Mobile Micro-Cloud: Application Classification, Mapping, and Deployment

Shiqiang Wang^, Guan-Hua Tu#, Raghu Ganti, Ting He*, Kin Leung^, Howard Tripp+, Katy Warr+, and Murtaza Zafer**

*# UCLA – US, * IBM – US, ^ Imperial College – UK, + Roke Manor – UK*

***Abstract*— Mobile micro-cloud envisions a logical network composed of two components, the core (e.g., the command and control center) - with access to large quantities of static (and possibly stale) information and the edge (e.g., the forward operating base) - with access to smaller quantities of more real-time and dynamic data. The edge and core are separated by dynamic and performance constrained networks with a many-to-one relationship between the core and the edge. The goal of the mobile micro-cloud is to deliver situational awareness to the small units (primarily interacting with the edge) in a timely and resource aware manner.**

Fundamental to this mobile micro-cloud paradigm is the flexibility for users to deploy varied applications dynamically as demands, capacity, connectivity, and mission requirements continuously evolve. This "runtime" approach is in contrast to historical systems that are provisioned based on fixed requirements for specific applications.

In this paper, we examine various aspects of the mobile micro-cloud. First, we present an approach to deriving semantics for consistent representation of application requirements in order to enable a generic approach to application deployment in the mobile micro-cloud environment. Second, we examine the advantages of migrating an application (or service) to the edge and quantify these gains through preliminary experimental results. Third, we examine the problem of mapping applications (identified for migration) to available resources that are changing dynamically in a Security-aware manner. Finally, we illustrate the prototype platform for the mobile micro-cloud and its characteristics.

Keywords*— *Mobile micro-cloud, application placement

I. INTRODUCTION

Mobile micro-cloud envisions that applications (or computing tasks) will be deployed in a *mobile micro-cloud*, a logical network composed of two components, the *core* (e.g., the command and control center) – with access to large quantities of static (and possibly stale) information and the *edge* (e.g., the forward operating base) – with access to smaller quantities of more real-time and dynamic data. The edge and core are separated by dynamic and performance constrained networks with a many-to-one relationship between the core and the edge. It is also possible for edge nodes to communicate with each other. Further, the (edge and core) nodes can belong to different coalition partners, raising the question of security and operational policies for handling of data and computation. Figure I-1 illustrates the vision of the micro-cloud for the delivery of situation awareness to the tactical edge and Figure I-2 illustrates a typical architecture of the mobile micro-cloud in the army coalition context.

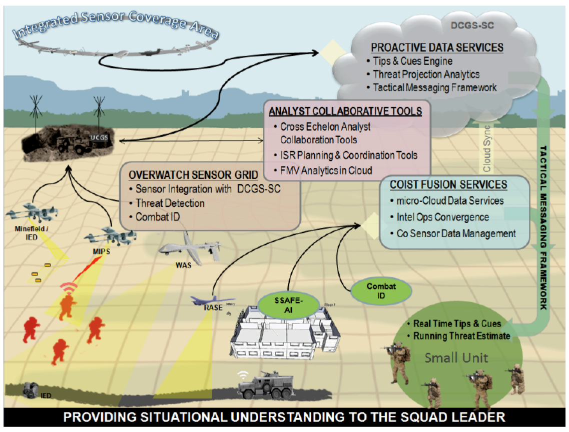
Integrated Sensor Coverage Area
DCGS-SC
PROACTIVE DATA SERVICES
• Tips & Cues Engine
• Threat Projection Analytics
• Tactical Messaging Framework
ANALYST COLLABORATIVE TOOLS
• Cross Echelon Analyst Collaboration Tools
• ISR Planning & Coordination Tools
• FMV Analytics in Cloud
UCGS
OVERWATCH SENSOR GRID
• Sensor Integration with DCGS-SC
• Threat Detection
• Combat ID
Cloud Sync
COIST FUSION SERVICES
• micro-Cloud Data Services
• Intel Ops Convergence
• Co Sensor Data Management
TACTICAL MESSAGING FRAMEWORK
Minefield / IED
MIPS
WAS
RASE
SSAFE-AI
Combat ID
• Real Time Tips & Cues
• Running Threat Estimate
Small Unit
IED
PROVIDING SITUATIONAL UNDERSTANDING TO THE SQUAD LEADER

Figure I-1: Using micro-clouds to deliver Situational Awareness to the tactical edge

