User Experience of Mobile Cloud Applications – Current State and Future Directions

Michael J. O'Sullivan, Dan Grigoras
Department of Computer Science
University College Cork, Cork, Ireland
{m.osullivan, grigoras}@cs.ucc.ie

***Abstract* – The increasing penetration rate of feature rich mobile devices such as smartphones and tablets in the global population has resulted in a large number of applications and services being created or modified to support mobile devices. Mobile cloud computing is a proposed paradigm to address the resource scarcity of mobile devices in the face of demand for more computing intensive tasks. Several approaches have been proposed to confront the challenges of mobile cloud computing, but none has used the user experience as the primary focus point. In this paper we evaluate these approaches in respect of the user experience, propose what future research directions in this area require to provide for this crucial aspect, and introduce our own solution.**

Keywords: mobile cloud, applications, services, user experience

I. INTRODUCTION

The rate of penetration of mobile devices is growing in the worldwide population. Smartphones and tablets are capable of providing much richer functionality to the user than traditional devices. They are equipped with multiple networking cards which allow the user to carry out tasks such as browsing the web and accessing online services while on the move. For many users, the mobile device has become their primary platform for browsing the web. Mobile devices can also download from stores and run third party applications, which can be used for tasks such as word processing, music playing, image editing, and playing games. Sensors now also commonly found on smartphones can include a location sensor, accelerometer, gyroscope, camera and microphone. They allow the applications to benefit from context awareness, by getting input about the context of the user and surrounding environment [1]. The combination of these new features has resulted in a demand for mobile devices to execute more complex tasks, including ones that may once have been performed on a desktop computer. Still, limited processing power, memory, and energy supply from the battery prohibit them from executing highly demanding tasks.

Mobile cloud computing is a new mobile computing paradigm that enables mobile devices to be used for more computation intensive work. By using the resources of cloud computing infrastructure [22], the resource scarcity of the mobile device can be addressed. Work can be moved from the mobile device to the cloud, executed there, and results returned to the device when complete.

Unfortunately, many factors combine to make this paradigm difficult to implement and therefore have a detrimental impact on the user experience. The varying quality and availability of network coverage, possibly resulting in disconnection, can inhibit communication between the cloud and the mobile device. The time and energy cost for communication of data over the network is also not negligible. Network operators charge a fee for transfer of data over their wide-area cellular networks, and if the energy used for communication is large, a drained battery will naturally render the device useless. Various approaches to solve some of these problems have been proposed, yet these solutions only tend to focus on one or few of the previously described problems, while the unaddressed problems still dominate. Unlike our work, none considered the user experience as a primary motivation.

Our research interest is in providing an integrated user experience for mobile cloud applications. This is an essential consideration, as demanding and mobile end-users with limited technical background are the consumers of mobile cloud applications and services. The aim of such an experience is to make the interaction between the user and the mobile cloud as seamless as possible, while rich new functionality and models of interaction are enabled by the mobile cloud. Such an experience must also confront all the aforementioned challenges. Client software of the mobile cloud embracing this experience observes the status and condition of the user and device, and intelligently caters its execution to the situation. Such an approach can foster better mobile applications

running in the cloud, all optimised for the user experience, and benefiting from the device sensors for context awareness.

In this paper, we evaluate several of the current approaches to solving the problems in integrating the device with the mobile cloud with respect to the user experience. From this, we determine what desirable properties future work in this area should focus on. We then present our own conceptual approach which can solve these problems.

The remainder of the paper is organised as follows. Section II evaluates the current work for mobile cloud integration with respect to user experience. Section III outlines future requirements for the user experience, and introduces our proposed conceptual work for these requirements. In section IV, we evaluate our solution in respect of the requirements. New mobile cloud application and service scenarios enabled by our approach are presented in section V, followed by our conclusions in section VI.

II. STATE OF MOBILE CLOUD

The mobile cloud model will be largely adopted and therefore successful if mobile users will have a positive experience in all respects. Considering our research interest in the user experience, we examine in this section the current work in solving some of the outstanding difficulties of mobile cloud computing.

