Architecting for Adaptive Resource Management in Mobile Augmented Reality Systems: Models, Metrics and Prototype Software Solutions

Mykola Tkachuk[1,2(✉)], Oleksii Vekshyn[1], and Rustam Gamzayev[1]

[1] National Technical University "Kharkiv Polytechnic Institute", Kyrpychova Str., 21, Kharkiv 61002, Ukraine
tka@kpi.kharkov.ua, alexeyvekshin@gmail.com, rustam.gamzayev@gmail.com
[2] Karazin National University, Majdan Svobody 4, Kharkiv 61077, Ukraine

Abstract. A 3-level architecting approach to adaptive resource management in mobile augmented reality systems (MARS) is elaborated, which is based on comprehensive data structuring and analyzing of their specific hard- and software features. At the conceptual modeling level an ontology of adaptive MARS resources is constructed, and at the logical modeling level a generic algorithmic model is proposed, which can be instantiated in the collection of specific methods and metrics. As a physical model the reference software architecture for adaptive resource management in MARS is designed, and this approach is implemented partly as a software prototype. It is tested successfully to solve the task of adaptive image resolution on mobile device, according to changes of computational load that finally enables better video stream quality in MARS.

Keywords: Mobile system · Architecture · Adaptation · Resource management · Augmented reality · Model · Metric · Case-based reasoning

1 Introduction

Nowadays mobile information systems can be recognized as more and more efficient and comfortable communication facilities for different kinds of their users. One of the most complex and dynamically grown kind of these systems are mobile augmented reality systems (MARS) [1]. Such systems require more hardware resources than standard mobile applications: e.g. social network clients, instant messengers etc., and this fact leads to maintenance problems of different mobile devices (MD) such as smartphones or tablets. One of the modern trends in software development is using of construction principles for complex systems, especially cybernetic adaptive control schemes for software components, including appropriate decision making models and quality evaluation metrics [2]. These methods are also useful in case of mobile information systems development, in particular for MARS. Such system development approach requires effective usage of restricted resources in MD, and on the other hand supposes to implement complex real-time computational algorithms.

© Springer International Publishing AG 2017
A. Ginige et al. (Eds.): ICTERI 2016, CCIS 783, pp. 17–35, 2017.
https://doi.org/10.1007/978-3-319-69965-3_2

In this paper we propose an adaptive model-based architecting framework for resource management in MARS that includes a collection of domain-specific algorithmic models and quantitative metrics to formalize and estimate some relevant parameters of MARS functioning. The rest of this paper is organized in the following way: Sect. 2 depicts briefly some modern trends in this problem domain and introduces a possible classification of MARS with respect to adaptation issues. Section 3 provides the common vision of model-based architecting concept for adaptive resource management in MARS, and Sect. 4 presents the MARS domain modeling issues using an ontological approach. In Sect. 5 the generic model for resources management in MARS is proposed that includes the collection of appropriate algorithmic models and quantitative metrics to provide an adaptation mode for several MARS subsystems. Section 6 describes the software prototype for adaptive screen resolution facility in MARS, and discusses the experimental data of its testing. Finally, in Sect. 7 a short outlook on the achieved results is discussed, and future work is presented.

2 Mobile Augmented Reality System Development: Classification and Short Review of Related Work

Augmented reality (AR) is a visualized representation form for a real physical environment that is extended by adding computer-generated data [3]. Currently AR supports several data sources: two-dimensional markers; data received from GPS modules (Global Positioning System) [4], and data from build-in gyroscopes. Additionally, MARS could use some modern technologies like image recognition without any markers and GPS data [5].

With respect to the supported data source types, it is possible to construct the MARS classification presented in Fig. 1.

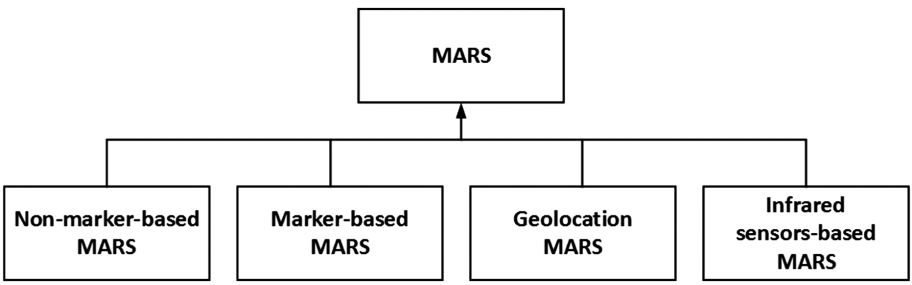

Fig. 1. Classification of MARS implementation technologies

There are four "top-level" types of MARS; each of them has some features and requirements to MD performance. Marker-based MARS operate with special markers which store some data and links to additional information. Non-marker based MARS are more complex type of these systems, they are based on image recognition algorithms, and requires more computational resources to find and recognize free-form objects in an input image. Geo-locational MARS use build-in GPS sensors in order to get

information about real environment and augment it with some virtual data. Finally, infrared-sensors based MARS operate with some infrared sensors, which are able to detect objects and move them in a real environment, subsequently this type of MARS is very useful in entertainment and simulators. In scope of this research we are focused below on marker-based MARS, because this class of MARS has the lowest technical complexity, is easy to implement and does not require any additional equipment apart from MD.

In Figs. 2 and 3 the examples of user interface in the demo marker-based MARS are presented [6]. This MARS detects source object in the input video stream recognize it, obtain 3D graphical model of this gear and finally augment source image with this model. Figure 2 demonstrates interface of the marker-based MARS with not augmented image. In current state MARS is ready to analyze source image and search for markers.

