Autonomic Service Management in Mobile Cloud Infrastructures

1 Aduragba Olanrewaju Tahir
Dept. of Electrical and Computer Engineering
Kwara State University,
Malete, Ilorin, Nigeria.
tahrlanre@hotmail.com

2 Adedoyin Adeyinka
Dept. of Information and Communication Science
University of Ilorin,
Ilorin, Nigeria.
yinkaadedoyin22@yahoo.com

ABSTRACT
With the introduction and advancement of mobile cloud computing, endless opportunities for the provision of mobile phone services have ensued. Mobile cloud computing technology has brought about a breakthrough in information management by enabling the use of central servers which act as hosts, to deliver the required services, providing flexibility and reliability to mobile users. Due to the dynamic nature of mobile devices, and various constraints such as limited bandwidth however, the use of a traditional central server for these devices continues to prove problematic, with a number of issues including congestion and server failure occurring time and time again, while the demand for mobile cloud services continue to rise. This research project seeks to address these problems by presenting an alternate approach for service management in mobile cloud infrastructures. This research presents a decentralised model that takes inspiration from the flocking behaviour of birds since birds are known to interact locally to produce a global behaviour without a central point of authority. Three variations of the proposed model were developed to support how mobile devices interact for service provision. As an exemplar case study, a simulation of mobile users participating in the opening ceremony of the London 2012 Olympics where multimedia contents are provided and consumed as services was performed. Additionally, experiments were conducted to evaluate the three variations of the models based on their performance on the set out aim. The results from the experiments indicate that the proposed mechanism was quite practicable since the mobile devices are able to select and provide services in a decentralised manner. The research has shown that mobile devices can be configured to provide their services in a decentralised manner while considering social issues.

KEYWORDS: Mobile Cloud Infrastructure, Autonomic Service, Ubiquitous Service-Oriented Network, Service Management and Birds flocking behaviour

1. INTRODUCTION
Since the first mobile phone was created, they have grown from simple phone-only devices to sophisticated multi-function devices with substantial resources. For example, Smartphone’s now have quad-core processors, significantly enhanced storage and memory capabilities, GPS, Wi-Fi/3G/4G radios for faster Internet connectivity. However, this rapid advancement in mobile computing technology has driven the need for mobile cloud applications and services (Chang et al, 2013). Mobile cloud computing can be described as a paradigm that allows resource constrained mobile users to adaptively adjust processing and storage capabilities by transparently partitioning and offloading the computationally intensive and storage demanding jobs on traditional cloud resources by providing ubiquitous wireless access (Khan et al, 2013). A large variety of mobile applications have taken advantage of this paradigm to provide applications as services to mobile users over the Internet. Examples of mobile cloud applications that illustrate the usefulness of mobile-cloud computing are Apple iCloud which provides cloud storage services for Apple mobile devices, Google’s Gmail for Mobile which provides email services to mobile devices users with Gmail accounts and Facebook Messenger for Mobile which provides messaging services to mobile devices users with Facebook accounts. Mobile cloud computing encourages a service-oriented computing paradigm where various kinds of resources are delivered as services. As an alternative to traditional remote servers, users can also offer services and content to other users while on the go via their mobile devices (Del Prete and Capra, 2008). For example a mobile user may provide sensing...
information from its embedded sensors as a service to other mobile phones within its environment. As this scenario shows, it is possible to have multiple service providers in this environment who offer mostly similar services but with different qualities. Therefore, mobile users will need to select services from more than one service provider, these services, will have to be arranged, composed and managed to satisfy user requirements. There are a number of ways to manage services in mobile cloud environments most of which are carried out in a centralized manner. Most of these approaches utilise a centralized server where services are maintained and accessed by every mobile device (Mohan et al, 2004). However, this architecture is not suitable in mobile environment due to the dynamic nature of mobile devices with services appearing and disappearing all the time. Another reason is that mobile devices have limited bandwidth and more mobile communications trying to connect to the same server can cause congestion. For example, the 2009 U.S Presidential Inauguration Park witnessed many service failures due to a large number of people trying access multimedia cloud services from central servers. As a result, mobile users were unable to access cloud services at the time when it is most interesting and relevant (Marinelli E, 2009). Also, there is a possibility that communication between the users’ devices and the central servers might not be available due to central server failure. For example, the terrorist events of September 11, 2001 in the United States caused a lot of damages to network facilities hence leaving majority of the people without cellular networks to make phone calls (Beard, 2005). Considering these problems, there is need for a decentralised and autonomic model for managing service provisioning in mobile cloud infrastructures.

2. RELATED WORK

A mobile cloud environment can be referred to as an enormous socio-technical system that comprise of devices and services interacting with each other in different ways (Al-Obiasat and Braun, 2007). These systems can be compared to swarm intelligent systems; they are composed of different entities interacting with one another and these interactions give rise to the complexity of mobile environments. Al-Obiasat and Braun (2007) identified various advantages of Swarm intelligent systems. These are:

1. Scalability: scalability is driven by the interactions of both local and distributed individuals. As mobile clouds are growing immensely in size, the population of swarm agents can easily adapt the size of the mobile clouds.
2. Adaptation mechanism: swarm agents are able to change their behaviour and reproduction according to the changes in mobile clouds.
3. Autonomy: no human intervention or minimal human intervention or supervision is required.
4. Fault tolerance: there is no need for control management to be centralized.

