A Review on Distributed Application Processing Frameworks in Smart Mobile Devices for Mobile Cloud Computing

Muhammad Shiraz, Abdullah Gani, *Senior Member, IEEE,* Rashid Hafeez Khokhar, *Member, IEEE,* and Rajkumar Buyya, *Senior Member, IEEE*

***Abstract*—The latest developments in mobile devices technology have made smartphones as the future computing and service access devices. Users expect to run computational intensive applications on Smart Mobile Devices (SMDs) in the same way as powerful stationary computers. However in spite of all the advancements in recent years, SMDs are still low potential computing devices, which are constrained by CPU potentials, memory capacity and battery life time. Mobile Cloud Computing (MCC) is the latest practical solution for alleviating this incapacitation by extending the services and resources of computational clouds to SMDs on demand basis. In MCC, application offloading is ascertained as a software level solution for augmenting application processing capabilities of SMDs. The current offloading algorithms offload computational intensive applications to remote servers by employing different cloud models. A challenging aspect of such algorithms is the establishment of distributed application processing platform at runtime which requires additional computing resources on SMDs. This paper reviews existing Distributed Application Processing Frameworks (DAPFs) for SMDs in MCC domain. The objective is to highlight issues and challenges to existing DAPFs in developing, implementing, and executing computational intensive mobile applications within MCC domain. It proposes thematic taxonomy of current DAPFs, reviews current offloading frameworks by using thematic taxonomy and analyzes the implications and critical aspects of current offloading frameworks. Further, it investigates commonalities and deviations in such frameworks on the basis significant parameters such as offloading scope, migration granularity, partitioning approach, and migration pattern. Finally, we put forward open research issues in distributed application processing for MCC that remain to be addressed.**

***Index Terms*—Mobile Cloud Computing, Application Offloading, Elastic Applications, Distributed Systems**

I. Introduction

THE MINIATURE nature, compact design, high quality graphics, customized user applications support and multimodal connectivity features have made SMDs a special choice of interest for mobile users. SMDs incorporate the computing potentials of PDAs and voice communication capabilities of ordinary mobile devices by providing support for customized user applications and multimodal connectivity for accessing both cellular and data networks. SMDs employ wireless network technologies for accessing the internet; such as 3G connectivity, Wireless Fidelity (Wi-Fi), Wi-Max, or Long Term Evaluation (LTE). SMDs are the dominant future computing devices with high user expectations for accessing computational intensive applications analogous to powerful stationary computing machines. Examples of such applications include natural language translators [1]-[2], speech recognizers [1]-[3], optical character recognizers, image processors [4]-[5], online games, video processing [6] and wearable devices for patients such as wearable device with a head-up display in the form of eyeglasses (a camera for scene capture and earphones) is a useful application that helps Alzheimer patients in everyday life [7]. Such applications necessitate higher computing power, memory, and battery lifetime on resource constrained SMDs [8]. On the other hand, SMDs are still low potential computing devices having limitations in memory, CPU and battery power. In spite of all the advancements in recent years, SMDs are constrained by weight, size, and intrinsic limitations in wireless medium and mobility.

A key area of mobile computing research focuses on the application layer research for creating new software level solutions. Application offloading is an application layer solution for alleviating resources limitations in SMDs. Successful practices of cloud computing for stationary machines are the motivating factors for leveraging cloud resources and services for SMDs. Cloud computing employs different services provision models for the provision of cloud resources and services to SMDs; such as Software as a Service, Infrastructure as a Service, and Platform as a Service. Several online file storage services are available on cloud server for augmenting storage potentials of client devices; such as Amazon S_3 [9], Google Docs [10], MobileMe [11], and DropBox [12]. In the same way, Amazon provides cloud computing services in the form of Elastic Cloud Compute. The cloud revolution augments the computing potentials of client devices; such as desktops, laptops, PDAs and smartphones. The aim of MCC is to alleviate resources limitations of SMDs by leveraging computing resources and services of cloud datacenters. MCC is deployed in diverse manners to achieve the aforementioned objective. MCC employs process offloading techniques [13], [14] for augmenting application processing potentials of SMDs. In application offloading intensive applications are offloaded to

Manuscript received 7 March 2012; revised 4 June 2012 and 16 September 2012.

M. Shiraz, A. Gani, R. H. Khokhar are with Mobile Cloud Computing Research Lab, Faculty of Computer Science and Information Technology , University of Malaya, Kuala Lumpur, Malaysia (e-mail: muh_shiraz@siswa.um.edu.my, abdullah@um.edu.my, rkhokhar@ieee.org, raj@csse.unimelb.edu.au).

R. Buyya is with Cloud Computing and Distributed Systems(CLOUDS) Lab, Department of Computer Science and Information Systems , The University of Melbourne, Australia.

Digital Object Identifier 10.1109/SURV.2012.111412.00045

remote server nodes. Current offloading procedures employ diverse strategies for the deployment of runtime distributed application processing platform on SMDs. A challenging issue in current DAPFs (Distributed Application Processing Frameworks) is the additional computing overhead on SMDs in the deployment and management of runtime distributed application execution. This paper reviews current DAPFs in SMDs within MCC domain and identifies challenges in the cloud based processing of mobile applications. We classify existing DAPFs by thematic taxonomy and investigate commonalities and deviations in such frameworks on the basis of significant parameters such as offloading scope, partitioning approach, migration support, migration granularity, application developer support, migration pattern and execution monitoring. The contribution of the paper lies in the categorization of frameworks on the basis of thematic taxonomy, analysis of current DAPFs by discussing the implications and critical aspects, identifying the issues in existing solutions for offload processing and challenges to cloud based application processing of mobile applications. The listing of challenges and open issues guide researchers to select the appropriate domain for future research and obtain ideas for further investigations.

The rest of the paper is organized as follows. Section 2 explains the fundamental concepts of MCC. It discusses the concept of cloud computing, mobile cloud computing and explains the different techniques to augment smart mobile devices resources based on resources available within the cloud. Section 3 presents thematic taxonomy of current DAPFs, reviews current application offloading algorithms on the basis of taxonomy and investigates the implications and critical aspects of current DAPFs. Section 4 compares current DAPFs by comparing the commonalities and deviations by using significant parameters presented in the taxonomy. Section 5 highlights the issues in current DAPFs and discusses challenges to distributed application processing for MCC which contributes toward the advancement of research and development of optimal distributed applications for MCC. Finally, section 6 draws concluding remarks with future directives. Table 1 shows the list of acronyms used in the paper.

TABLE I
LIST OF ACRONYMS

Symbol	Description
3G	Third Generation
CC	Cloud Computing
CPU	Central Processing Unit
DEM	Device Elasticity Manager
DAPFs	Distributed Application Processing Frameworks
DISHES	Distributed Shell System
DRAM	Dynamic Random Access Memory
DS	Developer Support
EM	Execution Management
HTTP	Hyper Text Transfer Protocol
IP	Internet Protocol
LTE	Long Term Evaluation
MAR	Mobile Augmented Reality
MAUI	Mobile Assistance Using Infrastructure
MCC	Mobile Cloud Computing
MG	Migration Granularity
MS	Migration Support
MP	Migration Pattern
OS	Offloading Scope
PA	Partitioning Approach
PDA	Personal Digital Assistant
SOA	Service Oriented Architecture
S3	Simple Storage Service
SD	Service Directory
SMDs	Smart Mobile Devices
SS	Security Support
TSP	Telecommunication Service Provider
UMSC	Universal Mobile Service Cell
URL	Uniform Resource Locator
VM	Virtual Machine
Wi-Fi	Wireless Fidelity
XML	Extended Markup Language

II. BACKGROUND

This section elaborates the concept of cloud computing and mobile cloud computing. Further, it explains the mechanism of augmenting smartphone through computational clouds.

A. Cloud Computing

Cloud computing is the latest distributed computing model that implements the utility computing vision [15] wherein computing services are provided on demand basis. Cloud service models enable with new IT business models such as on-demand, pay-as-you-go, and utility computing. The objective of the cloud computing model is to increase the capacity and capabilities of client devices by accessing leased infrastructure and software applications instead of owning them. Cloud computing has introduced new kind of information and services and new ways of communication and collaboration. Cloud has created online social networks in which scientists share data and analysis tools to build research communities [16]-[17]. In cloud computing, applications are delivered as services over the internet and user access computing resources from centralized cloud servers through service providers [18]. The examples of public utility computing include Amazon Web Services (AWS), Google AppEngine, Microsoft Azure and Aneka. AWS offers infrastructure as a service and software as service which enable to utilize the virtualize resources and services in cloud datacenters. It reduces the cost and efforts associated with the administration of computer hardware and software for organizations [19]. AWS are utilized to store personal data through its Simple Storage Service(S3) [20], and computation is performed using elastic cloud compute. Google App Engine provides application developmental and deployment platform in Googles data centers. It uses powerful infrastructure to provide services worldwide. App Engine provides an application development environment which uses well known application developmental tools (such as Java and Python). It provides a rich set of APIs to users whereas sustaining the security and isolation from other applications running the cloud infrastructure [21].Windows Azure is an extensible and open cloud computing platform which provides the services to develop deploy and operate applications and services in cloud datacenters. Windows Azure offers a simple, widespread, and powerful cloud computing platform for the creation of web applications and services [22]. Aneka offers the platform as a services model for cloud computing. It serves as an application development framework for building customized applications and deploying them on either public or private clouds [23].Computational clouds implement different types of service models for implementing the on demand computing vision [15]. Service providers provide services in

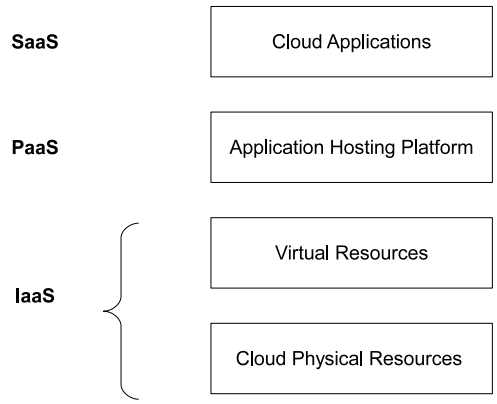

Fig. 1. Layered Cloud Computing Architecture

the form of various service models; Software as a Service (SaaS), Infrastructure as a Service (IaaS), and Platform as a Service (PaaS). Fig. 1 shows an abstract level layered cloud computing architecture.

The hardware resources in the cloud datacenters are the physical resources of computational clouds. Access to the physical resources is provided in the form of virtual machines. A middleware (hypervisor) masks access to the physical resources and is responsible for the deployment and management of virtual machines. The application hosting platform is composed of cloud programming environments and tools and monitoring tools such as QoS negotiation, admission control, pricing and billing. The cloud applications run on the virtual machine instances in complete isolation.

B. Mobile Cloud Computing

Mobile cloud computing is the latest practical computing paradigm that extends utility computing vision of computational clouds to resources constrained SMDs. Aepona [24] defines MCC as a new distributed computing paradigm for mobile applications whereby the storage and the data processing are migrated from the SMD to resources rich and powerful centralized computing data centers in computational clouds. The centralized applications, services and resources are accessed over the wireless network technologies based on web browser on the SMDs. Successful practices of accessing computational clouds on demand for stationary computers motivate for leveraging cloud services and resources for SMDs. MCC has been attracting the attentions of businesspersons as a profitable business option that reduces the development and execution cost of mobile applications and mobile users are enabled to acquire new technology conveniently on demand basis. MCC enables to achieve rich experience of a variety of cloud services for SMD at low cost on the move. [25]-[26]. MCC prolongs diverse services models of computational clouds for mitigating computing resources (battery, CPU, memory) limitations in SMDs. The objective of MCC is to augment computing potentials of SMDs by employing resources and services of computational clouds. MCC focuses on alleviating resources limitations in SMDs by employing different augmentation strategies; such as screen augmentation, energy augmentation, storage augmentation and application processing augmentation of SMD. In [27], we study mobile augmentation techniques and devise a taxonomy including three main approaches, namely high-end resource production, native resource conservation, and resource requirement reduction. We analyze a number of approaches and argue that MCC lessens need to high-end hardware, reduces ownership and maintenance cost, and alleviates data safety and user privacy.

The MCC model is composed of three major components; SMDs, internet wireless technology and computational cloud. SMDs use wireless network technology protocols such as 3G, LTE, or Wi-Fi to access the services of computational cloud in mobile environment. However, the connection is less reliable due to mobility requirements such as handoff processes. As SMD inherit its nature of mobility, it needs to execute location-aware services which consume resources and turned it to be a low-powered client. Fig. 2 shows a generic model of MCC in which the cloud that provides off-device storage, processing, queuing capabilities, and security mechanism is integrated with SMD via wireless network technologies.