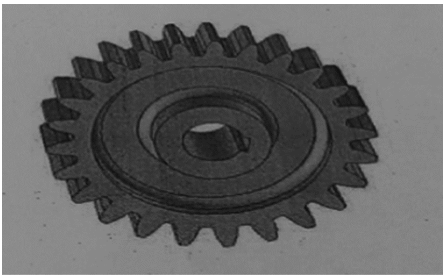

Fig. 2. The initial video image [6]

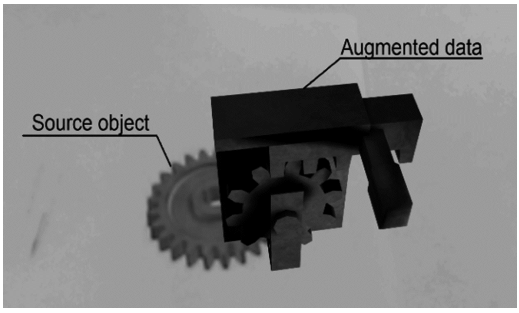

Fig. 3. The example of image with data augmented by marker-based MARS [6]

In Fig. 3 the result image is shown, where MARS finds a marker in the source image (in this case a gear), obtains related data and augments source image with these ones.

In terms of software development there are already some modern software frameworks to develop MARS: Metaio Mobile SDK (Software Development Kit) [7], D'Fusion Mobile [8] and Qualcomm [9].

It has to be noted that last few years a software adaptation became one of the common trends in software engineering, and especially in mobile application development. There are several approaches to adaptation in mobile systems, and some of them were

implemented in such projects as [10, 11]: Q-CAD (QoS and Context Aware Discovery), MADAM (Mobility and Adaptation Enabling Middleware) and IST-MUSIC. Recent development is already focused on cloud-based MARS, e.g. CloudRidAR (A Cloud-based Architecture for Mobile Augmented Reality) [12], Elastic Application Model [13], and on common problems of resource management in mobile cloud computing (see e.g. in [14]).

Q-CAD is a resource discovery framework that enables mobile applications to discover and to select resources best satisfied the user's needs. MADAM and ITS-MUSIC frameworks provide a model-driven development approach enabling to assemble applications through a recursive composition process [11]. CloudRidAR project [12] is a cloud-based framework to MARS development which provides development facilities to construct MARS using all advantages of could computing and code offloading, but on the other hand this framework forces developer to use more complex design solutions. Elastic Application Model [13] is based on code offloading, but flexible application architecture and models are built only on the server side called "weblets", and MD hosts only simple client application, which is connected to these components.

More general (not only focused on MARS domain) approaches to resource management problems in mobile cloud computing take into account cloud structure, code offloading and energy efficiency [14]. Besides that, they utilize some economic- and mathematical models: game theory, auction procedures, and optimization methods (e.g., Nash equilibrium concept using in [15], etc.) with corresponding algorithms to calculate the target parameters for adaptive solutions.

To sum up this short review of related work we can conclude that the most part of existing approaches do not provide any complete model-based framework to adaptive resource management in MARS, because they usually consider only some particular aspects of this problem.

3 Model-Based Architecting for Adaptive Resource Management in MARS

Taking into account the results of the provided analysis and based on the under-standing of modern trends in the domain of adaptive MARS-development (see Sect. 2), we can conclude that it is necessary to elaborate a comprehensive model-based framework for adaptive resource management in MARS. This assumption is completely corresponded with such well-proved and recognized approaches in modern software development as model-driven development (MDD) and model-driven architecture (MDA) [16]. Last time these issues are already discussed intensively in a lot of publications about resource management in distributed real-time systems (see e.g. in [17], in SOA- and cloud-centered applications [18, 19], but there is a lack on such work in the domain of MARS development. That is why we propose to provide such model-based architecting for adaptive resource management in MARS framework in the following way:

- to elaborate a domain model to specify all hard and software resources to be analyzed and considered in any adaptive procedure in MARS, and this model represents a conceptual level in the proposed framework;

- to propose some algorithmic-based approaches to manage these resources in adaptive mode, and this vision about the resources in MARS can be considered as a logical modeling level in our framework;
- to develop a collection of reference software architectures to implement a logical model of MARS with appropriate components and interfaces, and such architecting can be represented as physical modeling level in this framework.

The common scheme of these modeling levels is shown in Fig. 4.

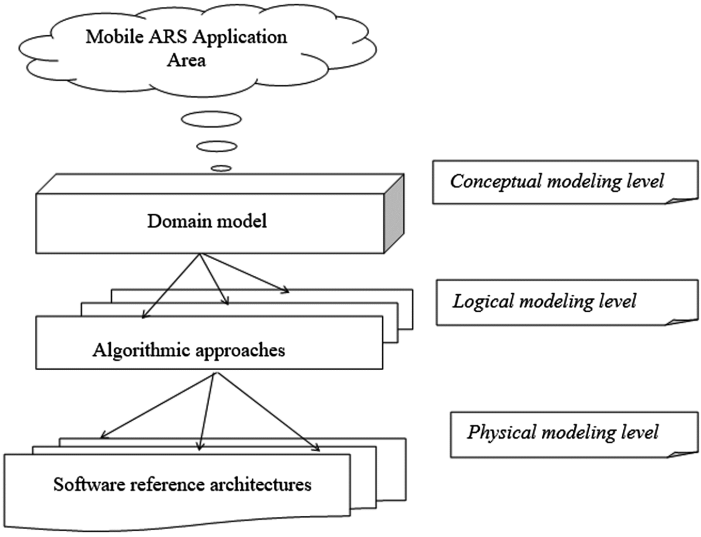

Fig. 4. 3-level modeling framework for adaptive resource management in MARS

According to this model-based view about resource management in MARS for the one and the same domain model a lot of different algorithmic approaches can be elaborated, and for any such approach some different reference software architectures might be implemented. It has to be mentioned that the proposed architecting approach to resource management in MARS completely corresponds, e.g. to well-known and fully recognized 3-level modeling scheme in traditional database development [20].

Basing on the elaborated 3-level modeling scheme (see Fig. 4) and taking into account some specific features of resource management in MARS mentioned in Sect. 2, we propose in this research to consider the following tasks to be solved within our architecting approach to MARS resource adaptation, namely:

1. to provide an adaptive screen resolution (ASR) on mobile devices;
2. to support an adaptive calculation balancing (ACB) between client and server components;
3. to manage an adaptive memory cache (AMC) in run-time mode of MARS.

To solve this list of tasks (1)–(3) it is necessary to elaborate a collection of knowledge-based models, algorithms, numerical metrics, and software solutions that support

an adaptive mechanism for resource management in MARS. They are presented in more details in the following sections.