In line with the trend in this idea, there have been many works that has applied swarm intelligence to propose autonomic service management framework in dynamic network environments. For example, Chang et al (2011) proposed autonomic network service management, which was inspired by the observations of swarming intelligence in biological systems. Their work focused on issues of autonomous agents in complex Ubiquitous Service-Oriented Networks.

3. METHODOLOGY

The study proposed a set of artefacts which include constructs and models for representing how a mobile cloud infrastructure can be configured to provide its services based on social relations and a new method for coordinating service provision activities in mobile cloud user devices in a decentralized manner. Together, they constitute an innovative process for representing and managing service provisioning in mobile clouds thus enabling mobile cloud user devices thrive in dynamic networks. The proposed artefacts draw up from existing knowledge base on how birds flock without a central coordination. The bird flocking behaviour was successfully simulated on computer by Reynolds (1986). To perform evaluation, the research presents a case of the 2012 Olympic Opening Ceremony where users participating in this event can deliver multimedia contents as services to nearby users’ devices. The discovery
and requisition of services from user devices is enabled by physical proximity but is subject to the existence of a social relation between the users. This event was simulated as a form of experiment to evaluate the proposed design artefacts. The simulation implements different service provision policies. Then a thorough analysis was done on the simulation results.

3.1 Birds Flocking Behaviour

The flocking and schooling behaviour of a collection of interacting agents such as birds, fish and penguins in natural systems have been widely studied by many scientists from diverse fields (Ahmed and Glasgow, 2012). These groups of agents maintain a harmonious and cohesive association by organizing themselves without a central coordination. Unlike most social groups among vertebrates that arrange themselves hierarchically based on a pecking order before they interact, all members of a flock or school are alike in influence and importance (Shaw, 1978). Ahmed and Glasgow (2012) defined bird flocking as the synchronized group movement of a large number of interacting birds with a common group objective. This objective are motivated by certain factors such as protection from predators, statistically improved chance of a survival of an individual bird in a case of attack from predators, gaining from a larger search for food and advantages for social and mating activities (Shaw, 1970: Reynolds, 1986). For a bird to participate in a flock, it must have certain behaviours that allow it to coordinate its movement with those of its flock mates. Fundamentally, natural flocks comprise of two balanced behaviours, a wish to stay close to the flock and a desire to avoid collisions within the flock (Reynolds, 1986). In 1986, Reynolds proposed a model to simulate the flocking behaviour of birds. He posited three simple behaviours to implement the simulated flocking behaviour of birds. The behaviours (Reynolds, 1986) are stated as follows:

(i) Flock Centring: This behaviour makes a flock member want to stay close to nearby flock mates by flying in a direction that moves them closer to the centroid of the nearby flock mates.

(ii) Collision Avoidance: This is the urge to avoid collision with an imminent impact. This behaviour makes a flock member avoid collisions with nearby flock mates based on their relative positions.

(iii) Velocity Matching: This behaviour makes a flock member want to match its velocity with nearby flock mates. Collision avoidance and velocity matching can be considered to complement each other.

Collision avoidance aims to create the minimum required separation distance between flock mates while velocity matching aims to maintain the separation distance during flocking. They both ensure that each flock member is free to fly around without colliding with nearby flock mates (Reynolds, 1986). It should be also noted that, these flocking behaviours are generally known as cohesion, separation and alignment rules in literature (Olfati-Saber, 2006).

3.2 Proposed Model

In order to enable mobile user devices manage services autonomously, they must collectively exhibit some behaviours to cope with the dynamic environment of the mobile cloud. In this study, we aim to do so by proposing a model, which takes inspiration from the flocking of birds to take care of the discovery, selection and provision of services to satisfy mobile cloud user requirements. Users’ devices that make up the mobile cloud mimics basic behaviours similar to that of birds in a flock to ensure effective composition and discovery of services. The model represents the way each mobile node interacts and combines with each other so as to serve their own individual needs as well as the sustainability of the overall mobile cloud. In order of decreasing importance, each mobile cloud users’ device is expected to exhibit the following behaviours:

i. Social relationship maintenance: attempt to maintain social relationship between the mobile cloud users.

ii. Load Balancing: attempt to match service provision rate with nearby mobile cloud user’s device.
3.3 Social relationship maintenance

Social relationship maintenance makes a mobile user want to stay connected with friends in the mobile cloud. This behaviour is similar to flock centre in the simulated flocking behaviour of birds described whereby flock members want to be near their nearby flock mates expecting to profit from a larger effective search for food and protection from predators amongst other benefits (Shaw, 1978). Likewise in humans, we aim to stay close to people that we know for certain benefits. Despite the benefits gained from the inception of the Internet, there has been lot of privacy and social issues concerning the handling of sensitive information about mobile cloud users. In this model, we aim to manage the risk of privacy issues by ensuring the mobile cloud users’ device select and provide services based on certain rules regarding their social relationships. The social relationship employed in this model takes advantage of pre-existing social graphs in a social network such as Facebook. Stated briefly below are the rules that lead to social relationship maintenance in the mobile cloud:

i. From the list of all available services, a mobile cloud user submits request for a service attached to the mobile cloud user a social relationship exist between them.

ii. If a mobile cloud user’s device is overloaded and needs to delegate a job to another mobile cloud user’s device, it selects the job of a mobile user that no social relationship exists between them.

iii. The job that is selected for delegation is transferred to a mobile cloud user’s device where a social relationship exists between the initial requestor of the service and the mobile cloud user the job is transferred to.