MCC utilizes cloud storage services [[9]-[12] for providing online storage and cloud processing services for augmenting processing capabilities of SMDs [14]. Processing capabilities of SMDs are augmented by outsourcing computational intensive components of the mobile applications to cloud datacenters. The following section discusses the concept of augmenting smartphones through computational clouds.

C. Augmenting Smartphones through Computational Clouds

MCC implements a number of augmentation procedures for leveraging resources and services of cloud datacenters. Examples of the augmentations strategies include; screen augmentation, energy augmentation, storage augmentation and application processing augmentation of SMD [27]. In MCC, two categories of the cloud services are of special interest to research community; cloud contents and computing power. Cloud contents are provided in the form of centralized storage centers or sharing online contents such as live video streams from other mobile devices. A number of online file storage services are available on cloud server which augments the storage potentials by providing off-device storage services. Examples of the cloud storage services include Amzon S3 [9] and DropBox [12]. Mobile users outsource data storage by maintaining data storage on cloud server nodes. However, ensuring the consistency of data on the cloud server nodes and mobile devices is still a challenging research perspective. SmartBox [28] is an online file storage and management model which provides a constructive approach for online cloud based storage and access management system. Similarly, the computing power of the cloud datacenters is utilized by outsourcing computational load to cloud server nodes. The mechanism of outsourcing computational task to remote server is called process offloading or cyber foraging. Smart mobile devices implement process offloading to utilize the computing power of the cloud. The term cyber foraging is introduced by Satyanarayanan [29] to augment the computing potentials of wireless mobile devices by exploiting available stationary computers in the local environment. The mechanism of outsourcing computational load to remote surrogates in the close proximity is called cyber foraging [30]. Researchers

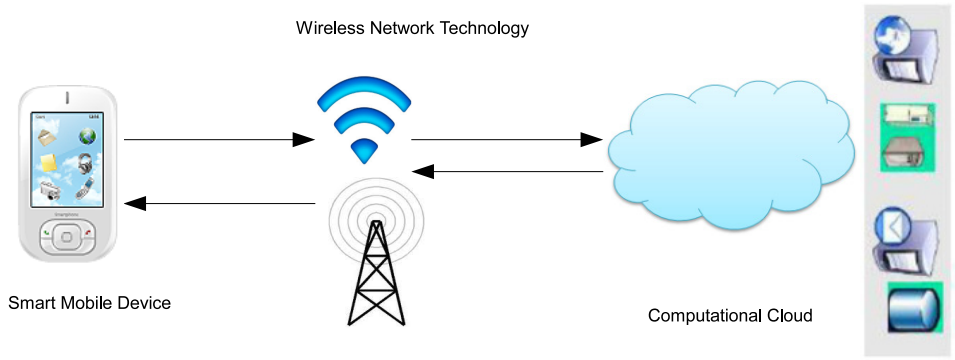

Fig. 2. Model of Mobile Cloud Computing

extend process offloading algorithms for Pervasive Computing [31], Grid Computing [32] and Cluster Computing [33]. In recent years, a number of cloud server based application offloading frameworks are introduced for outsourcing computational intensive components of the mobile applications partially or entirely to cloud datacenters. Mobile applications which are attributed with the features of runtime partitioning are called elastic mobile applications. Elastic applications are partitioned at runtime for the establishment of distributed processing platform.

Elastic mobile applications are attributed with the following features [34]. a) Ad-hoc platform creation is an important attribute of elastic mobile applications. Distributed application processing platform is established on ad-hoc basis at runtime in which elastic mobile application is partitioned dynamically and computational intensive components are migrated to remote server nodes. Mobile clients dynamically arbitrate with cloud servers or surrogates to determine appropriate server node for remote application processing. b) Elastic applications are designed in such a manner so that computational intensive components of the mobile application are separated dynamically at runtime. Applications are partitioned at different granularity level depending upon the design and partitioning policy of the offloading algorithm. c) Adaptive offloading of the intensive components of the applications is a significant attribute of elastic mobile applications. Partitions of the application are offloaded to remote machines for remote execution which augments the computing capabilities of SMDs. Application offloading occurs whereas keeping in view different objective functions; such as energy saving, processing power, memory storage, and fast execution. d) Transparency in the distributed execution platform is a significant attribute of elastic applications. Transparency assures that elastic mobile application executes transparently on remote surrogates/server nodes. A transparent distributed processing environment gives the notion as entire application is being executed locally on SMD. All the complexities of remote execution are concealed from mobile users. Researchers determine applications offloading as an appropriate software level solution for alleviating resources limitations in SMDs.

Currently application offloading is implemented in a number of ways. The application offloading frameworks outsource computational load of SMD at different granularity levels. The static application partitioning approach is used to separate the intensive components of mobile application only once. The dynamic partitioning approach is implemented to address the issue of dynamic application processing load on SMDs at runtime. Dynamic partitioning of the intensive mobile application at runtime is a robust technique for coping with the dynamic processing loads on SMD. In dynamic partitioning application is partitioned dynamically at runtime casually or periodically. In casual partitioning runtime profiling and solving mechanisms are activated in critical conditions to offload intensive components of mobile application. In periodic partitioning the runtime optimization mechanism evaluates computing resources utilization on SMD periodically. Current dynamic partitioning approaches analyze the resources utilization on SMDs, computational requirements of the mobile application and search for runtime solving of the problem of resource limitations on SMD. The profiling mechanism evaluates computing resources requirements of mobile application and the availability of resources on SMD. In critical condition (the unavailability of sufficient resources on SMD) elastic mobile application is partitioned and the computational intensive components of the application are offloaded dynamically at runtime. SMDs negotiate with cloud servers for the selection of appropriate server node. At that moment partitions of the application are migrated to remote server node for remote processing. Upon successful execution of the remote components of the application, result is returned to main application running on SMD.

Empirical analysis ascertains the significance of distributing processing loads to remote server nodes [13], [35]-[14]. However, the deployment of distributed application processing platform is obstructed by a number of unresolved challenges for MCC. The latest offloading models [14] focus on the establishment of dynamic distributed application processing platform at runtime. For the selection of cloud server node, SMDs arbitrate with cloud server node dynamically at runtime. Therefore, the configuration of distributed processing platform at runtime is a resources starving and energy hunting mechanism. Dynamic runtime offloading involves the issues of dynamic application profiling and solver on SMD, runtime application partitioning, migration of intensive components and continuous synchronization for the entire duration of runtime execution platform. Therefore, the development and deployment of intensive mobile applications on the basis of current algorithms is still a challenging research issue. Fig.

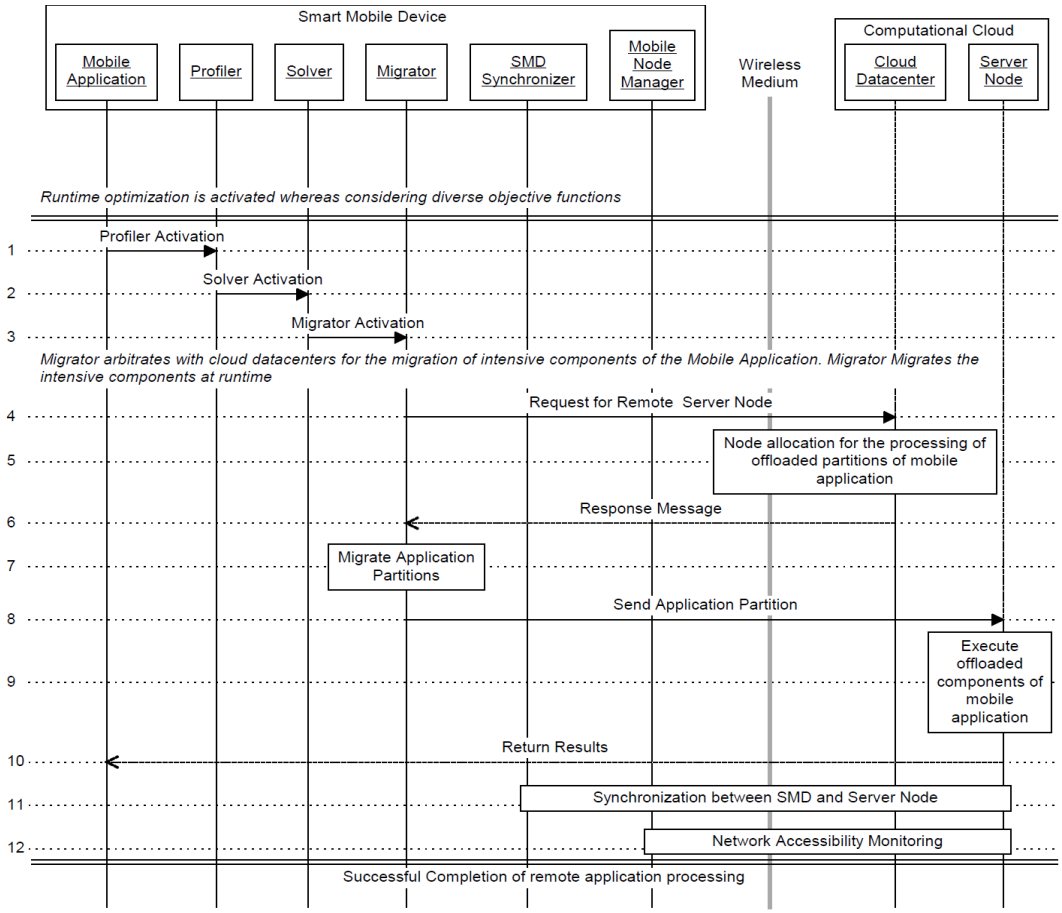

Fig. 3. Sequence Diagram for Dynamic Application Offloading in MCC

II-C shows general sequence of flow for dynamic offloading of intensive components of the mobile application.

III. Distributed Application Processing Frameworks (DAPFs) for Smart Mobile Devices

The current DAPFs for SMDs employ a number of strategies for the establishment of runtime distributed application execution platform. This section provides thematic taxonomy for current DAPFs and reviews the traditional DAPFs on the basis of framework nature attributes of the taxonomy. Further, it investigates the advantages and critical aspects of current DAPFs for SMDs.

A. Taxonomy of Distributed Application Processing Frameworks

Fig. III-A shows the thematic taxonomy of DAPFs, which are categorized on the basis of framework nature, migration pattern, migration support, partitioning approach and objective functions. The attribute of framework nature indicates the main mechanism employed for application offloading. We categorize current DAPFs on the basis of virtual machine migration, entire application migration and application partitioning. Virtual machine migration nature of DAPFs indicates that SMD offload application by encapsulating the running application in VM instance on SMD. The VM instance is outsourced to cloud server nodes. A number of current DAPFs employ VM migration based approach for offloading intensive components of the application [6], [30], [36]-[37]. The entire application migration nature of DAPFs indicates that SMDs offload entire processing job to remote server nodes. Current DAPFs offload entire application in two different manners. a) Mobile application starts execution of the smart mobile device and in critical condition the running instance of the mobile application is offloaded to remote server node [38]. b) The client component of the mobile application runs on the SMD where the processing load of the application is offloaded to cloud server for remote processing [39]. Application partitioning nature of the framework indicates that computational intensive components of the mobile are separated at runtime. Partitions of the application are offloaded to remote server nodes [14], [40]. The partitioning approach of a framework indicates the mechanism of separating intensive components of the application. Current DAPFs employ two different strategies for separating computational intensive components of the mobile application. In static application partitioning the intensive components of the application are separated either at compile time or runtime statically. Dynamic partitioning mechanism follows dynamic evaluation mechanism for assessing the computational load on SMD. The attribute of migration support indicates the level of support required for migrating application or partitions of the application at