4 Ontological Domain Modeling for Adaptive MARS

Taking into account the 3-level modeling framework for adaptive resource management in MARS proposed in previous Sect. 3, it is needed to elaborate a domain model for this purpose. This model should represent all relevant hard and software capabilities of MD included in any MARS, which in turn can be used as a collection of adaptable parameters for an appropriate algorithmic modeling approach. We have decided to utilize an ontology-based approach to form the domain model for adaptive resource management in MARS.

Ontology models are widely used to represent relationships between concepts in some application domain, and they can be applied for different purposes in software engineering, e.g.: for information sharing between human actors and machines in Semantic Web applications [21]; in software product line engineering [22], and recently ontological models are also constructed in MARS applications for different purposes.

For example, in [23] authors present the domain model including 4 local ontologies like User, Service, Environment and Device to describe general relationships that occur during MARS development. In [24] the ontology is used to build an information basis of educational institution that could be used for rapid MARS development. In [25] the ontologies are used to connect together the knowledge about the users, environment, and business aims in the given application domain.

Most of ontological models mentioned above describe different MARS components taking into account some static resource allocation, and they do not represent system features needed for adaptive resource management. To close this gap in our approach the new MARS domain model was created using the OWL notation, and the OWLGrEd tool is used for this purpose [26], which provides the UML-like graphical editor. In Fig. 5 the proposed ontological domain model for MARS is presented.

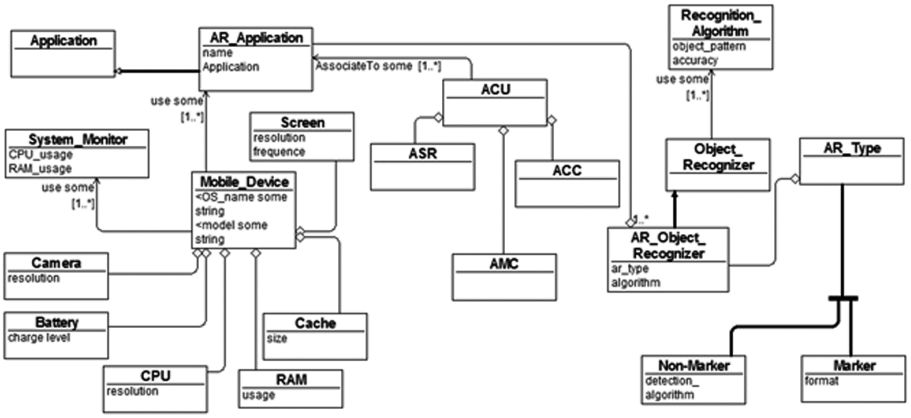

Fig. 5. Ontological domain model for adaptive resource management in MARS

There are the following main entities included in this domain model (see Fig. 5):

1. Augmented Reality Application is an application that uses a technology to analyze elements in the real physical environment and to extend them with additional virtual visual objects or with additional data.
2. Adaptive Control Unit (ACU) is a control facility that is responsible for adaptive resource management in ARS, where the following tasks have to performed: adaptive screen resolution (ASR), adaptive computation balancing (ACB), and adaptive memory cache (AMC).
3. Mobile Device is a small device (iPhone, PDA, notebook, tablet etc.), with restricted amount of system resources, and such a device is managed by an appropriate System Monitor.
4. For resource adaptation of Mobile Device, the following its sub-components (modeling parameters) have to be used: Screen (screen resolution); Cache (cache memory size); RAM (memory size); CPU (computational loading); Battery (charging level); Camera (screen resolution).
5. AR Object Recognizer is a system component to analyze and to extend extracted physical objects with additional or virtual information.

Below this domain model is used consequentially to elaborate all modeling and algorithmic issues for adaptive resource management in MARS.

5 Algorithmic Models and Metrics for Adaptive Resource Management in MARS

According to the proposed adaptive model-based framework (see Sect. 3) at its logical level a collection of algorithmic approaches to adaptive resource management has be constructed with respect to specific hard and software characteristics of mobile devices (MD) operating in a given MARS.

5.1 Generic Algorithmic Model for Adaptive Resource Management

In order to formalize the proposed framework to find adaptive solutions for resource management we can use an algorithmic modeling approach [27, 28], and the appropriate algorithmic model (AM) in a generic way can be defined as the following tuple

$$AM = \langle Workflow\,(Methods),\ InfoBase,\ Metrics \rangle, \tag{1}$$

where *Workflow(Methods)* are some algorithms which implement the given *(Methods)*, *Infobase* is an information base to be used to perform these methods, and *Metrics* is a collection of metrics to quality assessment of adaptation process in MARS. The choice of adaptation methods in (1) depend on the specific features of MARS resources that should be managed, and one of such possible solutions will be considered in the next subsections. The metrics in (1) also have to be defined taking into account the appropriate hard- and software properties of MD which are used in a target MARS (see domain model shown in Fig. 5).

5.2 Adaptive Screen Resolution Management with Case-Based Reasoning

With respect to the generic algorithmic model defined with formula (1) we propose to elaborate the approach to ASR management with case-based reasoning (CBR) as an adaptation technique.

Taking into account a complex and weak-formalized character of MARS functioning, namely: parallel and multi-threaded computational processes, turbulence overloading on MD, frequent changes on number of users etc., it is reasonable to use so-called soft computing methods [29]: neuronal net technologies, fuzzy-logic methods, generic algorithms, CBR and some others. In particular, exactly CBR-methods can be considered as an effective way to develop decision-making procedures for management of complex software systems (see e.g. in [30–32]). According to this statement the collection of (Methods) in formula (1) can be specified as follows

$$Methods = (NNM, kNNM, kwNNM), \tag{2}$$

where NNM is a Nearest Neighbor Method, kNNM is a k-Nearest Neighbors Method, and kwNNM is k-weighted Nearest Neighbors Method [30].

The main idea of all CBR-methods is that any new problem occurred in some application domain can be resolved using already existing solution for the similar situation (called precedent or case). The CBR-methods differ from each other in a search algorithm to find an appropriate precedent in the given database. For this purpose, it is also important to elaborate an adequate description for the precedents representation, which reflects all relevant issues of MARS functionality.

