources, but also the external service requirements. The roles of different architecture elements are as below:

- Service Manager (SM)
  - Provides an external interface to the Enterprise End User (EEU) and a list of available services (Service Catalog).
  - Responsible for deploying the MOBaaS Service Orchestrator (SO) and forwarding requests to SO for MOBaaS service instance deployment and disposals.
- Service Orchestrator (SO)
  - Responsible for handling requests from MOBaaS SM by instantiating or disposing MOBaaS service instance through the Cloud Controller Software Development Kit (SDK).
  - Responsible for retrieving monitoring information and taking scaling decision whenever needed.
  - SO main modules are Service Orchestrator Decision (SOD) and Service Orchestrator Execution (SOE), which are in charge of implementing decision logic based on external information performing corresponding execution actions (such as scaling).
- Cloud Controller (CC)
  - Supports service deployment/provisioning/disposal
  - Access to atomic and support services

Fig. 2 describes the interactions between EEU, MOBaaS SM, MOBaaS SO, Cloud Controller, and Monitoring as a Service (MaaS) while deploying and provisioning the MOBaaS

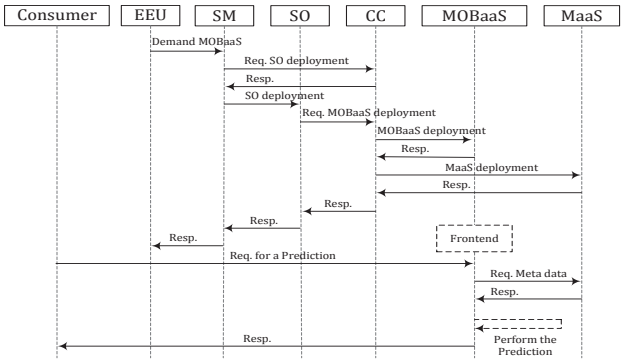

Fig. 2. MOBaaS Service Instance Deployment and Provisioning.

service instance. The procedure of requiring a MOBaaS instance has two stages: deployment and provisioning. When the EEU wants a MOBaaS instance, it firstly sends the request to MOBaaS SM (given that the endpoint of MOBaaS SM is already registered at Cloud Controller). This request will then be forwarded, from MOBaaS SM, to the Cloud Controller in the form of a heat template [5]. After getting the response from Cloud Controller, MOBaaS SM will then deploy the MOBaaS SO. After the MOBaaS SO is deployed and running, it will deploy the required MOBaaS virtual machine and start the provisioning of the MOBaaS internal logics, which means to start all the components (prediction algorithms). After the provisioning phase, a MOBaaS web server will be running at the instantiated instance, which waits for prediction requests.

IV. Demo Description

This section describes the MOBaaS demonstration scenario, which aims at facilitating the audience to understand the relationships between MOBaaS and its invoking services, such as other MCN services of ICNaaS (Information Centric Network as a Service) and EPCaaS (Envolved Packet Core as a Service). The demostration shows how MOBaaS service instance is created under requests and how the prediction results are returned. For the reason of simplicity, we take the ICNaaS as an example of the invoking service of MOBaaS.

A. MOBaaS - ICNaaS Integration

The integration between MOBaaS and ICNaaS is implemented using a request-based approach, in which a RESTful web server is running at the MOBaaS service instance side to receive prediction requests. The MOBaaS web server will be running as far as the service instance is instantiated under the request of an ICNaaS SO.

Whenever ICNaaS needs a prediction, its service orchestrator sends a deployment request to the MOBaaS SM, which will then trigger the instantiation of a MOBaaS service instance. After a MOBaaS service instance is deployed, its endpoint will be sent back to the invoking service such that ICNaaS knows where the mobility prediction web server is running. With the knowledge of the endpoint of the MOBaaS service instance, ICNaaS then sends its prediction request in a JSON (Jave Script Object Notation) message. JSON is a lightweight data interchange format, and it provides a set of standard API

to facilitate the object transmission over network protocols. The request message includes *User-ID, Current Time, Current Date, Prediction Period* (a time interval that the prediction should be repeated), and the endpoint of ICNaaS (to inform MOBaaS where the prediction results should be returned). Given this information, MOBaaS will make the mobility prediction of which cells the specified user will be located with certain probability at a certain future time. MOBaaS will return the prediction results also in a JSON message, which includes multiple pairs of *<Cell-ID, Probability>*, which means the potential cells the user might be with certain probability values. This prediction will be repeated with a time interval of every *Prediction Period*.

B. Evaluation Scenarios

The accuracy evaluation of the proposed mobility prediction algorithm can be found in our companion paper. Therefore, this section only contains the performance results of the MOBaaS SO evaluation, which includes two types of tests:

- Functional tests: to check whether the MOBaaS SO successfully deploys the service instance, and whether the MOBaaS web server could generate prediction results after receiving requests from ICNaaS.
- Performance tests: to measure a set of metrics that are not related to the accuracy of the service/algorithm, but rather to the performance of the environment, in which service is deployed and the latency of making a prediction.

