Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A novel pre-cache schema for high performance Android system

Hui Zhao [a], Min Chen [c], Meikang Qiu [b,*], Keke Gai [b], Meiqin Liu [d]

[a] Software School, Henan University, Kaifeng, Henan, 475000, China
[b] Department of Computer Science, Pace University, New York City, NY 10038, USA
[c] School of Computer Science and Technology, Huazhong University of Science and Technology, Wanhan, China
[d] College of Electrical Engineering, Zhejiang University, ZJ 310027, China

HIGHLIGHTS

- Propose a novel model for high-performance Heterogeneous Android systems.
- Use middleware-based pre-cache technology.
- Use middleware-based approach to save networking traffics.
- Generate Version Flags, which is a proposed new component of web pages.

ARTICLE INFO

Article history:
Received 12 March 2015
Received in revised form
7 April 2015
Accepted 5 May 2015
Available online 21 May 2015

Keywords:
Android systems
Pre-caching
Middleware services
High performance
Distributed computing systems

ABSTRACT

As a mobile operating system framework, Android plays a significant role in supporting mobile apps. However, current Android application model is not efficient by using current two common approaches, including *Activity+XML Layout Files* (AXLF) and *HTML+WebKit* (HWK) models. In this paper, we propose a novel middleware service solution that overcomes the drawbacks with using the pre-cache approach, *PrecAche Technology of Android System* (PATAS). The proposed method uses HTML to design the application interface and separately store the *Page Framework* (PF) and *Page Data* (PD). We create a new middleware of web pages, *Version Flags*, to indicate whether PF and PD are expired. Our experimental results represent that the proposed approach can improve the execution efficiency as well as reduce the networking costs, which can be broadly used in cloud-based distributed systems.

1. Introduction

With the rapid development of the mobile technologies, mobile applications have been broadly used in multiple domains. The implementations of the mobile apps have brought mobile users a variety of benefits, such as entertainments, communications, and personal information management. Mobile devices have become a platform supporting mobile apps and wireless communications [1]. The high-performance of wireless networks are also beneficial for the expansion of the mobile apps, such as distributed computing and mobile cloud computing [2,3]. The openness of Android systems brings a higher-level flexibility for implementing multiple wireless technologies [4]. However, current Android apps are facing a great challenge in efficiency and fast response to the user demands due to the inefficient execution models [5]. This paper addresses this issue and proposes a novel approach for high performance Android systems by using pre-cache technologies.

As a popular framework, Android is a mobile phone operating system that is established on the basis of the Linux kernel. Considering a complete open mobile platform, the emergence of Android has brought a large number of opportunities to smart phones as well as challenges. Android system has become a popular operating system on mobile embedded systems, which has been continuously increasing as the improvements of computation capability, wireless networks, and distributed deployments [6–8]. Cloud computing provides Android systems with a broad service delivery platform with a strengthened communication capability [9]. Content caching is an approach for optimizing performances of Android systems in distributed computing systems [10,11].

Nevertheless, increasing the efficiency of the Android apps is a challenging issue for current mobile developers. There are two traditional app approaches for Android, including *Activity*

* Corresponding author.
E-mail addresses: zhh@henu.edu.cn (H. Zhao), minchen2012@hust.edu.cn (M. Chen), mqiu@pace.edu (M. Qiu), kg71231w@pace.edu (K. Gai), liumeiqin@zju.edu.cn (M. Liu).

http://dx.doi.org/10.1016/j.future.2015.05.005
0167-739X/© 2015 Elsevier B.V. All rights reserved.

+ *XML Layout Files* (AXLF) and *HTML* + *WebKit* (HWK). First, as an official method recommended by Google, AXLF performs a lower-level response to user demands, even though it supports all great features of Android system. The other approach is HWK, which uses HTML technology to develop mobile apps. Despite the efficiency of HWK is better than AXLF, the drawback of using this model is that a lot of unnecessary documents will be generated by repeatedly downloading from remote servers. Therefore, there is an urgent demand for finding out an approach that can take all advantages of prior models as well as avoid disadvantages.

Addressing this issue, we propose a novel method named *PrecAche Technology of Android System* (PATAS). The proposed paradigm mainly consists of three algorithms, namely *Server-Side Execution Procedure Algorithm* (2SEPA), *On-Premise Load Page Algorithm* (OPLPA), and *Page Synchronization Algorithm* (PSA). Implementing the proposed schema aims to prevent from unnecessary downloading data base on the high efficiency performance. The main contributions of this paper are twofold:

1. We proposed a novel model for high-performance Android systems using the pre-cache-based technology.
2. Our schema uses middleware-based approach to save networking traffics by generating *Version Flags*, which is a proposed new component of web pages.