According to formula (1), *InfoBase* is an information base to apply the CBR-methods defined in (2). It includes a set of precedents, and any such precedent can be defined in the following way

$$c = (\vec{p}, \vec{s}), \tag{3}$$

where $\vec{p}$ is a vector of parameters to characterize a given problem situation, and $\vec{s}$ is a vector of parameters to represent an appropriate solution for this problem.

Taking into account the hard and software features of MD which are included in MARS (see the domain model in Fig. 5), vector $\vec{p}$ can be given as:

$$\vec{p} = (CPU, RAM, BAT, RES, FPS), \tag{4}$$

where CPU is a current level of a processor loading (in %), RAM is a current level of RAM usage, BAT is a current level of battery charging; RES is a number of possible screen resolution modes in MD, and FPS is a measure of a screen refresh rate.

Further, a vector $\vec{s}$ in formula (2) can be represented as the tuple

$$\vec{s} = (Width, Hight), \tag{5}$$

where *Width* and *Height* are respectively a width and a height of a video frame size on MD.

In [28] is mentioned that a performance of a MARS client application depends on its screen resolution, and accordingly to this reason a number of frame per second (FPS) can be used as one of the metrics from their set *(Metrics)* defined in formula (1). This factor also depends on some parameters: on MD processor performance, on size of its RAM, and on screen resolution of its video camera.

Therefore, a collection of metrics *(Metrics)* in (1) has the following definition

$$Metrics = (T, R), \tag{6}$$

where T is a number of FPS, and R is a total MD's productivity.

A value of metric T can be calculated using the standard function Count(), namely:

$$T = Count(FPS), \tag{7}$$

A value of metric R (named below as a load index) defines a complex characteristic of MD productivity, which is a dimensionless parameter and it can be defined as following

$$R = w_c \frac{CPU_{cur}}{CPU_{total}} + w_r \frac{RAM_{cur}}{RAM_{total}} + w_b \frac{BAT_{cur}}{BAT_{total}}, \tag{8}$$

where CPU_{cur} is a current MD processor loading ratio (in %); RAM_{cur} is a current RAM utilization (in Kb); $BATcur$ is a current battery utilization (in Ah) CPU_{total}, RAM_{total}, BAT_{total} are respectively the nominal values of these parameters; w_c, w_r, w_b are some weighting coefficients for these parameters, and the following condition must be fulfilled $w_c + w_r + w_b = 1$. In this work based on some empirical reasons, we have defined the following value ranges for the index R: it is critical if $R \geq 0.95$; it is high if $0.6 \leq R < 0.95$; it is normal if $0.25 \leq R < 0.6$; and it is low if $0 \leq R \leq 0.25$.

The metrics defined in formula (6)–(8) allow us to estimate the computational resources of an appropriate MD that is used in a target MARS with respect to our final goal: to provide adaptive recourses management in this ARS.

5.3 Complexity Metrics for Adaptive Load Balancing in MARS

As already mentioned above, MARS require more hard- and software resources than other mobile applications, and one of the possible solutions of this problem is an execution of complex business logic on the server side, where computational capabilities are higher than on MARS clients [28]. We propose the approach based on the estimation of computational complexity and analysis of MARS state in run-time. This approach allows to provide an adaptive computation balancing (ACB) in MARS, namely to decide which part of business logic should be executed on the mobile client side, and which one on the server side, depending on CPU (Central Processing Unit) - performance of a MD. In order to provide ACB the collection of metrics to estimate calculation complexity in MARS has been constructed [28].

With respect to the definitions in formula (1), the InfoBase component for ACB can be defined in the following way

$$c = (\bar{a}, \bar{c}),$$

where $\bar{a}$ is precision of calculation; $\bar{c}$ is calculation complexity.

Definition 1. A metric of calculation precision is a parameters vector

$$\overline{a} = (f,\ r,\ s), \tag{9}$$

where f is a number of decimal places after comma; r is an image resolution (in pixels); s is a ratio of image compression (possible values are from 0 to 1).

These values have to be taken into account in the procedure of computation complexity estimation. They implicitly describe possible amount of data for business logic processing, and subsequently affect amount of operations and calculation time on MD.

Definition 2. A metric of calculation complexity is a parameters vector

$$\overline{c} = \left(c_0,\ c_t\right), \tag{10}$$

where c_0 is an estimated amount of operations; c_t is an estimated calculation time (in seconds).

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

$$P = \frac{D_P \cdot 10^6}{\frac{c_0}{c_t}}, \tag{11}$$

where D_p is the estimated MD performance (in MIPS). It is to notice that because D_p is measured in MIPS, this value should be multiplied by 10^6 to transform its value to c_0 measurement.

Thus, the formula (11) defines the coefficient of the MD computation loading as a ratio of device performance to required amounts of operations in seconds. Taking into account this estimation value, we can conclude that a calculation on MD is possible if and only if $P \geq 1$, otherwise it is necessary to execute the appropriate business logic in MARS on its server side. To illustrate the proposed approach, the following example can be used [28]: let CPU-time's value is $D_p = 0.00961$ *MIPS*, the precision of calculations is $\overline{a} = \left(3,\ 96000,\ 0.6\right)$, and the estimated complexity of algorithm is $\overline{c} = \left(3241,\ 0.4\right)$. In this case, according to the Eq. (3), value of the coefficient of a MD computation loading is $P = 1.186$, and this calculation should be executed on the MD client side. These issues are discussed in more details in [28].

5.4 An Approach to Adaptive Memory Cache Management

In modern complex client-server software applications, the problem of data access time becomes more and more actual, and one of the possible ways to solve this problem is to use some kind of memory cache component [33]. On the other hand, apart from increasing of data access speed, such component stores most recently used data in random access memory (RAM) that leads also to increasing of a memory usage. Thus, we are facing with the task to choose the most effective memory cache size in order to provide an appropriate data access time with respect to effective usage of RAM.

In our approach to adaptive resource management in MARS we propose a generic vision for adaptive memory cache modeling, and Fig. 6 illustrates this model graphically. To analyze this model, the following parameters, have to be taken into account: free memory rate (M), and amount of unique queries to a database influence on cache capacity (Cs).