3.4 Load Balancing

Load refers to the service requests that are currently been served by a mobile device (service provider). The manner in which mobile devices select services from the list of available services tries to reduce the delay at the service provider due to on-going service provisioning. Equally, the service providers try to manage their service provision rate by wanting to match the service provision rate of nearby mobile devices. This behaviour is similar to velocity matching in the simulated bird flocking behaviour by Reynolds (1986) where flock members try to adjust their velocity to match that of nearby flock members to avoid collision. If a mobile device submits a service request to mobile device with the least service provision rate, it is unlikely that it will experience any service provision delays or mobile devices will be overwhelmed with service requests. The rules that formulate the load balancing behaviour are briefly stated below:

i. From the list of all available services, a mobile cloud user submits request for a service attached to a mobile cloud user with the least service provision rate.

ii. At regular intervals, if the service provision rate of a particular mobile cloud user is high compared to its neighbours, such mobile cloud user should transfer one of its jobs to a less busy mobile cloud user’s device given that it can provide the service.

3.5 Policies

Based on the decentralised model presented, mobile cloud users can adopt some policies to support autonomic management of services in the mobile cloud. A brief description of these policies is given below:

3.5.1 First policy

A policy to manage the services in a mobile cloud can be for mobile devices to submit service requests given preference to nearby mobile users that a social relationship exists between them. However, if there is no service attached to a nearby friend (mobile cloud user’s device) to select from, then it can submit the service request to another mobile cloud user that is idle and can serve the request even though no social relationship exists between them. Moreover, if this mobile user becomes overloaded and needs to delegate a job, it selects the job of the mobile cloud user it does not have a social relationship with in preference to all other requests. Furthermore, the selected job is transferred to a nearby mobile cloud user that
can provide the service and has a social relationship with the initial requestor. However, if there is no friend of the initial requestor that can provide the service then it transfers the job to a mobile cloud user with a lower service provision rate that can provide the service. This policy ensures higher chance of service discovery and provisioning as users do not necessarily have to submit request or provide services to friends only, they can also select and provide services to nearby non-friends. On the other hand, there is a greater probability that sensitive information will be shared with non-friends.

3.5.2 Second policy

An alternative policy can have users only submit service request to nearby users that a social relationship exist between them. This means a user selects from the list of available services only services attached to their friends within its neighbourhood. Then amongst the friends it selects the one with the least service provision rate. Additionally, if a user becomes overloaded with service request, it only transfers job to nearby users a social relationship exists between them and the initial requestor of the service. This policy ensures that higher trust levels and social relationship maintenance are ensured during service discovery and composition.

3.5.3 Third policy

This policy builds upon the previous policy and it employs service delegation as an additional action to ensure better service discovery and provisioning. In this case, if a mobile cloud user submits a service request and there is no friend that can serve the request, then it delegates such request to an idle user whom is a friend for it to search for users that can provide the service within its own neighbourhood. If the search is successful then it places a service request as a mediator on behalf of the initial requestor and then transfers this service as it receives it to the initial requestor. This policy ensures better service discovery and provisioning while high trust levels and social relationship are maintained between mobile cloud users.

4. IMPLEMENTATION

To implement the experiment, the study adopted the Opening Ceremony of the London 2012 Olympic Games as the simulation scenario and MASON a simulation toolkit written in java, was developed to support discrete event interaction between a large number of agents.

4.1 Simulation Scenario: The Opening Ceremony of the London 2012 Olympic Games

The study adopted, 2012 London Olympic Opening Ceremony as the case study for the simulation. Thousands of people with Smartphone’s and tablets attended the event equipped with mobile devices and tablets to call and text family and friends or share photos and videos. According to analysts, the London 2012 Olympics was the heaviest in terms of data (BBC News, 2012). They suggested that about 60 Gb of data, the equivalent of 3000 photographs would be shared amongst mobile users in the Olympic Park every second. The volumes of these data are very high and they are expected to strain the central servers. Instead of many mobile users downloading the same popular multimedia content from a central server, they can conduct a local search and place a request to download such content from other user devices that can provide them. In this research, this scenario is simulated whereby multimedia contents are provided as services to members of the mobile cloud via their mobile devices.

4.2 MASON: Multi-Agent Simulation Toolkit

MASON is a simulation toolkit written in java, developed to support discrete event interaction between large numbers of agents. It places special importance on simulation of many agents in a swarm (Luke et al, 2005). MASON has several features but certain features influenced the choice to use it for this research. For instance, the Neighbourhood lookup feature. In MASON, there is a representation of space known as grid and each agent has a coordinate, which represents its location within this grid. A grid has neighbourhood lookup functions that select all the coordinate locations in the grid lying within some region a distance away from a
provided point. Furthermore, MASON separates model and visualisation from each other allowing models to be detached or attached to be visualised dynamically (Luck et al. 2004). Another feature that influenced the choice of this toolkit is the “Network” feature. This feature allows the definition of both directed and undirected graphs. Graphs in MASON can be compared to that of the graph of a social network whereby any agent can be a node and connects to other agents via agents. An edge stores information that represents the social relationship between the connected nodes.