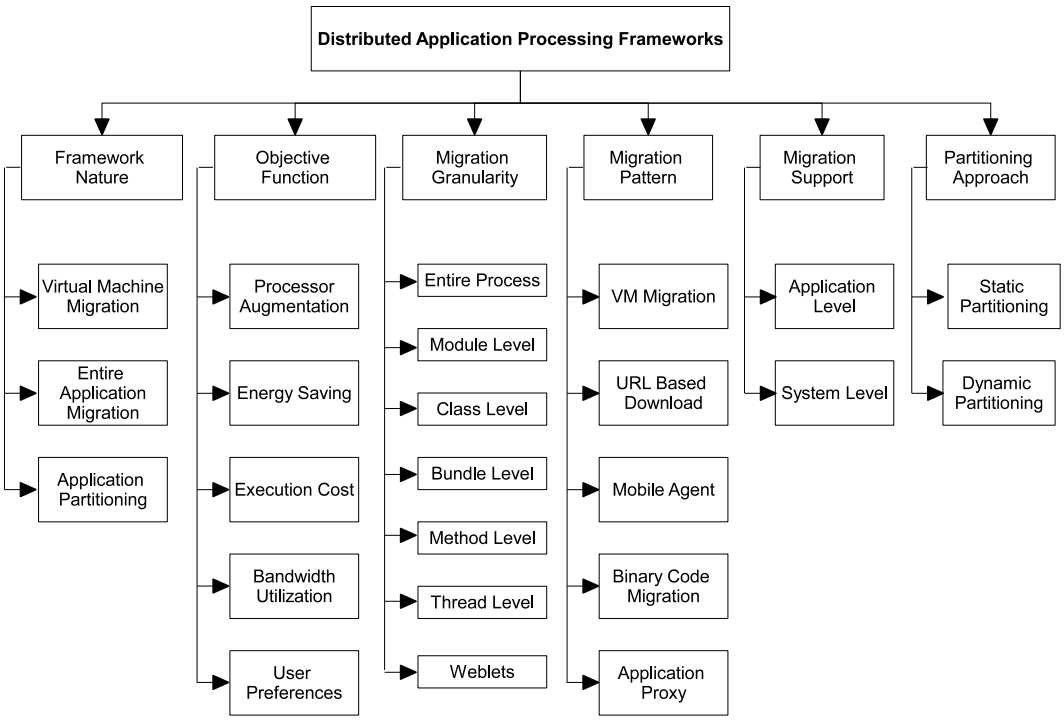

Fig. 4. Taxonomy of Application Processing Frameworks for SMDs

runtime. Currently, migration support is provided either at system level or application level. The attribute of migration granularity indicates the granularity level at which application is migrated. Current DAPFs offload intensive components of the application at different granularity level. For example thread level granularity indicates that running thread is offloaded for remote processing. In the same way, method level granularity indicates that methods of the application are offloaded for remote processing. Migration pattern represents the pattern or way by which application or partition of the application is migrated to remote server. Current DAPFs employ a number of migration patterns such as VM migration, download using URL on remote host, mobile agent serving as courier for application transfer, binary code transfer of the application or copying entire proxy of the application on distributed computing nodes. The objective function attribute indicates the primary objective of a framework for application offloading. Current DAPFs aim for a number of objective functions such as saving energy on SMD, efficient bandwidth utilization, saving processing power on SMD, user preferences for fast application processing, or execution cost parameter.

B. Review on Application Offloading Frameworks by Using Thematic Taxonomy

A classification of application offloading frameworks by using their attributes is shown in Fig. 4. This section analyzes current application offloading frameworks and investigates the implications and critical aspects of current DAPFs.

1) VM Migration Based Application Offloading: In [30] cyber foraging framework is employed to utilize computation resources of computing devices (stationary or mobile) in close proximity of SMD. The framework implements client/server architecture. Mobile devices request for process offloading and surrogate server provides the services on demand. The framework supports configuration of multiple surrogate servers simultaneously and employs virtual machine technology for remote application processing. A single surrogate server is capable to run a configurable number of independent virtual servers with isolation, elasticity, resource control and simple cleanup mechanism. Each offloaded application executes on isolated virtual server. The framework ensures secure communication by deploying cryptographic measures for communication between SMD and surrogate server. The framework includes the benefits of low latency, local accessibility of remote surrogates and fewer concerns of security and privacy. The critical aspects of such approach is the deployment of template based virtualization approach which is a highly time consuming and resources starving mechanism for VM deployment [41]. The framework requires the annotation of individual components of the application as local or remote which is an additional effort for application developers. Further, surrogate based cyber foraging is restricted to the availability of services and resources on local servers.

VM based cloudlets framework [7] differs from cyber foraging [30] by migrating image of the running application to the explicitly designated remote server. A cloudlet is a trusted resource rich computer or cluster of computers that is connected to internet and is accessible for SMDs. Mobile device serve as a thin client providing only user interface whereas actual application processing is performed on the cloudlet in distributed environment. The proposed framework is based upon transient customization of cloudlet infrastructure using hardware VM technology in which VM encapsulates and

detaches the temporary guest software environment from the cloudlet infrastructures permanent host software environment. The framework employs variant procedures for VM migration. The critical aspects are that the framework requires additional hardware level support for the implementation of VM technology and is based on cloning mobile device application processing environment to remote host which involves the issues of VM deployment and management on SMD, privacy and access control in migrating the entire execution environment and security threats in the transmission of VM.

Clone cloud based framework [6] is a significant approach for offloading application of diverse nature in different manners. Clone cloud differs from other approaches [7], [30] by employing three different offloading algorithms for different types of applications. However, the attribute of offloading image of the running states of the application to remote server resembles to the VM based Cloudlet [7] approach. The framework reduces the dynamic transmission overhead of application code by deploying a simple approach for synchronization. Clone cloud employs Primary functionality outsourcing by offloading computational intensive tasks to remote host whereas simple tasks such as user interfaces are executed on mobile devices. Examples of the applications include speech recognition, image processing and video indexing. Background augmentation is implemented for applications demanding no user interaction. Background augmentation offloads the entire application to remote host and communicates results from background process to the mobile device. Examples of the applications include; antivirus and file indexing for faster search. Mainline augmentation policy is deployed for applications having mixed nature; having some computational intensive computational load and need to interact with other parts of the applications such as debugging applications. Clone cloud [6] is a significant framework for offloaded processing which includes a simple approach for synchronization between SMD and remote server. The critical aspect of the Clone cloud is the migration of execution environment to remote server which involves the issues of security, privacy, access control, VM deployment and management on SMD. The deployment of variant strategies for application migration on the basis of application nature results in enlarged overhead on mobile devices. Clone cloud deploys a single thread approach which increases jitter in the execution time of the application components.

The elastic CloneCloud [36] extends the concept of local Clone cloud [6] to remote cloud datacenters. The framework is based on partitioning of the application on thread basis. Partitioning and integration of the application occurs at application level. The running states of the outsourcing components of the mobile application are encapsulated in VM instance and VM migration is employed for partition migration to cloud node. The framework is implemented at application level and centralized monitoring mechanism is used for the establishment and management of distributed application execution platform. CloneCloud [36] is a productive approach for extending the concept of VM based offloading from local distributed platform to centralized cloud servers. Offload processing is monitored through a centralized mechanism. The framework reflects on execution time and energy consumption at mobile device for cost metrics. CloneCloud implements a complicated architecture on SMD for the establishment and management of distributed platform. The framework is based on VM instance migration to the cloud node which involves the concerns of secure communication of running application states encapsulated in VM and privacy and access on remote server node. A major limitation of the architecture is that a single thread is migrated to the cloud at a time which reduces concurrency of the execution of application components. Virtualized execution environment for mobile applications [42] is a VM migration based framework for application offloading. The framework utilizes application level process migration and employs android platform for distributed application deployment. A running application is encapsulated in VM on SMD and VM is migrated to remote cloud computing environment. Cloud server creates fresh VM instance, and the offloaded application VM is cloned into the newly created VM instance on server. A synchronization mechanism is provided between SMD and cloud server. A middleware is placed between mobile device OS and hardware to support runtime workload migration and to better utilize the heterogeneous resources of mobile device and cloud servers. The framework deploys pause and resume scheme of the android platform for state transfer. The framework employs application level process migration strategy for offload processing and employs hardware base trusted platform module. The framework provides mechanism for storing encryption keys and performs cryptographic operations on sensitive data. The critical aspects are that the framework requires heavy and traffic intensive synchronization mechanism for ensuring consistency between SMD and cloud server. The framework entails a separate program called agent to be installed on SMD and cloud server which results in additional overhead on mobile device.

Mirror server [37] is a distinct augmentation framework which employs Telecommunication Service Provider (TSP) based remote services. A TSP is a type of communications service provider which provides voice communication services such as land line telephone and cellular phone call services. A mirror server is a powerful server configured in TSP backbone which maintains VM template for different mobile devices platforms. The VM template for each mobile device is kept with default settings. A new VM instance is created for offloaded component of the mobile application. The VM template for each mobile device is called its mirror and the server responsible for the deployment and management of the mirrors is called mirror server. The server creates fresh VM instance as per the platform of the requesting SMD. Mirror server is scalable and is capable to create hundreds of mirrors at a time. Mirror server augments smartphones by providing three different types of services; security (file scanning), storage (file caching) and computation offloading. The significant aspect of mirroring smartphone is that it provides reliable services through 3G network and addresses the challenging aspect of heterogeneity in SMDs platforms. The framework provides a lightweight protocol for SMDs for accessing remote services on mirror server and employs an optimized mechanism for downloading and offloading. The critical aspects of mirroring based DAPF is the deployment

of TSP based mirror servers which are not basically designed for data processing, for that reason limited services can be acquired as compared to cloud datacenters.

The following section describes the generic sequence of operations for VM migration based application offloading. 1) The first step for application offloading is to arbitrate for appropriate surrogate or remote server host. Subsequently, the running application is encapsulated in VM on SMD which involves the creation of VM instance, VM configuration for running application and encapsulating all the state information of running application in VM instance. 2) VM instance is migrated to the remote server through wireless medium. VM encapsulates either entire application or partition of the application. 3) A new VM instance is created on remote server and the migrated VM is cloned onto the newly created VM instance on remote server. Running states of the application are resumed and application is executed on remote server host. Finally, results are returned to the SMD. 4) Remote server ensures complete isolation of guest VM which means that the executing environment of guest VM is prevented from interference. Fig.5 shows abstract level flowchart of VM migration based application offloading.

2) Entire Application Migration Based Application Offloading: Lightweight secure cyber foraging infrastructure [30] employs Virtual Server Manager (VSM) which handles requests from SMDs for surrogate operations. SMD sends a request to VSM which is composed of URL to the program to be executed on surrogate. The entire program is downloaded on that URL and executed remotely. The background augmentation strategy of Clone cloud [6] employs entire application migration to remote host. The application is migrated to remote local servers using VM instance migration and results are returned from background process to the mobile device. Examples of the applications include; antivirus and file indexing for faster search. The virtual cloud computing provider solution for mobile devices [13] is an ad-hoc cloud framework focuses on the establishment of virtual cloud of SMDs. The virtual cloud computing environment is composed of SMDs in the proximity which are in the same locality and remain in stable mode. Mobile devices in the proximity set up an ad-hoc or virtual cloud environment and enables SMDs in the vicinity to share computational load. The framework is composed of different components. The context manager component of the architecture maintains information regarding volunteer SMDs for resource sharing. The offloading manger component is responsible for sending and receiving entire applications, management of runtime distributed environment and detecting failure and failure management. Offload manager coordinates with p2p component for application offloading and returning results. Universal Mobile Service Cell (UMSC) based framework is a unique mobile agent based optimization solution which focuses on virtual cloud of mobile devices [38]. The distinguishing features of the framework are the employment of mobile agent (UMSC) for application offloading and virtual cloud based service provision. The proposed architecture is composed of mobile hosts, UMSC, and mobile cloud units. Cloud unit support several services such as offload computing and remote storage. The mobile cloud is composed of two kinds of cloud units such as local cloud unit and remote cloud