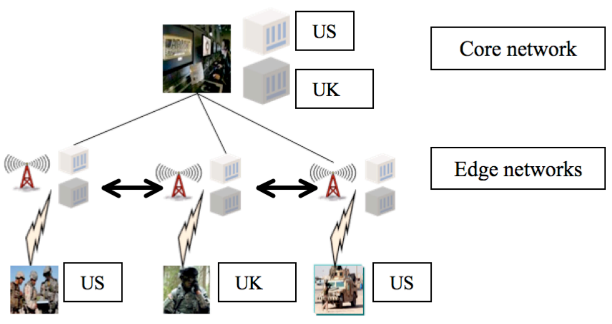

Figure I-2: A tactical network scenario – Enabling efficient computations over dynamic networks

The benefits of embedding storage and computation into such a micro-cloud tactical network are two fold: (i) Effective provisioning for diverse information requirements – the micro-cloud supports users with different latency requirements and access rights and (ii) Effective information exchange in a constrained environment – Complete shuffling of information is impractical in a tactical network and the micro-cloud reduces congestion by providing computation at the edge.

A natural question that arises is that of the applicability of traditional cloud computing technologies to the dynamic tactical network based micro-cloud environment. Traditional cloud computing has become quite popular in the recent years, providing utility computing in a flexible, agile, and scalable manner [1]. The micro-cloud tactical environment poses some fundamentally new challenges to the traditional cloud-computing paradigm. The underlying networks are dynamic and cannot be pre-planned whereas traditional cloud computing facilities provide and assume a homogenous and stable data center environment. Further, coalition operations require distributed and shared compute/storage capabilities with the additional constraint of adhering to the coalition security policies. In contrast, typical cloud computing infrastructure does not warrant the sharing of clouds across multiple parties (this can be thought of as two different enterprises collaborating to achieve a single end-goal).

In this paper, we propose preliminary steps toward realizing the mobile micro-cloud vision. We observe that an key first step in the tactical war fighter context is that of the initial assessment to establish that the cloud's capabilities are sufficient and appropriate to the application's requirements. The mobile micro-cloud presents additional challenges to this assessment that are not present in commercial cloud provisioning environments. For example, the micro-cloud may be highly constrained in terms of its processing power, network bandwidth, latency and reliability. In addition, in the hybrid coalition environment, security domain restrictions may prevent the deployment of an application to a particular micro-cloud. Because of the constraints of the mobile micro-cloud environment, it may not be possible to deploy an application in its entirety to a single edge or it may simply be best if the application is running at the core (e.g., DCGS core cloud). If the application is to be distributed, it may require division into sub-applications and subsequent deployment across multiple micro-clouds in order for its requirements to be satisfied.

Deployment to the mobile micro-cloud must also enable the flexibility and agility appropriate to the tactical environment. This presents a challenge due to the diversity of applications and the highly heterogeneous and dynamic nature of the clouds to which they may be deployed. We seek to establish whether an effective logical application requirements representation can be utilized for multiple application varieties in order to ease the complexity of this deployment and of subsequent application management.

This paper outlines our approach to researching how a tactical cloud's resources can be provisioned on-demand using a consistent language for representation of application structure and its requirements. Fundamental to this is the recognition that applications, or application parts, could have vastly different needs; for example, some tasks might be static triggered batch events, whereas others might be long-running continuous stream processing. We explore the benefits and challenges of developing taxonomy to represent such applications based on their multidimensional needs and the needs of the application parts. The goal is to develop a language to enable applications to be expressed in a normal form suitable for direct integration into the cloud placement and scheduling optimization algorithms. Such a language could be exploited in order to automate the cloud deployment by using the same generic techniques across multiple applications and types. Then, we will we examine the advantages of migrating an application (or service) to the edge and quantify these gains through preliminary experimental results. Further, we will present the problem of mapping applications (identified for migration) to available resources that are changing dynamically in a Security-aware manner. Finally, we illustrate the prototype platform for the mobile micro-cloud and its characteristics.

II. The Benefits of a Consistent Application Taxonomy for Mobile Micro-Cloud Deployment

In order to ascertain the applicability of a micro-cloud for a specific application, and then to aid the application's subsequent micro-cloud placement (IV), we propose consistent representation of application requirements through a consistent taxonomy. This approach is likely to bring the following benefits:

A. Application Placement Optmization

The ability to recognize common application requirement patterns (for example, requirements patterns that are common in MapReduce or Streams processing) and to optimize placement of applications with these requirements based previous assessments and heuristics.

