Association for Information Systems
AIS Electronic Library (AISeL)

2012 International Conference on Mobile Business

International Conference on Mobile Business (ICMB)

2012

MOBILE COMPUTATION OFFLOADING - FACTORS AFFECTING TECHNOLOGY EVOLUTION

Antero Juntunen
Aalto University, antero.juntunen@aalto.fi

Matti Kemppainen
Aalto University, kemppi@cs.hut.fi

Sakari Luukkainen
Aalto University, sakari.luukkainen@aalto.fi

MOBILE COMPUTATION OFFLOADING - FACTORS AFFECTING TECHNOLOGY EVOLUTION

Juntunen, Antero, Aalto University, PO Box 15400, FI-00076 AALTO, Finland, antero.juntunen@aalto.fi

Kemppainen, Matti, Aalto University, PO Box 15400, FI-00076 AALTO, Finland, kemppi@cs.hut.fi

Luukkainen, Sakari, Aalto University, PO Box 15400, FI-00076 AALTO, Finland, sakari.luukkainen@aalto.fi

Abstract

Compared to desktop devices, mobile devices have inherent constraints such as limited processing power, memory, and battery capacity. With the proliferation of resource-hungry applications, researchers are looking for new solutions to address these limitations. One such solution is mobile cloud computing (MCC), which uses cloud infrastructure to enhance the capabilities of mobile devices. This paper focuses on a related, emerging technology called mobile computation offloading (MCO), where the emphasis is on dynamically offloading computation from native applications running on mobile devices to outside surrogates such as cloud infrastructure. We use an exploratory approach to evaluate the business potential of MCO by identifying critical factors that influence the technology evolution of MCO. We base this evaluation on a literature review of MCO and utilize a research framework derived from the existing literature on technology evolution and MCO.

Keywords: mobile offloading, mobile computation offloading, mobile cloud computing, technology evolution

1 Introduction

Smartphones or high-end mobile phones with advanced computing capabilities and features have increasingly dominated the mobile phone landscape in recent years. In the second quarter of 2011, smartphone shipments exceeded the shipments of feature phones (normal phones) for the first time in Western Europe (IDC, 2011) and the yearly gain in worldwide smartphone sales was 47 percent from 2010 to 2011 (Gartner, 2012). The processor speeds as well as memory and display capabilities of mobile devices have rapidly increased in the last decade, allowing the devices to run more computationally demanding applications. This increase in technical capabilities has coincided with a proliferation of applications available on these devices. The increased availability of applications has been largely due to the rise of application stores that have simplified the process of finding and installing the applications for the end users, increasing the demand for mobile applications. In addition, a more open policy by device manufacturers has allowed small developers and hobbyists to develop and publish their applications for mobile devices, thus increasing the supply of mobile applications.

However, despite the advances in smartphone capabilities, mobile handsets fall short of desktop computers in the types of applications they can run. Compared to desktop computers, mobile devices possess less computing power, memory, storage capacity, and network bandwidth. These limitations are especially acute for resource-hungry applications such as video streaming and mobile games, which contributes to device manufacturers seeking to continually improve the capabilities of mobile devices. Moreover, one aspect of mobile devices that has not improved on pace with their processor speed and memory size is battery power. Although battery technology has continuously developed, this development has been eclipsed by increased battery capacity demands posed by power-hungry applications, display devices, and sensors. As a result, the gap between the demand and supply for power in mobile devices has been increasing, and will continue to increase in the foreseeable future barring unforeseen breakthroughs (Bakhshi, 2009).

These two challenges facing mobile devices today – limited computational resources and battery power – can be partially addressed by enhancing the capabilities of mobile devices with cloud computing infrastructure. We define this trend of mobile cloud computing (MCC) as using cloud-computing principles to deliver applications and services for mobile devices. MCC can also limit the negative effect of mobile operating system fragmentation with the help of browser-based applications and open APIs (Juntunen, Suikkola, Raivio and Luukkainen, 2011). Thus, MCC can be seen as a potential technological discontinuity in the mobile technology trajectory, moving intelligence from mobile devices into the cloud and offering tangible benefits for both end users and application providers.

