Contents lists available at Science-Gate

International Journal of Advanced and Applied Sciences

Journal homepage: http://www.science-gate.com/IJAAS.html

Architectural patterns for reuse-driven development of mobile cloud computing systems

Abdulrahman Alreshidi *

College of Computer Science and Engineering, University of Ha'il, Ha'il, Saudi Arabia

ARTICLE INFO

Article history:
Received 25 February 2020
Received in revised form
12 June 2020
Accepted 21 June 2020

Keywords:
Mobile cloud systems
Patterns and frameworks
Software architecture
Software reuse

ABSTRACT

Mobile and pervasive systems facilitate enterprises and their users to rely on portable and context-aware computing but suffer from lack of power, computation, and storage resources due to limited hardware of mobile devices. In comparison, the cloud computing model provides access to on-demand and virtually unlimited computing and storage services. Mobile Cloud Computing Systems (MCCS) that combines context-awareness features of mobile computing with access to cloud computing services can enable users to exploit systems that are portable, context-sensitive with backend computation and storage resources. MCCS enables mobility and context awareness with computation and storage services to provide systems that are portable, yet resource sufficient. In an architectural context for MCC systems that require context-awareness, security, and privacy, scalability, multitenancy, and service composition, etc., there is a need to capitalize on reusable solutions–utilizing patterns and best practices–to architect and develop mobile cloud software. This research aims to build and exploit a catalog of patterns that support reusable design knowledge for architecture-based development of MCC systems. We discovered three patterns as generic and reusable solutions and demonstrated pattern-driven architecting of mobile cloud-based on a case study. The results of a case study based evaluation suggest that pattern-based architecting supports reusability and efficiency of system design and development. The solution is the first attempt towards establishing the catalog as patterns repository–facilitating architects with reusable knowledge and practice to develop MCC systems.

1. Introduction

Mobile or handheld systems as a subclass of pervasive computing have emerged as a pervasive technology based on anywhere, anytime–portable, context-sensitive, and connected-mobile devices. Mobile devices are equipped with hardware sensors and installed software applications that empower users to perform a variety of tasks ranging from mobile commerce to health and fitness monitoring (Satyanarayanan, 2011; Picco et al., 2014). However, a mobile device is considered as a resource-constrained computer that lacks the energy, efficiency, and quality of service for computation and memory-intensive tasks (Simanta et al., 2012; Lewis et al., 2013). Cloud computing technology represents the state-of-the-art for client-server computing with on-demand access to hardware and software services corresponding to computation and storage resources (Armbrust et al., 2010). The synergized unification of mobile and cloud are known as Mobile Cloud Computing Systems (MCCS) represent a class of systems where front-end portable and context-aware mobile devices can utilize the backend processing and storage capabilities of the cloud computing (Cox, 2011; Dinh et al., 2013). Despite the benefits of MCCS technology, a number of challenges must be addressed while architecting and developing MCC systems (Satyanarayanan, 2011; Picco et al., 2014; Cox, 2011). Moreover, a rapid demand for developing mobile cloud-based software requires a number of highly knowledgeable and experienced architects who may not be widely available as MCCS have recently emerged as an innovative technology (Picco et al., 2014).

Architectural styles and patterns (The terms patterns and styles are often used interchangeably (Dinh et al., 2013). We focus on patterns (Buschmann et al., 2007) that 'provide a generic,

* Corresponding Author.
Email Address: ab.alreshidi@uoh.edu.sa
https://doi.org/10.21833/ijaas.2020.10.015
Corresponding author's ORCID profile:
https://orcid.org/0000-0002-9034-3909

reusable solution to recurring problems of architectural design,' while 'styles (Pahl et al., 2009) provide a constrained composition of elements for architectural organization and restructuring') have been used for providing packaged knowledge about well-known design solutions to both experienced and novice architects (Buschmann et al., 2007; Pahl et al., 2009). Specifically, design and/or architectural patterns provide a concentrated knowledge and wisdom of engineering practices to provide solutions that are efficient and reusable (Harrison et al., 2007). In recent years, pattern-based approaches resulted in (i) promoting reuse while (iii) decreasing the efforts required during the architectural design and evolution processes (Buschmann et al., 2007; Cámara et al., 2013). In addition, pattern-oriented solutions (Harrison et al., 2007) enhance quality by applying the best practices and knowledge to resolve recurring problems of architectural design (Lewis et al., 2013).

In this research, we aim for pattern-based architecting of MCCS that can accelerate the process of gaining knowledge and experience in successfully modeling and evolving the system's structure and behavior at higher abstractions (Simanta et al., 2012; Dinh et al., 2013). Specifically, we focus on building and exploiting the catalog as a collection of architectural patterns that promote the reuse of design knowledge for architecting MCCS that is currently lacking in existing research (Simanta et al., 2012; Lewis et al., 2013; Kim, 2010). We model and utilize MCCS patterns that support the rationale for reusability and best practices for system engineering and development. While architecting MCCS, one exploits dynamically composed services to develop systems that are portable, yet they have backend cloud services with sufficient computing and storage resources (Lewis et al., 2013; Armbrust et al., 2010; Dinh et al., 2013). In contrast to the research and development in the past (objects, components, and software service systems) (Buschmann et al., 2007; Pahl et al., 2009), patterns for mobile MCCS architectures are characterized by specific requirements such as mobility, context-sensitivity for (front-end) mobile computing with service composition, elasticity, scalability, and multi-tenancy of (backend) cloud services.

One of the key challenges in providing pattern-based architectural knowledge is a systematic discovery and detailed documentation of design or architectural patterns that can be exploited as reuse-driven knowledge and practices to address frequently occurring problems for systems design and development (Buschmann et al., 2007; Harrison et al., 2007). In this paper, we report (i) our empirical effort towards establishing a catalog of architectural patterns for MCC applications; and (ii) demonstrate how the discovered architectural patterns can be applied to architect a mobile cloud system. Our approach consisted of three simple steps, including pattern discovery, pattern documentation, and pattern application. The following three step-process is adopted to develop the proposed solution.

- **Step 1**: Systematic and empirical discovery of architectural patterns as reusable knowledge to engineer and develop MCCS.
- **Step 2**: Template-based documentation of the discovered patterns that can be analyzed and reused during system development.
- **Step 3**: Case study based application of the documented patterns to demonstrate the applicability and reusability of patterns to develop MCCS effectively and efficiently.