B. Application Partitioning

An application may need to be deployed across multiple micro-clouds due to coalition security or other constraints defining what and where processing and storage can be located. This approach will ease the the ability to optimize application partitioning and subsequent distribution across multiple micro-clouds.

C. Standard Treatment of Policies and SLAs

A generic mechanism for application representation will enable policies (such as resilience, performance or coalition security) to be associated with the application or its parts in a standard way. This capability may also ease other aspects of application management such as the Policy-driven cross-coalition Service Provision (reference task 5.2).

The following figure depicts the methodology for deploying applications using such a classification mechanism.

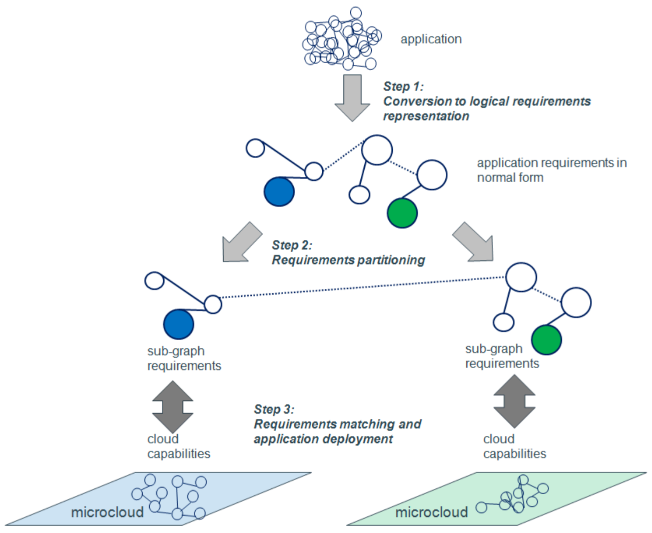
application
Step 1: Conversion to logical requirements representation
application requirements in normal form
Step 2: Requirements partitioning
sub-graph requirements
sub-graph requirements
Step 3: Requirements matching and application deployment
cloud capabilities
cloud capabilities
microcloud
microcloud

Figure II-1: Deployment to Mobile Micro-Cloud Environment exploiting Application Requirements Classification

The steps illustrated in the diagram are as follows:

Step 1 Application Requirements Generation: The application is converted to a normal form depicting its key graphical structure and its requirements. Requirements may be generated automatically from the application itself, from policies associated with the application or from manual applied hints pertaining to application placement. This is discussed below in section III.

Step 2 Application Partitioning: If the application requirements cannot be satisfied by a single micro-cloud (for example, the micro cloud application spans security domains), the application requirements graph may be partitioned into sub-graphs prior to deployment. This is discussed below in section III.

Step 3 Exploitation of Application Requirements During Micro-Cloud Deployment: The application requirements in normal form are matched to micro-cloud capabilities. The compatibility of the micro-clouds(s) with the application requirements are considered in order to deploy the application to the appropriate micro-cloud(s). The subsequent placement of the application is discussed below in section IV.

III. Approach for the Development of an Application Requirements/Constraints Taxonomy

In this section, we present existing application taxonomies and classifications that may be exploited for deployment of applications to cloud environments. We discuss the additional challenges presented by the mobile micro-cloud environment and assess the applicability of existing classifications to this environment. Where the classification is not applicable or is insufficient for the mobile micro-cloud environment, we consider how the classification is likely to require extension/alteration.

A. Application Classification in the Constrained Tactical Environment

A distributed application can be represented as a logical graph of communicating nodes where the graph nodes denote 'computations' and graph links denote 'message passing'. The graph provides an indication of the topology of the application but does not indicate the requirements of that topology in terms of, for example, processing or network requirements. Such requirements may apply to the complete application (for example, represent a profile of the complete anticipated workload), to individual nodes or links, or to portions of the graph.

The approach presented will assess the feasibility of augmentation of an application description and its logical graph representation with additional metadata describing the application's requirements. We will also consider whether specific nodes or sub-graphs of the application can be labeled in terms of their service provision requirements (for example, required connectivity to a service requester).