This paper focuses on what could be considered a subset of mobile cloud computing – mobile computation offloading (MCO). For the purposes of this paper, we define MCO as concerning purely offloading computation into surrogate devices such as cloud servers, typically aiming to enhance the computational capacity of the mobile device or save the battery power of the device. Thus, content offloading – offloading user data and other content into the cloud – is outside the scope of MCO, except when it is needed to perform computation. Although there are existing solutions such as Assisted GPS (A-GPS) that offload computation from a mobile device, these solutions are tied to specific hardware such as GPS receivers. In our definition, MCO focuses more generally on native applications, the processing of which is dynamically executed either in surrogates or on the mobile device. MCO can be seen as a potentially disruptive enabling technology, which could shift the focus from increasing computing capabilities to a new performance dimension (Christensen, 1997).

Because MCO is an emerging technology, we use exploratory research to depict the phenomenon in more detail. Thus, our research goal is to identify critical factors that affect the technology evolution of MCO. We base our work on a review of recent literature on MCO, which is at a basic research state and has concentrated on different technical solutions (for example; Chun et al., 2011; Cuervo et al., 2010; Kemp, Palmer, Kielmann and Bal, 2010). We analyze this literature on MCO using our own research framework as a research focus and base the framework on existing literature of technology evolution and MCO. Section 2 describes the framework and its theoretical background as well as the previous literature on MCO. We analyze MCO using this framework in section 3, summarize and discuss the results in section 4, and give our conclusions in section 5.

2 Theoretical Background

2.1 Technology evolution

Technologies evolve through periods of incremental change punctuated by major technological advances (Tushman and Anderson, 1986). Tushman and Anderson make a distinction between the incremental improvements of existing technologies and the technological discontinuities that break incremental improvements. Anderson and Tushman (1997) later defined the technological discontinuities as innovations that advance the technological state-of-art of the industry by an order of magnitude. Furthermore, technological discontinuities are based on new technologies with smaller technical limitations than those of the previous dominant technology.

Industries evolve through a sequential development of technology cycles. These cycles are initiated by technological discontinuities that emerge through scientific advance or through a unique convergence of existing complementary technologies, which eventually substitutes the existing products (Anderson and Tushman, 1990). At some point, diminishing returns begin to surface as the technologies begin to reach their limits and new, substitute technologies start to emerge (Anderson and Tushman, 1997). The threat of substitute products depends on a number of factors, including relative price, new features and added value, performance, and switching costs (Porter, 1985).

The success of many new entrants has lead to coining a phenomenon called the “attackers’ advantage”. This term refers to those new entrants who are better than the incumbents in developing and commercializing emerging technologies because of the smaller size of the new entrants, limited path-dependent history and no commitment to the value networks of the previous technology (Foster, 1986; Christensen and Rosenbloom, 1995). New entrants can also be successful despite the incumbents’ greater resources and experience with the existing technology. However, industries have barriers to entry, which protect the existing profit levels of the incumbents and hinder new entrants from entering the market. Barriers to entry are unique to each industry and include factors such as cost advantage, economies of scale, brand identity, switching costs, capital requirements, learning curve, regulation, access to inputs or distribution, and proprietary products (Porter, 1985).

Christensen (1997) states that the incumbents improve their technological performance on an existing trajectory and finally exceed even the most demanding customers’ needs. Simultaneously, new, more cost-effective technologies are developed by new entrants first for the needs of the customers of other industries. These new technologies start to increase their market share among less-demanding customer segments. These technologies that were originally ignored by the incumbents because of their small market penetration will later enter the existing mainstream market. Christensen refers to these technologies and the related innovations as ‘disruptive’, which is an extension to the concept of technological discontinuity previously discussed in this section. Similarly, disruptive innovations significantly change the current market structures, customer usage patterns, and value propositions. If the markets of disruptive technologies develop fast, new entrants gain advantages due to economies of scale. If the development is slower, the incumbents will have more time to react.

Rogers (2003) considers the most important factor affecting innovation diffusion to be the relative advantage (price and performance) over competing technology substitutes, which changes over time. The second factor affecting the diffusion is the compatibility with the values, norms, and experience of the end-users. Innovations that are connected to older, already proven technology, have an advantage. Complexity then hinders this diffusion, because the end users have difficulty using and understanding the product. Trialability relates to the extent to which the product can be experimented with a low risk. Easy trialability for the early adopters enhances the diffusion. This is also supported by Gaynor (2003), who emphasizes the importance of experimentation, especially in times of great market uncertainty. The last factor of Rogers' model is the observability of innovation, which means the visibility among the user community. In addition to the factors directly related to the product, other issues also affect the diffusion, such as the characteristics of the implementation decision.