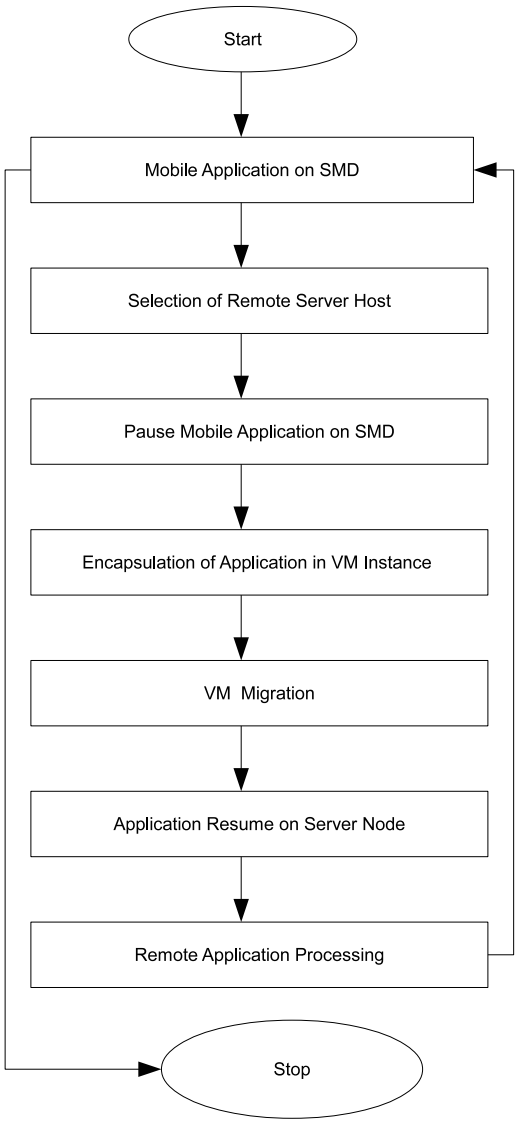

Fig. 5. Flowchart for the VM Migration Based Application Offloading

unit. The framework uses UMSC for the offloading of entire application to remote cloud unit. UMSC serves as a mobile agent and works as a proxy for transmission between mobile cloud and mobile host. UMSC is implemented as an intelligent software module which carries the requests of users. UMSC does not send request or responses to the network; instead UMSC itself migrates into the cloud to search response. The framework is composed of mobile agents in local mobile cloud computing environment for mobile devices and uses a genetic algorithm based scheduling policy for UMSC. The mobile environment is divided into a large number of cell regions. Each cell region is composed of several mobile cloud units. The cloud units in the cloud regions collectively form the virtual mobile cloud environment. Cloud units are the mobile support stations for providing services. The framework addresses the intrinsic issues associated with mobile computing; mobility, heterogeneity, and low bandwidth. UMSC based approach is a hybrid solution that combines mobile agent technology with virtual cloud model composed of SMD. UMSC employs mobile IP to compensate the problem of

mobility and provides a mechanism to overcome the problem of mobile host disconnection. UMSC guarantees the quality and stability of wireless connections. The critical aspects are that distributed services are restricted to the availability of mobile nodes in virtual distributed wireless environment. The framework exploits localized approach for accessing distributed resources and involves a decentralized monitoring mechanism on SMDs which increase computing resources demands on SMDs. The framework implements management of mobile agents on mobile devices, which is a sophisticated and resource consuming mechanism. Distributed Shell System (DISHES) [43], is the extension of UNIX kernel shell to support ubiquitous distributed computing platform for SMDs. The architecture is composed of a centralized server, which contains a Service Directory (SD). The ambient computers register with the server which employs SD for maintaining database of all the nodes willing for sharing resources. Mobile clients make use of SD services for tracking appropriate remote server. DISHES serves as an interface middleware between a mobile user and network computers. SMD request for the availability of remote host for application processing, SD responds with the IP address of appropriate volunteer remote host for application process. SMD offloads entire application to remote host for remote processing and results are returned to SMD on successful completion of the remote processing. DISHES includes performance optimization mechanism to monitor network traffic and provides remote execution services in transparent manner. The critical aspects of DISHES are the decentralized approach, unavailability of centralized mechanism for the establishment and management of distributed platform and entire application migration for offload processing. DISHES imposes additional assistant process creation on SMDs which involves intensive monitoring overhead on SMDs.

Misco [35] is deployed for SMDs which extends the concept of MapReduce to the distributed cloud environment which is composed of centralized server and mobile worker nodes. MapReduce is a flexible distributed data processing framework which automatically parallelizes the processing of long running applications in cluster environment [44]. In Misco, master server is a centralized monitoring entity which is responsible for the implementation of MapReduce framework. A distinctive feature of the framework is that SMDs are the worker nodes which serve as serving components for remote application processing. All the worker nodes coordinate with the master server for getting workload and returning result. The communication between worker and master server occurs through HTTP Server. All downloads and uploads between master server and workers are performed in the form of XML files. Map and reduce functions are identified by the developers during the application development process. The framework provides a distributed platform for mobile applications. Misco provides a centralized monitoring mechanism for monitoring of the distributed execution platform. The critical aspects are that the framework consists of worker mobile nodes which are intrinsically resources poor and for that reason the availability of computing services is restricted to the computing potentials of SMDs. Misco requires the developers to annotate the methods as map or reduce functions and does not perform any centralized processing of application which results in communication overhead repeatedly between worker nodes and master server. Communication overhead increases jitter in the process execution, bandwidth consumption and energy consumption.

Tradeoff between energy savings and privacy protection [45] addresses the issue of data privacy in offloading processing. The focus of research is on the privacy of data sent to the grid powered server. Stenographic techniques [46]-[47] are exploited for disguising actual image from grid powered servers. The authors focus on the fact privacy based offloading in which the contents of the offloaded components are hidden from the cloud node. The authors investigate the tradeoff between energy consumption and privacy of offloading and performed analysis of different execution patterns of the applications. Different parameters are involved in the energy consumption of mobile application processing; computation power of mobile, network power, idle power of mobile device, the speed of mobile system, speed of server, and bandwidth of network. Cogniserve [39] is an optimized architecture for the cloud server. The framework focuses on the processing of recognition applications such as speech recognition and image processing applications of mobile phones. Cogniserve addresses the evolving feature of Mobile Augmented Reality (MAR) for MCC. Cogniserve architecture is composed of three main components: Application cores for processing over cloud server; Application specific recognition accelerators for performance improvement and decreasing latency; Architectural support for general purpose programming and efficient communication between small cores and accelerators. The recognition server is composed of small cores connected via interconnect to an integrated memory controller for attaching it to DRAM. The design is composed of simple chip multi-processor. Application specific accelerators are used to further enhance the recognition execution time. Three types of accelerators are deployed in the architecture; Gaussian mixture model for speech recognition, match accelerator and interest point detection for image recognition. The resulting architecture is heterogeneous by integrating small cores. In traditional models accelerators are treated such as input/output devices, and are configured with drivers. Accelerators are programmed using physical addresses in memory. For this reason accelerators are difficult to program and the architecture becomes inefficient for the reason that of the overhead involved in user space and kernel space memory transitions. CogniServe deploys the concept of instruction set architecture, in which there is direct user access from the small core to accelerator. For that reason the user to kernel mode transitions has been eliminated. The architecture also utilizes the concept of common memory management units which lead the accelerator share virtual memory space with the core, as a result of this it eliminated data movement overhead with the kernel to user mode transition. CogniServe is an optimized architecture for the processing of recognition applications such as speech recognition and image processing applications of SMDs. The framework deploys heterogeneous server architecture for recognition applications of mobile devices. CogniServe provides direct access between server cores and accelerators instead of kernel to user space transition which

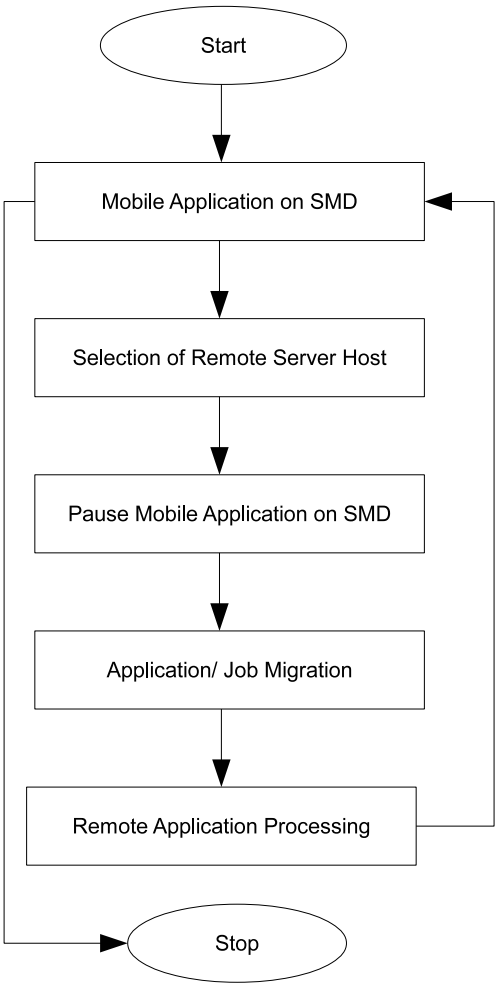

Fig. 6. Flowchart for Entire Application Migration Based DAPFs

eradicates address space transition and results in a low cost and energy efficient architecture. The critical aspects are that the framework requires special hardware level support for the implementation and is specially designed for recognition applications such as image processing and speech recognition applications.

In [42] the framework employs application level process migration and uses android platform for the deployment. A running application is encapsulated in VM on SMD and VM is migrated to remote cloud computing environment for remote processing. Mirror server based approach [37] involves the migration of the state of the entire running application on the smartphone device to mirror instance on server. Application is executed in the mirror VM instance and result is return to the smart mobile device. Fig. 6 shows the abstract level flow of offloading entire application/job to remote server node. SMD arbitrate with cloud server nodes for the selection of remote server node, at that moment entire application or job is migrated to remote server node. Upon the successful execution of the application on remote server ultimate results are returned to SMD.

Virtual machines lead to high CPU utilization. VMs share the same CPU/core which increases the CPU scheduling latency for each VM significantly [41]. VM migration based offloading requires additional computing resources and time for the deployment and management of VM on SMD. Consequently such approaches increase the execution cost and time of the application. Migration of running application along with its data and states is susceptible to security breaches and attacks. Further, a number of other research challenges [48] such as privacy and access control are still addressable which obstruct the goals of optimal VM based migration algorithms for MCC.

3) Application Partitioning Based Application Offloading: Partitioning of the mobile application at runtime is a prominent approach for outsourcing the intensive components of mobile applications. Elastic mobile applications are capable to be partitioned at runtime for coping with the resources constraint on SMD. Elastic applications are partitioned either statically or dynamically at runtime. The following section classifies and reviews existing approaches on the basis of static or dynamic offloading.

Static Partitioning Based Application Offloading: In static application partitioning the application is partitioned in fixed number of partitions either at compile time or runtime. The computational intensive partitions of the applications are outsourced to remote servers for offload processing. Current DAPFs deploy a number of approaches for making the decision of partitions outsourcing. In the primary functionality offloading [7]; application is statically partitioned in two major partitions. Such applications involve two types of processing; user interface required on mobile device; and computational intensive parts of the application are offloaded to remote surrogates or cloud servers. In MISCO [35] the application is statically partitioned into two types of functions; map and reduce. Map function is applied on the set of input data and produces intermediary ¡key, value¿ pairs; such pairs are grouped into a number of partitions. All pairs in the same partition are passed to a reduce function which produces the final results. Application developers are responsible for implementing the map and reduce functions and the system handles all the remaining mechanism. The worker nodes process map and reduce functions and results are returned to master server.

Dynamic Partitioning Based Application Offloading: Dynamic partitioning of the intensive mobile application at runtime is a robust technique for coping with the dynamic processing loads on SMD. Current dynamic partitioning approaches analyze the resources consumption of SMDs, computational requirements of the mobile application and search for runtime solving of the problem of resource limitations on SMD. In [49] a middleware framework is introduced for sharing the application processing load on SMD dynamically between cloud server node and mobile client. Objective of the framework is to deploy the application in optimal mode by automatically and dynamically determining the execution location for modules of an application. Application profiling component of the architecture partitions the application in modules on the basis of its behavior and represents modules in the form of data flow graph known as consumption graph. The framework exercises existent module management such as R-OSGi [48] and deployment tool such as AlfredO [50]. The framework implements both static partitioning and dynamic

partitioning strategies. K-Step and ALL algorithms are used for application partitioning. K-Step is deployed for dynamic partition at runtime whereas ALL is employed for static partitioning of the application. Preprocessing of the consumption graph is performed before running the algorithm to reduce search space. Preprocessing separates local and remote bundles of the application. It looks for the bundles (application modules) that produce a very high cost in offloading and for that reason are not feasible for offloading. The framework employs both static and dynamic partitioning algorithms for the establishment of runtime distributed platform between SMDs and cloud datacenters. A significant aspect of the framework is that SMDs are assigned application processing load on the basis of the availability of memory and processing capacity. The framework derives optimal solution for optimization problem in order to optimize different objective functions such as interaction time, communication cost, and memory consumption. The critical aspect of the framework is the runtime partitioning strategy which obliges SMDs for additional computing resources consumption in the dynamic analysis, profiling, synthesizing, partitioning and migration. The framework requires SMD to continuously synchronize with the cloud server node which requires maintaining SMD in active state for the entire duration of distributed platform. Active state of mobile device for a longer period of time results in high energy consumption [51]-[52].