In the broadest sense, the application may be classified according to existing classification mechanisms that consider the application in terms of broad CPU and IO requirements ([2], [3]). Its primary logical parts are then represented and associated with requirements. Logical parts may be logical nodes, logical communications paths or sub-graphs within the application graph.

The research will begin by the specification of the appropriate application requirements and the association of these requirements with the logical application graph. Potential requirements that may be considered within the context of this research include:

- *Whole application categorization* using an existing application categorization mechanism. This will enable an application to be classified in terms of (for example) IO bound, CPU bound, discrete batch, continuous task.
- *Security constraints* derived from coalition policies. For example, domain and conflict constraints (further discussed in I.B) and minimum encryption level. The security constraints may also specify a subset of modules in the application that has to be run on the same server, in which case these modules are merged into one logical node.
- *Required Qualities of Service* (derived from policies) applied to the complete application or its logical parts. For example, bandwidth and latency requirements between different parts of the application, or resilience requirements. This is important for assessing whether a micro-cloud is capable of supporting an application as well as potentially aiding the application's subsequent placement on that cloud (see I.C).
- *Hints* to aid realization of Qualities of Service applied to the complete application or to its parts. For example, service requester proximity or data

collector proximity or anticipated temporal workload variation (such as expected data reduction at night).

We will take the following approach in order to develop application requirements taxonomy:

1) Derivation and Theoretical Assessment of an Application Requirements Taxonomy

In order to define the appropriate application taxonomy, we will derive and apply a categorization for a number of contrasting example applications informed by tactical scenarios. We will enumerate the similarities and differences between the applications and consider how applications may be logically partitioned and how the sub-parts may be represented within the taxonomy.

We will then consider theoretical physical networks in order to assess whether the information provided in the taxonomy would be sufficient to enable appropriate application placement.

We anticipate that this initial process of defining the application taxonomy will be iterative as we consider broader application examples and different topologies. The output of this step will be an assessment as to the feasibility of defining a broad range of applications in a generic language and recommendations as to the structure and key taxonomy of that language.

2) Automation of Application Requirements Classification and Application Partitioning

We will apply our classification method to a number of applications in order to assess the ease and feasibility of classification automation and what information is required to perform this classification process. Automation of the classification step might be performed based on known classification patterns (for example, a MapReduce job tends to have a particular classification), application of policy information, or monitoring of the application on a non-network constrained system.

Using example logical application graphs, we will assess the use of heuristics for optimum sub-graphing of the application when its requirements cannot be fulfilled by a single micro-cloud.

IV. Application Placement

In this Section, we examine two key problems that arise in the mobile micro-cloud scenario – (i) What are the advantages of migrating an application to the edge and (ii) Provided application migration is advantageous, how do we map the application to the available resources (in the micro-cloud)? We will examine the advantages of migrating an application to the edge through experimental observations in the first part of this section and then tackle the problem of mapping the application to existing resources in a security-aware manner in the latter part.

A. Advantages of Migration to the Edge: A Case Study

A key question that arises in the context of mobile micro-cloud is the quantification of the advantages that the edge offers. These are clearly dependent on the application and its characteristics, where we elaborated on the taxonomy. This section focuses on observations drawn from a real-world scenario and illustrates preliminary results obtained from analysing a large dataset consisting of accessing a large range of services from a cellular network (across a population of about 800k users). We note that the cellular network is representative of a typical three-tiered architecture consisting of the mobile device layer, an edge layer, and a core layer – which is similar to the mobile micro-cloud architecture (depicted in Figure I-2). We use observations drawn from one such network to showcase the advantages that an edge can offer.

In today's cellular networks, all the services run at the core (or in the Internet, accessed via the core). In this context, we compute the average response time to access a service from the mobile devices and compare that with emulated access to services (residing at the edge). Since, there are no services running at the edge in today's deployments, we emulate it by accessing these services without going through the mobile network (e.g., accessing google.com over the WiFi/Ethernet interface as opposed to using the GPRS network).

Based on preliminary experimental results, we observed that the delay incurred in accessing a service in the core is significantly higher than the delays incurred when accessing the same service at the edge. This is a preliminary indicator of the advantages of migrating services to the edge. Further, we also observe that the response delays increase with varying time of day, due to network load and usage patterns.

B. Application Placement with Security Considerations

As we observed in the pervious Section, changes in network characteristics can affect the performance of applications (e.g., increased response times), thus raising a key question – given an application, how do we map this application the resources provided by the mobile micro-cloud environment?