5. SYSTEM ARCHITECTURE

Figure 1 illustrates the system architecture. There are two levels: the mobile cloud user level and the user device or system level. Each mobile cloud user is associated to a mobile node.

![Figure 1: The System Architecture](image)

Within the mobile cloud user level, certain relationship exists between the mobile cloud users. The relationship between individuals in a social network can be used to deduce that a trust relationship exists between them. This relationship determines how service is provided to other nodes. It is up to the user to set up rules for service provisioning. A user may decide to only provide services to social network friends while another user may decide not to provide services to only social network friends. In this case, the user provides services to users that no social relationship exists between them.

At the system level, each mobile node has neighbour nodes. The neighbourhood logic is based on proximity awareness between mobile nodes in the system. A mobile node is considered to be a neighbour node if it is within a certain proximity metric from a particular node. Essentially, the discovery and provision of services is based on this proximity. A mobile node can place a service request to a node or provide services to another node within the cloud. Mobile nodes also have the potential to transfer service requests to other mobile nodes. Although discovery and request for services is based on proximity awareness, service provisioning is subject to existence of social network relationships between two mobile nodes.

5.1 Neighbourhood

As stated above, the sharing of multimedia contents as well as discovery and composition of services is enabled by physical proximity between two nodes. In this research, the existence of wireless connections between the mobile devices is considered in order to define the neighbourhood relations. A mobile device can connect wirelessly to other devices through a Bluetooth connection of Wi-Fi connection. These connections vary in terms of their data transfer rate and broadcasting range. The broadcasting range of these connections determines the proximity metric used to define neighbours of a particular mobile device in the system. Wi-Fi covers a larger broadcasting range than Bluetooth. Hence, a Wi-Fi connection will mean a larger neighbourhood for each mobile device in the system.

5.2 Social Graph

In addition to the neighbourhood relations, the system employs the concept of social network to determine how multimedia resources are shared to form better discovery and composition of services. In a social network, nodes and edges connected to the nodes constitute a social graph where the nodes represent the mobile cloud users and the edges describe the social relationship between two mobile cloud users. The act of adding a person as a friend in a social network
shows that a user has some degree of knowledge of the person being added. More so, it can be a real world relationship such as family members, friends or colleagues. On this basis, mobile cloud users’ devices are configured to provide services in order to ensure high degree of trust between mobile cloud users. Social networks such as Facebook, Twitter already exists and is popular amongst users. Hence, the system can easily utilize these social networks to explore social relations between mobile cloud users. Figure 2 illustrates the social graph of mobile cloud users in the simulation of the system.

Figure 2: A social graph showing the relationship between mobile nodes

5.3 Java Classes and Agent Types

The simulation system was implemented with six java classes. Figure 3 shows the class diagram for the implementation.

Figure 3: UML Class Diagram for the multimedia content exchange simulation

- **MobileCloud**: this is the main class. It sets up the simulation and schedules the agents

- **MobileCloudWithUI**: an optional wrapper for MobileCloud. It handles the visualization and allows for real-time change of simulation parameters. Figure 8 shows a screenshot of the simulation window during a simulation demo.

- **Node**: contains all behaviours for agents representing the mobile cloud user’s devices. This class handles the various behaviours of the mobile cloud users’ devices modelled. User device behaviours include searching for media content with neighbours, placing request for a service, providing services to other nodes within the mobile cloud and delegating jobs to idle agents. All these behaviours build up to form the two main behaviours: social relationship maintenance and load balancing.

- **Request**: this class represents service placed by nodes in the simulation. It contains
information about the node that places a request, the file requested for and the level of neighbourhood search.

- **File**: this class represents all the files present in the cloud simulation. It contains information about the file size and the file ID.

- **Service**: this class represents services provided by nodes. The class also contains information about the node that requests for a service and the requested file.

Figure 4: Screenshot of a simulation window

### 4.4 Anonymous Agents

MASON allows for the use of anonymous class to add a single agent to the schedule. An anonymous class in MASON refers to a class that has no name rather it exists as a subclass of another class and creates a single instance (Luck et al. 2004). The designed system makes use of four anonymous agents for simulation:

- **RandomiseRequest**: this anonymous agent is implemented in the *MobileCloud* class and it will occur at the same time step as the other agents but it will always be stepped before the other agents. The function is to randomly generate a number of requests and then it randomly assigns the generated requests to the mobile nodes.

- **ComputeDownloads**: All requests are served at a constant rate therefore a download lasts a particular time depending on the file size. The function of this anonymous agent is to compute downloads for each node after a time point. This agent is stepped immediately after the mobile agents.