Case study based demonstration and results of evaluation suggest that pattern discovery is an exhaustive and time-consuming process. However, once patterns have been discovered and documented, their reuse can save time and effort with design reusability. Pattern-based design and development of MCCS also support the flexibility of design and efficiency of system development. With regards to the existing research in Lewis et al. (2013), Armbrust et al. (2010), Cox (2011), and Dinh et al. (2013), our contributions are to:

- Empirically discover patterns that specifically address mobile cloud requirements to architect context-sensitive, resource-constrained yet scalable and elastic MCCS.
- Exploit discovered patterns as elements of reuse knowledge that guides a step-wise process of pattern-driven and reuse-oriented architecting of MCCS.

This paper is organized as follows. Section 2 presents related research to position the proposed contributions. Section 3 overviews the research method and the proposed solution. Section 4 presents a reference architecture and discovered patterns for MCCS. Section 5 discusses a case study for evaluating the pattern-based architecting of the MCCS. Fig. 6 concludes the paper with a discussion about future research.

2. Related research

In this section, we discuss the most relevant related research to the proposed solution. First, we discuss related work on the design and architectural patterns for mobile and computing systems. We then illustrate a reference architecture that acts as a blueprint for the development of MCCS.

2.1. Design and architectural patterns for mobile cloud systems

Cloud Computing: One of the thorough work on cloud architecture patterns (Wilder, 2012) reports best practices for scalability, big data, fault handling, and distributed services on Windows Azure (Platform as a Service: PaaS). Patterns in Wilder (2012) provided guidelines and practical solutions

to address the scalability and elasticity in cloud-native applications for the Windows Azure platform. Also, the work reported in Harrison et al. (2007) established a catalog of patterns for cloud computing services (e.g., SaaS, PaaS, IaaS), that can be operationalized by means of service deployment models (e.g., community, hybrid, public, private). The patterns are organized in a framework to guide developers to systematically select and apply these patterns. Erl et al. (2015) and CDP (2016) reported a community-driven development of patterns including (1) Cloud Computing Design Patterns (Erl et al., 2015) present a collection of 39 patterns to address the scalability, reliability, security and monitoring issues of cloud applications, (2) Cloud Design Patterns (CDP, 2016) present a collection of patterns created by various (cloud) architects based on the type of problems, and their generic design patterns.

Mobile Computing: Compared to the research on cloud computing patterns, there is less work on pattern-based designing and architecting of mobile solutions. In Lewis et al. (2013), three architectural patterns are presented for mobile computing in the context of a tactical-edge scenario that provides mobile and cloud-based services to military personal in a combat and emergency management context. The proposed architecture patterns, namely data source integration, group-context awareness, and cyber foraging that support availability, elasticity, multi-tenancy, performance, and response time for systems at the tactical edge. The proposed patterns allow system designers and developers to engineer systems in a reusable fashion while ensuring the required functionality and desired quality. In a similar work (Roth, 2002), a collection of architectural solutions, namely mobility patterns, are presented. The proposed mobile system patterns provide fundamental building blocks for mobile systems that require high interactivity, portability, connectivity, and context-sensitivity.

In contrast to the existing research (Harrison et al. 2007; Wilder, 2012; Lewis et al., 2013; Roth, 2002), the proposed patterns in our catalog are focused on architecture-centric development, execution, and management aspects of MCC systems. Moreover, our work aims to establish a pattern catalog-continuously evolving pattern repository-based on an incremental discovery and specification of new architectural patterns guided by Buschmann et al. (2007), Fehling et al. (2011), and Ahmad et al. (2018).

2.2. Reference architectures and patterns for MCC systems

A number of reference architecture (Simanta et al., 2012; Lewis et al., 2012) and a pattern-based solution for mobile service-oriented architecture (SOA) (Kim, 2010) are presented. The technical distinction between a reference architecture and architectural patterns are detailed in bass et al. (2003). In Simanta et al. (2012), the authors have described a reference architecture that enables offloading of (computing and storage-specific) mobile code to cloud computing servers in the context of hostile environments. This paper also presents the most feasible implementation scenarios, along with the highlight of trade-offs mobile cloud systems (i.e., efficiency, response time, availability). A similar work (Lewis et al., 2012) presents a robust architecture that orchestrates connected mobile devices to create a device eco-system to support first responder teams in smart emergency management. In the context of mobile SOA, a taxonomical hierarchy of architectural patterns, namely, standalone, full offloading, partial offloading, SaaS-based, CaaS-based, and offloaded CaaS-based is presented by Kim (2010). For each of the proposed architectural patterns, the authors have organized architectural patterns and their runtime configurations. Different architectural patterns address various quality attributes and architecturally significant requirements, such as efficiency, performance, security, and response time.

This section presents an overview of the most relevant research on the patterns-based architecting of mobile and cloud computing systems. Specifically, the work (Simanta et al., 2012; Lewis et al., 2012; Kim, 2010) on reference architecture and patterns (bass et al., 2003) is most relevant to our proposed solution. Based on an overview of existing research and development along with current challenges for mobile computing (Satyanarayanan, 2011; Picco et al., 2014; Cox, 2011), we claim that the proposed solution is the first attempt to establish a catalog for pattern-driven, architecture-centric (reusable) development of MCCS.

3. Research method and proposed solution

In this section, we first discuss the research methodology that had been adopted to conduct this research (Section 3.1). We also overview the proposed solution for pattern-based architecting of MCCS (Section 3.2).

3.1. Methodology for pattern discovery and documentation

Pattern discovery is based on the design review method (Chen, 1998) that primarily focuses on reviewing the most frequent design space of recurring challenges to architect MCCS (Cox, 2011; Dinh et al., 2013). In our effort, the design review team was comprised of 3 members with experience of (a) conducting the systematic review (Satyanarayanan, 2011; Erl et al., 2015), (b) pattern mining (Ahmad et al., 2018), and (c) development of mobile and cloud systems.

- **Step I:** Conducting Systematic Review on Architectural Solutions for MCC Systems: The review was conducted to investigate the recurring challenges, design problems, and existing solutions

to develop mobile cloud architectures. A systematic review (Alreshidi and Ahmad, 2019) is expected to minimize the potential bias in the review and has a protocol that guides the process. Based on the research questions (RQs) below and the review protocol, we selected 69 studies (problem-solution map) as primary sources of pattern discovery.

