Algorithmic Software Adaptation Approach in Mobile Augmented Reality Systems

Oleksii Vekshyn
Computer Aided Management Systems department
National Technical University "Kharkov Polytechnic Institute"
Kharkov, Ukraine
alexeyvekshin@gmail.com

Mykola Tkachuk
Computer Aided Management Systems department
National Technical University "Kharkov Polytechnic Institute"
Kharkov, Ukraine
tka@kpi.kharkov.ua

***Abstract*—Growing complexity of mobile software systems leads to problems with productivity and multiple devices support in such applications, especially in augmented reality systems. In this paper, the new approach to algorithmic adaptation for mobile augmented reality systems is proposed. This approach is based on complexity estimation of business logic and separation of computation loading between client and server sides that increases software performance and usability in augmented reality systems. This paper illustrates the main concept and provides some design issues of the proposed approach.**

Keywords-augmented reality; software; mobile systems; algorithmic adaptation

I. INTRODUCTION

Nowadays, mobile information systems become more and more popular. One of the most complex and dynamically grown type of these systems are augmented reality systems (ARS) [1]. Such systems require more hardware resources than standard mobile applications for social networks, and this fact leads to supporting problems of different devices such as mobile phones and tablets. One of the possible solutions is an execution of complex business logic on the server side, where computational capabilities are higher than on the mobile client side; but on the other hand this, could lead to problems with application response time and energy efficiency because of more intensive usage of wireless networking technologies. In this paper we, propose an approach which is based on computational complexity estimation on design-time and analysis of ARS (Augmented Reality System) [1] state in run-time for algorithmic adaptation of mobile ARS. The elaborated approach helps to define which part of business logic should be executed on mobile the client side, and which one on the server side, depends on CPU-performance (Central Processing Unit) of a mobile device.

The paper is structured in the following way: Section 2 depicts briefly some modern trends in this research domain, w.r.t. software adaptation issues; Section 3 provides the review of mobile ARS reference architectures; the formal definitions of the proposed approach are introduced in Section 4; in Section 5, the elaborated algorithm and the appropriate software architectural solution for proposed approach are represented. Finally, Section 6 concludes the paper and gives a short outlook on some future works on this research.

II. SURVEY OF AUGMENTED REALITY SYSTEMS AND ADAPTATIONAL SOLUTIONS FOR MOBILE SYSTEMS

An Augmented reality (AR) is a representational form for a real physical environment, which is extended by adding of computer generated data [2]. AR registers physical objects in three dimensions and combines them with the virtual ones. Unlike the concept of virtual reality, which completely replaces the real world with the virtual one, AR uses a combination of them both.

The ARS operate with such data sources as: two-dimensional markers; data received from GPS-modules (Global Positioning System) [3] and from build-in gyroscopes; they use technologies like images recognition without any markers and GPS data [4].

For implementation of mobile ARS's several frameworks could be used, e.g., Metaio Mobile SDK (Software Development Kit) [5], D'Fusion Mobile [6] and Qualcomm [7].

Nowadays, software adaptation is one of the common trends in modern software engineering (see, e.g., in [8]), and especially in mobile application development. There are several approaches to adaptation in mobile systems, some of them are represented in projects like Q-CAD (QoS and Context Aware Discovery), MADAM (Mobility and Adaptation Enabling Middleware), IST-MUSIC (Self-Adapting Applications for Mobile Users in Ubiquitous Computing Environments), etc [9].

Q-CAD is a resource discovery framework which enables mobile applications to discover and to select resources best satisfied the user's needs. MADAM and ITS-MUSIC frameworks provide model-driven development approach enabling to assemble applications through a recursive composition process. In this case, variability is achieved by plugging into the same component type different component's implementation with similar functional behavior [9]. In [10], a new approach to the composition of mismatching components in context-aware systems is introduced.

Summarizing the described approaches, we can conclude they do not take into account possibility of ARS algorithmic adaptation based on their complexity business logic estimation. This way, it is possible to separate computation loading between client and server sides in order to increase ARS productivity in run-time mode.

III. REFERENCE ARCHITECTURES FOR MOBILE ARS WITH RESPECT TO ADAPTATION ISSUES

Nowadays, mobile ARS could have two types of reference architectures. The simplest type of Mobile ARS architecture is Standalone architecture [11], which is presented in Figure 1. Such systems work in the following way: mobile devices register the environment using the camera, camera handler processes obtained images and adds data to images with information from local database or GPS-data. This architecture contains several components as: 1) Camera handler is a component, which process data from mobile device's camera; 2) AR objects handler is a component, which processes data from the Camera handler and injects information in images; 3) GPS component is a component, which provides information from GPS controller of the mobile device; 4) Local Database is a component, which provides access to application-specific data stored on mobile devices locally. Standalone architecture has the following features: problems with scalability, maintainability, updating information and extensibility; denial of a whole system in case of single element's denial, high performance in case of low-complex calculations and on the other hand: low performances in case of high computational load; low level of security and easier to manage.