A. Cloudlets

Cloudlets are a form of minimal cloud computing infrastructure proposed by Satyanarayanan et al [2] that is targeted at solving the latency and bandwidth concerns for mobile cloud applications. Cloudlets are deployed in an area near where mobile users gather such as a coffee shop or an airport. The user is then only one Wi-Fi hop away from the Cloudlet, thereby reducing the latency. The Wi-Fi network should have higher bandwidth than a WAN. Latency is an important factor for the user experience, specifically perception [21]. Cloudlets run a base virtual machine, which contains the core functionality of a desktop operating system like Windows. It is combined with an overlay VM, located on the user's mobile device. This contains the user's personal applications and settings. When combined, the full operating system can be used. Latency sensitive functionality can run on the Cloudlet, while other tasks are sent to the remote cloud infrastructure.

Cloudlets may not be appropriate for the user experience. Desktop operating systems are not designed with the mobile user in mind. Input and output required may be difficult to handle, such as displaying the visual elements on the small screen, and providing touch input from large fingers to small point and click elements. Such operating systems are not optimised to receive the unique input afforded by mobile devices, such as context data from the sensors. The time required to actually move the VM-overlay to the Cloudlet and synthesize with the base VM is not negligible. The overhead of this approach may be too demanding for simple tasks. A study by Clinch et al [3] showed that latency for less demanding applications did not adversely affect performance to a large degree. We believe that bandwidth will in fact be a concern as the number of users of the Cloudlet grows.

B. Application Partitioning

An application is partitioned into distributable components. These components are then deployed onto various nearby devices, where they execute. The distribution is decided with the aim of application performance meeting certain requirements.

An example of such an approach is Alfredo [4]. The Alfredo system uses a graph representation of an application's components to find a cut to maximize or minimize an objective function. Based on the cut, some components are then distributed in the cloud. Another example of this approach is the Dynamic Cloudlet by Verbelen et al [5]. They modify Satyanarans Cloudlet concept, so that a Cloudlet is made up of a collection of nearby computing devices. Their software splits an application up into components on a constraint basis. The constraints specify that execution of a component must have completed by a certain time. Distribution is based on the probability that constraints will be met.

The authors of Alfredo [4] suggest that components that make up the user interface be left on the mobile device, while the other computationally intensive components are distributed. The role of the device is a viewer, to simply display output on the screen, and take input from touch events to interact with the interface. As with Cloudlets, applications built for such an approach cannot utilise any context information. A partition of components may optimise one objective function, and render another sub-optimised; for example, it maximises throughput, but may increase energy usage. For the user experience, all objectives need to be optimised to as large an extent as possible. They also found the deployment time not negligible. For the Dynamic Cloudlet implementation, constraints were often not met.

C. Remote Display Using Thin Protocols

Remote viewing approaches are similar in some sense to Alfredo. Using a thin client protocol, the mobile device is used as a thin client viewer. It displays the visual output of virtual machines running on cloud servers. Users then use the device to run and interact with desktop applications running on the cloud, as described by Simoens et al [6]. An example implementation of the concept is described by Deboosere et al [7].

This concept suffers from the same drawbacks as Cloudlets and partitioning approaches. As described in

[6], the current state of remote viewing is not optimal for mobile devices. Authors point to solutions for some of these problems, such as caching, buffering, and NIC wake/sleep intervals [8-9].

D. Code Offload

Computation heavy and resource intensive application code is offloaded to the cloud infrastructure, executed there, with the results then returned to the device. Mobile device energy is not used up by such code, and the greater computation power on the cloud could execute the code faster. This requires a consistent connection to the cloud infrastructure. Typically, an optimization solver makes the decision for code to be either executed locally on the mobile device, or remotely on the cloud infrastructure. Examples of code offload solutions include MAUI [10], CloneCloud [11], MACS [12], and exCloud [13].