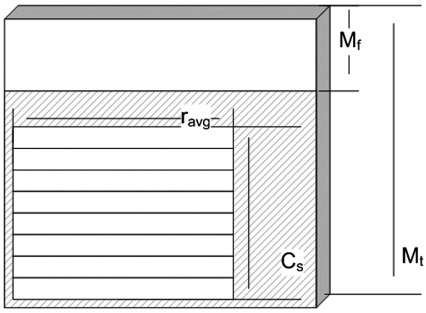

Fig. 6. Graphical interpretation for the approach to Adaptive Memory Cache modeling

We proposed to use CBR method in Adaptive Memory Cache (ACM) as an adaptive management mechanism of, this method allows us to resolve effective cache capacity problem by reusing already existing solution, so it will significantly reduce time for solution finding.

As mentioned before (see Sect. 5.2) a precedent could be defined in the following way:

$$c = (\vec{p}, \vec{s}), \tag{12}$$

where $\vec{p}$ is a vector of parameters to characterize a problem situation, and $\vec{s}$ is a vector of parameters to describe an appropriate solution for problem situation.

In scope of our research we could build $\vec{p}$ and $\vec{s}$ vectors with our cache-specific parameters:

$$\vec{p} = \{ R, M \}, \tag{13}$$

$$\vec{s} = \{ C_s \}, \tag{14}$$

where: R is an amount of unique queries (number of items); M is a free RAM rate and Cs is cache capacity.

But on the other hand, it is necessary to elaborate a way to predict cache capacity with respect to MD state. The most significant parameter for this method is free memory rate M, which is calculated by the following expression:

$$M = \frac{M_f}{M_t}, \tag{15}$$

where M_f – free RAM capacity (MB); M_t – total RAM capacity (MB).

Subsequently, adaptive cache capacity can be calculated with expression below:

$$C_s = R \cdot r_{avg} \cdot M, \tag{16}$$

where R is an amount of unique queries (number of items), r_{avg} is an average size of data record stored in cache (in bytes); M is a free RAM rate.

Expression 16 gives us a possibility to deal with situation when there is no appropriate precedent in database and we need to create a new one.

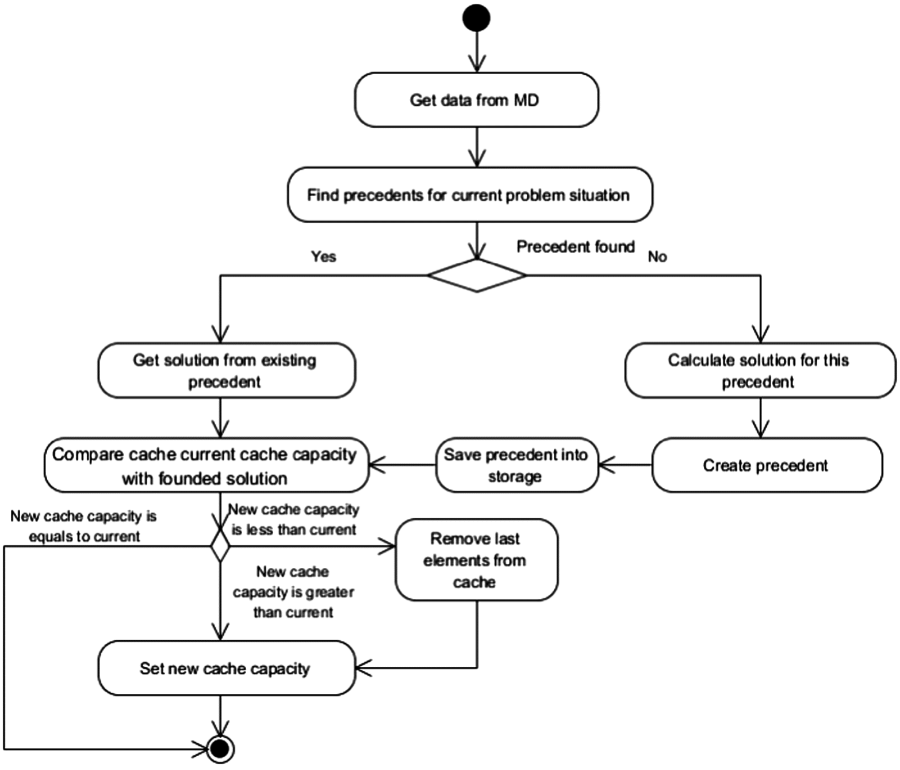

Fig. 7. Algorithm to support AMC management

Basing on CBR method the algorithm for adaptive memory cache management is constructed (see Fig. 7). This algorithm considers the case, when for current problem situation there is no appropriate precedent and a new precedent should be generated, additionally covers actions to perform in a case of increasing or decreasing cache capacity.

The proposed approach for adaptive cache management is quite generic and does not cover all aspects of cache modeling and implementation, but in our future research we will improve this model with more precise components based on modern approaches to cache modeling and construction (see e.g. in [34, 35]).

6 Feasibility Study for Adaptive Screen Resolution Facility: Software Prototype and Experimental Results

6.1 Software Prototype Design and Implementation

In order to depict a feasibility study for the proposed approach the MARS prototype with integrated ACU was developed, but currently with ASR facility only. The core functionality of this prototype is to recognize a marker on some cinema poster in the