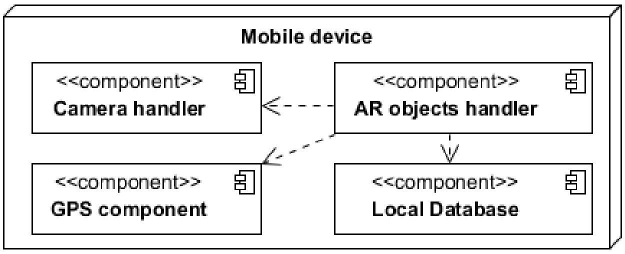

Figure 1. Stand-alone architecture of mobile ARS [11]

Second type of Mobile ARS implementation is Two-tier client-server architecture [12] which is shown in Figure 2 as a UML 2.0 [13] deployment diagram. There are two tiers in this architecture: application or user interacts with client-side software, in which most cases just provides interface to a server-side application. The client side application invokes server on demand, in some cases for modern systems (e.g., all necessary data are available in local database) data could be processed on the mobile client side. For mobile system's client node is a mobile device, which in case of standalone architecture registers objects from an environment. Two-tier architecture in addition to standalone contains following components: 1) AR objects processing web-service is a component witch provides functionality to process AR object on the server side with better productivity; 2) Server-side Database is a centralized storage of AR objects. Two-tier architecture have the following features: high scalability, maintainability, update of information and extensibility, high performance in case of complex calculations and on the other hand: low performance in case of slow network connection.

All ARS types could be designed with either Standalone or Two-tier architecture, so, the reference architecture for Mobile ARS could be selected with respect to software requirements and available resources.

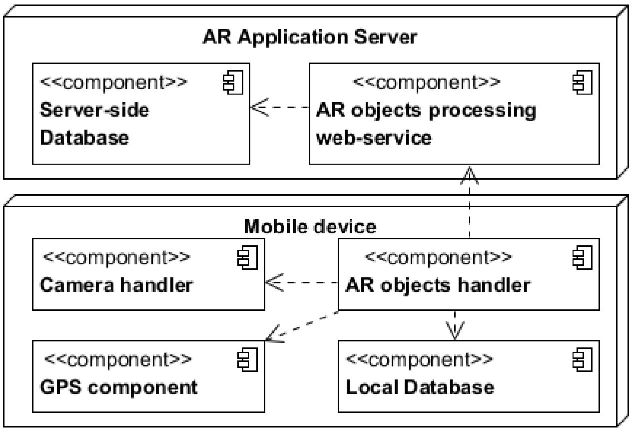

Figure 2. Two-tier architecture of mobile ARS [12]

To design and implement the proposed approach, we choose Two-tier architecture, because of some constraints of Standalone architecture such as: scalability problems and server node's absence, where complex business logic should be executed and where the centralized database should be deployed.

IV. FORMAL DEFINITION OF ALGORITHMIC ADAPTATION FOR MOBILE ARS

The proposed approach to algorithmic adaptation of Mobile ARS can be represented in a formal way using the following definitions.

To measure time of processor work the special parameter: CPU-time has been used. One of the measurement units of CPU-time is MIPS (Million Instructions per Second) [14]. This value has some restrictions, but it could be used in the proposed approach because it is necessary to compare performance of algorithms on the one and the same device.

Definition 1. Precision of calculation for AR is a parameters vector

$$\bar{a} = (f, \quad r, \quad s), \tag{1}$$

where f – a number of decimal places after comma;

r – image resolution (in pixels);

s – ratio of image compression (possible values are from 0 to 1).

These values should be taken into account in the procedure of computation complexity estimation. They implicitly describe possible amount of data for business methods and these influence on amount of operations and calculation time.

Definition 2. A calculation complexity is a vector

$$\bar{c} = \left(c_0,\ \ c_t\right), \tag{2}$$

where c_0 – estimated amount of operations;

c_t – estimated calculation time (is seconds).

Definition 3. A coefficient of a mobile device loading can be estimated with following expression:

$$P = \frac{D_P \cdot 10^6}{\frac{c_0}{c_t}}, \tag{3}$$

where D_p – estimated mobile device performance, MIPS.

Need to notice, that D_p is measured in MIPS, so this value should be multiplied with 10^6 to transform its value to c_0 measurement. This equation illustrates the coefficient of a mobile device load as a ratio of device performance to required amounts of operations in a second.