The goal of application placement is to place the workloads onto the physical cloud, to satisfy the constraints of available resource, delay, security, etc., as well as to balance the load of servers and communication links. Without loss of generality, we model the workload as a logical graph with nodes representing modules in the application, and edges representing communications between modules. The physical mobile micro-cloud can also be modelled as a graph with nodes representing servers and edges representing communication links. The logical graph is associated with resource demands and the physical graph is associated with available resources at servers and communication links. The problem of application placement is therefore converted into a graph mapping problem, which maps each logical node to at least one physical node and each logical link to at least one physical path between the hosts of corresponding logical nodes, such that all resource demands are satisfied within physical node/link capacities. Compared with application placement in traditional clouds, challenges in application placement in mobile micro-clouds include the security requirement, time variability of communication links, and the distributive nature of multiple micro-clouds and their coordination in between.

The security requirement in a coalition environment can involve restrictions such as in data sharing, remote access, resource allocation, etc. While the specific security requirements may be protocol-dependent, many of such constraints can be classified into one of the following categories [4]:

- *Domain constraint*, which specifies the subset of physical nodes and edges that each logical node or edge can be placed to.
- *Conflict constraint*, which specifies a set of logical nodes or edges that cannot be placed onto the same physical node or edge.

In a coalition environment, the domain constraint can be used to capture cases where a subset of the logical graph can only be processed in a specific subset of physical nodes in the micro-cloud that is own by a particular team. Such a domain-constrained assignment can reduce the risk of sensitive data being eavesdropped by other teams. The conflict constraint is significant for distinguishing different security levels. For instance, a server that is processing sensitive data for a particular team may not wish to share its resource with logical nodes from other teams, because sharing resources imposes potential eavesdropping risks.

The domain and conflict constraints, together with other resource availability constraints, can be expressed as a set of linear constraints with some binary variables. As a result, the application placement problem can be formulated as a mixed-integer linear program [5], which can be solved with standard optimization tools for small-scale networks. Because the problem is NP-hard in nature, approximation algorithms need to be developed for efficiently solving the problem in large-scale networks.

C. *Time-varying Communication Links*

Another challenge for the application placement in mobile micro-clouds is the dynamically changing communication links. This is particularly obvious for nodes that are connected via wireless links, due to the random fading in the wireless environment and the mobility of nodes. Therefore, the application placement algorithm also needs to consider such dynamics, and performs the application placement accordingly. The application placement algorithm may exploit the application requirements classification described in III in order to place the application appropriately.

1) Link Performance Monitoring

The first step towards the solution to link dynamics is to develop an efficient mechanism to monitor link variations. Depending on the traffic load of the link, the monitoring can be performed either by inserting additional probing packets or by observing the actual data packets that are sent via the link. To fully utilize the available information, a cross-layered approach may be used, which considers information from multiple layers from the physical layer to the application layer. Mobility information that contains nodes' speeds and locations can also be gathered, because the mobility of nodes can impact the variation of link states.

2) Future Performance Prediction

After gathering historical information that reflects link dynamics, a mechanism needs to be developed to predict the future link performance. This could be done using techniques in time-series analysis. The challenge here is to develop an appropriate model to capture information from multiple domains and perform prediction accordingly.

3) Decision on Application Placement

Based on the prediction result, the application placement algorithm needs to decide where to place the logical nodes and edges. Link stability issues need to be considered in addition to other factors. Upon a link failure, the algorithm also needs to decide whether to migrate the affected logical nodes to other physical nodes or to wait for link recovery. A related problem has been studied in [6].

D. *Distributed Application Placement*

Because different micro-clouds may belong to different teams in the coalition, for a large workload which needs to be run across different micro-clouds, all the involved micro-clouds need to cooperate in application placement. However, they may not wish to reveal their complete information to each other. Therefore, in this case, we need to have a distributed algorithm. Some existing distributed optimization techniques may be used to tackle this case. In cases where the shared information is very limited or only a limited number of iterations is allowed, the distributed solution may not achieve the same performance as the global solution. In such cases, we need to study how well the distributed solution can approximate the global solution, as well as what is the best trade-off between information exchange and performance of application placement.