- **CalculateAverageDownloadRate**: This agent captures output parameters in form of statistics. At each nth step, it calculates the average bandwidth of all nodes, the number of connected friends, the number of unconnected friends and the number of failed download attempts. These

- **GraphPlotter**: is the last stepped agent; it is stepped immediately after agents have been stepped. The function is to update the graph data at each time step.

### 5.5 Parameters

The simulation has several parameters that can be provided by the user or are calculated as an aggregate statistics. For example, *numNodes* and *NumFiles* control the initial number of mobile cloud user devices and number of files in the simulation. There are two set of parameters, the input and output parameters.

#### 5.5.1 Input Parameters

The input parameters are the initial parameters needed to run the simulation. The values for these parameters can be entered via the simulation console or input files. Though, default values are automatically set for each parameter when the simulation is loaded. These parameters are shown Figure 5.
A detailed description of the input parameters is illustrated in table 2:

**Table 2: Simulation input parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numNodes</td>
<td>1000</td>
<td>Number of mobile cloud user devices</td>
</tr>
<tr>
<td>numFiles</td>
<td>100</td>
<td>Number of files in the simulation</td>
</tr>
<tr>
<td>maxNumFiles</td>
<td>20</td>
<td>Maximum number of files a user device can own</td>
</tr>
<tr>
<td>neighbourhood</td>
<td>10</td>
<td>The range in which a mobile node’s neighbourhood relationship with other mobile nodes is defined</td>
</tr>
<tr>
<td>downloadSpeed</td>
<td>196</td>
<td>The rate at which service (file downloads) is delivered in kilobit per seconds. This is a constant rate.</td>
</tr>
<tr>
<td>maxDownloadSlots</td>
<td>3</td>
<td>The maximum number of service provision slot for a mobile node</td>
</tr>
<tr>
<td>maxSizeofFile</td>
<td>19968</td>
<td>Maximum size of a file in kilobytes</td>
</tr>
<tr>
<td>minSizeofFile</td>
<td>512</td>
<td>Minimum size of file in kilobytes</td>
</tr>
<tr>
<td>requestArrivalRate</td>
<td>30</td>
<td>The average arrival rate of service requests at each time step. This can be considered lambda in a Poisson distribution.</td>
</tr>
<tr>
<td>maxNoOfFriends</td>
<td>130</td>
<td>Maximum number of friends a mobile cloud user associated to a device can have</td>
</tr>
<tr>
<td>policyNo</td>
<td>1</td>
<td>The policy no to be implemented</td>
</tr>
</tbody>
</table>

**5.5.2 Output Parameters**

The output parameters are calculated as aggregate statistics. For example, the average download rate at each time step. These output parameters are used as values to plot graphs for simulation results. MASON has graphing capabilities, which makes it easy to plot and update time series graphs at each time step. A detailed description of the output parameters is illustrated in the table below.

**Table 3: Simulation output Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>noOfConnectedFriends</td>
<td>Total number of times service request is placed to friend’s user device at each time step</td>
</tr>
<tr>
<td>noOfUnconnectedFriends</td>
<td>Total number of times service request is placed to non-friend’s user device at each time step</td>
</tr>
<tr>
<td>noOfFailure</td>
<td>Total number of service request failure at each time step</td>
</tr>
<tr>
<td>averageBandwidth</td>
<td>The average service provision rate of all mobile nodes in the simulation at each time step</td>
</tr>
<tr>
<td>standardDeviation</td>
<td>The standard deviation of the loads of all mobile nodes in the simulation at each time step</td>
</tr>
</tbody>
</table>

**5.6 Initialization**

When the simulation starts, the first step is to set up its agents and their locations based on the parameters. Initially, the space in which the mobile cloud area is going to occupy is created. This involves setting up the simulation area. Then, the files that will be present in the simulation are created. Each file is assigned an ID and its size is randomly generated. Though, there is a minimum and maximum file size controlled by the parameters.
Then, the mobile nodes (agents) are created and a random number of files are allocated to them. Files are chosen at random to fill the required numbers of files each node have initially. Also, the social network graph in the simulation is defined. Each mobile node is allocated a random number of friends. Friends of a mobile node are randomly selected and there is a limit to the number of friends it can have controlled by a parameter. The scheduler takes over after and it steps the simulation through time allowing agents to carry out their tasks.

5.7 Simulation Walkthrough

The first agent stepped is the anonymous agent that randomly generates file requests. The number of requests generated at each time is drawn from a Poisson distribution taken account the average rate of request arrival parameter. Then, the generated file requests are randomly assigned to the mobile nodes in the cloud. The mobile nodes in this simulation have basic operations: they either operate as a service provider (server) to provide files to other mobile cloud user devices or a service requestor (client) that requests to download a file from list of available service providers. If a mobile node needs a multimedia content, it conducts a local search to see which node it can download from. If the search is successful then it places request to download the media file to a node that can serve it. A request is placed depending on the policy employed based on the social relationship between the requesting mobile node (client) and the mobile node (server) that can serve the request. A mobile node (server) serving the request transfers the file to the mobile node that requested the file (client) at a constant rate. The size of the file is to taken so that the download lasts for a particular period of time. Figure 6 outlines these activities.