Although security considerations are relevant for all computer systems, mobile communications and network-enabled services can be especially vulnerable, and security should be carefully considered in developing mobile applications (Siau and Shen, 2003). Moreover, cloud computing poses challenges concerning also privacy, and trust (Robinson et al., 2011). Thus, all these three aspects are relevant when considering a technology such as mobile computation offloading that combines both mobile applications and cloud computing.

2.2 Mobile computation offloading

In the 1990s, visions of ubiquitous computing described technologies that would bring human-computer interaction to a completely new level. They were to integrate to the daily lives of ordinary people indistinguishably, providing means for augmented reality and many other things considered only a dream back then. As mobile computing has taken enormous evolutionary steps over the last two decades, these ideas have gotten a platform on which to evolve. Furthermore, thanks to open APIs and online application markets for mobile devices, designing mobile applications is nowadays the playground not only for seasoned professionals working with the device vendors but also for independent entrepreneurs and hobbyists. Mobile application sphere is nowadays extremely active and exploits the ever-increasing resources and contextual information of the mobile devices such as smart phones and tablets in numerous creative ways.

The foremost obstacle for bringing the ideas of ubiquitous computing into reality has traditionally been the poorness of mobile resources. Specifically, there is a notable disparity in technological advances between battery life and compute power. Since battery is the component that enables the luxury of mobility in the first place, we obviously need to optimize energy usage in order to prevent mobile devices becoming stationary due to our resource-hungry applications. Satyanarayanan (2001) proposed augmentation of the mobile resources by external ones as the solution, calling this cyber foraging. While the paper left many questions open, later research has supported the idea even on the modern mainstream mobile devices (Kemp et al., 2010; Cuervo et al., 2010; Chun et al., 2011; Zhang, Jeong, Kunjithapatham and Gibbs, 2010).

More recently, mobile computation offloading has established itself in the terminology, as the research has focused into smaller areas. MCO is an enabling technology that attempts to overcome the challenges of mobility by migrating parts of computation outside the mobile device. Offloading transfers the control data and the application state information over the network to a server machine called surrogate that will complete the computation task and send the results back to the mobile client. In contrast to web-based applications, MCO resembles a low-level distributed systems architecture such as remote procedure calls (RPC). As mobile network connectivity is often limited, offloading is an opportunistic operation for the application process, rendering it an optimization method. In other words, offloading should take place only when it is beneficial; that is, the costs of offloading are less than the performance gain.

Mobility by its nature requires live decisions for offloading, since the overall environment is usually undefined until it develops dynamically. Moreover, accuracy in the decision-making analysis determines the maximum level of overall system performance. Certain device-specific functionality is not migratable at all, such as low-level user interface handling, user input, or local file system access.

Mobile networking environment lays its own challenges, as well. First, modern mobile devices communicate through a number of different networking interfaces, such as WiFi, 3G/LTE and Bluetooth, all of which may be considered valid paths to the surrogate. A networking abstraction layer such as proposed by Kemp et al. (2010) would help the burden of an application developer. Second, considering universally accessible surrogates, we would like such an abstraction to cope with host mobility in terms of network topology. Third, network should be used in an optimal way, so as to minimize energy consumption.

There are a number of recently-developed offloading frameworks that can help the application developer. In fact, the frameworks discussed in this paper generally make a promise of simplicity to the application developers, yet all of them require at least some developer interaction. To maintain focus on the development of mobile applications, we specify three arbitrary levels at which offloading frameworks work: features, methods and system level. We emphasize that this categorization gives only hints about the actual implementation of the frameworks, and it is merely used for supporting the viewpoint of an application developer.

Feature offloading intercepts those parts of the code that a software developer has manually set up for offloading. The developer obviously has to be able to provide the framework with meaningful functionalities, which stands out especially in Elastic Application Model (Zhang et al., 2010) that selects the execution location for each application component. On the other hand, a framework called Cuckoo (Kemp et al., 2010) integrates into Android applications by creating a proxy inside the application for the interfaces that the application developer has defined. The proxy then decides whether to invoke its corresponding local method or to migrate the computation to the surrogate. If the application contains any UI or other high-priority tasks, the functionalities need to run in parallel with the rest of the process in order to reduce the severity of the downsides of mobility. Since manual labor defines the optimization baseline, this approach is also highly dependent on the domain expertise of the application developer. The application developer may also choose to provide different interface implementations for the client and the surrogate.