V. DEVELOPMENT OF A MICRO CLOUD EXPERIMINTATION FACILITY

In this section, we discuss the various options for development of a micro-cloud prototype and experimentation facility. There are several "programming models" that can be adopted to deploy applications in the micro-cloud environment, which consist of (a) VM – providing a full scale operating system for execution of applications at the edge, (b) Remote procedure execution, and (c) Web services, these different models are further described in [7]. We note that each of these models provides varying degrees of flexibility (VMs being the most flexible). In contrast, higher flexibility implies heavyweight execution units (e.g., VMs require entire state migration to move the execution unit). We argue in this paper for a more flexible execution unit in the form of Virtual Machines that are the basis for the generic cloud computing environment.

Virtual Machines (VMs) provide a platform for the generic execution of applications along with storage, networking, and other fundamental computing resources. The natural sandboxing of the virtual machines from each other enables easy enforcement of security policies in coalition scenarios. Further, recent work on VM migration such as SnowFlock [8] shows that state migration of VMs can be achieved through novel mechanisms that are very fast (order of few seconds). Given these observations, we chose to use IBM ASPN (Application Service Platform for Networks) for the initial prototype development of the mobile micro-cloud.

ASPN provides a cloud-like virtualization environment for deployment into distributed resource-constrained environments. Applications run in distributed cloud-like containers, isolated from one another but able to communicate with each other (and the platform) in a secure manner. The main components of ASPN consist of (i) Base platform that includes a heartbeat monitor, network configuration service, and libvirt for low-level virtualization features, (ii) ASPN platform VM for secure communication, diagnostic service for collection of diagnostics, and component service for configuration of the deployed application VMs, and (iii) Virtual appliance that are deployed to run on the ASPN base platform (and in which applications reside). Figure V-1 illustrates the various components of ASPN architecturally.

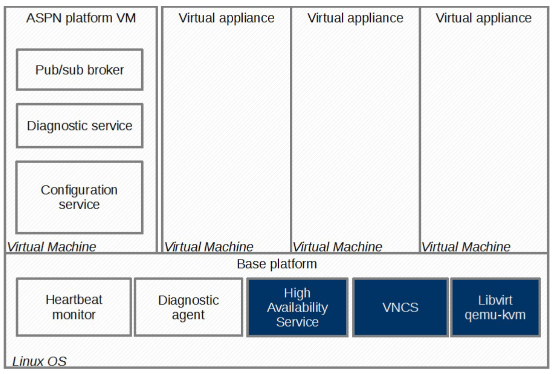

Figure V-1: Architectural components of ASPN

VI. RELATED WORK

A. *Existing Approaches to Application Classification*

Existing approaches to application deployment into commercial cloud assume reliable and homogeneous provision of cloud resources. Application classification techniques consider the complete application as there is no need to partition the application across multiple cloud deployments.

Reference [2] presents an application categorization specifically applicable to cloud provisioning based on the Forward and Lethbridge taxonomy [3]. It is anticipated that this taxonomy, or a similar equivalent, will provide the basis to the application requirements classification at the broadest level of classification for this work.

B. *Application Placement*

Existing works on the application placement problem mainly focus on traditional clouds in a wired network environment [4]. Due to the NP-hardness of the problem, many heuristics have been proposed. Recently, there is increasing attention on optimization problem formulation [9] and approximation algorithms [10] for application placement. However, there are still many open issues, such as incorporating the security constraints and communication link variations into the problem formulation, as well as improving the performance bounds of approximation algorithms. Some preliminary work has also been done in the area of mobile clouds, but mainly focus on cases where there is a single mobile device and a single cloud, such as [11]. The problem of partitioning the application graph into multiple clouds (and each cloud subsequently performs the remaining application placement individually) has been studied in [12], but no optimization guarantee was given. Therefore, the challenges discussed in Section IV follow.

C. *Prototype Development*

The concept of mobile micro-cloud in the form of "cloudlets" was proposed by the Elijah project [13]. This introduced the idea of a new architectural element arising from the convergence of mobile computing and cloud computing. The premise of this project is that in order to enable novel bandwidth hungry and real-time applications on mobile devices, "data center in a box" needs to be brought closer to the device. Four key attributes are identified as part of these cloudlets: (i) Soft-state maintenance for the mobile device generated data, (ii) Powerful, well connected, and safe, (iii) Logical proximity to mobile devices, and (iv) Builds on standard cloud technology such as Amazon EC2 [14] and OpenStack [15]. Contrary to these key assumptions, this mobile micro-cloud project is targeted toward poorly connected environments with heterogeneous nodes and varying resource availabilities.