On the other hand, if the search is unsuccessful, the mobile node can delegate a request to an idle mobile node to conduct a local search within its own neighbourhood. Upon accepting to conduct a search as a mediator, a routing record is created with information about the originator of the request. At each pass, the routing record is updated with the information about the intermediate requestor. At some point, if the search is successful then the delegated mobile node places a request to download this file on behalf of the initial requesting mobile node. As the node is downloading the file, it channels the download to the node that initially placed the request. The delegation of request stops after a search has been conducted in the neighbourhood a level above the requesting mobile node’s level. While a mobile node is providing multimedia contents (service) to other mobile nodes, there may be some nearby mobile nodes that are idle or have much lower load. In this case, the mobile node transfers one of its jobs to the idle mobile node. Figure 7 illustrates these activities. After the download finishes, the file is added to the list of files the initial requestor owns.
6. EVALUATION

In this section, the evaluation results from running the simulation of mobile cloud users participating in the 2012 London Olympic opening ceremony is presented. Mainly, the evaluation of different policies of the model is based on the requirements (social maintenance and load balancing) that way established earlier in this research by tuning the important parameters.

6.1 Experimental Setup

The experimental setup models multimedia sharing between a number of mobile cloud users participating in the 2012 London Olympic opening ceremony. A mobile node can either request to download a media file from a nearby node or provide media files to a number of nodes within the cloud. At each time step, a random number of requests drawn for a Poisson distribution are generated. If a mobile node successfully selects a node to receive content from then the request is considered successful while if it does not find a node to download from, it is considered the service failed. For each policy, a simulation is run with mobile nodes implementing that policy and the results are compared with the other policies. Each simulation is run for 500 time steps with an average of 30 service requests arrival. The retrieved results from the simulation are collected and represented as a graph. Due to the wide variations in the results, which can be attributed to random arrival of requests at each time step, a 10-point moving average is calculated to see a trend in the results. This is calculated for all results in this section. This experiment makes certain assumptions about the simulation runs. The assumptions with brief justification are given below:

- The data transfer rate is 196 Kbit/s. theoretically, the maximum data transfer rate for Bluetooth communication is 3 Mbit/s but the real world experience is usually half of this.
- The social graph of Facebook is used for simulation. This assumption is based on Facebook being the most popular social network in the world with 1 billion monthly active users.
- An average Facebook user has 130 friends.

6.1 Social Relationship Maintenance

One of the main aims of the proposed model is to maintain social relationships in order to achieve high trust levels while providing services. The three policies of the model are evaluated to see how many social relationships are maintained during service provisioning. This is measured by counting the number of friends a mobile node connects for service provisioning at each time step.

6.1.1 Neighbourhood Distance

An important aspect of the proposed model that enhances service discovery and selection is the neighbourhood distance. In this section, we study how the neighbourhood distance affects the social relationship performance of the different policies of the model. For this simulation, the neighbourhood distance is set to 10. This is based on the assumption that the mobile nodes transfer files using Bluetooth connection. The broadcasting range in which mobile devices can effectively transfer files is 10 meters. The first simulation was run with following parameters as shown in Table 4.

Table 4: Parameters used for the first simulation

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM_NODES</td>
<td>1000</td>
</tr>
<tr>
<td>REQUEST_ARRIVAL_RATE</td>
<td>30</td>
</tr>
<tr>
<td>MAX_NUM_FRIENDS</td>
<td>130</td>
</tr>
</tbody>
</table>
Based on the experimental test scenario described above, the social connection performance between agents adopting a particular policy is presented in figure 8.

![Social Connection Plot](image)

**Figure 8:** Social connection comparison for all policies when neighbourhood distance is 10

As shown in figure 8, the trend of social connections is lower where the mobile nodes adopt policy 1 for service selection, discovery and provision. In policy 1, mobile nodes do not only download media contents from friends alone, they can also choose non-friends to download from as long as they are able to provide the content. In the simulations where mobile nodes adopt policies 2 and 3 where mobile nodes only download from friends have a higher number of social connections. Results from policy 3 are significantly better than that of policy 2. This could be attributed to the delegation task used in policy 3 where mobile nodes can delegate request to friends to conduct a further search on their behalf. If search is successful, service provisioning is initiated between the service provider and the mediator (delegate node) while channelling to the initial requestor.

![Download Failure Plot](image)

**Figure 9:** Service requests failure comparison for all policies when neighbourhood distance is 10

When a mobile node is unable to select a suitable node to serve its request then such service request is considered to fail. Figure 9 shows the trend in the total numbers of service failures at each time step. The simulation where mobile nodes adopt policy 1 has a lower trend in the number of failed downloads than policies 2 and 3. This can be attributed to policy 1 allowing mobile nodes to download from non-friends, which increase the chance for a successful service discovery and selection. The trend in the number of downloads in Policy 2 is the highest. It is slightly higher than the trend in policy 3. The low performance in Policy 2 is due to the strict adherence of only downloading from friends without extra attempt to increase the chance of service discovery similar to the delegation function in policy 3.

Another set of simulations was done for all policies while increasing the neighbourhood distance to 50. As was stated earlier, mobile devices can also provide services using Wi-Fi connection, which covers a larger broadcasting range hence, a larger neighbourhood relation. All parameters used for the first sets of simulation except for the increase in neighbourhood to model a larger neighbourhood relationship. The table below presents the list of parameters used for this simulation.