RQ1– What methods/techniques/ frameworks/ solutions are provided in existing (research and practices) to model/develop/evolve MCC system architectures?
RQ2– What are the patterns/styles/frameworks to support reusable design knowledge for architecting MCC systems?

- **Step II:** Identification of Pattern Data Sets: Once the studies were identified, we extracted the data sets in Table 1 from selected studies. Datasets refer to mapping the existing architectural design problems and their solutions. For objective evaluation, we derived 7 items in Table 1 (I1 to I7– item collection is referred to as datasets) driven by the convention and guidelines from (Chen, 1998), our past experience with architecture pattern mining (Ahmad et al., 2018), and classification of reusable architectural solutions and patterns (Alreshidi et al., 2019). Items in Table 1 guided the pattern mining team to objectively review the problem (P) and solution (S) mapping, the attributes (A) that affect the solution, and the occurrence frequency (T) of the repeatable solution by analyzing the pattern datasets. Once a decision (D) is reached, the results are documented as pattern elements (E) for a peer-review before finalization.
- **Step III:** Thematic Analysis to Investigate Pattern Datasets: After identification of the datasets, thematic analysis as the final step helps to 'identify, analyze, and report' patterns from datasets by following three steps. A theme is a possible solution, or method or a mechanism to resolve problems.

(1) Data Analysis process comprises of (a) analyzing datasets, (b) to extract the design attributes from problem-solution mapping (I5 in Table 1).
(2) Pattern Discovery process involves (a) searching of the recurring themes based on data analysis, and (b) reviewing the identified themes. To discover patterns, we reviewed studies and aimed at discovering design problems (I2), and they relate solutions (I3). We consider a recurring theme as a discovered pattern (I6).
(3) Pattern Documentation is the last process that includes (a) classification of related themes based on design attributes (I5) and documented them in a template (I7). Fig. 1 shows a three-step process for pattern-driven architecting.

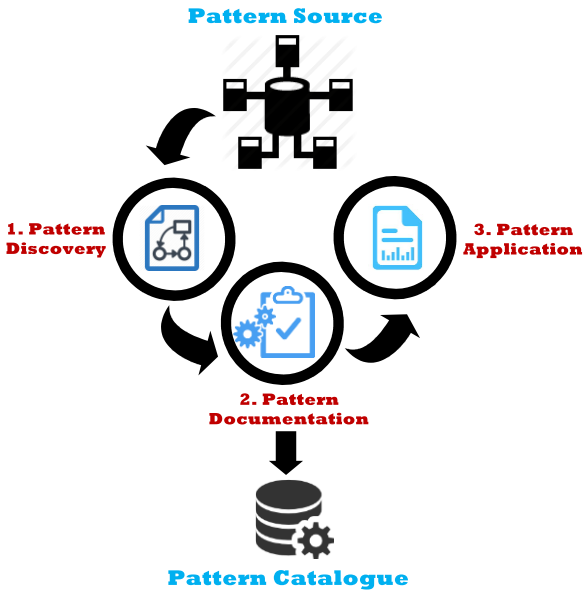

Fig. 1: A three-step processes for pattern-driven architecting

3.2. Solution overview for pattern-driven architecting

We propose pattern-driven architecting as a 3-step process with underlying activities and repositories in Fig. 1. Pattern discovery involves pattern discovery and pattern documentation (in Section 5). Pattern documentation involves pattern classification (in Section 6). Pattern application involves selection and instantiation (in Section 7). If a designer finds suitable patterns from the catalog, then the first two steps are skipped.

4. Reference architecture and pattern catalog for MCC Systems

In this section, we first discuss the reference architecture for mobile cloud computing systems (Section 4.1). We then present the pattern catalog for architecting for mobile cloud computing systems (Section 3.2).

4.1. Reference architecture for mobile cloud computing

In Fig. 2a; we illustrate the reference architecture for the mobile cloud systems that consists of two distinct layers of architectural component, (i) Cloud Computing Layer (resource sufficient-backend), and (ii) Mobile Computing Layer (portable and context-aware-front-end). The discussion and presentation of the reference architecture are vital before presenting the architectural patterns and their presentation (bass et al., 2003). Specifically, the reference architecture in Fig. 2a; acts as a blueprint or simply a reference to derive advanced architectural solutions. Further details about the reference architectures, architectural solutions, and architectural patterns are provided in Simanta et al. (2012), Lewis et al. (2013), and bass et al. (2003).

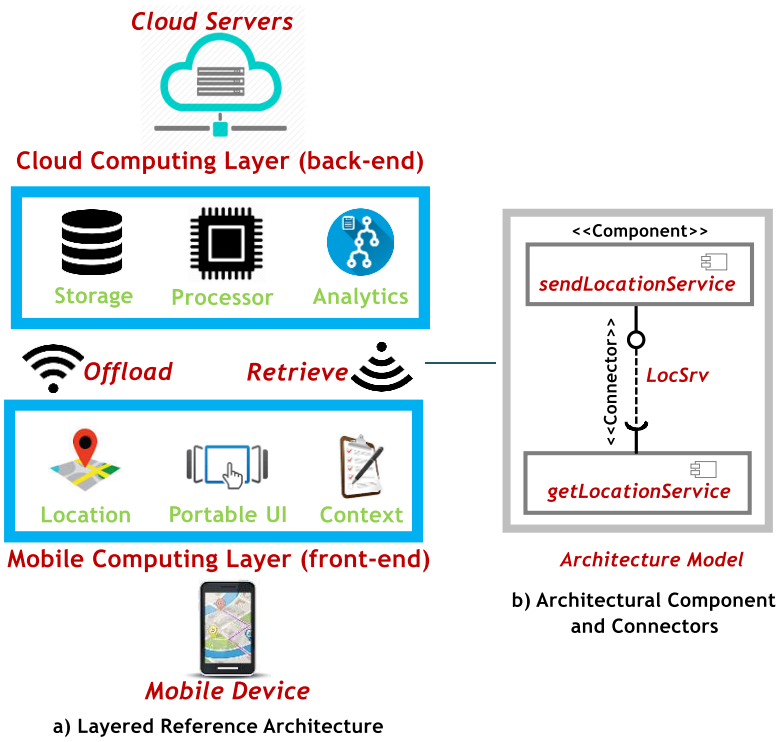

Fig. 2: Reference architecture for mobile cloud systems

- (1) Portability and Context-Sensitivity of User Interface: Mobile computing layer acts as a front-end that allows a mobile user to manipulate the data by exploiting the context and location information of the mobile device. However, the front-end mobile device lacks computation and storage-intensive resources.
- (2) Resource Sufficient and Elastic Cloud Layer acts as a backend to the offloaded data by a mobile device that allows scalable and virtually unlimited storage and processing resources (Picco et al., 2014; Dinh et al., 2013).