AIDE [34] establishes distributed platform composed of different computing devices such as laptops, PCs, PDAs, and smartphones. The framework is composed of surrogate server and mobile device client. SMD searches for suitable surrogate to share application processing load. The partitioning component of the AIDE partitions the application by following a partitioning policy. The framework exercises class level granularity for partitioning of application. An application profiling component establishes the feasibility of offloading. Application profiler reflects on two parameters, the execution history of the application and prediction of the future resources required for the application. Profiler aims for offloading the components that could improve the performance of the system. Partitions are offloaded to the remote surrogates. AIDE provides a transparent distributed application deployment framework for mobile applications. The sophistication of application migration and remote execution are masked from mobile users. AIDE employs a dynamic partitioning and migration approach for offload processing and employs computing services of the remote hosts in the local distributed environment. AIDE implements distributed execution platform in transparent manner to give the user the notion of application being executed on local device. AIDE incorporates the option to use multiple surrogates for remote execution. The critical aspects of AIDE are that the runtime partitioning of the application requires additional computing resources exploitation for the establishment of distributed platform. AIDE is a decentralized distributed platform for dynamic partitioning and migration; for that reason SMDs need to monitor the execution environment which imposes heavy monitoring overhead on SMD.

Mobile Assistance Using Infrastructure (MAUI) [40] is a dynamic partitioning framework which focuses on energy saving for SMD. MAUI partitions the application dynamically at runtime in which the computational intensive components of the application are offloaded to the cloud server nodes. Programmers annotate the individual methods of the application as local or remote. MAUI profiler determines the remote methods of application to be offloaded to cloud server. Each time a method is called, the profiler component assesses it for energy saving which exploits additional computing resources (CPU, energy) on SMD. MAUI solver decides the destination of execution for the method annotated as remote. The decision of MAUI Solver is based upon the input of MAUI profiler. Proxies of the application are created for execution on both cloud server node and mobile device for communication between local methods and remotely executable methods. MAUI generates a wrapper for each method marked as remote at compile time. The wrapper has similar method type signature, however with two changes; one additional input argument, and one additional return type. Input argument is required for the state transfer of smartphone to MAUI Server through client application proxy. The additional return value is used to transfer the application state back from the server to smart mobile device using server proxy. State of the method is transferred in serialized form.

MAUI is a cloud server based dynamic partitioning framework which considers energy saving on SMD as the main objective function for offload processing. MAUI masks the complexity of remote execution from mobile user and gives the notion as the entire application is being executed on SMD. The framework is based upon method state migration as a substitute of method code migration. MAUI copes with the mobility of the mobile user and provides optimized solution periodically to adapt to the changes in network and user location. The critical aspect of MAUI is the dynamic partitioning of the application at runtime which activates the profiler and solver component dynamically to determine execution point for application partitions. Development of the applications on the basis of MAUI requires additional developmental efforts for annotating the execution pattern of each individual method the application. MAUI deploys full proxies of the application on both SMD and cloud datacenter. MAUI obliges the overhead of dynamic application profiling, solving, partitioning, migration, and reintegration on SMD.

CloneCloud [36] employs dynamic partitioning of the application at runtime. Partitioning and integration of the application occurs at application level dynamically. Partitioning phase of the framework involves; static analysis, application dynamic profiling, and optimization solution. Mobile device uses preprocess migratory thread to assist a process with suspending, packaging, resuming and merging thread states. In [39] the issue of application partitioning between mobile devices and clouds is addressed. Authors model the partition optimization problem through a mathematical expression, which includes cost of execution of each module on mobile device, cost of execution of each module on cloud, and the cost of communication between the two modules. Variant objective functions are considered for partitioning; minimize execution time, minimize battery power consumption or cost of execution of the application. Elastic application model for augmenting the computing capabilities of mobile devices

[14] provides a middleware framework for elastic mobile applications. Application is partitioned into weblets and migrated dynamically between mobile device and remote cloud server. Variant elastic patterns are used for the replication of weblets on the remote cloud. The execution destination for the weblet is determined dynamically at runtime. The framework considers different parameters for offload processing of the weblets such as status of the mobile device, cloud, application performance measures and user preferences which comprise power saving mode, high speed mode, low cost mode and offload mode. The framework considers an optimal cost model for the execution configuration of the weblets. The cost model considers different costing factors such as power consumption, monetary cost, performance attributes and security and privacy.

The framework proposes a security mechanism for ensuring the integrity of communication between SMD and Cloud server [53]. Upon downloading weblet on SMD, the integrity of each weblets is ensured by the installer of the device by recomputing hash value for each weblet and comparing it with the hash value stored in the weblet. The installer registers the application with Device Elasticity Manager (DEM). The DEM maintains a table of installed applications on the device which need elasticity manager support. The table maintains detailed information about weblets such as signed hashed values and migration settings. Several parts of the elastic application are installed on Cloud Elasticity Service (CES). CES maintains installed applications for users. For this purpose users register with CES and authenticate with CES during installation. The cloud based application manager is able to download the application code from an application store instead of uploading from mobile device. The node manager executes the weblet binary provided by application manger. The local weblet can query DEM to obtain the list of all active weblets in the same session. The local weblet can broadcast the URLs returned by DEM to any other weblet that needs to communicate.

The implications of elastic application model [14] are that the framework accomplishes application level partitioning and migration of applications. The framework employs a comprehensive cost model to dynamically adjust execution configurations and optimizes application performance in terms of a set of objectives and user preferences. The framework provides a security mechanism for the authentication and authorization of weblets migration and reintegration and provides support for synchronization between application on mobile device and weblets running on cloud node. The critical aspect is the establishment of runtime distributed platform for SMD which necessitates additional computing resources exploitation for the establishment and management of distribute platform. The framework deploys replication of the application both on the mobile device and application manager of the cloud server. The framework implements a sophisticated mechanism for the migration of weblets between SMD and remote cloud nodes. The framework imposes extensive overhead of application profiling, dynamic runtime partitioning, migration, reintegration, and rigorous synchronization on mobile devices for offload processing. Fig. 7 shows a generic flowchart for application partitioning based offloading frameworks. The profiling mechanism evaluates computing resources requirements of mobile application and the availability of resources on SMD. Profiling mechanism works differently in different frameworks. The critical situation indicates the unavailability of sufficient computing resources on SMD. Therefore, the computational intensive components of the application are separated at runtime. SMD negotiate with cloud servers for the selection of appropriate server node. At that moment partitions of the application are migrated to remote server node for remote processing. Upon successful execution of the remote components of the application, result is returned to main application running on SMD.

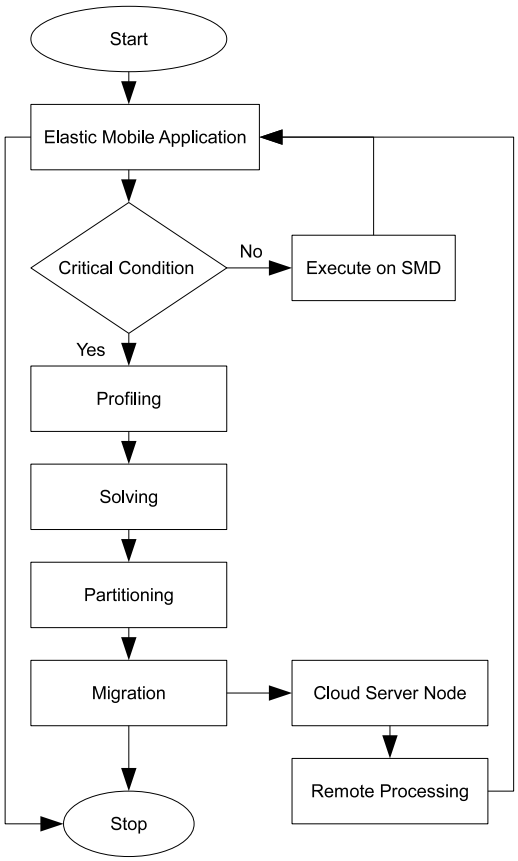

Fig. 7. Flowchart Partitioning Migration Based DAPFs

IV. Comparison of Application Offloading Frameworks by Using Thematic Taxonomy

This section compares current DAPFs on the basis of taxonomy presented in Fig. 4. We classify application offloading frameworks in two categories: (1) local resources utilization frameworks and (2) centralized server resources utilization model for application offloading. It investigates commonalities and deviations in such frameworks on the basis parameters presented in the taxonomy. The comparison parameters considered are: Offloading Scope (OS), Partitioning Approach (PA), Migration Support (MS), Migration Granularity (MG), Developer Support (DS), Migration Pattern (MP) and Execution Monitoring (EM). Table 2 compares local application offloading frameworks, whereas table 3 compares server based application offloading frameworks on the basis of such parameters.

TABLE II
COMPARISON OF LOCAL RESOURCE SHARING BASED APPLICATION OFFLOADING FRAMEWORKS

Framework	OS	PA	MG	MS	MP	SS	DS	EM
Distributed Platform for Resources Constrained Devices [34]	Local Distributed Environment	Dynamic	Class Level	Application Level	Application Transfer	No	Not Required	Decentralized
Secure Cyber Foraging [30]	Local Distributed Environment	n/a	Entire Application	System Level for VM	Binary File Download	Yes	Required	Centralized
Clone cloud [6]	Local Distributed Environment	Static	Entire Application/ Partitioning	System level	VM Instance	No	Required	Decentralized
Optimized Solution for Mobile Devices [38]	Ad-Hoc Cloud of Mobile devices	n/a	Entire Application	Application Level	UMSC	No	Not Required	Decentralized
VM-Based Cloudlets [7]	Local Distributed Environment	n/a	Entire Application	System Level	VM Instance	No	Not Required	Decentralized
DISHES [43]	Local Distributed Environment	n/a	Entire Application offloading	System Level	Binary File Download	No	Not required	Centralized
Virtual Cloud Computing [13]	Ad-hoc Mobile Cloud	n/a	Entire Application	Application Level	Application Transfer	Yes	Not required	Decentralized
MISCO [35]	Cloud of mobile nodes	Static	Method level	Application Level	Binary File Download	No	Required	Centralized

The offloading scope a application offloading framework indicates the scope of distributed platform established at runtime. Current DAPFs implement application offloading by accessing the services of local computing devices or remote cloud server nodes. In the local resources utilization model the services and resources of the local computing nodes are utilized. The traditional local DAPFs implement application offloading by employing the following three different types of decentralized local computing resources utilization approaches. a) Surrogate based distributed model is composed of remote servers which are accessible in the local environment. The surrogate servers can be stationary or mobile remote computer which is accessible in the local environment of SMD. Mobile device is enabled to select an appropriate surrogate at runtime for application offloading. In such model a centralized monitors the establishment of distributed platform and provision of computing resources [30]. b) Mobile devices based distributed platform is a virtual or ad-hoc cloud computing model in which distributed SMDs in the close proximity participate in resources sharing [13], [38]. The peer SMDs are enabled to share computing resources and provide remote services. In such an environment, sharing of the computing resources and services is restricted to the computing capabilities mobile devices and services shared by SMDs involved in the virtual cloud environment. The unavailability of centralized mechanism for the establishment and management of virtual cloud is a challenging aspect of virtual cloud model. c) In centralized server based mobile devices distributed platform, the computing services are provided by mobile worker nodes. However, a centralized server monitors the established and management of distributed application execution platform [35]. In such a distributed computing model, distributed resources and services provision are restricted to the computing potentials and services of worker nodes (SMDs). Fig. 8 highlights different models employed for the distributed application processing platforms in application offloading.

The partitioning approach attribute of the DAPF represents the partitioning strategy of the framework. Current DAPFs implement application partitioning in two different ways; static partitioning and dynamic partitioning. a) In static partitioning the application is partitioned in fixed number of partitions only once [6], [35], [49]. Static application partitioning is a simple and lightweight approach for application offloading; however it lacks the features of addressing the issue of dynamic workload on SMDs. b) In dynamic partitioning the elastic mobile application is partitioned dynamically at runtime. The dynamic portioning approach is implemented to cope with the issue of dynamic application processing load on SMDs at runtime. Dynamic partitioning of the intensive mobile application at runtime is a robust technique for coping with the dynamic processing loads on SMD [14], [34], [36], [49], [40]. The entire application migration frameworks which do not involve application partitioning are represented as n/a in PA column of Table 3 and Table 4.