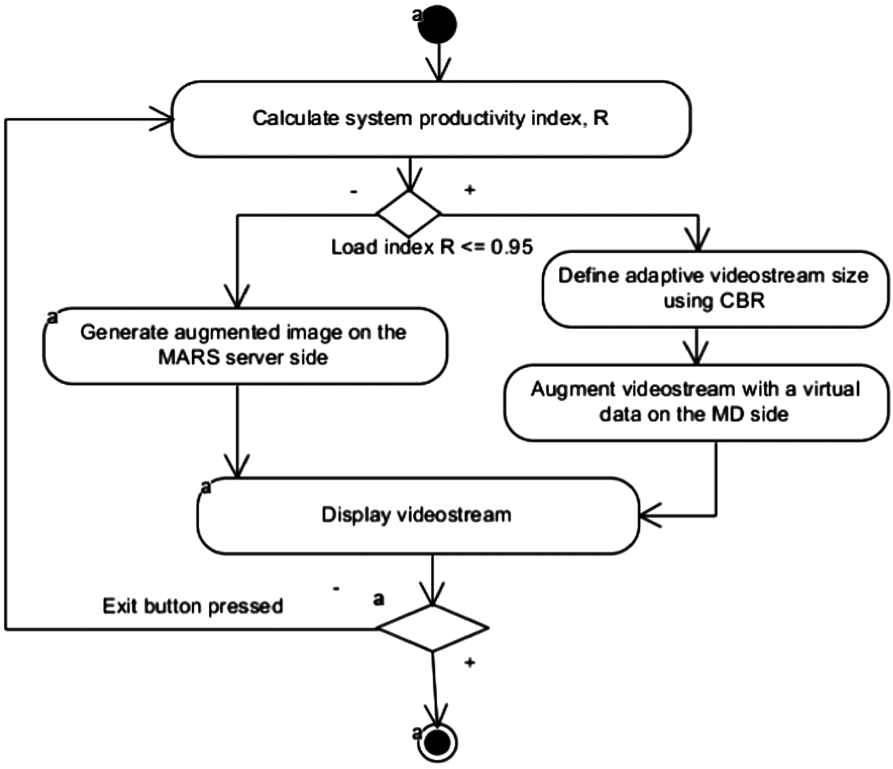

Fig. 8. Algorithm to provide ASR facility in MARS

input video stream, to search some additional data about this cinema in the appropriate database, and to augment recognized cinema with this additional information in real-time mode. Thus, such MARS prototype with integrated ACU is able to analyze environmental parameters and to adapt screen frame size taking into account the MD current state that is given by its load index (see formula (8)). In Fig. 8 the algorithm to provide ASR facility in MARS is presented in form of UML activity diagram [36].

The first step in this algorithm is to calculate the current value of MD's load index. If this value is less than 0.95 ($R \leq 0.95$), it is possible to perform data augmentation on the MD side, so the next steps are to define an adaptive video stream size, using the appropriate CBR method, and to augment the source video stream on MD side, and finally to display the augmented video stream to user. If index $R > 0.95$ it is not possible to augment data on the MD side, and in this case it is necessary to use external resources to augment image from input video stream (e.g. to perform data augmentation on a

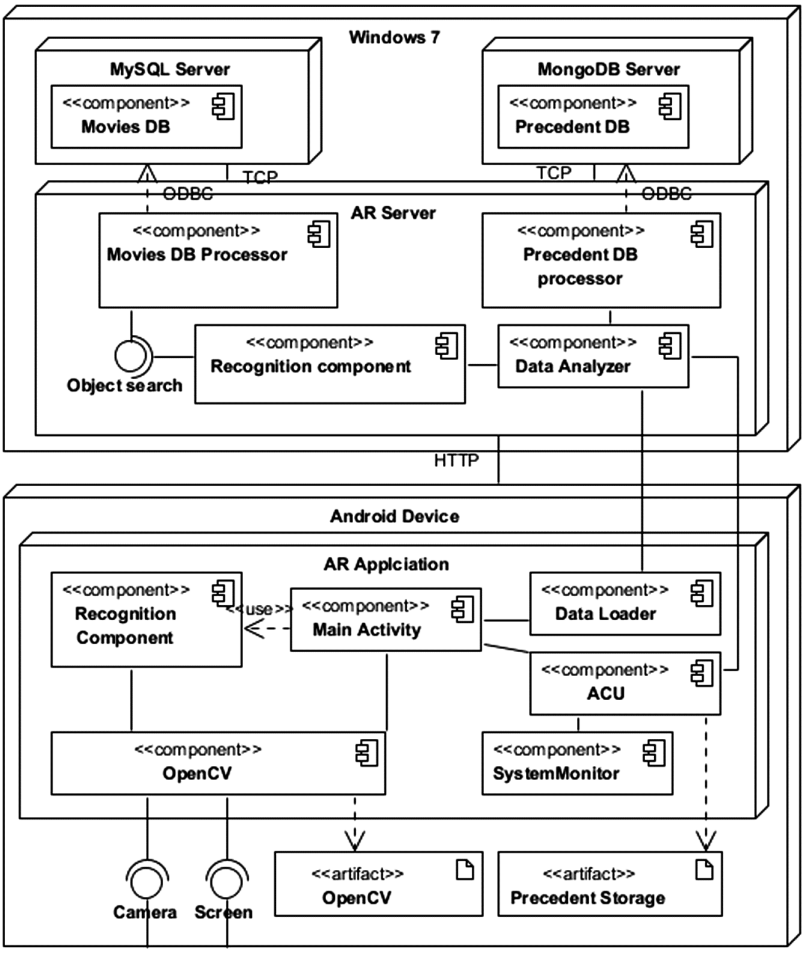

Fig. 9. 3-tier adaptive MARS software architecture

MARS server) and to show the result to a user. The algorithm workflow in both cases can be interrupted by user (in case of Exit button pressed).

To implement the elaborated algorithm, the 3-tier software architecture of MARS was chosen that is presented in Fig. 9 as UML component deployment diagram [36]. This architecture provides some crucial advantages, e.g.: high scalability, data processing security, and lower resource requirements for clients MD.

The client components were implemented with Android platform [37], using embedded Berkeley DB engine [38], and OpenCV library [39]. To develop the server-side components PHP programming language [40], Apache Web-server [41], MySQL [42], and DBMS MongoDB [43] have been chosen.

With respect to the proposed architectural solution, 3 databases have been implemented (see Fig. 9): (1) the local precedents DB (Precedent Storage), this DB is used by the control block ACU; (2) the movies DB (Movies DB), this one stores information about movies, which are processed in MARS prototype; (3) the remote centralized DB (Precedent DB) to store all given precedents, and all local DBs have to be synchronized with this centralized DB. The conceptual data model of the Precedent DB is presented in Fig. 10 as the UML class diagram [36]. This data model takes into account the following entities: Device, Precedent, Param, ListPrecedent, Platform, TypeParameter, to handle all data required by CBR method used in our approach.