As in Fig. 2, the mobile-cloud computing can empower its users by unifying the features such as context-sensitivity, geo-location information, and portability of mobile computing with virtually unlimited computation and storage resources of cloud computing. However, such system integration and operation require continuous network connectivity and involve latency along with security and privacy of data that communicates between mobile and cloud. The communication between the architectural layers is enabled by means of connectors that interconnect the architectural components (Harrison et al., 2007; Alreshidi et al., 2019). As illustrated in Fig. 2b, the architectural component, and connector view supports the functionality for location information service. The service named getLocationService running on a mobile device requests queries the location provision service from sendLocationService component that is being executed on a cloud server. The connector names LocSrv interconnects the two components, namely, getLocationService (mobile computing layer) sendLocationService (cloud computing layer).

4.2. Architecture patterns catalog

We now present the pattern catalog that contains patterns and their related information in a structured format. A catalog essentially documents and maintains the patterns as a repository of reusable architectural solutions (Harrison et al., 2007; Fehling et al., 2011; Erl et al., 2015). In the context of a catalog (as a repository), the pattern template (structured representation) provides the necessary elements to capture and represent the individual patterns (Harrison et al., 2007). Due to space constraints, we only present the necessary elements of a pattern template that include:

(A) **Pattern Name** provides a descriptive identity for each of the patterns in the catalog.
(B) **Intent** that describes the primary purpose and goals of the pattern.
(C) **Classification** that classifies patterns in pre-defined classes.
(D) **Reference Diagram** provides a visual representation of the pattern that is also called as pattern thumbnail. Table 2 shows the template-based specification of the mobile-sensing and cloud-analytics pattern, and Table 3 shows the template-based specification of the mobile cloudlet pattern.

5. Case study-based validation: Pattern-based architecting of MCCS

After introducing the patterns, we now present a case study based validation to demonstrate the

applicability of the pattern. In this section, we first present the case study on pattern-based reuse-driven architecting of the mobile cloud system in Section 5.1. We then present the evaluation in terms of reuse of existing expertise to architect the mobile cloud system in Section 5.2.

Table 1: Template-based specification of the adaptive mobile-cloud offloading pattern

Pattern I: Adaptive Mobile-Cloud Offloading
A) **Pattern Intent:** To enabling a mobile device to dynamically determine *what, how,* and *where* to offload its data to enhance efficiency and of mobile computing.
B) **Design Problem:** How to enable a (resource-constrained) mobile device to delegate its memory and computational-intensive data and tasks to (resource-sufficient) computers?
C) **Solution:** Integrate the offloading logic between Mobile Computing and Cloud Computing Layers. Such an integrated logic enables a mobile device to exploit dynamic parameters - such as energy efficiency, computational overhead, and storage requirements - to determine and offloading data and tasks to cloud computing servers
D) **Architecture Elements:** Mobile Computing Layer with (resource-constrained) mobile devices. Cloud Computing Layer (resource Sufficient) server is integrated with offloading knowledge.
E) **Reuse Design Knowledge:** Integration of offloading logic/knowledge to delegate mobile computing data and tasks to cloud-based servers.
F) **Quality Characteristics:** • *Elasticity* of cloud services (acquiring and releasing resources) based on dynamically determined offloading. • Q*oS-driven Offloading* to ensure that dynamic parameters such as energy storage and computational efficiency
G) **Reference Diagram:** Fig. 3b represents a concrete instance of the abstract pattern representation in 3a. Specifically. Fig. 3b illustrates a scenario of pattern application where mobile devices are used as portable computers to capture contextual images that need analytics and processing to gather information. Image processing must be delegated to the cloud servers that have Image databases to match and process the image. The offloading logic is integrated between the mobile device and cloud server to enable a mobile device to selectively and dynamically offload the images to the appropriate cloud-based server based on energy, computational or storage efficiency

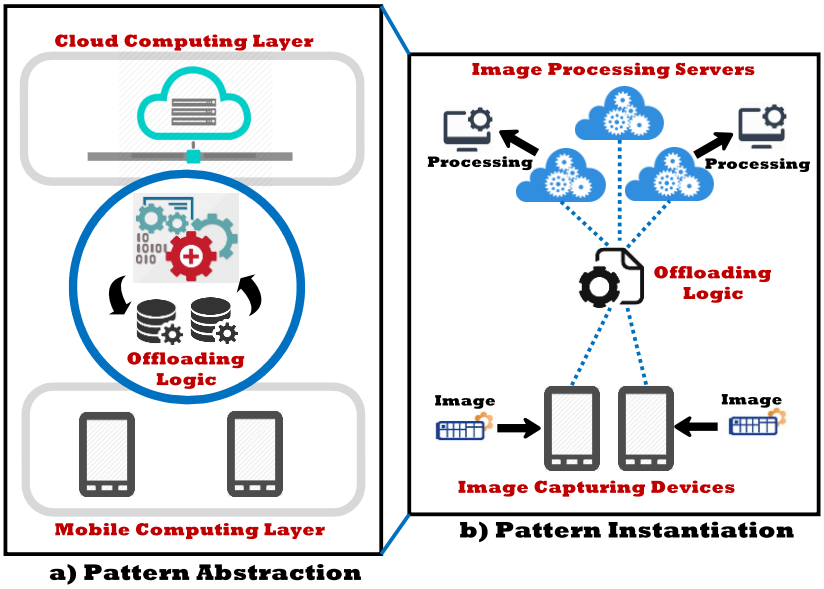

Fig. 3: Adaptive mobile cloud offloading pattern (Pattern abstraction and pattern instantiation)

Table 2: Template-based specification of the mobile-sensing and cloud-analytics pattern