Rather than requiring a separate interface or component, method offloading uses per-method annotations and wraps methods directly for proxying. The most well-known of the frameworks that implements this ideology is MAUI (Cuervo et al., 2010). This approach is less intrusive from application developer's viewpoint in the sense that it does not conceptually require strict separation of offloadable code parts. On the other hand, the same application code runs on the surrogate as on the mobile device, and therefore the runtime environments need to be interoperable at least to the extent of the migrated method invocations. Compared to just reducing the offloading burden of an application developer, system-level or process image offloading promises to take the most of it. CloneCloud (Chun et al., 2011) uses a modified virtual machine implementation of Android to intercept running threads at byte-code level and to migrate them for distributed concurrency. To ensure appropriate thread synchronization, it uses special migration and re-integration points that are pointed in the code by means of static analysis. Migratability limits differ from other approaches, as the object abstraction is flattened into memory representation. Interestingly, CloneCloud also supports use of native function calls local to the system of execution, which may increase portability and efficiency of software even further. As a side effect in reducing burden to the application developer, image-level offloading frameworks are required to be more sophisticated.

Recent literature uses energy and computation time measurements as the way to observe the benefits of offloading. Kemp et al. (2010) modified an object recognition application, reporting a computation speed-up by a factor of 60, reducing battery consumption by a factor of 40 while delivering results at a better quality. In contrast to these very impressive numbers, the authors of MAUI give somewhat more moderate figures (Cuervo et al., 2010). They demonstrate energy savings of 27% for a video game and 47% for chess. The frame rate of video game effectively doubled when offloading over WiFi connection. The quality of the connection played a key role. As the round-trip time in the network increased, lesser consumption of energy and processing speed-up eventually turned into overhead in certain examples. Finally, Chun et al. (2011) report factors up to 20 both in speed-up and energy savings for the tested applications.

2.3 Research framework

Based on the above literature review on technology evolution, we created the following framework for the empirical part of this study. The most important factors affecting the technology evolution of mobile computation offloading are summarized in Table 1. The 'added value' category focuses on the value of the MCO service over existing solutions and focuses on the viewpoints of the main actors – end users and software companies. 'Ease of experimentation' concentrates on the ability of developers to create new MCO applications or services. The category 'complementary technologies' examines supporting technologies required by MCO. 'Incumbent role' focuses on the roles of major incumbent actors, including device manufacturers, mobile OS providers and mobile network operators. The chosen categories were considered especially useful for a new, emerging technology and the categories arose from both the literature on technology evolution and MCO.

Table 1. Research framework.

Dimension	Meaning
Added value	The relative advantage over existing technologies
Ease of experimentation	The threshold of end users or third parties (developers) to experiment with new services
Complementary technologies	The interdependence between complementary technologies
Incumbent role	The product strategy of existing players
Security, privacy, trust	The acceptable level of security and sufficient privacy for end user

3 Analysis

In this section, we apply the research framework of Table 1 to mobile computation offloading (MCO). Section 3.1 examines the added value provided by MCO, while section 3.2 evaluates the ease of experimentation with MCO from the point of view of third-party application developers. In section 3.3, we analyze the role of complementary technologies, and section 3.4 examines the role of incumbent actors such as software companies, mobile network operators, and handset manufacturers. Finally, section 3.5 evaluates the security and privacy considerations of MCO.

3.1 Added value

Offloading computation from mobile devices to surrogates such as cloud is a logical way to increase the limited computing power of the mobile devices. By utilizing the higher resources available at the surrogates, it is possible to bypass some of the resource limitations of mobile devices and enhance their performance. When offloaded, an application could offer enhanced functionality (a turn-based strategy game offers a better artificial intelligence) or better responsiveness (an image recognition algorithm runs faster). MCO may also enable developing applications that would not normally function on even the most advanced smartphones or tablets, but which utilize the memory and processing power available in cloud in order to run on mobile devices. This would allow providing mobile users with applications previously only available on fixed devices with higher resources. Furthermore, MCO may enable older mobile devices to run applications currently targeted at high-end devices.