Applications must first be profiled to discover what code is suitable for offload. These approaches work on user code at the virtual machine level. Some code, such as native code and code that accesses resources on the device, cannot be offloaded. In addition, code should only be offloaded if it actually results in some time or energy gain. With variable quality network connections, it may take more energy or time to offload the code than to execute it locally, therefore profiling of the state of the network also occurs. Both profiles are given as inputs to the optimization solver to make the decision. When offload occurs, all object state that the offloaded code accesses must also be serialized and offloaded, which can be costly.

Profiling has an adverse effect on the user experience, requiring time and energy. Code offload also depends on the quality of the network. If the network connection drops while code has been offloaded, then the code that was being executed remotely must now be re-executed on the device, wasting time and resources (MAUI). Network quality on 3G networks tends to be lower due to high latency and low bandwidth. For MAUI, the optimization solver always refused to offload on a 3G network. Users constantly move and change between networks of differing quality. Applications that could not run on the device due to resource constraints have been shown to be executable with code offload in tests for MAUI, but the user experience still suffers.

E. Cloud-Based Middleware

Middleware approaches can bring advantages for the user experience. Many companies provide cloud-based services accessed by different APIs. These libraries are incompatible with each other. Developers are facing the difficulty of programming applications with each API individually. Users may want to use cloud services from different companies for different tasks. For example, users may wish to store photos on a cloud storage service such as Amazon S3, link them onto a social network such as Facebook, and apply some facial recognition from Face.com. This scenario is recognized by Flores et al [14]. The authors developed a cloud-based middleware, which can be used to access different cloud provider APIs with ease. Requests for a service go to the middleware, and an appropriate adapter is substituted for each request to each cloud provider respectively. The middleware waits for the response, freeing up the device to carry out other tasks. When the middleware receives the result from the service, it will relay it back to the device.

The holding of the result in the middleware until the device is ready to receive it will ensure the result is safe if the device is disconnected. The APIs for cloud providers are often in the form of web services such as SOAP or REST that can take context information from the device.

Another cloud-based middleware is that of a web-service mash-up approach by Wang & Deters [15]. Several web service APIs provide services to clients, such as SOAP based services, normally XML based. It is not optimal for mobile devices. REST based services can use the efficient JSON format, resulting in reduced data transfer and lowering charges for the user. This can be parsed faster on the device. The middleware takes REST based requests to access web services, converting the request to SOAP if required. It invokes the service, saves the result, and returns it to the device using REST. This approach can also use context information from the device, and store the result until the device is ready to receive it. The implementation requires the user to input the WSDL URL, and to indicate if it is a SOAP or REST service they wish to call [16]. With a broad range of people, most users will not know what WSDL, SOAP and REST are.

F. Context Awareness

Various projects have been undertaken to bring context awareness to the mobile device. They often involve middleware to collect, process, categorise, and infer the context, before taking appropriate action. Chen & Kotz [1] define two types of context aware application, active and passive. Active applications actively alter their behaviour based on the context, passive applications may or may not provide some additional features from context awareness. Gu et al [17] designed a middleware approach that could take context data from sensors, process it, and provide it to context consuming applications. It uses a rules based system to infer context, which is represented as ontologies. Work by Hong et al [18] aimed to build on this approach by using context history, enhancing the quality of context. Beach et al [19] uses social networking sites to collect data for context awareness, and provides enhanced context to friends with similar interests found on the social networks.

Context awareness allows applications to provide a better user experience. The ability of context to be shared among others or inferred from history will be

useful in the event that new context is difficult to gather. Only the middleware approaches we described can use context, depending on the web service.

III. FUTURE APPROACHES TO MOBILE CLOUD INTEGRATION

The evaluation of the current results of mobile cloud integration shows that user experience is not the primary goal. In particular, each design focuses on one or a few aspects that are important to the user experience. Some of these solutions may contradict others. Aside from the user experience perspective, several of the works highlight challenges that will not be easy to resolve, such as the time and energy cost of profiling, offloading, and distributing of code and application components in the partition and code offload approaches. The Cloudlet approach is very dependant on the users position relative to the Cloudlet, along with the network state, and current technologies used in VM synthesis. In order to be adopted on a large scale, any future approach to mobile cloud computing has to be centered on the user experience. As such, a given integration solution will have to overcome all of the problems that were discussed in the previous section, not just one or a few, otherwise the user experience cannot be optimal.