<table>
<thead>
<tr>
<th>Table 5: Parameters used for this simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM_NODES = 1000</td>
</tr>
<tr>
<td>REQUEST_ARRIVAL_RATE = 30</td>
</tr>
</tbody>
</table>


Figure 14 displays the trends in total number of social connections for all policies with the above parameters. The trends in this graph is similar to the trend in the result of the first set of simulations shown in figure 8, although the difference in the trends total number of social connections between all policies in this simulation is lesser. It shows that increase in the neighbourhood relation between nodes increases the chance of service discover, selection and provision. The trend in the total number of social connections in the simulation where mobile nodes adopt policy 3 remains higher than that of the other policies.

Figure 11: Service requests failure comparison for all policies when neighbourhood distance is 50

This result confirms the above observations: when mobile nodes adopt policy 3 for service management, it is expected to have larger social connections between mobile nodes due to the delegation capability. This means a mobile node requesting for a service always have a larger neighbourhood apparently because it can conduct search for service providers farther than its own neighbour via mobile nodes nearby. Furthermore, when mobile nodes adopt policy 2 for service management, it is expected that there would be last download failures because of a larger pool of service providers compared to other policies where mobile nodes only select friends to provide services.

6.1.2 Number of Nodes

Another set of simulation was done reducing the initial number of 1000 nodes by half to 500 nodes. This is to test how the systems react to scalability, as the numbers of nodes are expected to change dynamically. For example, mobile devices enjoy the convenience of mobility hence mobile cloud users leave and enter the network. Also, a mobile node may leave the cloud due to battery exhaustion.

Table 6: Parameters used for the simulation

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM_NODES</td>
<td>500</td>
</tr>
<tr>
<td>REQUEST_ARRIVAL_RATE</td>
<td>30</td>
</tr>
<tr>
<td>MAX_NUM_FRIENDS</td>
<td>130</td>
</tr>
<tr>
<td>MAX_SIZE_OF_FILE</td>
<td>19968</td>
</tr>
<tr>
<td>MIN_SIZE_OF_FILE</td>
<td>512</td>
</tr>
<tr>
<td>NEIGHBOURHOOD</td>
<td>10.0</td>
</tr>
</tbody>
</table>

Figure 10: Social connection comparison for all policies when neighbourhood distance is 50

Figure 11: Social connection comparison for all policies when neighbourhood distance is 50

Figure 11: Service requests failure comparison for all policies when neighbourhood distance is 50

This result confirms the above observations: when mobile nodes adopt policy 3 for service management, it is expected to have larger social connections between mobile nodes due to the delegation capability. This means a mobile node requesting for a service always have a larger neighbourhood apparently because it can conduct search for service providers farther than its own neighbour via mobile nodes nearby. Furthermore, when mobile nodes adopt policy 2 for service management, it is expected that there would be last download failures because of a larger pool of service providers compared to other policies where mobile nodes only select friends to provide services.

6.1.2 Number of Nodes

Another set of simulation was done reducing the initial number of 1000 nodes by half to 500 nodes. This is to test how the systems react to scalability, as the numbers of nodes are expected to change dynamically. For example, mobile devices enjoy the convenience of mobility hence mobile cloud users leave and enter the network. Also, a mobile node may leave the cloud due to battery exhaustion.

Table 6: Parameters used for the simulation

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM_NODES</td>
<td>500</td>
</tr>
<tr>
<td>REQUEST_ARRIVAL_RATE</td>
<td>30</td>
</tr>
<tr>
<td>MAX_NUM_FRIENDS</td>
<td>130</td>
</tr>
<tr>
<td>MAX_SIZE_OF_FILE</td>
<td>19968</td>
</tr>
<tr>
<td>MIN_SIZE_OF_FILE</td>
<td>512</td>
</tr>
<tr>
<td>NEIGHBOURHOOD</td>
<td>10.0</td>
</tr>
</tbody>
</table>
Figure 12 displays the result when the simulations are run with 500 mobile nodes. The behaviour of all policies in the presented result is similar to the results from running the simulation with larger number of mobile nodes. Though, the trend in number of social connections generally is low compared to when the numbers of nodes are larger. Essentially, this can be attributed to the lower number of service providers. Since the number of nodes has reduced, there are fewer mobile nodes to choose from to provide services.

In line with the trend in the above graph, the service request failures are significantly demonstrating that lesser number of mobile nodes reduces the chance of service discovery. Figure 13 shows the trend in service request failure when there is lesser number of mobile nodes.

### 6.2 Load Balancing

Another aspect of the model is load balancing behaviour, which aims to reduce delays at the service provider during service provisioning. For instance, a mobile user will prefer to select a service attached to a mobile user that is less occupied compared to another user is more occupied. In this section, the viability of the load balancing approach presented in the proposed model is evaluated. All the three policies employ the same approach for service selection with regards to load balancing, therefore it is expected that load is distributed among all mobile devices. From the results presented above, the trend of service request failure is significantly lower when mobile nodes adopt policy 1 for service management than the other policies. This shows that mobile devices generally achieve higher utilization rate when they adopt policy 1 compared to when mobile nodes adopt either policy 2 or policy 3. Thus, a simulation was run with all mobile nodes adopting policy 1 to see how load is balanced between mobile nodes. The table 7 shows the list of parameters used for this simulation.