Virtual machines are quite popular as the success of Amazon EC2 and other cloud computing environments illustrate. Migration of virtual machines using novel encodings of the state differences have been proposed in [8] and [16]. We plan on exploring the applicability of these techniques in the tactical network scenarios.

VII. CONCLUSIONS

In conclusion, we have described the various key problems that we are currently exploring to realize the vision of mobile micro-cloud. The first is to come up with taxonomy of application characteristics to identify and define a unified framework to specify application needs, the second is to examine and quantify scenarios in which applications should migrate to the edge, where we provided detailed analysis results from a mobile cellular network. Based on the input about whether to migrate or not, we then explore the problem of mapping applications to a set of logical resources in a dynamic environment taking security and coalition scenarios into account. Finally, we concluded with the illustration of a platform prototype for the mobile micro-cloud, its architecture and characteristics.

Acknowledgments

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] M. Armburst et. al., "Above the Clouds: A Berkeley View of Cloud Computing", Technial Report UCB/EECS-2009-28, UC-Berkeley, 2009.

[2] A. Milenkoski, A. Iosup, S. Kounev, K. Sachs, P. Rygeilski, J. Ding, W. Cirne and F. Rosenberg, "Cloud Usage Patterns: A Formalisation for Description of Cloud Usage Scenarios", Technical Report: SPEC-RG-2013-001 Version 1.0.1, SPEC RG Cloud Working Group, May 2013

[3] A. Forward and T. Lethbridge, "A Taxonomy of Software Types to Facilitate Search and Evidence-Based Software Engineering" in Proceedings of the 2008 Conference for Advanced Studies on Collaborative Research: Meeting of Minds (CASGON), New York, pp. 14:179-14:191, 2008.

[4] A. Fischer, J. Botero, M. Beck, H. De Meer, and X. Hesselbach, "Virtual network embedding: A survey," IEEE Commun. Surveys Tuts., Feb. 2013, accepted.

[5] S. Wang, M. Zafer, K. K. Leung, and T. He, "Security-aware application placement in a mobile micro-cloud," in Proc. of Annual Fall Meeting of ITA (AMITA) 2013.

[6] T. He, S. Chen, H.l Kim, L. Tong, and K. Lee, "To migrate or to wait: bandwidth-latency tradeoff in opportunistic scheduling of parallel tasks," in Proc. of IEEE INFOCOM mini-conference, Orlando, FL, March 2012.

[7] P. Bahl, R. Han, L. Li, and M. Satyanarayanan, "Advancing the State of Mobile Cloud Computing", In Proceedings of MobiSys workshops, MCS, 2012.

[8] H. Cavilla, J. Whitney, A. Scannell, P. Patchin, S. Rumble, E. Lara, M. Brudno, and M. Satyanarayana, "SnowFlock: Rapid Virtual Machine Cloning for Cloud Computing", In Proceedings of EuroSys, 2009.

[9] M. Chowdhury, M. Rahman, and R. Boutaba, "Vineyard: Virtual network embedding algorithms with coordinated node and link mapping," IEEE/ACM Trans. Netw., vol. 20, no. 1, pp. 206–219, 2012.

[10] N. Bansal, K. Lee, V. Nagarajan, and M. Zafer, "Minimum congestion mapping in a cloud," in Proc. of PODC 2011.

[11] M. Ra, A. Sheth, L. Mummert, P. Pillai, D. Wetherall, and R. Govindan, "Odessa: Enabling Interactive Perception Applications on Mobile Devices," in Proc. of MobiSys 2011.

[12] A. Leivadeas, C. Papagianni, S. Papavassiliou, "Efficient Resource Mapping Framework over Networked Clouds via Iterated Local Search-Based Request Partitioning," IEEE Transactions on Parallel and Distributed Systems, vol.24, no.6, pp.1077-1086, June 2013

[13] CMU, Project Elijah, http://elijah.cs.cmu.edu

[14] Amazon EC2 cluster, http://aws.amazon.com/ec2/

[15] OpenStack Cloud Software, http://www.openstack.org/

[16] K. Ha, P. Pillai, W. Richter, Y. Abe , and M. Satyanarayanan, "Just-in-Time Provisioning for Cyber Foraging", In Proceedings of MobiSys 2013.