Battery capacity has been a limiting factor in mobile devices for a long time and there are several different ways to address this limitation. Aside from innovations in the battery technology itself, mobile hardware manufacturers build processors capable of switching to energy-saving mode, operating systems are designed to be more energy efficient, and there are even ways to gather energy from outside sources such as vibrations, heat, light, and radio waves (Kumar and Lu, 2010). Offloading computation into cloud can help save energy on a mobile device because the energy-draining computation is performed outside the device. However, this decrease in computation energy consumption is balanced by the communication required to offload the computation, which in turn requires energy and depends on factors such as the amount of data transferred and the burstiness of the traffic. Thus, offloading is most suitable for applications or methods that require significant energy in processing but only limited energy in communication (Kumar and Lu, 2010; Miettinen and Nurminen, 2010).

When evaluating the added value of MCO it is important to consider the degree to which this value is perceived by the end users or observed by others (Rogers, 2003). The ability to utilize new applications or enhanced application functionality is directly observable, but more subtle benefits such as slightly reduced energy consumption or increased computational capacities may be harder to appreciate for an end user. Assuming offloading happens on a per-application basis, it may be difficult for the user to appreciate the energy savings gained by offloading any one application when compared to the energy consumption of the mobile devices as a whole. In addition, the value perceived by the end users is affected by the fees the users have to pay both for the MCO service and for the increased mobile network data traffic. This effect can be mitigated with flat rate pricing for mobile data and bundling the MCO service fee with other services.

3.2 Ease of experimentation

Although mobile offloading can benefit end users and software companies, the trialability or ease of experimentation of MCO depends on how easy it is for application developers to move to an offloading model. It is important to consider the effort required to both develop new applications that can utilize mobile offloading and to modify existing applications for mobile offloading. Balan, Gergle, Satyanarayanan, and Herbsleb (2007) have demonstrated that software modifications for offloading take only a minimal amount of time, taking domain expertise and appropriate partitioning of the software for granted. On the other hand, software not originally designed for offloading may be suboptimal in terms of partitioning that the offloading schemes generally require. Some current solutions, such as CloneCloud (Chun and Maniatis, 2009; Chun et al., 2011), aim to eliminate the need for developer involvement in MCO as completely as possible. This is especially important for low-margin, long-tail applications that cannot realistically be manually optimized for mobile offloading. Other solutions try to facilitate the development of offloadable software by integrating with existing development tools and automating parts of the development process (Kemp et al., 2010) or requiring the developers to engage in only higher-level (method level) partitioning (Cuervo et al., 2010). On a conceptual level, system-level offloading schemes (Chun et al., 2011) require the least input from the developer, feature offloading (Kemp et al., 2010) the most, with method offloading (Cuervo et al., 2010) residing between these extremes. Nevertheless, at least in the current state of MCO, developer involvement cannot be completely avoided at some part of the offloading process.

3.3 Complementary technologies

Mobile network technologies are very important to determining whether a given application or method is suitable for offloading from an energy consumption point of view. The most important factor in the energy consumption of a wireless modem is the amount of time the interface remains active (Miettinen and Nurminen, 2010). Thus, the more data that needs to be offloaded, the more costly it is in terms of energy consumption. In addition, a smooth traffic pattern can consume more energy than bursty traffic, as the wireless modem remains active for longer. Furthermore, minimization of communication between the client and the surrogate requires efficient observation methods for the active state of the application and the related changes to the state (Cuervo et al., 2010). On the other hand, high bandwidth can alleviate the energy cost of communication, as this allows transferring larger amounts of data in a shorter period of time, reducing the amount of time the wireless interface remains active.

In addition to bandwidth, latency is another significant characteristic of a wireless network. As the latency of the network increases, the interactivity of an offloaded application is negatively impacted. The user may not notice the effect of latency in some offloaded applications or tasks such as web browsing, but immersive applications requiring fast response times can become noticeably sluggish. In addition, even a small latency can cause a considerable drop in frame rate compared to a thick client (Satyanarayanan, Bahl, Caceres and Davies, 2009).

The potential of mobile offloading is heavily dependent on mobile network characteristics such as bandwidth, latency, and coverage. Thus, mobile network technologies complement mobile offloading, which is why advances in mobile networking, such as 3GPP Long Term Evolution (LTE), are instrumental for the viability of mobile offloading. In addition, using WLAN access points and local surrogates can help provide excellent conditions for offloading, but the availability of these access points is currently limited.