Our approach to realising this vision and solving these problems focuses on the cloud carrying out more intelligent work for the mobile user. By using more cloud based applications, services and computing resources, rather than mobile based, we can relieve the device of the burden of having to carry out extra work as in the previous projects. More applications and services are now developed and deployed in the cloud, taking advantage of the resources available.

A. Requirements for User Experience

For the mobile user, the question is how to achieve this goal, while optimizing the experience. By analysing the previous research work, we can identify a set of core requirements for a user experience oriented design.

1. *The approach has to address the latency between the device and the cloud infrastructure*

As mobile devices are used more for applications and services that require real time data and actions, the latency between the cloud and the mobile device must be minimal. In the event that minimal latency cannot be guaranteed, approaches must be optimised and lightweight.

2. *The approach has to minimise bandwidth utilisation*

The time and energy cost over networks such as cellular 3G, is typically high due to the low bandwidth. Furthermore, cellular network operators often charge per kilobyte of data transferred. The rate is even higher if the user is roaming. Therefore the utilisation of unnecessary bandwidth must be minimised, especially if the user regularly carries out tasks such as uploading of photos to social networks.

3. *Device workload overhead must be minimized*

The actual workload the device carries out to enable such operations must also be minimal. An example is the profiling activities we have seen in other approaches. The energy drain required by such tasks runs contrary to the idea of the user experience.

4. *The approach must gracefully handle mobility aspects such as disconnection*

As the user is mobile, they will switch between networks of differing quality, and will at times be disconnected. Applications and services used must be able to handle such occurrences, by safeguarding any tasks the user was working on and associated data.

5. *Provisioning for context awareness must play a central role*

The unique ability of mobile devices to provide information about the users context such as physical, social, work and environment will be central to the development of future mobile applications and services, as their operation can be personalised to provide a greater user experience.

6. *The solution must uniquely cater for the mobile user, rather than the desktop user*

Existing applications and services may not be optimised for the mobile user. They provide services that are costly to the user on a mobile network, and may be costly for the device to work on. They are designed with the desktop computer in mind. New approaches must consider the requirements of the mobile user, and the different input and output capabilities mobile devices provide to the user.

7. *The thin client on the mobile must provide an adaptive UI and services*

The thin client of the mobile cloud must provide seamless access to applications and services on the cloud through a UI that adapts to the current state of the mobile and user. The client may also provide services to other local applications through an API (see section V).

8. *A standards-based solution must be used*

Currently there are many different kinds of applications and services that can be used, all heterogeneous. They differ in regards to input, output and APIs. A common interface adhering to a standard

defining how to describe different types of services and their utilisation must be developed and implemented in an automatic discovery service solution.

B. Context Aware Mobile Cloud Services

Our approach is a cloud-based middleware system that is deployed in the cloud, sitting between the mobile user and cloud applications and services. We call it the Context Aware Mobile Cloud Services (CAMCS) middleware system - see figure 1. Its main goal is to provide discovery and access to cloud based applications and services for the mobile user and applications. Cloud services include third party applications such as e-commerce, entertainment, productivity, and services that enhance the device capabilities by providing access to cloud based resources like storage. The middleware has several features that specifically cater for enhancing the user experience with respect to the requirements we have outlined. Specifically, the middleware consists of several key components:

- *The Cloud Personal Assistant*

The cloud personal assistant (CPA) [23] is a service, which runs in the cloud and can carry out tasks asynchronously for the user on his/her behalf. The assistant can discover and invoke services to carry out these tasks, get a result, and save it in the event the user has become disconnected. When subscribing, each user of the middleware receives his/her own CPA. An authentication and authorisation mechanism is also provided. The existing implementation will be extended to move between different cloud infrastructures.