Pattern III: Mobile Sensing and Cloud Analytics
Pattern Intent: To enable a mobile device in a hostile environment to frequently offload data to servers that are in close proximity to mobile devices that they serve.
Solution: The proposed architecture integrates an intermediate layer (based on localized cloud known as cloudlets) between the enterprise cloud and the mobile device. The solution assumes that connectivity to the main cloud (enterprise) cloud is either not reliable or commonly unavailable
Reference Diagram: This architecture inserts an intermediate layer between the central core (i.e., enterprise cloud) and the mobile devices (Fig. 4). At the heart of this architecture is a large centralized core that could be implemented as one of Amazon's data centers or a private enterprise cloud. At the edges of this architecture are offload elements for mobile devices. These elements, or cloudlets, are dispersed and located close to the mobile devices they serve. This architecture decreases latency by using a single-hop network and potentially lowers battery consumption by using Wi-Fi or short-range radio instead of broadband wireless which typically consumes more energy

5.1. Mobile cloud architecture for safe campus emergency management system

In order to demonstrate pattern usage, we present a case study on the architectural design of a first responder emergency management system that is implemented using the MCC technologies. First responder teams are a group of emergency management personal whose job is to tackle emergency scenarios such as medical situations, fighter fighting, rescues, and disaster management. In order to work effectively, individuals in the first responder team(s) must be able to coordinate effectively with other team members, such as to communicate coordinates and context of team members in a search and rescues mission. In this context, the first responder team(s) deployed in the field have team members with handheld mobile devices to support team coordination and communicate the contextual information (location,

images, videos, etc.) from the emergency team to the backend server. The emergency server is deployed in the cloud that contains all the backend data to assist the on-ground teams with their operations. Based on the identified patterns, as in Fig. 6, the most appropriate pattern is the Mobile Sensing and Cloud Analytics pattern that can be used to design the architecture of the first responder emergency management system. After presenting the discovered patterns, we now demonstrate the pattern applicability to an architectural case study. The case study is based on an ongoing project name 'Safe Campus' that aims to provide student and staff safety at the campus. The pattern supports two layers that include:

- Mobile Sensing Layer: It enables the on-ground first responder emergency teams to collect contextual information that can be shared with the team. Moreover, the contextual information can be shared with the backend emergency management server for analytics. This layer corresponds to the Team Coordination Layer in the architecture model.
- Cloud Analytics Layer: It provides data storage, computations, and analytics based on the data provided by the Mobile Sensing layer. Such information can guide the teams about their operations and required actions during emergency management. This layer corresponds to Emergency Analytics Layer.

A partial architectural view of the system is presented in Fig. 6 that illustrates an incident on the campus. In Fig. 6, we have presented the component and connector architectural view of the system. For example, the component named Incident Sensor at the mobile computing layer senses an incident and reports it to the component Incident Report using the connector sendIncident Data on the server layer. The incident can be captured (normally an image, textual details, location of the incident, etc.) with a mobile device that runs the Safe Campus. The incident captured by any individual using their mobile device can be communicated with the incident reporting server that manages the incident reports and their processing. A server is a cloud-hosted machine that stores retrieves and processes the incident details.

We utilize one of the discovered patterns to architect the above-mentioned scenario. For demonstration purposes, we have selected and presented the scenario that has an associated pattern available. There are various scenarios where a relevant architectural pattern is not available. In such a case, the designer/architect must develop the architecture without any patterns. Patterns only provide a packaged and reusable design rationale for architecting the systems. We illustrate the pattern application based on Fig. 6. Fig 6 shows a partial architectural overview of the Safe Campus system

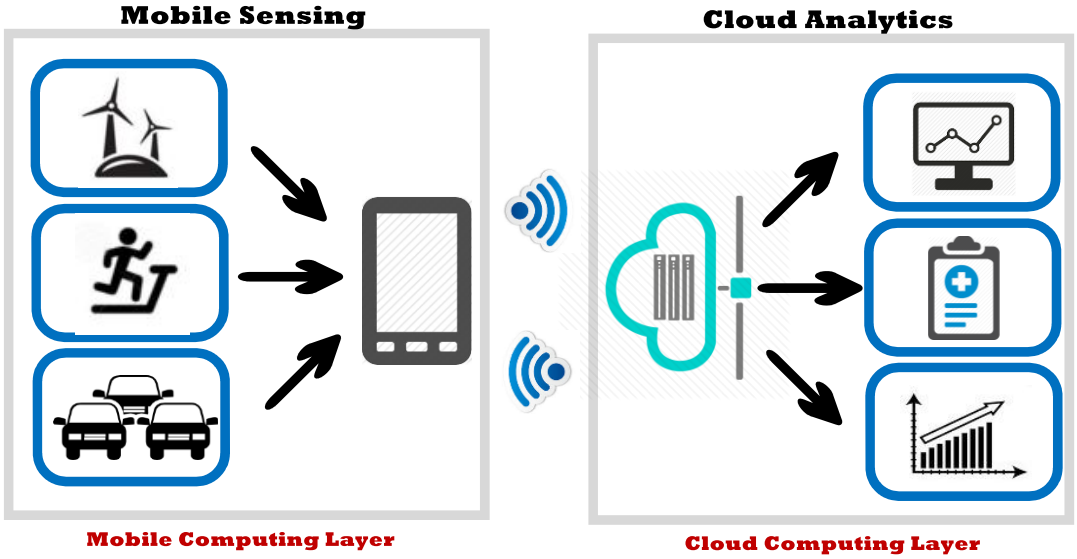

Fig. 4: Mobile sensing and cloud analytics pattern

Table 3: Template-based specification of the mobile cloudlet pattern

Pattern III: Mobile Cloudlet Pattern
Pattern Intent: To enable a mobile device in a hostile environment to frequently offload data to servers that are in close proximity to mobile devices that they serve.
Design Problem: How to minimize the offloading latency (on a remote server) to a single-hop network while maximizing the performance and QoS for mobile computing tasks?
Solution: The proposed architecture integrates an intermediate layer (based on localized cloud known as cloudlets) between the enterprise cloud and the mobile device. The solution assumes that connectivity to the main cloud (enterprise) cloud is either not reliable or commonly un-available.
Reference Diagram: As illustrated in Fig. 5. This architecture inserts an intermediate layer between the central core (i.e., enterprise cloud) and the mobile devices. At the edges of this architecture are offload elements for mobile devices. These elements, or cloudlets, are dispersed and located close to the mobile devices they serve.