We consider two different client side application types, which can be classified with UML class-diagram given in Figure 3.

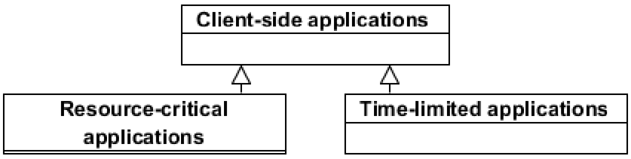

Figure 3. Client-side applications classification

In case of resource-critical applications, we consider the availability of resources for execution of application on the mobile device. For this type of application a calculation on a the mobile device is possible if and only if $P \geq 1$; otherwise it is necessary to execute this on a server side.

The second type of client applications, namely so-called time-limited applications (see Figure 3) could be executed on client side with respect to existing recourses, but due to, time constraints they have to be transferred to server side. In case of time-limited applications calculation on a mobile device side possible if the equation (4) is satisfied.

$$t_{mobile}(\bar{a}) > t_0\ , \tag{4}$$

where $t_{mobile}(\bar{a})$ – data processing time on mobile client side (in seconds);

t_0 – time constraint for calculation (in seconds).

Additionally, for time-limited applications time efforts should be estimated for network communication. If a network connection speed is slow, transferred calculations could reduce application performance. To estimate the time in such a situation, equation (5) is introduced:

$$t_{mobile}(\bar{a}) > t_{request}(\bar{a}) + t_{server}(\bar{a}) + t_{response}(\bar{a}) \tag{5}$$

where $t_{mobile}(\bar{a})$ – data processing time on mobile client side (in seconds);

$t_{request}(\bar{a})$ – time for request sending (in seconds);

$t_{server}(\bar{a})$ – data processing time on server side (in seconds);

$t_{response}(\bar{a})$ – time for response getting (in seconds).

Taking into account the time constraints in time-limited applications we can make the conclusion: calculation transfer is possible if and only if then the equation (4) and the equation (5) are both satisfied.

Below, we consider only resource-critical applications, the time-limited applications and its constraints do not take into account in the proposed algorithm and approach.

V. ALGORITHM OF ADAPTATION PROCESS AND PROTOTYPE ARCHITECTURE

Based on the given definitions (see Definition (1) - (3)) the algorithm of the adaptation process in mobile ARS has been elaborated and presented as UML 2.0 activity diagram in Figure 4.

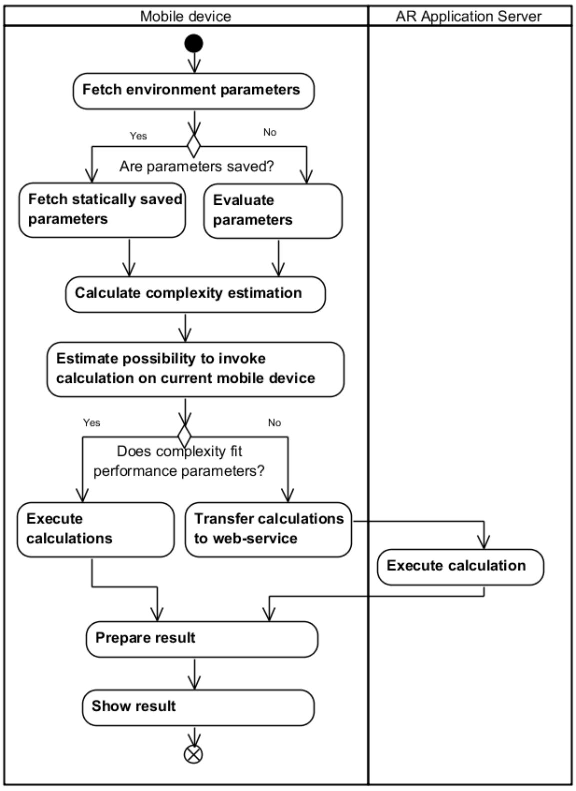

Figure 4. Algorithm of the adaptation process

This algorithm illustrates the process of a node selection where calculation will be executed.

With respect to proposed algorithm, environment's parameters should be obtained and evaluated. Basing on values of this parameters complexity estimation should be calculated (see definition (2)). If complexity estimation requires more resources, than the mobile device performance (see definition (3)), calculations should be transferred to AR Application Server, in other case calculations should be executed on mobile client side. After execution a result will be prepared to visualization and presented to an user.

In case of mobile system, it is possible to apply algorithmic adaptation for selection of node where business logic should be executed. This could improve data processing speed because of complex calculations will be transferred to web-service which provides better computational facilities.

Additionally, it should be noticed that the transfer of calculations requires an estimation of business logic complexity and definition of possibility to execute these calculations on the mobile client side.