- *Task History*

The CPA has access to task and service history that has been utilised in the past, which it can use to automatically make intelligent decisions about future actions it can carry out on the users behalf, with minimal intervention required from the user. History data shows preferred services and providers, what inputs are typically used, and what times invocation

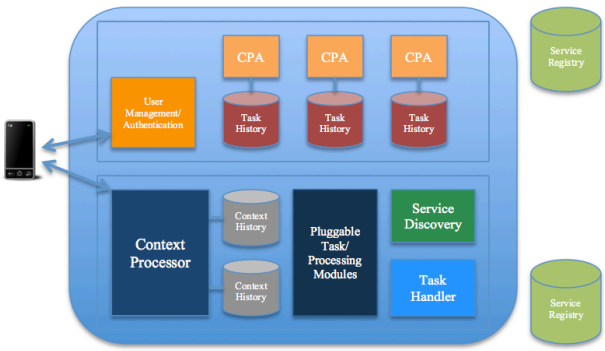

Fig. 1 The main components of the Context Aware Mobile Cloud Services (CAMCS) middleware system.

typically occurs at, enabling independent action when disconnected from the user; therefore it can either act alone in response to a specific event or time, guided by the history, or the user can signal it to execute a task with minimal data transfer from the thin client.

- *The Context Processor*

The context processor is another cloud service that handles context aware information from the device. Based on user preferences or other factors such as current network quality, it can choose when to request new context data from the device. It can locally infer context from these resources of information and cater for the responsibility of passing context data to the applications and services that require it.

- *Context History*

The history of inferred context can contribute to a new context as it is gathered. This is also useful if for some reason new or complete context cannot be determined. It can also be used to recommend information to a user if they have not indicated they should be alerted to new but relevant context, thereby enhancing their experience of applications and services.

- *Discovery Service*

Our middleware incorporates a distributed discovery service, which can be used by the CPA to discover cloud applications and services on behalf of the user. The discovery service uses a distributed registry of standards based cloud applications and services, which can provide efficient discovery, along with service negotiation based on user preferences.

- *Pluggable Modules*

Plug-ins can be developed and added into the middleware to further enhance operation with custom functionality. More information is given in section V.

IV. EVALUATION

Our proposed middleware can address the requirements that were set, and therefore provides a comprehensive solution for many of the problems outlined in previous projects, while placing user experience as the primary motivation behind the design and approach decisions - see table 1.

The movement of the CPA can address latency by moving between cloud infrastructures so that it is closest to where the user is situated, and it can handle disconnections by storing results for the user. Bandwidth utilisation is low, as the user only needs to signal the CPA to carry out work and pass required information. The CPA in the cloud carries out all the complex work of discovering and invoking services,

Table 1: A summary of the problems that the various approaches we have examined can solve, including the problems our own approach can solve. ✓ indicates an approach solves a problem, * indicates dependence on the individual services.

	Cloudlets	Partitioning	Code Offload	Remote Display	Other Middleware	CAMCS
Latency	✓	✓	✓			✓
Bandwidth			✓		N/A	✓
Disconnection					✓	✓
Energy Aware		✓	✓	✓		✓
Context Aware					*	✓

freeing the mobile device from continuous data transfer, and freeing it up to run other tasks. This is made easier by the standards based operation of the distributed service registry.

To minimize the workload on mobiles, a thin client application can be used to communicate with the middleware. In addition, the added benefit of such an approach is that applications and services are located in the cloud. Resource intensive applications are not executed on the device anymore, which reduces its workload and in turn will benefit from time and energy savings. The user does not have to spend the time searching for and downloading specific applications for specific services, and removes the need for users to leave an application running and undisturbed while waiting for a task to finish.

The ability of the middleware to include context awareness processing will enhance the user experience. Indeed, it can intelligently gather various new contexts in suitable conditions, and infer context from task and context history, all the while taking advantage of the cloud resources such as datastores and processing power. In addition, the middleware specifically provides services to cater to the mobile device, such as optimised request/response data formats, communication architectures with REST, and utilisation of cloud infrastructure to supplement the operation and features of the mobile device.

Consider the following example to demonstrate the operation of the middleware and the interactions between components. An employee of a goods supplier company is making a site visit to a client's store. The client wishes to stock up on sold out goods, but is also interested in selling new ranges of products from the supplier. Having discussed the details with the client, the employee uses his/her mobile device, which runs a thin client to communicate with the middleware, to instruct his CPA to carry out the required work.