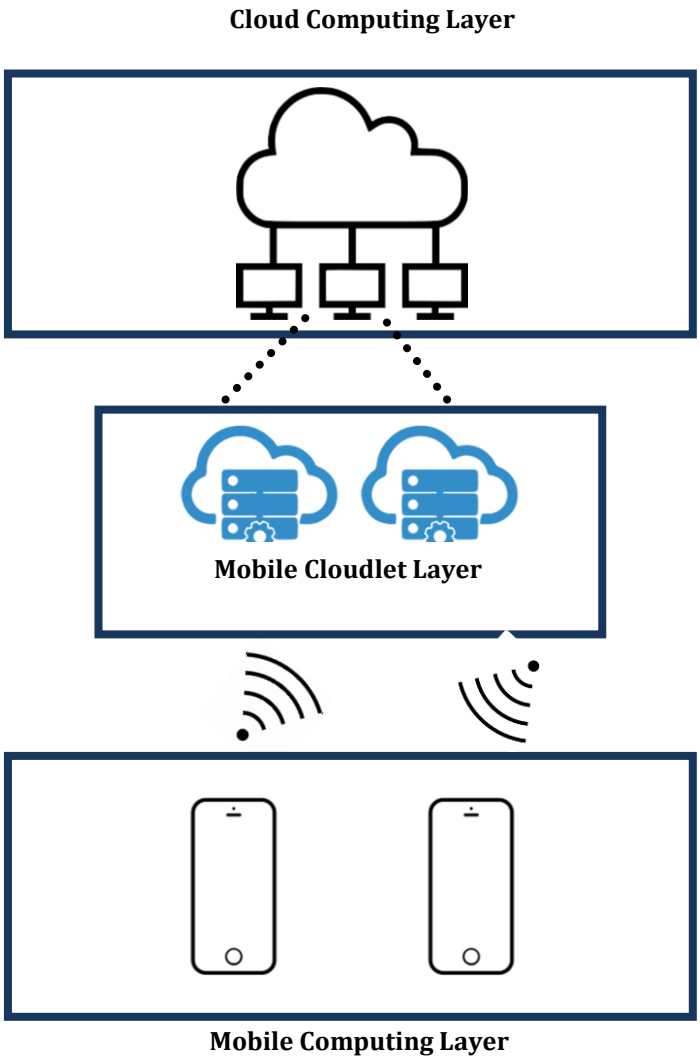

Fig. 5: Mobile cloudlet pattern

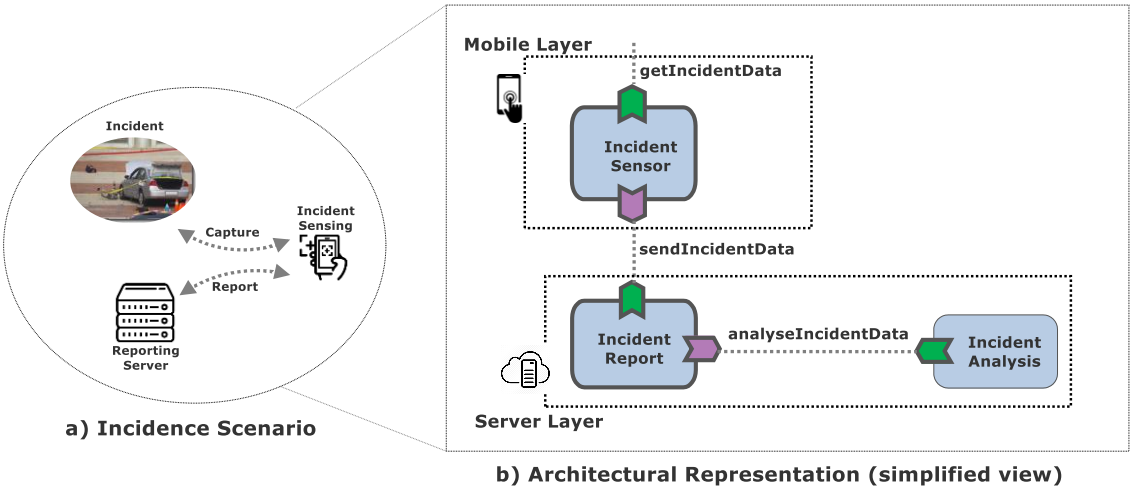

Fig. 6: A partial architectural overview of the safe campus system

As illustrated in Fig. 6, architectural requirements are the foundation for designing the desired architecture. The designer/architect, based on the requirements selects the most appropriate pattern from the catalog and apply it to achieve reusability in the architectural design process to achieve the architecture (Kim, 2010; Wilder, 2012; Fehling et al., 2011; Alreshidi et al., 2019). In Fig. 6, the Mobile Sensing and Cloud Analytics pattern has been applied (Table 4). The pattern enables the integration of the mobile computing layer that senses the incident and sends the details to the cloud server. The cloud-based layer manages the processing and analytics of the incident. Fig. 7 shows

an overview of the pattern-based architecting process.