The attribute of migration granularity represents the level of granularity of offloading intensive components of the mobile application. The finer granularity level results in outsourcing computational load at refined level. However, refined level granularity requires highly intensive monitoring mechanism on SMD at runtime [6], [7], [38], [35], [14], [34], [36], [40]. The refined level granularity requires resources intensive synchronization mechanism between SMD and cloud server node. Further, finer granularity level has the issue of ensuring consistency in the distributed execution of mobile application. On the other side, the abstract level of granularity results in simple offloading mechanism and requires low monitoring overhead on SMD [6], [13], [30], [42]-[37], [43], [45], [39], [40]. However, abstract level of granularity results in increased data transmission overhead and therefore increases

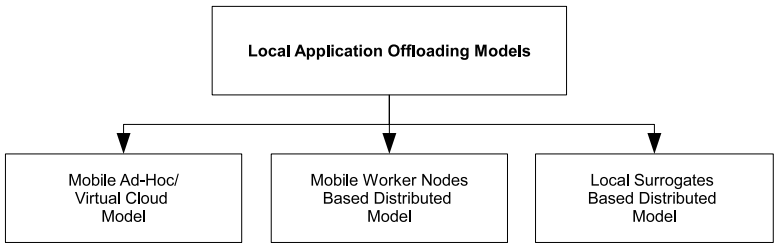

Fig. 8. Local Application Offloading Models

security threats of the outsourcing components of the mobile application. For example, migration of an entire application [7], module or component [49], [54] of the mobile application is more vulnerable to network threats as compared to the outsourcing thread [36] or method [40] of the mobile application. For the reason that eavesdropping of finer level code is less meaningful to the attackers as compared to the entire module or component of the mobile application. Current DAPFs implement the following granularity levels for application offloading. a) Module level migration represents that entire module or bundle of the application is migrated to remote environment [49], [54]. b) Method level migration represents that partitioning occurs at the method level and intensive methods of the application are migrated to remote server [40]. c) Object level migration represents that entire object is migrated to remote environment for outsourced processing [45], [39]. d) Thread level migration represents thread level partitioning and migration of the application to remote environment [36]. e) Entire application migration in which case entire application is offloaded to remote server [6], [7], [13], [30], [42], [37], [38]. The attribute of migration support represents the level of support required for the migration of application. The mechanism of application offloading increases resources utilization on SMD. Specifically, the resources utilization for the operating system increases in application offloading mechanism. Currently, a number of application offloading frameworks require additional support from the operating system; such as VM deployment and management [6], [7], [30], [43], [36]. Such offloading frameworks are represented as System Level in Table 3 and Table 4. On the other hand, the latest application offloading frameworks implement component offloading at application level and do not require additional support for component outsourcing [13], [35], [14], [34], [42]-[37], [38], [45], [39], [49], [55], [47]]. Such frameworks are repressed as Application Level in Table 3 and Table 4.

The attribute of migration pattern represents the mechanism of transfer intensive applications to remote server nodes at runtime. Current DAPFs implement the mechanism of application offloading in a number of ways. The following migration patterns are practiced for traditional application offloading frameworks. a) Application transfer is a migration pattern in which the binary code of the application is outsourced to remote server [13], [34], [40]. b) URL download based migration pattern represents a migration pattern in which a URL is provided to remote host and application is downloaded from that URL as a substitute of transferring the application directly from SMD [35], [14], [30]. c) VM Instance represents a migration pattern in which the application is encapsulated in VM instance (partially or entirely) and the VM instance is migrated to remote server. A fresh VM instance is created on the remote server and guest VM instance is copied to the freshly created VM on remote server [6], [7], [36], [42], [37]. d) UMSC is a migration pattern in which mobile agent is employed for the migration of outsourcing application [38]. UMSC serves as a courier for the migration of the application between SMD and remote server. e) Module/Bundle transfer represents a migration pattern in which binary files of the modules of the application are migrated to remote servers [49]. f) Application proxy is a migration pattern in which entire proxies of the application are maintained on the local SMD and remote cloud server node [40]. g) Object transfer is a migration pattern in which entire object is outsourced to remote server at application level [45], [39]. The attribute of security support represents the security provision attribute of the DAPFs. Security is an important parameter for application offloading frameworks. Current DAPFs which implement security mechanisms for application offloading are represented with the value Yes, whereas the frameworks which lack in the provision of security mechanism for application offloading are represented with the value No. A number of current DAPFs required developers support for defining execution scope of the components of application at different granularity level. Such approaches restrict application developers to classify and annotate the components of mobile application as local or remote. The attribute of DS shows the requirement of additional support required for the development of the application. Therefore, the traditional application offloading models which require developer support [6], [35], [14], [30], [40] are represented as required, whereas the frameworks which do not require developers support [7], [13], [30], [33], [36], [38]-[43], [49], [40] are represented as Not required. The attribute of execution management shows the management policy for the deployment and management of runtime distributed application platform. Current DAPFs are classified in two categories from the perspective of execution management. Decentralized management represents the deficiency of centralized mechanism for the deployment and management of distributed platform [6]-[7], [13], [30], [38]. Therefore, SMDs are responsible for monitoring distributed platform and distributed application execution. Therefore such frameworks results in larger overhead of the distributed application deployment at runtime. Centralized management represents that a centralized management and monitoring mechanism is provided for the establishment of distributed platform and monitoring of application execution [35]-[14], [36], [42], [37],

TABLE III
COMPARISON OF SERVER BASED APPLICATION OFFLOADING FRAMEWORKS

Framework	OS	PA	MG	MS	MP	SS	DS	EM
Calling the Cloud [49]	Cloud Server	Static/ Dynamic	Modules (Bundle)	Application Level	Bundles Transfer	No	Not Required	Centralized
MAUI [40]	Cloud Server	Dynamic	Method Level	Application Level	Application Proxy	No	Required	Centralized
Dynamically Partitioning Applications [54]	Cloud Server	Dynamic	Module level	Application level	Application Transfer	No	Not Required	Centralized
Energy Savings and Privacy Protection [45]	Grid Server	n/a	Image Object	Application Level	Object Migration	No	Not Required	Centralized
CloneCloud [36]	Cloud Server	Dynamic	Thread	System Level	VM Instance	No	Not Required	Centralized
COGNISERVE [39]	Cloud Server	n/a	Entire Job	Application Level	Object Migration	No	Not Required	Centralized
Elastic Application Model [14]	Cloud Server	Dynamic	Bundles (Weblets)	Application level	URL download	Yes	Required	Centralized
Mirroring Smartphones [37]	TSP Based Server	n/a	Entire Application	Application Level	VM Instance	Yes	Not Required	Centralized
Virtualized Execution Environment [42]	Cloud Server	n/a	Entire Application	Application Level	VM Instance	No	Not Required	Centralized

[43], [45], [39], [49], [40]. The following Table shows the comparison of local DAPFs for smart mobile devices.

Localized application offloading frameworks employ decentralized monitoring approach for process offloading which results in the extensive involvement of SMDs for the management of distributed processing. Further, local offloading frameworks are deficient in the centralized management and the availability of resources for the provision of remote services. In the scenario of unavailability of local remote service provider, remote services become inaccessible which hinders the objectives of availability and scalability of services in distributed computing paradigm. To cope with the issues of decentralized DAPFs centralized server based solutions are exercised. The server based resources utilization models utilize computing resources and services of centralized servers. The availability of centralized servers lessens the monitoring overhead on SMD in the deployment and management of runtime distributed platform. Further, centralized servers assist in ensuring the availability of resources and services. The server based DAPFs implement application offloading by employing the following three different types of centralized computing resources utilization approaches. a) Grid server based distributed platform in which remote services are provided by Grid servers [45]. b) Telecommunication Service Provider (TSP) based distributed platform in which remote services are configured at TSP servers [37]. c) A cloud datacenter based distributed platform is established by leveraging computing power potentials of cloud datacenters. Distributed computing services are provided through service providers by employing the vision of utility computing paradigm [14], [36], [42], [39], [49], [40]. Server based application offloading frameworks accomplish outsourced application processing in diverse modes. Several approaches exploit VM cloning; others focus on part(s) of the application to be offloaded. A number of approaches implement dynamic application partitioning whereas other focus on entire job migration. Diverse objective functions are considered; saving processing power, efficient bandwidth utilization, saving energy consumption, user preferences, and execution cost. Table 3 compares server based offloading frameworks in which centralized resources are available for the provision of remote services.

Server based DAPFs provide centralized management and ensure availability of remote services. However, a number of obstacles obstruct optimization goals of server based remote application processing. In the following section, we discuss issues in current DAPFs and identify challenges for leveraging the application processing services of computational clouds.

V. CHALLENGES AND ISSUES FOR DISTRIBUTED APPLICATION DEPLOYMENT

Table 4 summarizes challenges to current DAPFs and open research issues in cloud datacenters based distributed application processing. Issues indicate the unresolved problems in current DAPFs whereas challenges indicate the issues of research in distributed application processing for MCC that remain to be addressed. The following section discusses issues in current offloading frameworks and identifies challenges to the cloud based application processing of resources intensive mobile applications.

A. Scalability and Availability of Services and Resources

Scalability of services is a challenging aspect of distributed application processing in mobile cloud computing. The traditional local DAPFs [6], [7], [13], [16], [35], [34] for remote application processing are deficient in centralized management of the distributed platform. A challenging issue in local DAPFs is the unavailability of centralized resources. For example; in the scenario of unavailability of remote service provider, remote services become inaccessible which hinders the objectives of availability of services in distributed computing paradigm. Similarly, local resources are accessible to limited number of mobile devices in the local environment. Therefore,

TABLE IV
CHALLENGES AND ISSUES IN DISTRIBUTED APPLICATION PROCESSING FOR MCC

	Challenge to Current DAPFs	Open Issues in Distributed Processing for MCC
Unavailability of centralized resources and services in local distributed models.		Yes
Successful execution of remote processing and returning result to SMD in ad-hoc/ virtual distributed models.	Yes	Yes
Optimal operating procedure with minimum possible computing resources utilization on SMD for the establishment and management of distributed Platform.	Yes	Yes
Scalability of Services and resources.	Yes	Yes
Minimum resources exploitation on SMD while sustaining elasticity of mobile applications.	Yes	
Minimum dynamic runtime profiling and solving overhead on SMD for the establishment of runtime distributed processing platform.	Yes	Yes
Lightweight distributed management mechanism on SMD.	Yes	Yes
Ensuring trustworthy remote processing environment.	Yes	Yes
Providing authorized access to legitimate mobile users.	Yes	
Integrity of offloading components of the application.	Yes	
Convenient developmental and deployment procedures.	Yes	
Deficiency of design level support for distributed processing of mobile application.		Yes
Analogous extension of cyber foraging to centralized cloud based remote processing		Yes
Transparent distributed platform.	Yes	Yes
Coping with intrinsic limitations of wireless medium.	Yes	
Seamless Connectivity to cloud servers.	Yes	
Heterogeneity of mobile device architecture and operating system platforms.	Yes	Yes

the possibility of inaccessibility of the remote services always remains associated in local distributed models. Whereas, scalable systems ensure the provision of services irrespective of the number of clients accessing the services. Therefore, unavailability of centralized resources and services and scalability of services is a challenging research issue for ad-hoc and virtual distributed models of MCC [13], [38]. It is challenging to implement peaceful degradation policy on SMDs in the critical conditions of unavailability of remote services. Scalable systems sustain the provision of services and resources for large number of clients whereas availability of services ensures the provision of remote services. It is imperative to ensure the scalability of services in cloud datacenters so that SMDs are enabled to access centralized services for distributed application deployment with high aim of scalable remote services. In centralized datacenter based computational cloud are resources rich and computational resources and services are provided on demand basis. Cloud resources and services are accessible to both stationary computer clients and SMDs. However, the unique architecture, compact design, operating platforms, low computing potentials, and portable mobile nature of smart mobile devices require special services for ensuring the availability of cloud services. The mobile nature and the intrinsic limitations associated with the wireless access medium of SMD necessitate availability of cloud services and resources homogeneously worldwide. It is challenging in cloud based processing of mobile applications to ensure the availability of services and identical access to cloud services over different types of wireless network technologies (Wi-Fi, 3G and LTE). Therefore, sustaining uninterrupted provision of cloud services and resources to SMDs is a challenging research perspective of mobile cloud computing.