The thin client sends a signal of small data size to the CPA for this task, which has been undertaken before. Also sent with the request are the details of the new product ranges the client will stock, which were provided by the CPA previously to show the client. Once the request has reached the cloud, the employee is authenticated to their CPA by the user management/authentication module. The CPA consults the context processor to discover the latest location information. The context processor knows the location of the user and can infer what client they are with, and from the context history, knows why they are there (re-supply). The task history of the CPA is then consulted to get the details of the previous orders the employee has placed for this client. This information, along with details of the new products, is passed to the task handler. Here, the CPA is in the domain of the company, and knows how to instruct the task handler to call the service to put in an order for a re-supply. If it is unaware of the correct company service, the discovery service module could consult a company specific service registry, accessible within the company domain.

The service is invoked, with details of the order stored with the CPA in the task history. The task history is updated with the new products the client will be stocking for future reference. The CPA then notifies the employee on their mobile device of a successful order placement. This can even contain a price for the customer. The calculation of the price and placing of the order may have required some time due to having to consult multiple database tables for product information, and updating client records. The asynchronous approach of the CPA in the middleware frees the mobile device from having to wait for these operations to complete, wasting time and energy.

Several challenges must be overcome in developing the CAMCS middleware:

- The gathering and inferring of context is a complex process not well defined.
- No standards currently exist for applications and services at the cloud level. Work is being carried out in standardisation in the form of the Open Cloud Computing Interface (OCCI), which has an aim of providing standards for cloud services at the IaaS, PaaS, and SaaS levels.
- Security and privacy concerns, unaddressed in the other approaches evaluated, will need to be addressed aside from the basic user authentication and authorisation proposed by this middleware, to protect user and data privacy and security. This is something that will be addressed in a future work.

V. NEW MOBILE CLOUD USAGE SCENARIOS AND ENABLED FEATURES

Several new kinds of applications and services can be enabled by our middleware approach to mobile cloud integration for the user experience.

A. *Real-Time Context Aware Services*

Real-time services, which can respond to the users current and inferred context, possibly from context history, can be enabled by CAMCS. Such applications and services can actively or passively recommend activities or a course of action to users based on context like location, such as reminders or information about tasks they have to carry out such as the purchase of items from a shopping list, or if in an unfamiliar location, can suggest places of interest based on past activities. In the case of the shopping list reminder scenario, the CPA could actively check store stock for the user, and if unavailable, direct to a new store nearby that has the desired item(s) in stock, without user intervention. As another example, based on the current time of day or even year, such as during the day during school-term, it can notify parents to a routing of least traffic congestion to a school. This would stop occurring out of school term. The CPA can actively consult with traffic services, and perhaps could even make the information available to other applications or devices in the car such as a SatNav.

- *Collaborative Services*

Each user of CAMCS has a corresponding CPA in the cloud for their mobile device. Each will have invoked and discovered different services. In a vision of the next generation of mobile applications by Sankaranarayanan et al [20], they propose collaborative applications and services that share information to provide a seamless user experience, rather than the current generation of applications and services that work in isolation. An example the authors used was a traveller attending a conference. A calendar application with details of the conference such as location and time/date, could share this information automatically with airline and hotel services to find the best flights and accommodation to a location near the conference venue. CAMCS could enable such an approach, if the CPA can share details of discovered services with each other. This could be further enhanced if the CPA's users who share some relationship such as friends or work colleagues are made known to each other, in which case they share similar interests and context. Any collaboration scenario would be subject to strict privacy control.

- *Additional CPA Services*

The CPA concept can be expanded so that the CPA can carry out its own work, rather than simply looking up and invoking services. The functionality of the CPA can be extended to carry out other tasks for the user, such as data processing and synchronisation with multiple services that the user is interested in, such as the synchronisation of data among social networks or sharing of files in a workplace domain.