3.4 Incumbent role

Software companies could benefit from MCO in several ways. First, decreased energy consumption of an application can make that application more desirable for the end user, although this effect is small if the energy savings aren’t apparent for the end user. Second, MCO can increase the performance of an existing application. Third, software companies can develop applications that would not be possible without MCO, allowing them to target new users. Finally, MCO can allow software companies to target mobile devices that would not normally have the capability of running the application. Thus, older generations of smartphones could run the same application as the newest versions, provided that the appropriate offloading capacity is available. As a result, mobile offloading can help reduce hardware fragmentation in the mobile space. Similarly, software companies may be able to target lower-end phones or feature phones, gaining access to an even larger potential target group. However, the decision for a software company to utilize MCO depends on the terms set by a MCO provider because too high fees may negate the benefits of MCO for a software company.

Mobile offloading has several implications for incumbent mobile device manufacturers and operating system (OS) providers. First, the general benefits of mobile offloading can make the mobile devices more attractive for end users. Second, similarly to software companies, the mobile device manufacturers and OS providers can also benefit from decreased hardware fragmentation, with older-generation devices potentially capable of running new, higher-performance applications due to MCO. This benefit may also extend to feature phones, which may be able to run more of the applications currently aimed for today’s smartphones. Third, mobile device manufacturers and OS providers compete as part of business ecosystems (Moore, 1993), where applications and application stores have a lock-in effect for end users. Providing support for MCO can enhance the desirability of the ecosystem for application developers and, ultimately, end users.

Mobile offloading also offers possibilities for mobile network operators (MNOs). First, they already have an existing billing relationship with their subscribers and they have considerable experience and the necessary technical infrastructure for micropayments. This allows them to handle the billing from the end user and then forward the payment to the offloading infrastructure provider. Second, MNOs could decide to act as offloading infrastructure providers themselves, especially if they already have access to cloud infrastructure capabilities. Because the current trend with network infrastructure providers involves offering MNOs cloud infrastructure that can be used for purposes such as reducing costs, increasing flexibility, and allowing MNOs to offer cloud content services (see, for example [1], [2]), this infrastructure could then be used to support MCO as well. This could be especially useful when the cloud infrastructure is near the edge of the network, because it would reduce latency, thus resulting in more beneficial offloading. Generally speaking, MNOs can utilize their reputation as a provider of secure services to enhance the users' trust in the offloading services. In addition, MCO should increase the amount of data traffic for the MNOs, which can have both negative and positive impacts for the MNOs depending on the utilized pricing schemes.

3.5 Security, privacy, and trust

Security and privacy are significant issues in offloading data and computation from the user's mobile device to outside devices, or surrogates. The user has to be able to trust that the computation performed on the surrogate is trustworthy and that the privacy and integrity of the offloaded data is not compromised. On the other hand, the surrogate's system also has to be secured from malicious program code that could be offloaded on the system. Two basic methods for establishing trust between the user and the surrogate are trust establishment and reputation-based trust (Satyanarayanan et al., 2009). In trust establishment, the user or the user device checks the validity of the surrogate before offloading. In reputation-based trust, the user verifies the identity of the surrogate and decides based on certain criteria whether the surrogate can be trusted. These two approaches could be compared to drinking water from a tap: either you test whether the water is drinkable before use and/or boil the water (trust establishment) or you consider the water safe to drink based on where you are (reputation-based trust). However, an added consideration is that because mobile offloading is very conscious on the energy requirements imposed on the client, the security and privacy mechanisms should be as energy-conscious as possible (Liu, Kumar and Lu, 2010).

4 Summary of Results and Discussion

Table 2 summarizes our analysis on mobile computation offloading and displays the critical factors that affect the technology evolution of mobile computation offloading (MCO). The added value (Porter, 1985) of MCO or relative advantage over competing technologies (Rogers, 2003) is instrumental in assessing the value of the technology. MCO has the potential of providing end users with completely new or enhanced applications that would not be possible without the technology. In addition, MCO can increase the computational capabilities and decrease the energy consumption of existing applications. However, especially small energy savings may be difficult for the end users to perceive or others to observe (Rogers, 2003).

Ease of experimentation (Gaynor, 2003) or trialability (Rogers, 2003) in the context of our MCO analysis refers to how much workload is required of application developers to implement new applications using MCO or modify old ones to support MCO. At the moment, developers can face hurdles in adopting MCO, although the amount of effort required depends on the chosen MCO architecture.