Two testbeds are used, and detailed information of the testbeds are listed as follows:

TABLE I
TESTBED PARAMETERS

Testbeds	Allocated Resource for MOBaaS Test
OpenStack Juno at UBERN	100 Instances, 100 VCPUs, 100 GB of RAM, 80 Public IPs, 1 TB Storage
OpenStack Juno at ZHAW	10 Instances, 20 VCPUs, 20 GB RAM, 8 Public IPs, 1 TB Storage

C. Evaluation Results

To show the result of functional test, we configure ICNaaS to send a request containing the following information:

```
{"user_id":"0","prediction_period":"20",
"current_time":"3:35","current_date":"Monday",
"reply_url : http://130.92.70.142:8080   }
```

which will trigger MOBaaS and a prediction result message will be sent back:

```
{"multiple_user_predictions":
[{"user_id": 72, "next_time": "3:55", "cell": 67,
  "predictions":
   [ {"cell_id": "70", "probability": "0.75"},
     {"cell_id": "79", "probability": "0.25"}]},
 {"user_id": 78, "next_time": "3:55", "cell": 80,
  "predictions":
   [ {"cell_id": "80", "probability": "1.00"}]},
 {"user_id": 92, "next_time": "3:55", "cell": 89,
  "predictions":
   [ {"cell_id": "89", "probability": "1.00"}]},
 {"user_id": 63", "next_time": "3:55, "cell": 37,
  "predictions":
   [ {"cell_id": "29", "probability": "0.68"},
     {"cell_id": "38", "probability": "0.32"}]},...]
```

Fig. 3 shows the results of MOBaaS performance test, which includes the service instantiation/disposal latency and mobility prediction latency. We could see that service instantiation and disposal latencies are shorter in UBERN testbed, which is due to the fact that more resources are allocated at UBERN testbed for the test. Prediction latency depends on two factors: day of the week, and size of the trace file. It is clearly that the prediction using a trace with more users will take much longer time than the prediction using a trace with few number of users. We can also observe that predictions of weekdays take less time than prediction of weekend, which is consistent with the conclusion we have in the companion submission that user mobility is more regular in the weekday than in the weekend.

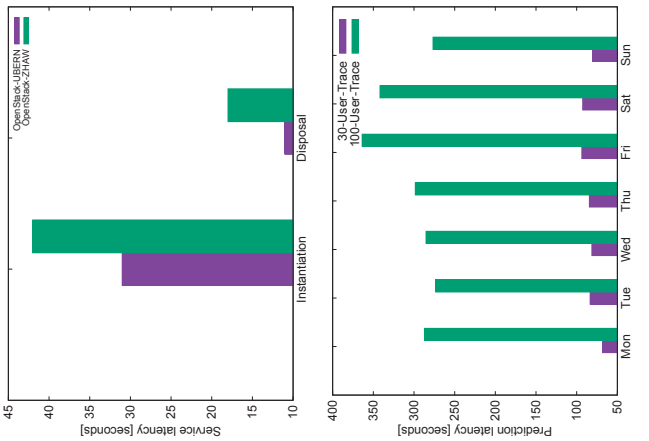

Fig. 3. MOBaaS Service Instantiation/Disposal Latency (left) and Prediction Latency (right)

V. CONCLUSIONS

From the evaluation results of both functional and performance tests, we could conclude that a cloudified mobility prediction mechanism is feasible and it could deliver good performance on cloud platforms. Future works will include integration with other MCN service, i.e. EPCaaS, such that real-time user trace can be collected (e.g., from EPC MME) to make prediction.

VI. ACKNOWLEDGMENT

This work is supported by MCN, the EU FP7 Mobile Cloud Networking project (FP7-ICT-318109).

REFERENCES

[1] EU FP7 Mobile Cloud Networking project, July 2015. http://www.mobile-cloud-networking.eu/site/.
[2] Triadimas Arief Satria, Morteza Karimzadeh, and Georgios Karagiannis. Performance evaluation of icn/ccn based service migration approach in virtualized lte systems. In *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*, pages 461–467. IEEE, 2014.
[3] Hoang T Dinh, Chonho Lee, Dusit Niyato, and Ping Wang. A survey of mobile cloud computing: architecture, applications, and approaches. *Wireless com. and mobile computing*, 13(18):1587–1611, 2013.
[4] Mcn deliverable 2.2 and 2.5. In *European Commmission, EU FP 7 Mobile Cloud Network, public deliverable*, 2013.
[5] Openstack heat orchestration template (hot) guide, July 2015. http://docs.openstack.org/developer/heat/template_guide/hot_guide.html.