- *Services for Environmental Context*

In this age where the "internet of things" is of growing interest, the CPA could be extended to either read state from or give instructions to objects in the environment either on behalf of the user over the network, or receive this information through the mobile device, which in turn can receive this information from its own sensors or over a local network such as Bluetooth.

- *Extensible Middleware*

The CAMCS functionality can be extended by adding in components on a plug-in basis to carry out other tasks on behalf of the mobile user, possibly utilising cloud based services, and taking advantage of the cloud infrastructure. Such functionality could include support for computationally intensive batch processing jobs, or a client for long running queries on databases within a workplace or scientific domain.

- *Public API for Developers*

The thin client that will run on the mobile device can present an interface to developers to enhance their applications to easily use services provided by the mobile cloud. A minimal interface can be published as follows:

```
public void findService(Details details);

public void executeTask(TaskInfo taskInfo);

public void updateContext(Context context);

public Context getContext();

public void enterDomain(Context context);

public void store(File[] files);

public void sync(Details details);
```

The interface can be used to access cloud services, and applications can use/adapt behaviour due to the current context. The CPA can execute such work on behalf of the client applications asynchronously. Methods are also provided to allow applications to store data on cloud-based storage, and to synchronise details among different cloud networks. We foresee such actions could even be built into the operating system functionality of the device so native operations and stock applications can benefit as well.

- *Multi-Location Deployments*

The ability of the CPA to move between various cloud infrastructures such that it is as close to the user's location as possible has already been mentioned. We

believe that public and private cloud providers who offer PaaS and SaaS services can deploy CAMCS into their service offerings to fully enable this functionality and compatibility for the user as they move.

VI. CONCLUSIONS

In this paper we evaluated the state of several current approaches to providing integration between the cloud and mobile devices, with respect to the user experience. This is particularly relevant considering the ever-increasing penetration rate of mobile devices as well as their users' expectations in terms of applications and services.

We determined that these approaches do not place the user experience at the centre of design and implementation, often only focusing on few individual aspects of mobile cloud integration such as latency or energy savings. Any future approaches will need to tackle several of the issues that are dominating the current approaches. After determining a set of requirements for future work in this area, we introduced our middleware solution to this problem, CAMCS, which can meet this set of requirements. We evaluated the approach and determined new scenarios that this middleware could enable. As the CPA is already implemented and successfully tested in Amazon EC2, our future work will involve implementing the other system components introduced in this paper, including the context processor, the task and context history, the service discovery and negotiation mechanisms, and the thin client mobile application. Subsequently, we will experiment with its user experience oriented functionality and performance.

ACKNOWLEDGMENT

The PhD research of Michael J. O'Sullivan is funded by the Embark Initiative of the Irish Research Council.

REFERENCES

[1] G. Chen and D. Kotz, "A survey of context-aware mobile computing research", Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, 2000.

[2] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The Case for VM-Based Cloudlets in Mobile Computing", IEEE Pervasive Computing, 2009; 8(4), pp. 14-23.

[3] S. Clinch, J. Harkes, A. Friday, N. Davies and M. Satyanarayanan, "How close is close enough? Understanding the role of cloudlets in supporting display appropriation by mobile users", 2012 IEEE International Conference on Pervasive Computing and Communications (PerCom), 2012, pp. 19-23.

[4] I. Giurgiu, O. Riva, D. Juric, I. Krivulev, and G. Alonso, "Calling the cloud: Enabling mobile phones as interfaces to cloud applications", Middleware 2009, pp. 83-102.

[5] T. Verbelen, P. Simoen, F. De Turck and B. Dhoedt, "Cloudlets: bringing the cloud to the mobile user", Proceedings of the third ACM workshop on Mobile cloud computing and services, Low Wood Bay, Lake District, UK. 2307858, ACM, 2012, pp. 29-36.

[6] P. Simoens, F. De Turck, B. Dhoedt, and P. Demeester, "Remote Display Solutions for Mobile Cloud Computing", Computer, 2011, 44(8), pp. 46-53.

[7] L. Deboosere, B. Vankeirsbilck, P. Simoens, F. De Turck, B. Dhoedt, and P. Demeester, "Cloud-Based Desktop Services for Thin Clients", IEEE Internet Computing, 2012, 16(6), pp. 60-67.