Anderson and Tushman (1990) noted that technological discontinuities can emerge through a convergence of existing complementary technologies, such as cloud computing and mobile networking in the case of MCO. Specifically, mobile networking technologies play a key role in the viability of MCO. Spotty coverage, limited bandwidth, and high latency of mobile networks can

[1] http://www.nokiasiemensnetworks.com/portfolio/liquidnet

[2] http://www.alcatel-lucent.com/new-thinking/market-growth/cloud.html

significantly limit the benefits of MCO. Nevertheless, advances in mobile networking technologies such as LTE or opportunistic use of WLAN access points can help in the adoption of MCO.

Table 2. Case analysis summary.

Dimension	Analysis
Added value	New and enhanced applications, increased processing capacity, energy savings
Ease of experimentation	Developer effort depends on MCO architecture
Complementary technologies	Mobile networking issues; LTE, WLAN as solutions
Incumbent role	Software companies: enhanced applications, reduced HW fragmentation Device vendors / OS providers: benefits from increased end user value, reduced fragmentation; ecosystem benefits MNOs: opportunities as billing provider, offloading provider
Security, privacy, and trust	Required on both mobile device and surrogate side; energy-efficient solutions required

The role and actions of incumbents are important in determining whether they can weather the attacks of new entrants (Foster, 1986), especially in the case of disruptive innovations (Christensen, 1997). Software companies benefit from MCO by being able to differentiate with new and enhanced applications as well as providing the end users with energy savings and enhanced computation. In addition, MCO can reduce the hardware fragmentation of mobile devices, increasing the potential target group for software companies. Incumbent mobile device manufacturers and operating system providers may reap benefits from MCO. These benefits may include increased end user value, reduced hardware fragmentation, and a more attractive ecosystem for developers. In addition, MCO offers opportunities for mobile network operators, who could function as billing providers or offloading providers.

Sufficient security, end user privacy, and trust are necessary preconditions for the diffusion of MCO (Siau and Shen, 2003; Robinson et al., 2011). Security solutions are needed for on both mobile device and surrogate sides, and the energy-efficiency of these solutions is an important added consideration.

Although mobile offloading holds much promise, the technological solutions are still in a very early phase. The current research is still relatively sparse and the presented offloading frameworks are not available for wider use. Moreover, the evaluated test cases do not cover applications commonly in use, which is why the wider applicability of mobile offloading for common use needs to be much more thoroughly tested.

5 Conclusions

The convergence of cloud computing and mobile technology is one possible solution to the constraints of mobile devices, such as limited computation capability and battery power. This trend, called mobile cloud computing, may result in transferring data and computation from mobile devices into the cloud, using either browser-based applications residing wholly in the cloud or native mobile applications that may be partly offloaded into the cloud. In this paper, we have examined the latter case, which is realized by an enabling technology called mobile computation offloading (MCO). We have aggregated the technical literature on MCO and analyzed the technology evolution of MCO using our own research framework derived from technology evolution and MCO literature. We identified several factors affecting the technology evolution of MCO, including both drivers and restraints.

Because the MCO technology is at a very early stage and the literature on MCO is focused on different technological solutions, the conclusions we can draw from this literature is limited. More specifically, we only identify factors affecting the technology evolution of MCO, but we do not comment on the interrelations and causalities of the factors. Moreover, we do not attempt to predict the success of MCO due to the exploratory nature of our analysis.

In the future, more research is needed on the technological development of MCO, the end user value MCO can provide, and discovery of the related possible early-phase applications. In addition, the business potential of MCO can be evaluated more accurately as the technology matures. Furthermore, it could be beneficial to examine energy saving solutions that seek to optimize the end-to-end energy savings including server-side solutions, not just the energy savings in mobile devices.

6 Acknowledgment

The work is supported by Tekes (the Finnish Funding Agency for Technology and Innovation, www.tekes.fi) as a part of Cloud Software Program (www.cloudsoftwareprogram.org) of Tivit (Strategic Centre for Science, Technology and Innovation in the Field of ICT, www.tivit.fi).

References

Anderson, P. and Tushman, M.L. (1990). Technological discontinuities and dominant designs: A cyclical model of technological change. Administrative Science Quarterly, 35 (4), 604-633.

Anderson, P. and Tushman, M.L. (1997). Technology Cycles, Innovation Streams and Ambidextrous Organizations. In Managing Strategic Innovation and Change. Oxford University Press, New York.

Bakhshi, S. (2009). Mobile Industry Confronts the Device Energy Gap, IDC. Available at: http://www.mirasoldisplays.com/sites/default/files/resources/doc/MobileDeviceEnergyGap%20Feb09%20pdf.pdf [Accessed February 28, 2012].