<table>
<thead>
<tr>
<th>Table 7: Parameters used for this simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUM_NODES = 1000</td>
</tr>
<tr>
<td>REQUEST_ARRIVAL_RATE = 30</td>
</tr>
<tr>
<td>MAX_NUM_FRIENDS = 130</td>
</tr>
</tbody>
</table>
To analyse the service selection with regards to load balancing in a quantitative fashion, the standard deviation of the load (service provisioning bandwidth) for all mobile nodes is analysed at each time step. The results are presented in Figure 14.

Figure 14: Plot showing average download rate and standard deviation of loads of mobile nodes while adopting policy 1

Figure 14 shows the average download rate and the standard deviation of the load for a simulation when nodes adopt policy at each time step. The average standard deviation over the 500 time steps is 71.44kb/s (Max: 103.39kb/s; Min: 25.35kb/s). Mostly, the standard deviation at each time step was below the average standard deviation across all time steps. Although this standard deviation appears high, it is less than 20% of the average download rate for all average time steps. This shows that load is fairly balanced. The high standard deviation is expected because of certain conditions that affect service provisioning. For example, an idle mobile node may not be able to provide a particular service at a particular time due to service unavailability therefore the service requestor needs to select another node even though it is serving other requests. Moreover, an idle node that is able to provide a service might not be within the service requestors neighbourhood to select for service provisioning. Based on the results presented in this section, it can be suggested that the achieved social connections are marginally dependent on the policy adopted and the neighbourhood relationship between mobile cloud users’ devices. Furthermore, load balancing performance is similar for all policies since they adopt the same approach to balance node.

6.3 Discussion

The proposed model expresses a step forward on realising autonomic service management in mobile cloud infrastructures. It provides a mechanism in which mobile devices collectively exhibit certain behaviours autonomously for more effective service discovery and composition. This model was shown to be well suited for mobile cloud environment as mobile cloud users’ devices are able to interact locally to produce emergent behaviours that successfully manage service provisioning in a decentralised manner.

The model presented in this study demonstrates how mobile cloud infrastructures can be configured to provide services while considering social and privacy issues. To support this, the model exploits the graph of an existing social network to infer trust relationships between user devices. The model ensures that priority is given to social relationship maintenance over any other behaviour. This forms the basis of interactions between user devices in the mobile cloud environment. A benefit this gives is that it makes it easy to control social interactions without requiring re-configuration of services. Also, service discovery and composition are more effective given that selection of services rely on social existence between users’ devices. Therefore, a mobile device only provides services to nearby friends and it does not have to cater for the whole mobile cloud. Since social interactions are based on social network relationship there is lower privacy concerns, as most users will be interacting with mobile cloud users’ who are their real-life friends. In this case, users’ devices select service providers whom
they are comfortable sharing sensitive information such as user location, service usage pattern with.

On the other hand, the experiments conducted above shows that the rates of service request failures are much higher when mobile nodes are only restricted to interact with friends. There are always a lesser number of mobile devices to select for service provisioning compared to when mobile nodes are allowed to non-friends. However, in most cases the service requests are relatively low for all policies.

As a supplementary behaviour, the model also aims to reduce delay at the service provider due to on-going service provisioning by distributing load across all mobile devices through load balancing. Load balancing is supported in the presented model by establishing that mobile agents select a service provider with the least download rate for provision of services and that a service provider tries to match the downloading rate of nearby service providers. Load balancing ensures that the utilisation rate of each user’s device is reduced to the minimum level. Overall, this improves average waiting queue for service requests because service requests may be delayed at the service provider due to on-going service provisioning.

7. CONCLUSION

From the study, a number of key insights were generated and all of the research questions that were proposed at the outset of the study have been addressed. Firstly, the study presented a decentralized model for autonomic service management in mobile cloud infrastructures. The model that was presented as the mechanism for efficient service discovery and composition within this study was inspired by the flocking mechanism designed by Reynolds (1986) to capture how a large number of birds interact autonomously to produce an emergent behaviour (flocking). Secondly, the proposed mechanism takes advantages of pre-existing trust relationships in a social network so as to enhance the autonomous discovery and composition of services in mobile cloud infrastructures while accounting for social and privacy issues. In order to discourage delays at the service provider due to on-going service provisioning, the model also employs load balancing between the mobile devices. Working independently, mobile users try both to ensure social relationship are maintained with one another and avoid overloading during service provisioning in the mobile cloud.

To demonstrate the proposed mechanism, a local media content exchange system for the opening ceremony of the London 2012 Olympic was simulated. Different variations presented as policies of the proposed mechanism were created to support how mobile users exchange media content between one another. By comparing these variations, the study has shown empirically that services can effectively be discovered and composed autonomously based upon the social relationship of mobile users. In addition, the study has shown that even though service provision is subject to social relations between mobile users, load is fairly distributed amongst all mobile devices to satisfy the requirement of minimum delay at the service provider due to on-going service provision.

8. REFERENCES