[8] B. Vankeirsbilck, P. Simoens, J. De Wachter, L. Deboosere, F. De Turck, B. Dhoedt, et al, "Bandwidth Optimization for Mobile Thin Client Computing through Graphical Update Caching", Telecommunication Networks and Applications Conference, ATNAC 2008 Australasian, pp. 385-390.

[9] P. Simoens, P. Praet, B. Vankeirsbilck, J. De Wachter, L. Deboosere, F. De Turck, et al, "Design and implementation of a hybrid remote display protocol to optimize multimedia experience on thin client devices", Telecommunication Networks and Applications Conference, ATNAC 2008 Australasian, pp. 391-396.

[10] E. Cuervo, A. Balasubramanian, D.-K. Cho, A. Wolman, S. Saroiu, R. Chandra, et al. "MAUI: making smartphones last longer with code offload", Proceedings of the 8th international conference on Mobile systems, applications, and services, San Francisco, California, USA. 1814441, ACM, 2010, pp. 49-62.

[11] B-.G. Chun, S. Ihm, P. Maniatis, M. Naik and A. Patti, "CloneCloud: elastic execution between mobile device and cloud", Proceedings of the sixth conference on Computer systems, Salzburg, Austria, 1966473, ACM, 2011, pp. 301-314.

[12] D. Kovachev, Y. Tian and R. Klamma, "Adaptive Computation Offloading from Mobile Devices into the Cloud", 2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications (ISPA), 2012, pp. 784-791.

[13] R. K. K. Ma, L. King Tin and W. Cho-Li, "eXCloud: Transparent runtime support for scaling mobile applications in cloud", 2011 International Conference on Cloud and Service Computing (CSC), 2011, pp. 103-110.

[14] H. Flores, S. N. Srirama and C. Paniagua, "A generic middleware framework for handling process intensive hybrid cloud services from mobiles", Proceedings of the 9th International Conference on Advances in Mobile Computing and Multimedia, Ho Chi Minh City, Vietnam, 2095715, ACM, 2011, pp. 87-94.

[15] Q. Wang and R. Deters, "SOA's Last Mile-Connecting Smartphones to the Service Cloud", Proceedings of the 2009 IEEE International Conference on Cloud Computing, 1632969, IEEE Computer Society, 2009, pp. 80-87.

[16] Q. Wang, "Mobile Cloud Computing", MSc Thesis, Department of Computer Science, University of Saskatchewan, 2011.

[17] T. Gu, H. K. Pung and D. Q. Zhang, "A service-oriented middleware for building context-aware services", Journal of Network and Computer Applications, 2005, 1, 28(1), pp. 1-18.

[18] J. Hong, E.-H. Suh, J. Kim and S. Kim, "Context-aware system for proactive personalized service based on context history", Expert Systems with Applications. 2009, 5, 36(4), pp. 7448-7457.

[19] A. Beach, M. Gartrell, X. Xing, R. Han, Q. Lv, S. Mishra, et al. "Fusing mobile, sensor, and social data to fully enable context-aware computing", Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications, Annapolis, Maryland, 1734599, ACM, 2010, pp. 60-65.

[20] J. Sankaranarayanan, H. Hacigumus, and J. Tatemura, "COSMOS: A Platform for Seamless Mobile Services in the Cloud", 12th IEEE International Conference on Mobile Data Management (MDM), 2011, pp. 303-312.

[21] N. Tolia, D. G. Andersen, and M. Satyanarayanan, "Quantifying interactive user experience on thin clients", Computer, 2006, 39(3), pp. 46-52.

[22] R. Buyya, C.S. Yeo, S. Venugopal, J. Broberg and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility", Future Generation Computer Systems, 2009, 6, 25(6), pp. 599-616.

[23] M. J. O'Sullivan, D. Grigoras. The Cloud Personal Assistant for Providing Services to Mobile Clients, IEEE MobileCloud, Redwood City, San Francisco Bay, USA, 2013, pp. 477-484.