Balan, R.K., Gergle, D., Satyanarayanan, M. and Herbsleb, J. (2007). Simplifying cyber foraging for mobile devices. In Proceedings of the 5th international conference on Mobile systems, applications and services, New York.

Christensen, C.M. (1997). The Innovator's Dilemma: When New Technologies Cause Great Firms to Fail. Harvard Business School Press, Boston.

Christensen, C.M. and Rosenbloom, R.S. (1995). Explaining the attacker's advantage: Technological paradigms, organizational dynamics, and the value network. Research Policy, 24 (2), 233-257.

Chun, B.-G. and Maniatis, P. (2009). Augmented smartphone applications through clone cloud execution. In Proceedings of the 12th conference on Hot topics in operating systems, Monte Verità, Switzerland.

Chun, B.-G. et al. (2011). CloneCloud: elastic execution between mobile device and cloud. In proceedings of the sixth conference on Computer systems, Salzburg, Austria.

Cuervo, E. et al. (2010). MAUI: making smartphones last longer with code offload. In Proceedings of the 8th international conference on Mobile systems, applications and services, San Francisco, CA.

Foster, R.N. (1986). Innovation: The Attacker's Advantage. Summit Books, New York.

Gartner (2012). Gartner Says Worldwide Smartphone Sales Soared in Fourth Quarter of 2011 With 47 Percent Growth. Available at: http://www.gartner.com/it/page.jsp?id=1924314 [Accessed February 20, 2012].

Gaynor, M. (2003). Network services investment guide: maximizing ROI in uncertain times. Wiley Publishing, Indianapolis, IN.

IDC (2011). Smartphones Outstrip Feature Phones for First Time in Western Europe as Android Sees Strong Growth in 2Q11, Says IDC. Available at: http://www.idc.com/getdoc.jsp?containerId=prUK23024911 [Accessed February 28, 2012].

Juntunen, A., Suikkola, V., Raivio, Y. and Luukkainen, S. (2011). Innovation in Mobile Clouds - Analysis of an Open Telco application. In Proceedings of The First International Conference on Cloud Computing and Services Science, Noordwijkerhout, The Netherlands.

Kemp, R., Palmer, N., Kielmann, T. and Bal, H. (2010). Cuckoo: a Computation Offloading Framework for Smartphones. In Proceedings of The Second International Conference on Mobile Computing, Applications, and Services, Santa Clara, CA.

Kumar, K. and Lu, Y.-H. (2010). Cloud Computing for Mobile Users: Can Offloading Computation Save Energy? Computer, 43 (4), 51-56.

Liu, J., Kumar, K. and Lu, Y.-H. (2010). Tradeoff between energy savings and privacy protection in computation offloading. In Proceedings of the 16th ACM/IEEE international symposium on Low power electronics and design, Austin, TX.

Miettinen, A.P. and Nurminen, J.K. (2010). Energy efficiency of mobile clients in cloud computing. In Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, Boston, MA.

Moore, J.F. (1993). Predators and Prey: A New Ecology of Competition. Harvard Business Review, 71 (3),75-86.

Porter, M.E. (1985). Competitive Advantage. Free Press, New York.

Robinson, N. et al. (2011). The Cloud - Understanding the Security, Privacy and Trust Challenges, RAND Corporation. Available at: http://www.rand.org/pubs/technical_reports/TR933.html [Accessed February 28, 2012].

Rogers, E.M. (2003). Diffusion of Innovations. Fifth edition. Free Press, New York.

Satyanarayanan, M. (2001). Pervasive computing: vision and challenges. IEEE Personal Communications, 8 (4), 10-17.

Satyanarayanan, M., Bahl, V., Caceres, R. and Davies, N. (2009). The Case for VM-based Cloudlets in Mobile Computing. IEEE Pervasive Computing, 8 (1), 14-23.

Siau, K. and Shen, Z. (2003). Mobile communications and mobile services. International Journal of Mobile Communications, 1 (1), 3-14.

Tushman, M.L. and Anderson, P. (1986). Technological Discontinuities and Organizational Environments. Administrative Science Quarterly, 31 (3), 439-465.

Zhang, X., Jeong, S., Kunjithapatham, A. and Gibbs, S. (2010). Towards an Elastic Application Model for Augmenting Computing Capabilities of Mobile Platforms. In Proceedings of the Third international ICST conference on mobile wireless middleware, operating systems, and applications, Chicago, IL.