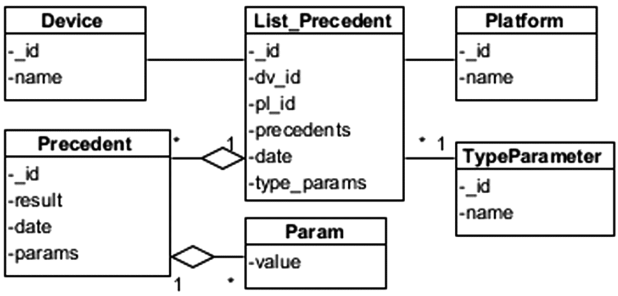

Fig. 10. Conceptual data model of the Precedent DB

To develop this database, we have selected non-relational (No-SQL) database management system MongoDB [43]. This DBMS provides high-speed data processing and stores the data in object-oriented form.

6.2 Test Results and Their Analysis

In order to estimate efficiency of the implemented MARS prototype the experiments have been performed using the following scheme: (1) select mobile devices for experiments; (2) generate precedents DB; (3) run MARS prototype with different operating modes for ACU (with enabled and disabled ACU).

To test MARS prototype two types of MD were selected, the detailed characteristics of these devices are presented in Table 1.

Table 1. Technical characteristics of testing MD

Device name	Processor	RAM	Maximal resolution
Nexus 7	Qualcomm APQ8064 (1.5 GHz)	2 GB RAM	1920 × 1080
Fly IQ4416	MT6572 (1.3 GHz)	512 MB RAM	800 × 600

In Fig. 11 the example of tuples, which describes particular precedents included in the precedents DB, is presented

№	Result	CPU	RAM	BATTERY	RESOLUTION	FPS
1	640x480	0.7	0.7	0.4	800x600	13
2	800x600	0.2	0.5	0.9	640x480	17
3	1024x768	0.3	0.8	0.5	1920x1080	2

Fig. 11. Example of precedents tuple in precedents DB

Two series of experiments have been performed with these mobile devices. The first experiment has been provided with disabled ACU (i.e. without adaptation mode), and the second one with enabled ACU. During these experiments, the image resolution of MD screen in case of different values of productivity index (see Eq. 8) had been measured. In these experiments we took into account 2 intervals from normal and high ranges of the load index: $0.4 \leq R < 0.6$ and $0.6 < R \leq 0.8$. The results of these experiments are presented in Tables 2 and 3 respectively.

Table 2. Experimental results in case of disabled ACU

Mobile device	ёResolution	T for $0.4 \leq R < 0.6$	T for $0.6 < R \leq 0.8$
Nexus 7	640 × 480	15	13
Nexus 7	800 × 600	14	10
Fly IQ4416	800 × 600	10	9
Fly IQ4416	480 × 320	15	14

Table 3. Experimental results in case of enabled ACU

Mobile device	Resolution	T $0.4 \leq R < 0.6$	Resolution	T $0.6 < R \leq 0.8$
Nexus 7	1024 × 768	11	800 × 600	13
Nexus 7	800 × 600	14	640 × 480	16
Fly IQ4416	640 × 480	13	640 × 480	14
Fly IQ4416	800 × 600	9	640 × 480	13

The data from Table 2 show that in case of the fixed image resolution on MD, and if the value of load index R is increased from the values range $0.4 \leq R < 0.6$ to the range $0.6 < R \leq 0.8$, then the T metric (see Eq. (7)) value is decreased, namely, these values are placed in interval {9, 14}; this interval covers values of T metric for both devices: Nexus 7 and Fly IQ4416. In other words, the maximum value's difference T is about

35.7% apart from difference 33,3% for case of {10, 15} and $0.4 \leq R < 0.6$. The reason of such trend in this experiment is the disabled mode of ACU.

Table 3 represents experiment results with the enabled ACU. ACU component monitors computational load on a mobile device and in case of its increasing correct image resolution; such correction leads to stabilization of T metric: this metric changes only in 18.75%. Additionally, in this experiment value of the maximal difference for T in case of $0.4 \leq R < 0.6$ is 37.5%. That is why we can make conclusion that the proposed adaptive resource management approach enables better video stream quality for mobile device in MARS.

7 Conclusions and Future Work

In this paper we have presented the model-based architecting approach to adaptive resource management in mobile augmented reality systems (MARS), which is based on the 3-level data modeling and analyzing of their specific hard- and software features. At the conceptual modeling level, the appropriate ontology-based domain mod-el of all MARS resources was elaborated, at the logical modeling level a collection of algorithmic models and quantitative metrics are proposed, and as a physical model to provide an adaptive resource management in MARS the 3-level reference software architecture is designed and implemented. This approach was successfully applied in order to solve the task to adaptive management of screen image resolution on mobile device according to changes of its computational loading that finally enabled better video stream quality in MARS.

In future we are going to extend a collection of decision search methods in order to improve an adaptation process and compare its accuracy with CBR implementation. It also is supposed to develop a more sophisticated domain model for adaptive MARS with wider amount of system features that should enable a more configure options in the proposed approach. Besides that, using this ontological model we could apply some domain specific modeling methods and appropriate CASE - tools in order to improve finally such important software quality attribute as code reusability (see e.g. in [44]). From the technological point of view, our next step to be done is to apply our architecting for adaptive resource management in cloud-based mobile augmented reality systems.

References

1. Lopez, H., Navarro, A., Relano, J.: An analysis of augmented reality systems. In: Proceedings of the 2010 Fifth International Multi-conference on Computing in the Global Information Technology (ICCGI 2010), pp. 245–250 (2010)
2. Joao, W.C., et al.: Software cybernetics. In: Encyclopedia of Computer Science and Engineering. Wiley (2008)
3. Furth, B.: Handbook of Augmented Reality. Springer, New York (2011)
4. Official U.S. Government information about the Global Positioning System (GPS) and related topics. http://www.gps.gov
5. Official site of CraftAR Service: The Ultimate AR Toolbox. https://catchoom.com/product/craftar/augmented-reality-and-image-recognition/