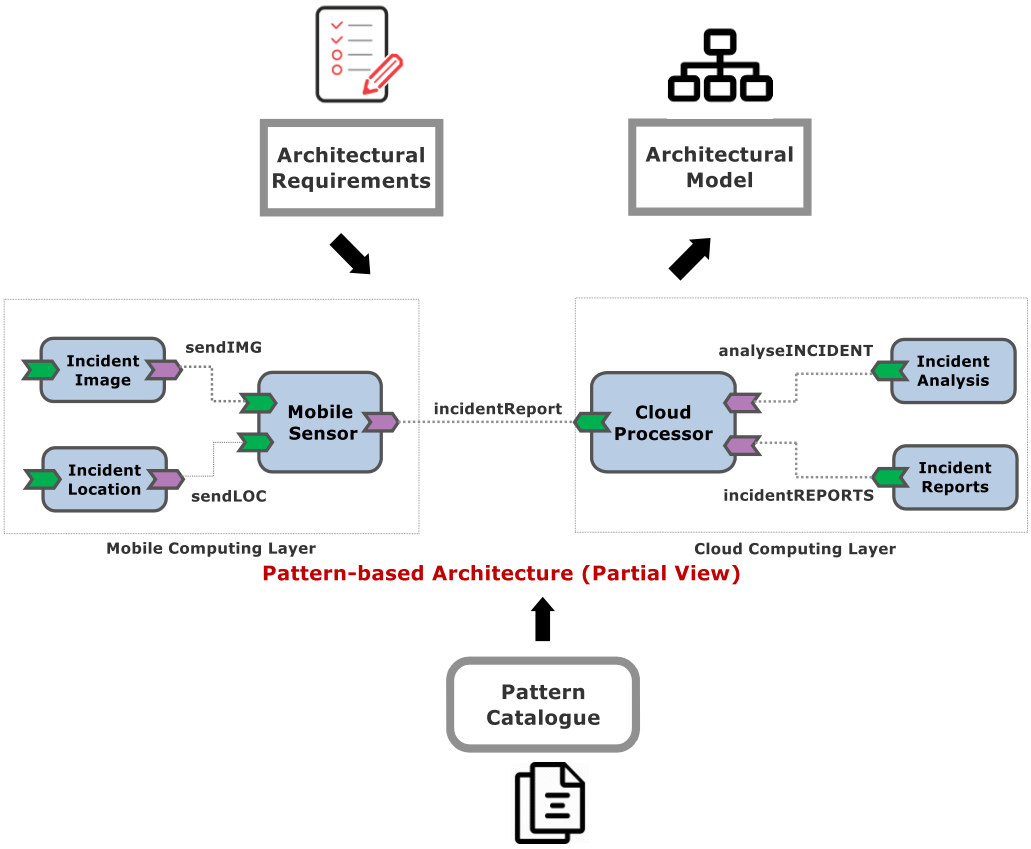

Fig. 7: An overview of the pattern-based architecting process

5.2. Evaluating pattern-based reusability in developing MCCS

We now evaluate the pattern-based architecting of the mobile cloud systems. Specifically, we aim to evaluate (i) the extent to which patterns promote reusability in terms of architecting operations, (ii) time taken to perform the architecting operation. We have used the following parameters for the evaluation. The following are four parameters that have been used during the evaluation.

- **Total Architecting Operations (TAO):** It refers to the total steps or actions that are required to implement a particular component in the architecture. For example, in order to add and interconnect two components in the architecture, a total of 03 architecting operations are required in terms of addition or two components and the addition of a connector that interconnects those components. TAO is the fundamental unit of measuring the number of actions for an architecting activity.
- **Architecting Time Taken (ATT):** It refers to the total time taken to complete a specific architecting operation. ATT is a measure of the efficiency of the architecting operation.
- **Reusability of Architecting Operations (RAO)**: It refers to the measure of the reusability that can be attained by pattern-based architecting.
- **Efficiency of Architecting Operations (EAO):** It refers to the measure of the efficiency that can be attained by pattern-based architecting.

Table 4: Summary of the data for the evaluation of pattern-based architecting

Pattern Name	TAO (Number of Steps)	RAO (Pattern Steps)	Architecting Reusability (1–[RAO/TAO])	AAT (Total Millisecond)	EAO (Total Millisecond)	Architecting Efficiency (1–[EAO/AAT])
Adaptive Mobile-Cloud Offloading	25	11	56%	1185	658	45%
Mobile Cloudlets	21	10	53%	1095	785	29%
Mobile-Sensing and Cloud-Analytics	31	14	55%	1546	911	42%
Average	15.6	11.6	54.6%	1275.3	1177	38.6%

Table 4 and Fig. 8 complement each other to present the results of the evaluation. Specifically, Table 4 provides a quantification of the reusability and efficiency of three patterns. In comparison, Fig. 8 provides an illustrative and comparative view of the architecting reusability and architecting efficiency of the pattern-based architecting process. Based on the illustration in Fig. 8, we can claim that pattern-based

architecting supports reusability of changes and, on average, supports approximately 55% of efforts in terms of architecting operations. Moreover, patterns-based architecting also supports the time efficiency of implementing the architecting process. Pattern-based architecting, on average, saves approximately 39% of the time when compared to non-pattern based architecting.

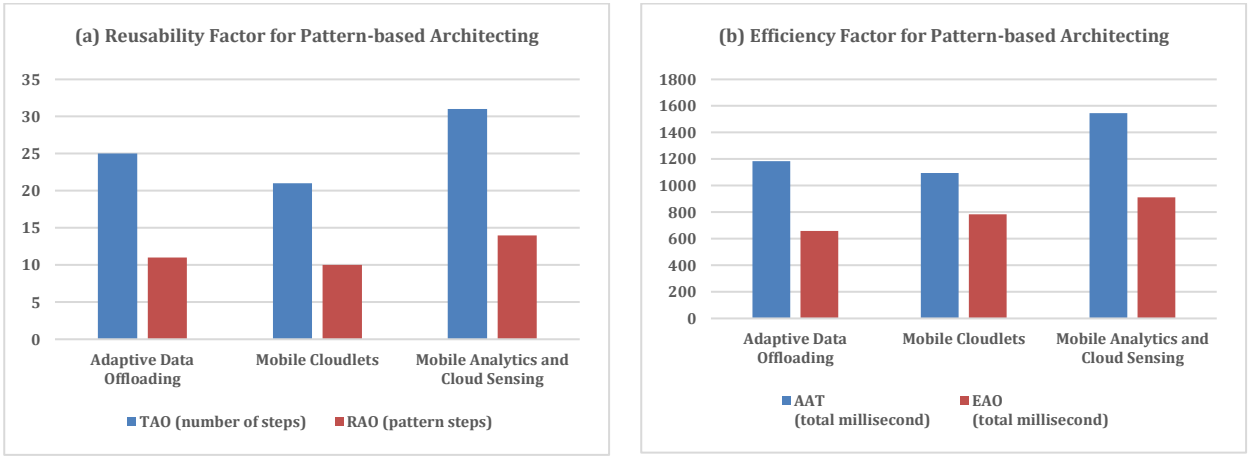

Fig. 8: Measure of the reusability and efficiency of pattern-based architecting

6. Conclusions and future research

We present a pattern-based approach to architect mobile cloud computing systems. Patterns, as empirically discovered knowledge, represent reusable knowledge and best practices for system design and development. The proposed solution follows an incremental approach to (i) discover patterns that can be (ii) documented using pattern template, and (iii) applied to support reusability of the architecting process. We discovered three patterns, namely Adaptive Mobile-Cloud Offloading, Mobile Cloudlets, Mobile Sensing, and Cloud Analytics Pattern. We have used Mobile Sensing and Cloud Analytics pattern for pattern-based architecting of the case study. The case study is based on a Safe Campus scenario that enables first responder teams to exploit context-sensitive mobile devices to capture details of any incidents on the campus. The backend cloud server is used to perform data storage and analytics, such as incident analysis and reporting. We have also validated the role of patterns in support reuse of frequent architecting activities.