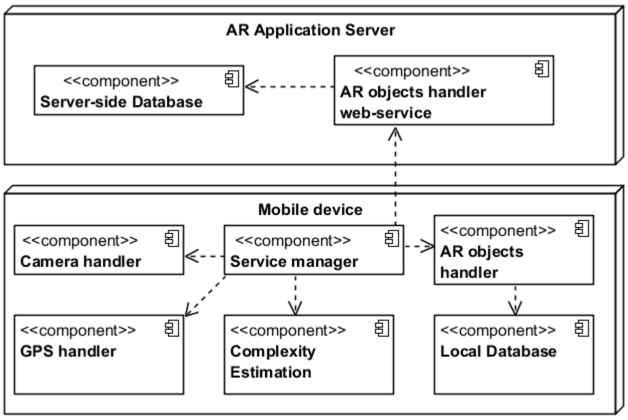

Figure 5. Adaptive mobile ARS architecture prototype

From an architectural point of view, the proposed approach is shown in Figure 5 in the form of UML 2.0 deployment diagram.

The selected two-tier reference architecture has been extended with new two components: 1) Service manager, which analyses context of mobile device, obtains data about business logic complexity and basing on these values generates decision about suitable target node (mobile client or server) for given calculations, and 2) Complexity Estimation, which calculates business logic complexity based on time complexity estimation gathered on mobile application's design time.

To prove the proposed approach, the following example could be used: let CPU-time's value is $D_p = 0.00961\ MIPS$, the precision of calculations is $\bar{a} = (3,\ \ 96000,\ \ 0.6)$, and the estimated complexity of algorithm calculated by Complexity Estimation component is $\bar{c} = (3241,\ \ 0.4)$. In this case, according to the equation (3), value of coefficient of mobile device load is $P = 1.186$, according to proposed approach this calculation could be executed on the mobile client side.

VI. CONCLUSION AND FUTURE WORK

We have presented the approach to algorithmic adaptation in mobile ARS, which increases performance and usability of mobile ARS. The proposed approach allows use of mobile systems in more efficient way with respect to system resources. For now, it is not a fully specified process to measure complexity of business logic in application design time and component to fetch complexity estimation on run-time is considered as "black box". In future work, we plan to specify the procedure to estimate a computational complexity of business logic for mobile ARS and integrate it in the proposed approach. Also, formal definitions will be extended with such mobile device characteristics, like free Random-Access Memory (RAM) size and battery capacity.

REFERENCES

[1] H. Lopez, A. Navarro, J. Relano, An Analysis of Augmented Reality Systems", Proc. of the 2010 Fifth International Multi-conference on Computing in the Global Information Technology (ICCGI '10), 2010, pp. 245-250

[2] B. Furth, Handbook of Augmented Reality, Springer New York Dordrecht Heidelberg London, 2011, p. 746.

[3] Official U.S. Government information about the Global Positioning System (GPS) and related topics, http://www.gps.gov/ 01.11.2012

[4] A. A. Macwilliams Decentralized Adaptive Architecture for Ubiquitous Augmented Reality Systems, 2005, 186 p.

[5] Metaio Mobile SDK, http://www.metaio.com/software/mobile-sdk/ 23.09.2012.

[6] D'Fusion Mobile, http://www.t-immersion.com/products/dfusion-suite/dfusion-mobile 23.09.2012.

[7] Qualcomm AR SDK, http://www.qualcomm.com/solutions/augmented-reality 23.09.2012.

[8] S. Kell, A Survey of Practical Software Adaptation Techniques, J.UCS, vol. 14, 2008, pp. 2110-2157.

[9] K. Kakousis, N. Paspallis, G. A. Papadopoulos, "A survey of software adaptation in mobile and ubiquitous computing," Enterprise Information Systems, vol. 4, Nov. 2010, pp. 355–389, doi:10.1080/17517575.2010.509814.

[10] J. Camara, G. Salaun, C. Canal: "On run-time behavioural adaptation in context-aware systems," Proc. 1st Workshop on Model-driven Software Adaptation (M-ADAPT'07 at ECOOP 2007), 2007, pp. 26–34.

[11] B. Butchart: "Architectural Styles for Augmented Reality in Smartphones", Third International AR Standards Meeting, 2011.

[12] C. Woodward, M. Hakkarainen, M.Billinghurst, "A Client/Server Architecture for Augmented Assembly on Mobile Phones", Research on Mobile Software Engineering: Design Implementation and Emergent Applications, IGI Global Publishing, 2010.

[13] UML 2.0 Specification, http://www.omg.org/spec/UML/2.0/ 23.09.2012.

[14] D. J. Lilja, Measuring Computer Performance: A Practitioner's Guide, Cambridge University Press, 2005, p. 278.