6. Tkachuk, M., Vekshin, V., Gamzayev, R.: A model-based framework for adaptive resource management in mobile augmented reality system. In: Proceedings of the ICTERI-2016: 12th International Conference on ICT in Education, Research and Industrial Applications: Integration, Harmonization and Knowledge Transfer, Kyiv, Ukraine, 21–24 June, vol. 1614, pp. 41–56. CEUR-WS.org (2016)
7. Official Web-site of Metaio Mobile SDK. http://www.metaio.com/software/mobile-sdk/
8. Official Web-site of D'Fusion Mobile project. http://www.t-immersion.com/products/dfusion-suite/dfusion-mobile
9. Official Web-site of Qualcomm AR SDK. http://www.qualcomm.com/solutions/augmented-reality
10. Kell, S.: A survey of practical software adaptation techniques. J. Univ. Comput. Sci. **14**, 2110–2157 (2008)
11. Kakousis, K., Paspallis, N., Papadopoulos, G.A.: A survey of software adaptation in mobile and ubiquitous computing. J. Enterp. Inform. Syst. **4**, 355–389 (2010)
12. Huang, Z., Li, W., Hui, P., Peylo, C.: CloudRidAR: a cloud-based architecture for mobile augmented reality. In: Proceedings of MARS 2014, pp. 29–34 (2014)
13. Zhang, X., et al.: Towards an elastic application model for augmenting the computing capabilities of mobile devices with cloud computing. J. Mobile Networks Appl. **16**(3), 270–284 (2012)
14. Ionescu, A.: Resource management in mobile cloud computing. Informatica Economica **19**, 55–66 (2015)
15. Alyfantis, G.: Resource management in mobile communication systems and distributed computer systems. Ph.D. thesis, Department of Informatrics and Telecommunication, University of Athens (2012)
16. Sommervile, I.: Software Engineering. Addison Wesley, Boston (2011)
17. Rafiliu, S.: Stability of adaptive distributed real-time systems with dynamic resource management. Ph.D. thesis, Department of Computer and Information Science, Linkoeping Univeristy, Sweden (2013)
18. Ghanbari, H.: Model-based dynamic resource management for service oriented clouds. Ph.D. thesis, York University Toronto, Ontario (2014)
19. Sun, Y., White, J., Eade, S.: A model-based system to automate cloud resource allocation and optimization. In: Dingel, J., Schulte, W., Ramos, I., Abrahão, S., Insfran, E. (eds.) MODELS 2014. LNCS, vol. 8767, pp. 18–34. Springer, Cham (2014). doi: 10.1007/978-3-319-11653-2_2
20. Batini, C., Ceri, S., Navathe, S.: Conceptual Database Design: An Entity-Relationship Approach. Benjamin Publishing Company, Redwood City (1992)
21. Fensel, D., Kerrigan, M., Zaremba, M.: Implementing Semantic Web Services: The SESA Framework. Springer, Heidelberg (2008)
22. Tenório, T., Dermeval, D., Bittencourt, I.: On the use of ontology for dynamic reconfiguring software product line products. In: ICSEA 2014, Proceedings of the 9th International Conference on Software Engineering Advances, pp. 545–550 (2014)
23. Hervas, R., et al.: Achieving adaptive augmented reality through ontological context-awareness applied to AAL scenarios. J. Univ. Comput. Sci. **19**(9), 1334–1349 (2013)
24. Chuan-Jun, S., Tzu-Ning, Y., Yu-Ming, Y.: Ontology-based mobile augmented reality for personalized U-Campus. In: Proceedings APIEMS 2012, pp. 2037–2046 (2012)
25. Hatala, M., Wakkary, R.: Ontology-based user modeling in an augmented audio reality system for museums. J. User Model. User-Adap. Inter. **5**(3–4), 339–380 (2005)
26. Bārzdiņš, J., et al.: OWLGrEd: a UML Style Graphical Notation and Editor for OWL 2. In: Proceedings of 7th International Workshop "OWL: Experience and Directions" (2010)

27. Ramesh, K., Karunanidhi, P.: Literature survey on algorithmic and non-algorithmic models for software development effort estimation. Int. J. Eng. Comput. Sci. **2**(3), 623–632 (2013)
28. Vekshyn, O., Tkachuk, M.: Algorithmic software adaptation approach in mobile augmented reality systems. In: Proceedings of 7-th International Conference on Software Engineering Advances, Lisbon, Portugal, pp. 40–43 (2012)
29. Aliev, A.: Soft Computing and its Applications. World Scientific (2001)
30. Maiden, N., Sutcliffe, A.: Case-based reasoning in software engineering. In: IEE Colloquium on Case-Based Reasoning, Digest No. 036, pp. 1–3 (1993)
31. Limthanmaphon, B., Zhang, Y.: Web service composition with case-based reasoning. In: Proceedings of 14th Australian Database Conference (ADC 2003), vol. 17 (2004)
32. Tkachuk, M., Polkovnikov, S., Bronin, S.: Adaptive control framework for software components: case-based reasoning approach. In: Proceedings of 6th International Workshop on Software Cybernetics (IWSC–2009), Seattle, USA, pp. 47–56 (2009)
33. Hankins, R., Patel, J.: Data morphing: an adaptive, cache- conscious storage technique. In: Proceeding of the 29th VLDB Conference, Berlin, Germany, pp. 417–428 (2003)
34. Press, A., et al.: Caching algorithms and rational models of memory. In: Proceedings of the 36th Annual Conference of the Cognitive Science Society, Quebec City, Canada (2014)
35. Bender, M., et al.: Cache-adaptive analysis. In: Proceeding of the 28th ACM Symposium on Parallelism in Algorithms and Architectures, Pacific Grove, California, USA, pp. 135–144 (2016)
36. Official Web-site of OMG Consortium. http://www.uml.org/
37. Official Web-site of Android platform. https://www.android.com/
38. Official Web-site of Oracle Berkeley DB. http://www.oracle.com/technetwork/database/database-technologies/berkeleydb
39. Official Web-site of OpenCV project. http://www.opencv.org
40. Official Web-site of PHP language. http://www.php.net
41. Official Web-site of Apache project. http://www.apache.org
42. Official Web-site of MySQL project. http://www.mysql.com
43. Official Web-site of MongoDB. http://www.mongodb.org
44. Tkachuk, M., et al.: An integrated approach to evaluation of domain modeling methods and tools for improvement of code reusability in software development. In: Heinrich, C., Mayr, Pinzger, V. (Hrsg.) INFORMATIK 2016, Lecture Notes in Informatics (LNI), vol. P-259, pp. 143–156. Kollen Druck+Verlag GmbH, Bonn (2016)