As part of future work, we aim to extend the approach and its applicability to more case studies and scenarios in the context of mobile-cloud systems. An important aspect of future research is to discover new patterns.

- Establishing Pattern Catalogue: Pattern catalog refers to a repository of discovered patterns that could be queried to retrieve the most relevant patterns. We focus on discovering more patterns from different systems that provide the foundations for pattern catalog.
- Towards a Pattern Language for Architecting MCCS: The pattern language represents an interconnected network of patterns with necessary syntax, semantics, and grammar for pattern usage. As part of future research, we aim to develop a pattern language as an interconnected network of patterns that can be applied in an incremental manner to design and architect the mobile cloud system.

Compliance with ethical standards

Conflict of interest

The authors declare that they have no conflict of interest.

References

Ahmad A, Pahl C, Altamimi AB, and Alreshidi A (2018). Mining patterns from change logs to support reuse-driven evolution of software architectures. Journal of Computer Science and Technology, 33(6): 1278-1306. https://doi.org/10.1007/s11390-018-1887-3

Alreshidi A and Ahmad A (2019). Architecting software for the internet of thing based systems. Future Internet, 11(7): 153. https://doi.org/10.3390/fi11070153

Alreshidi A, Ahmad A, B Altamimi A, Sultan K, and Mehmood R (2019). Software architecture for mobile cloud computing systems. Future Internet, 11(11): 238. https://doi.org/10.3390/fi11110238

Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, and Zaharia M (2010). A view of cloud computing. Communications of the ACM, 53(4): 50-58. https://doi.org/10.1145/1721654.1721672

Bass L, Clements P, and Kazman R (2003). Software architecture in practice. Addison-Wesley Professional, Boston, USA.

Buschmann F, Henney K, and Schimdt D (2007). Pattern-oriented software architecture: On patterns and pattern language. Volume 5, John Wiley and Sons, Hoboken, USA.

Cámara J, Correia P, De Lemos R, Garlan D, Gomes P, Schmerl B, and Ventura R (2013). Evolving an adaptive industrial software system to use architecture-based self-adaptation. In the 8th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, IEEE, San Francisco, USA: 13-22. https://doi.org/10.1109/SEAMS.2013.6595488

CDP (2016). What are AWS cloud design patterns? Cloud Design Patterns. Available online at: https://bit.ly/2XwIYOK

Chen KZ (1998). Integration of design method software for concurrent engineering using axiomatic design. Integrated Manufacturing Systems, 9(4): 242-252. https://doi.org/10.1108/09576069810217847

Cox PA (2011). Mobile cloud computing: Devices, trends, issues, and the enabling technologies. IBM Developer Works. Available online at: https://ibm.co/2C2pXfu

Dinh HT, Lee C, Niyato D, and Wang P (2013). A survey of mobile cloud computing: Architecture, applications, and approaches. Wireless Communications and Mobile Computing, 13(18): 1587-1611. https://doi.org/10.1002/wcm.1203

Erl T, Cope R, and Naserpour A (2015). Cloud computing design patterns. Prentice-Hall Press, Upper Saddle River, USA.

Fehling C, Leymann F, Retter R, Schumm D, and Schupeck W (2011). An architectural pattern language of cloud-based applications. In the 18th Conference on Pattern Languages of Programs, Association for Computing Machinery, Portland, USA: 1-11. https://doi.org/10.1145/2578903.2579140

Harrison NB, Avgeriou P, and Zdun U (2007). Using patterns to capture architectural decisions. IEEE Software, 24(4): 38-45. https://doi.org/10.1109/MS.2007.124

Kim J (2010). Architectural patterns for service-based mobile applications. In the IEEE International Conference on Service-Oriented Computing and Applications, IEEE, Perth, Australia: 1-4. https://doi.org/10.1109/SOCA.2010.5707181

Lewis G, Novakouski M, and Sánchez E (2012). A reference architecture for group-context-aware mobile applications. In the International Conference on Mobile Computing, Applications, and Services, Springer, Seattle, USA: 44-63. https://doi.org/10.1007/978-3-642-36632-1_3

Lewis GA, Simanta S, Novakouski M, Cahill G, Boleng J, Morris E, and Root J (2013). Architecture patterns for mobile systems in resource-constrained environments. In the MILCOM 2013-2013 IEEE Military Communications Conference, IEEE, San Diego, USA: 680-685. https://doi.org/10.1109/MILCOM.2013.121

Pahl C, Giesecke S, and Hasselbring W (2009). Ontology-based modelling of architectural styles. Information and Software Technology, 51(12): 1739-1749. https://doi.org/10.1016/j.infsof.2009.06.001

Picco GP, Julien C, Murphy AL, Musolesi M, and Roman GC (2014). Software engineering for mobility: Reflecting on the past, peering into the future. In the on Future of Software Engineering, Association for Computing Machinery, Hyderabad, India: 13-28. https://doi.org/10.1145/2593882.2593884 **PMid:24138709**

Roth J (2002). Patterns of mobile interaction. Personal and Ubiquitous Computing, 6(4): 282-289. https://doi.org/10.1007/s007790200029

Satyanarayanan M (2011). Mobile computing: the next decade. ACM SIGMOBILE Mobile Computing and Communications Review, 15(2): 2-10. https://doi.org/10.1145/2016598.2016600

Simanta S, Ha K, Lewis G, Morris E, and Satyanarayanan M (2012). A reference architecture for mobile code offload in hostile environments. In the International Conference on Mobile Computing, Applications, and Services, Springer, Seattle, USA: 274-293. https://doi.org/10.1007/978-3-642-36632-1_16

Wilder B (2012). Cloud architecture patterns. Oreilly and Associate Series, Sebastopol, USA.