B. Distributed Application Deployment

In current DAPFs, resources intensive distributed platform is established at runtime. Mobile applications offloading frameworks are developed on the basis of standalone application architecture, whereas the processing of application is performed in the distributed fashion. As a result, current DAPFs establish a resources intensive and complex computing environment at runtime. Application offloading techniques are primarily based on either entire application/job migration or application partition migration to remote servers. The implementation of distributed architecture for virtual mobile cloud is hindered by the following obstructs. a) Local distributed processing models lack in the availability of centralized management; for that reason it is difficult to configure explicitly defined client and server components for the mobile applications. b) Virtual clouds necessitate special requirements for the establishment of distributed platform which is challenging to maintain for mobile devices which are participating in ad-hoc cloud. The special requirements include; SMDs remain in the close proximity, follow the same movement patterns, voluntariness for service and provision, implementation of specific service architecture [13]. SMDs in the virtual cloud exploit additional computing resources for the configuration of distributed platform and management of distributed services provision to the requesting client devices. Further, shorter battery life time of SMDs is major challenge in virtual/ad-hoc distributed application processing models. Therefore, the ad-hoc and virtualized nature of local distributed platform is another obstacle in explicitly defining client and server components of the mobile application. However, the availability of centralized resources and services and centralized management mechanism in cloud datacenters are the motivating factors for incorporating distributed architecture for the intensive mobile applications. The implementation of client/server model can be a potential alternative for the traditional standalone intensive mobile applications for mobile cloud computing. On the other hand, traditional client/server model has the limitations of reliability of client application on server application. Applications are configured in such a manner so that client applications remain dependent on the

server application. Whereas, the wireless access medium is the main inhibiting factor for implementing highly dependent client/server model for intensive mobile applications in MCC. Hence, it is challenging for distributed mobile application to incorporate the principles of distributed applications in such a manner so that mobile applications can operate in the situations of inaccessibility of cloud server nodes.

C. Seamless Connectivity and Consistent Distributed Platform

Mobility is an important attribute of SMDs. Mobile users enjoy the freedom of computing and communication on move. However, a number of obstacles hinder the goals of seamless connectivity and consistency in the distributed platform of mobile applications; for example handoffs, traveling with high speed, diverse geographical locations and different environmental conditions. As a result, providing seamless connectivity and uninterrupted access to the centralized cloud datacenters in distributed application processing is a serious research issue for MCC. It is important that distributed application model provide versatile access to cloud resources and services on move with ubiquitous attributes and high degree of transparency. However, it is challenging to ensure the transparency of distributed environment. In particular to SMD, the issues and limitations in wireless medium hinder the transparency goals of distributed processing of mobile application. The seamless and transparent deployment of distributed platform for computational intensive applications is a challenging aspect for mobile cloud computing. It is mandatory for distributed model to mask the complexities of distributed environment from mobile user and give the notion as the entire application is being processed locally on SMD. Similarly, it is important to ensure successful execution of remote processing and returning results to SMD. Sustaining consistency of the offloaded components of the application with lightweight implementation procedures is a challenging aspect of DAPFs. Consistency is an issue for the components offloaded at runtime [14], the replicated applications using proxies [40], and transactions involving related updates to different objects. It is important that the distribution and replication of intensive mobile applications and data should be transparent to the mobile users and application running client device. Cloud based distributed processing of mobile application are required to fulfill Atomic, Concurrency, Isolation and Durability (ACID) properties of the distributed systems. It is challenging to provide location transparency, replica transparency, concurrency transparency, and failure transparency in cloud based application processing of mobile applications.

D. Homogenous and Optimal Distributed Platform

Homogenous and optimal cloud based application processing is an important research perspective in mobile cloud computing. Heterogeneity of SMD architecture and operating platform is challenging for distributed application processing in MCC. Mobile device vendors employ different hardware architecture and operating system platforms for the specific mobile product. Traditional application offloading frameworks focus on the implementation of platform dependent procedures for outsourcing computational intensive loads. For example, Weblets [14] and MAUI [40] are application offloading frameworks which are applicable for .Net framework, whereas virtualized execution framework [42] and mirror server [37] are suitable frameworks for android platform. Therefore, homogenous access to cloud services are highly expected wherein SMD are enabled to access widespread computing services of computational clouds irrespective of the concerns about operating hardware architecture and operating system platform. A homogenous distributed application deployment solution for the heterogeneous available SMDs platforms is a challenging issue for MCC. In [56], we propose a tripod of requirements with three legs of trust, energy efficiency, and ubiquity. It describes important metrics such as heterogeneity, under this tripod which are crucial for the success of cloud-mobile applications. Similarly, the deployment of distributed application processing platform at runtime [13], [14], [30], [41], [40] is a resources intensive mechanism. It uses computing resources on SMDs for the evaluation of computing resources utilization on SMDs and partitioning of intensive mobile applications at runtime. Current, DAPFs necessitate continuous assessment of application execution requirements on SMD which is a resource intensive operation. DAPFs employ runtime profiling and solving mechanism on SMDs periodically or casually to evaluate application processing requirements and the availability of computing resources on SMD [14], [40]. The centralized distributed application deployment models require arbitration of SMD with centralized server for the selection of appropriate server node. As a result, computing resources (CPU, battery power) of SMD are exploited abundantly for the entire process of application profiling and solving. The deployment of distributed platform, management and operation of remote application processing in the optimal possible fashion is an important perspective of cloud based application processing. It is challenging to provide homogenous solution for heterogeneous devices, operating platforms and network technologies with minimum possible resources utilization on the SMDs.

E. Security and Privacy in Cloud Based Application Processing

Privacy in the distributed platform and security of data transmission between mobile device and cloud server node are important concerns in cloud based application processing. Privacy measures are required to ensure the execution of mobile application in isolated and trustworthy environment, whereas security procedures are required to protect against network threats. Security and privacy are very important aspects for the establishing and maintaining the trust of mobile users in cloud based application processing. Security in MCC is important from three different perspectives: security for mobile devices, security for data transmission over the wireless medium and security in the cloud datacenter nodes. SMDs are subjected to a number of security threats such as viruses and worms. SMDs are the attractive targets for attacker. According to a report [57] the number of new susceptibilities in mobile operating systems increased 42 percent from 2009 to 2010. The number and sophistication of attacks on mobile phones is

increasing speedily as compared to the countermeasures. Data transmission over the wireless networks is highly vulnerable to network security threats. For example, using radio frequencies, the risk of interruption is higher than with wired networks therefore attacker can easily compromise confidentiality [58]. Similarly, in cloud datacenters the security threats are associated with the transmission between physical elements on the network, and traffic between the virtual elements in the network, such as between virtual machines within a single physical server. Therefore, in order to leverage the application processing services of computational clouds, a highly secure environment is expected at all the three entities of MCC model. In current DAPFs, transmission of the running states of mobile application which is encapsulated in VM [6], [7], [49], [42] or binary transfer of the application code at runtime [35], [30], [43], [49] is continuously subjected to security threats at mobile device, wireless medium and cloud datacenters. Therefore, secure transmission of the entire components of the application is a challenging issue for MCC. It is imperative to implement reliable security measures for the data transmission, and synchronization between SMD and cloud datacenters in distributed processing platform. Similarly, access control, fidelity and privacy of distributed application components in the remote cloud datacenters is an important consideration for the distributed application processing in MCC. Cloud datacenters provide augmentation services which are unapproachable to mobile users. Therefore, it is highly demanding to ensure the privacy of data and computing operations in remote server nodes. A trustworthy distributed application model is highly expected to cope with such important issues and ensure the trustworthiness of remote computing environment. A reliable distributed environment is expected to provide authentic access to authorized mobile user for legitimate operations on cloud server nodes. Considering the aforementioned research issues and challenges for distributed application deployment in MCC, lightweight and optimal distributed application deployment solution is extremely important. Such a solution should incorporate optimal procedures for the development, deployment and management of runtime distributed platform for MCC. In [59] we propose a lightweight distributed model for computational intensive mobile applications.

VI. Conclusions and Future Directions

The paper discusses the concept of cloud computing, mobile cloud computing and explains the different techniques to augment smart mobile devices resources based on resources available within the cloud. It analyzes current DAPFs by using thematic taxonomy and highlights the commonalties and deviations in such frameworks on the basis of significant parameters. It discusses issues in current DAPFs and highlights challenges to optimal and lightweight distributed application frameworks for MCC. Current DAPFs accomplish process offloading in diverse modes. Several approaches exploit entire application migration; others focus on part(s) of the application to be offloaded. A number of approaches employ static partitioning, others exercise dynamic partitioning. Variant migration patterns are used; downloading application by providing URL to remote host, VM cloning, Mobile agent such as USMC, application binary transfer and use of proxies. Diverse objective functions are considered; saving processing power, efficient bandwidth utilization, saving energy consumption, user preferences, and execution cost. Objective of all approaches is to augment the application processing potentials of resources constrained SMDs. We conclude that current DAPFs for MCC are the analogous extensions of traditional cyber foraging frameworks for pervasive computing or local distributed platforms. Hence, current DAPFs are deficient in the deployment of distributed system standard architectures. As a result, additional complications arise in the development, deployment and management of distributed platform. Current frameworks focus on the establishment of runtime distributed platform which results in the resources intensive management overheads on SMDs for the entire duration of distributed platform. SMDs exploit computing resources in arbitration with cloud servers for the selection of appropriate remote node, dynamic assessment of SMDs resources consumption and application execution requirements at runtime, dynamic application profiling, synthesizing and solving for application outsourcing, application migration and reintegration and rigorous synchronization with cloud servers for the entire duration of distributed platform. As a result, additional computing resources of the SMDs are exploited for the runtime orchestration of distributed platform. Therefore, current distributed application deployment algorithms employ heavyweight procedures for distributed application deployment and management.

The mobile nature, compact design, limited computing potential and wireless medium attributes of SMDs necessitate for optimal, lightweight and rich local services procedures for distributed application deployment in MCC. The incorporation of standardized design and development principles of distributed systems seem to be an optimal solution for coping with the challenges of lightweight distributed application deployment for MCC. The incorporation of distributed client/server architecture of distributed applications with the elastic features of the traditional offloading frameworks appears to be an appropriate optimal solution for addressing the issues of current DAPFs for MCC. The development of such lightweight model will result in reducing developmental efforts and enhancement in overall performance of application deployment, management and processing in mobile cloud computing.

Acknowledgment

This work is carried out as part of the Mobile Cloud Computing research project funded by the Malaysian Ministry of Higher Education under the University of Malaya High Impact Research Grant with reference UM.C/HIR/MOHE/FCSIT/03. A research in CLOUDS Lab at the University of Melbourne is funded by the Australian Research Council (ARC) under its Discovery and Linkage Projects programs.

References

[1] R. Balan, D. Gergle, M. Satyanarayanan, and J. Herbsleb, "Simplifying cyber foraging for mobile devices," in *Proc. 5th international conference on Mobile systems, applications and services*. ACM, 2007, pp. 272–285.

[2] J. Flinn, S. Park, and M. Satyanarayanan, "Balancing performance, energy, and quality in pervasive computing," in *Distributed Computing Systems, Proceedings. 22nd International Conference on*. IEEE, 2002, pp. 217–226.

[3] Y. Su and J. Flinn, "Slingshot: Deploying stateful services in wireless hotspots," in *Proc. 3rd international conference on Mobile systems, applications, and services*. ACM, 2005, pp. 79–92.

[4] M. Kristensen and N. Bouvin, "Developing cyber foraging applications for portable devices," in *Portable Information Devices, 2008 and the 2008 7th IEEE Conference on Polymers and Adhesives in Microelectronics and Photonics. PORTABLE-POLYTRONIC 2008. 2nd IEEE International Interdisciplinary Conference on*. IEEE, 2008, pp. 1–6.

[5] J. Porras, O. Riva, and M. Kristensen, "Dynamic resource management and cyber foraging," *Middleware for Network Eccentric and Mobile Applications*, vol. 1, p. 349, 2009.

[6] B. Chun and P. Maniatis, "Augmented smartphone applications through clone cloud execution," in *Proc. 8th Workshop on Hot Topics in Operating Systems (HotOS), Monte Verita, Switzerland*, 2009.

[7] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[8] M. Sharifi, S. Kafaie, and O. Kashefi, "A survey and taxonomy of cyber foraging of mobile devices," *IEEE Commun. Surveys Tuts.*, 2011.

[9] (Accessed on 20th July 2011) Amazon s3. [Online]. Available: http://status.aws.amazon.com/s3-20080720.html

[10] (Accessed on 15th July 2011) Google docs. [Online]. Available: http://docs.google.com

[11] (Accessed on 15th June 2011.) Mobileme. [Online]. Available: http://en.wikipedia.org/wiki/MobileMe

[12] (Accessed on 15th July 2011.) Dropbox. [Online]. Available: http://www.dropbox.com

[13] G. Huerta-Canepa and D. Lee, "A virtual cloud computing provider for mobile devices," in *Proc. 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*. ACM, 2010, p. 6.