The remainder of the paper is organized as the following order: In Section 2, related works in Android domains are reviewed. Section 3 describes the models and concepts used in our proposed schema. A motivational example is provided in Section 4. Next, the details of our proposed algorithm is given in Section 5. The experimental results and the conclusions are represented in Sections 6 and 7.

2. Related works

Prior research has addressed Android research in multiple perspectives. This section reviews the related works concerning Android models by using AXLF and HWK. Being aware of the Android systems, the *Open Handset Alliance* (OHA) was established by Google and its partners in November 2007 when released the first Beta version of the Android operating system, *Software Development Kit* (SDK). More than 80 firms in mobile domains attach to the organization and most mobile apps follow the standard formed by OHA.

Due to the restrictions of the computing capabilities, current mobile apps are required to perform low energy costs [12]. Many previous research has addressed the issue of energy consumption [13]. Leveraging mobile cloud-based solutions is an option for mobile apps to achieve low energy costs, high performance, and optimal task scheduling [14–16]. Using heterogeneous distributed computing has a positive impact on mobile computation offloading and connections across multiple platforms [17]. The benefits of heterogeneity in mobile distributed computing enable many mobile systems to migrate the focuses into interface design and data display with implementing human–computer interactions.

AXLF is one of the classic methods supported by Android systems. Applications use this model can straightly use the components provided by Android as well as maximize the usage of the Android system resources to achieve a dazzling visual effects. Multiple functionality-aimed apps can be executed by this approach, such as games, multimedia, and online communications. Two crucial parts in this model include *Activity Manager* (AM) and *View System* (VS). Using XML layer files to describe user interface layout is able to reduce the complexity of the system [18]. The interconnections between platforms can enable the apps communications based on the cross-platform tools [19].

However, some mobile apps do not need fashionable user interfaces but need complex system structures, such as e-dictionary, *Office Automation* (OA) systems, and E-commerce-related apps. Using AXLF approach can result in heavy workload and interface executions. In addition, the default component provided by Android needs users to have a strong background in computing, which results in the difficulties in workload controls. There are some system which need to be provided traditional computing network access and mobile access both, it will take twice as much time to developed respectively webpage design and mobile applications, and these two works cannot be shared by each other, this also greatly increased the workload of developers.

HWK is the other optional approach because the AXLF model is not efficient in running apps delivering information services. When using this model in Android applications, HTML technology is applied to develop webpage so as to achieve the applications user interface and response functions of each component. Then WebKit components of Android are used to load the page to display complex user interface and useful information in the form of the page [20]. When programmers develop applications with using this model, such as writing a webpage, the pages in the website can be directly used as the mobile user interface. Loading the page for the application user interface can achieve human–computer interactions. Currently, this model has gradually become one of the mainstream models for Information-as-a-Service mobile apps. However, implementing HWK model usually need a lot of extra execution time and can cause heavy networking traffic. Due to the workload of updating data, users have to download many extra contents.

Applications can store the data on-premise when the information is not needed to be changed, such as e-dictionaries, which implies that the wireless networks are not required if the source is stored locally. A middleware who can temporarily or permanently store the resource is an optional method when the resources are migrated to the clouds [17,21,22]. Using a middleware-based model has a potential to solve the problems we mentioned before. However, apps implementing HWK model must download all pages when apps needs to update the details of the information, even though most page information is already stored in mobile devices. This results in a longer execution time and unnecessary network traffics. Therefore, implementing HWK model usually needs extra execution time and can cause heavy networking traffics. Users have to download extra contents due to the workload of data updates.

3. Models and concepts

In this section, we introduce the basic models and concepts of our proposed schema, PATAS. Two crucial aspects of PATAS are covered in this section, including HTML page separation and execution procedure re-design. The whole HTML page is separated into two components, including *Page Framework* (PF) and *Page Data* (PD). A new component of web pages is introduced in our proposed schema, which is set of *Version Flags*. A version flag is a middleware that indicates whether PFs or PD are expired. Fig. 1 shows the architecture of PATAS.

Page Framework and Page Data: A PF refers to the framework of web pages, such as page layout, configuration information, and components and executions. A PD represents the real-time data that need load into the page, such as prices and images in e-commerce. Using pre-caching technology can allow applications to avoid downloading the re-stored PF and PD. Workload can be reduced by using this model without increasing networking traffics.

Page Separation: we separate the page contents into two parts, PF and PD. The purpose of page separation is that the mobile apps can respectively download and cache the two parts to the local device when running apps. Two parts are combed into a complete