[14] X. Zhang, A. Kunjithapatham, S. Jeong, and S. Gibbs, "Towards an elastic application model for augmenting the computing capabilities of mobile devices with cloud computing," *Mobile Networks and Applications*, vol. 16, no. 3, pp. 270–284, 2011.

[15] R. Buyya, C. 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*, vol. 25, no. 6, pp. 599–616, 2009.

[16] K. Kumar and Y. Lu, "Cloud computing for mobile users: Can offloading computation save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, 2010.

[17] R. Barga, D. Gannon, and D. Reed, "The client and the cloud: Democratizing research computing," *IEEE Internet Computing*, vol. 15, no. 1, pp. 72–75, 2011.

[18] A. Fox, R. Griffith *et al.*, "Above the clouds: A berkeley view of cloud computing," *Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Tech. Rep. UCB/EECS*, vol. 28, 2009.

[19] (Accessed on 15th September 2012.) What is aws. [Online]. Available: http://aws.amazon.com/

[20] M. Kristensen, "Enabling cyber foraging for mobile devices," in *Proc. 5th MiNEMA Workshop: Middleware for Network Eccentric and Mobile Applications*. Citeseer, 2007, pp. 32–36.

[21] C. Wesley. (Accessed on 15th September, 2012) What is google app engine? [Online]. Available: https://ep2012.europython.eu/conference/talks/google-app-engine-best-practices-latest-features

[22] (Accessed on 15th September, 2012) What is windows azure? [Online]. Available: http://www.microsoft.com/bizspark/azure/

[23] R. Calheiros, C. Vecchiola, D. Karunamoorthy, and R. Buyya, "The aneka platform and qos-driven resource provisioning for elastic applications on hybrid clouds," *Future Generation Computer Systems*, vol. 28, no. 6, pp. 861–870, 2012.

[24] "White paper, mobile cloud computing solution brief, aepona," November 2010.

[25] M. Ali, "Green cloud on the horizon," pp. 451–459, 2009.

[26] H. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless Communications and Mobile Computing*, 2011.

[27] S. Abolfazli, Z. Sanaei, and A. Gani, "Mobile cloud computing: A review on smartphone augmentation approaches," in *Proc. 1st International Conference on Computing, Information Systems and Communications*, 2012.

[28] W. Zheng, P. Xu, X. Huang, and N. Wu, "Design a cloud storage platform for pervasive computing environments," *Cluster Computing*, vol. 13, pp. 141–151, 2010.

[29] M. Satyanarayanan, "Pervasive computing: Vision and challenges," *IEEE Pers. Commun.*, vol. 8, no. 4, pp. 10–17, 2001.

[30] S. Goyal and J. Carter, "A lightweight secure cyber foraging infrastructure for resource-constrained devices," in *Mobile Computing Systems and Applications, 2004. WMCSA 2004. Sixth IEEE Workshop on*. IEEE, 2004, pp. 186–195.

[31] J. Oh, S. Lee, and E. Lee, "An adaptive mobile system using mobile grid computing in wireless network," *Computational Science and Its Applications-ICCSA 2006*, pp. 49–57, 2006.

[32] C. Li and L. Li, "Energy constrained resource allocation optimization for mobile grids," *Journal of Parallel and Distributed Computing*, vol. 70, no. 3, pp. 245–258, 2010.

[33] Y. Begum and M. Mohamed, "A dht-based process migration policy for mobile clusters," in *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on*. IEEE, 2010, pp. 934–938.

[34] A. Messer, I. Greenberg, P. Bernadat, D. Milojicic, D. Chen, T. Giuli, and X. Gu, "Towards a distributed platform for resource-constrained devices," in *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*. IEEE, 2002, pp. 43–51.

[35] A. Dou, V. Kalogeraki, D. Gunopulos, T. Mielikainen, and V. Tuulos, "Misco: a mapreduce framework for mobile systems," in *Proc. 3rd International Conference on PErvasive Technologies Related to Assistive Environments*. ACM, 2010, p. 32.

[36] B. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: elastic execution between mobile device and cloud," in *Proc. sixth conference on Computer systems*, 2011, pp. 301–314.

[37] B. Zhao, Z. Xu, C. Chi, S. Zhu, and G. Cao, "Mirroring smartphones for good: A feasibility study," *Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pp. 26–38, 2012.

[38] Q. Liu, X. Jian, J. Hu, H. Zhao, and S. Zhang, "An optimized solution for mobile environment using mobile cloud computing," in *Wireless Communications, Networking and Mobile Computing, 2009. WiCom'09. 5th International Conference on*. IEEE, 2009, pp. 1–5.

[39] R. Iyer, S. Srinivasan, O. Tickoo, Z. Fang, R. Illikkal, S. Zhang, V. Chadha, P. Stillwell, and S. Lee, "Cogniserve: Heterogeneous server architecture for large-scale recognition," *IEEE Micro*, vol. 31, no. 3, pp. 20–31, 2011.

[40] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: making smartphones last longer with code offload," in *Proc. 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 49–62.

[41] K. Wang, J. Rao, and C. Xu, "Rethink the virtual machine template," in *ACM SIGPLAN Notices*, vol. 46, no. 7. ACM, 2011, pp. 39–50.

[42] S. Hung, T. Kuo, C. Shih, J. Shieh, C. Lee, C. Chang, and J. Wei, "A cloud-based virtualized execution environment for mobile applications," *ZTE Communications*, vol. 9, no. 1, pp. 15–21, 2011.

[43] C. Lai and R. Ko, "Dishes: A distributed shell system designed for ubiquitous computing environment," *International Journal of Computer Networks & Communications*, vol. 2, no. 1, pp. 66–83, 2010.

[44] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[45] J. Liu, K. Kumar, and Y. Lu, "Tradeoff between energy savings and privacy protection in computation offloading," in *Proc. 16th ACM/IEEE international symposium on Low power electronics and design*. ACM, 2010, pp. 213–218.

[46] D. Wu and W. Tsai, "A steganographic method for images by pixel-value differencing," *Pattern Recognition Letters*, vol. 24, no. 9, pp. 1613–1626, 2003.

[47] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, G. Pelosi, and P. Samarati, "Preserving confidentiality of security policies in data outsourcing," in *Proc. 7th ACM workshop on Privacy in the electronic society*. ACM, 2008, pp. 75–84.

[48] J. Rellermeyer, O. Riva, and G. Alonso, "Alfredo: an architecture for flexible interaction with electronic devices," in *Proc. 9th ACM/IFIP/USENIX International Conference on Middleware*. Springer-Verlag New York, Inc., 2008, pp. 22–41.

[49] 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, 2009.

[50] O. Alliance, "Osgi service platform, core specification, release 4, version 4.1," *OSGi Specification*, 2007.

[51] I. Kelényi and J. Nurminen, "Bursty content sharing mechanism for energy-limited mobile devices," in *Proc. 4th ACM workshop on Per-*

formance monitoring and measurement of heterogeneous wireless and wired networks. ACM, 2009, pp. 216–223.

[52] M. Pedersen and F. Fitzek, "Implementation and performance evaluation of network coding for cooperative mobile devices," in *Communications Workshops, 2008. ICC Workshops' 08. IEEE International Conference on*. IEEE, 2008, pp. 91–96.

[53] X. Zhang, J. Schiffman, S. Gibbs, A. Kunjithapatham, and S. Jeong, "Securing elastic applications on mobile devices for cloud computing," in *Proc. 2009 ACM workshop on Cloud computing security*. ACM, 2009, pp. 127–134.

[54] B. Chun and P. Maniatis, "Dynamically partitioning applications between weak devices and clouds," in *Proc. 1st ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond*. ACM, 2010, p. 7.

[55] B. Nguyen, S. Yoon, and H. Lee, "Multi bit plane image steganography," *Digital Watermarking*, pp. 61–70, 2006.

[56] Z. Sanaei, S. Abolfazli, A. Gani, and R. Khokhar, "Tripod of requirements in horizontal heterogeneous mobile cloud computing," 2012.

[57] (Accessed on 15th September,2012) Protecting portable devices: Physical security. [Online]. Available: http://www.us-cert.gov/cas/tips/ST04-017.html

[58] M. Choi, R. Robles, C. Hong, and T. Kim, "Wireless network security: Vulnerabilities, threats and countermeasures," *International journal of Multimedia and Ubiquitous Engineering*, vol. 3, no. 3, 2008.

[59] M. Shiraz, A. Gani, and R. Khokhar, "Towards lightweight distributed applications for mobile cloud computing," in *Computer Science and Automation Engineering (CSAE), 2012 IEEE International Conference on*, vol. 1. IEEE, 2012, pp. 89–93.

Muhammad Shiraz is an Assistant Professor at Department of Computer Science, Federal Urdu University of Arts, Sciences and Technology Islamabad, Pakistan. He completed under graduation from CECOS University Peshawar Pakistan with the distinction of Gold medal. He has Completed Masters in Computer Science from Allama Iqbal Open University Islamabad Pakistan in 2007. Currently, he is pursuing his PhD Candidature under Commonwealth Scholarship Program at Faculty of Computer and Information Technology University of Malaya, Malaysia. He is an active researcher in the Mobile Cloud Computing Research Group at Faculty Computer Science and Information Technology University Malay Kuala Lumpur. His areas of interest include distributed applications design for Ubiquitous Networks, Distributed Systems, Lightweight Applications, Smart Client Applications and Optimization Strategies, Mobile Cloud Computing.

Abdullah Gani is an Assoc. Prof. at the Dept of Computer System and Technology, University of Malaya, Malaysia. His academic qualifications were obtained from UK's universities - bachelor and master degrees from the University of Hull, and Ph.D from the University of Sheffield. He has vast teaching experience due to having worked in various educational institutions locally and abroad - schools, teaching college, ministry of education, and universities. His interest in research started in 1983 when he was chosen to attend Scientific Research course in RECSAM by the Ministry of Education, Malaysia. More than 100 academic papers have been published in conferences and respectable journals. He actively supervises many students at all level of study - Bachelor, Master and PhD. His interest of research includes self-organized system, reinforcement learning and wireless-related networks. He is now working on mobile cloud computing with High Impact Research Grant of 1.5 M for the period of 2011-2016.

Rashid Hafeez Khokhar did his M.S (Statistics) and M.S (Information Technology) from University of the Punjab (Pakistan) and Preston University (Pakistan) respectively. He received his M.S (by research) and PhD in Computer Sciences from Universiti Teknologi Malaysia (Malaysia). Currently, he is working as Senior Lecturer in the Faculty of Computer Science and Information Technology, University of Malaya. His areas of interest are Mobile Cloud Computing, Vehicular & Mobile Ad Hoc Network (Architectures, Protocols, Security, and Algorithms), Artificial Intelligence and Real-Time Communications, and High Performance Computing.

Rajkumar Buyya is Professor of Computer Science and Software Engineering; and Director of the Cloud Computing and Distributed Systems (CLOUDS) Laboratory at the University of Melbourne, Australia. He is also serving as the founding CEO of Manjrasoft Pty Ltd., a spin-off company of the University, commercialising its innovations in Grid and Cloud Computing. He received B.E and M.E in Computer Science and Engineering from Mysore and Bangalore Universities in 1992 and 1995 respectively; and Doctor of Philosophy (PhD) in Computer Science and Software Engineering from Monash University, Melbourne, Australia in 2002. He was awarded Dharma Ratnakara Memorial Trust Gold Medal in 1992 for his academic excellence at the University of Mysore, India. He received Richard Merwin Award from the IEEE Computer Society (USA) for excellence in academic achievement and professional efforts in 1999. He received Leadership and Service Excellence Awards from the IEEE/ACM International Conference on High Performance Computing in 2000 and 2003. He received "Research Excellence Awards" from the University of Melbourne for productive and quality research in computer science and software engineering in 2005 and 2008. He is one of the highly cited authors in computer science and software engineering worldwide (h-index=57, g-index=124, 17000+ citations). The Journal of Information and Software Technology in Jan 2007 issue, based on an analysis of ISI citations, ranked Dr. Buyya's work (published in Software: Practice and Experience Journal in 2002) as one among the "Top 20 cited Software Engineering Articles in 1986-2005". He received the Chris Wallace Award for Outstanding Research Contribution 2008 from the Computing Research and Education Association of Australasia, CORE, which is an association of university departments of computer science in Australia and New Zealand. Dr. Buyya received the "2009 IEEE Medal for Excellence in Scalable Computing" for pioneering the economic paradigm for utility-oriented distributed computing platforms such as Grids and Clouds.