CARL: A Language for Modelling Contextual Augmented Reality Environments

Dariusz Rumiński, Krzysztof Walczak

CARL: A Language for Modelling Contextual Augmented Reality Environments

Dariusz Rumiński, Krzysztof Walczak

Poznań University of Economics
Niepodległości 10, 61-875 Poznań, Poland
{ruminski, walczak}@kti.ue.poznan.pl

Abstract. The paper describes a novel, declarative language that enables modelling ubiquitous, contextual and interactive augmented reality environments. The language, called CARL – Contextual Augmented Reality Language, is highly componentised with regards to both the structure of AR scenes as well as the presented AR content. This enables dynamic composition of CARL presentations based on various data sources and depending on the context. CARL separates specification of three categories of entities constituting an AR environment – trackable markers, content objects and interfaces, which makes the language more flexible and particularly well suited to building collective awareness systems based on ubiquitous AR-based information visualisation.

Keywords: Augmented reality, AR, AR services, contextual services, CARL

1 Introduction

Augmented reality (AR) technology enables superimposing rich computer generated content, such as interactive 3D animated objects and multimedia objects, in the real time on a view of a real environment. Augmented reality enables building advanced localised information visualisation systems and therefore forms a solid basis for the development of collective awareness systems.

Widespread use of the AR technology has been enabled in the recent years by remarkable progress in consumer-level hardware performance, in particular, in the computational and graphical performance of mobile devices and quickly growing mobile network bandwidth. Education, entertainment, e-commerce and tourism are notable examples of application domains in which AR-based systems become increasingly used.

There are a variety of tools available for authoring AR content and applications. These tools range from general purpose computer vision and graphics libraries, requiring advanced programming skills to develop applications, to easy-to-use point-and-click packages for mobile devices, enabling creation of simple AR presentations. However, the existing tools are designed for manual authoring of AR presentations – either through programming or visual design. To enable more widespread use of AR technology for visualisation of different kinds of up-to-date data, needed in collective

awareness systems, a different paradigm is required. The interactive presentations must be created automatically based on the available data sources, through selection of data and automatic composition of AR scenes. A key element enabling selection of information for visualisation is context, which incorporates aspects such as user location, time, privileges, preferences, device capabilities, environmental conditions and others. To enable meaningful contextual selection and automatic composition of non-trivial interactive AR interfaces, semantic web technologies may be used.

In this paper, we present foundations of a novel high-level programming language, called *CARL – Contextual Augmented Reality Language*, which enables building ubiquitous, contextual and interactive AR presentations. In the presented approach, the AR content that is presented to users is created by selecting and merging content and rules originating from different service providers, without the necessity to explicitly switch from one to another. Moreover, in CARL, the rules for tracked markers (where the information will appear), content objects (what will be presented) and interfaces (how the information will be accessible) are separated, enabling flexible composition of presentations meeting real business requirements.

The reminder of this paper is organised as follows. Section 2 presents related works in the context of building AR environments, in particular focusing on the use of AR for building collective awareness systems. Later, in Section 3, the concept of building ubiquitous contextual AR presentations is explained. In Section 4, the CARL language is presented. In Section 5, the current implementation of CARL is outlined. Finally, in the last section, conclusions and directions for future research are presented.

2 AR Environments and Collective Awareness Systems

A number of research works have been devoted to the problem of modelling and building AR environments combining the view of real-world objects with multimedia content. One of well-known toolkits that supports rapid design and implementation of AR applications is DART – Designer's Augmented Reality Toolkit [1]. Many AR systems are based on the ARToolKit library, which however requires advanced programming skills and technical knowledge to create AR applications [2].

GUI-based authoring applications that enable mixing virtual scenes with the real world have been presented in [3-8]. These systems provide user-friendly functionality for placing virtual objects in mixed reality scenes without coding and can be used for building AR environments, but they target only desktop computer environments and do not provide functionality to build open, interoperable and dynamic AR systems.

Since mobile devices have gained popularity, a number of mobile AR toolkits and applications have been developed. For example, ComposAR-Mobile is a cross-platform content authoring tool for mobile phones and PCs based on an XML scene description [9]. Lee and Seo have presented the U-CAFÉ framework, which can be used to build ubiquitous AR environments [10]. It supports contextual presentation and collaboration between users on various devices. However, it does not provide means for combining content from different services and modelling user interactions within the AR presentations.

Stiktu is a mobile AR authoring application that enables attaching ("sticking") simple content, such as text messages and images, to particular views of real places [11]. It enables users to express and exchange opinions with other users who have created virtual objects. The main limitation of this application is the lack of a possibility of creating complex objects, such as interactive 3D models.

Aurasma is an application that enables augmenting arbitrary real-world views using a mobile device's camera and then sharing the created AR content with other users [12]. Layar, Junaio and Wikitude are other examples of AR applications that enable users to share AR content [13-15]. These platforms are good examples of collective awareness systems, where users contribute to "augmenting" the real world with virtual objects. Growing popularity of such applications indicates that there is public interest in new forms of information visualisation, even if the functionality of these systems is still highly limited.

Several studies have been performed in the domain of declarative languages for building virtual environments. InTml is an XML-based language for describing 3D interaction techniques and hardware platforms in VR applications [16]. MPML-VR enables describing multimodal presentations in 3D virtual spaces [17]. VR-BML is a high-level XML-based behaviour programming language developed as a part of the Beh-VR approach [18]. These languages, however, have been designed for 3D/VR applications without taking into account AR-specific functionality.

A number of high-level languages have been also designed specifically for building AR applications. APRIL is an XML-based scripting language for authoring AR presentations within the Studierstube framework [19]. SSIML/AR is a language for structured specification and development of AR applications [20]. AREL is a JavaScript binding of the Metaio/Junaio SDK API allowing development of AR applications based on web technologies, such as HTML5, XML and JavaScript [21]. MR-ISL is a high-level language for defining mixed reality interaction scenarios [22].

A common motivation for developing novel platforms and declarative AR languages is to simplify the process of programming AR applications. However, the existing languages are not intended for building dynamic contextual AR information visualisation systems. In such systems, different types of data must be retrieved in the real time from distributed sources and automatically composed into interactive AR scenes.

3 Contextual Augmented Reality Environments

Existing AR platforms support two main forms of augmentation. In the first form of augmentation – *directional augmentation* – information is associated with points of interest in physical locations in the real world and is presented on mobile devices as directions on how to navigate to these locations (e.g., hotels and restaurants in Wikitude or Junaio). Coordinates of a mobile device are monitored through GPS or other sensors. This form of augmentation is usually functionally and graphically very simple – it rarely goes beyond displaying simple graphical objects with textual descriptions and the content is not geometrically registered in the real environment.

The second form of augmentation – *natural augmentation* – is more powerful. Position and orientation of content to show is identified directly through a device's camera by tracking known markers (patterns) in the captured images. This kind of augmentation can be based on complex interactive 3D content, which can additionally incorporate other multimedia objects, such as images, sounds and videos. The presented content can be geometrically integrated with particular objects in the real environment extending them with presentation of additional virtual objects or information. However, a common problem with the development of systems based on natural augmentation is their limited scope with regards to the number of markers tracked and – consequently – content objects presented. Systems of this type can be successfully used as interactive extensions to printed advertisements or AR additions to video games, but their wider applicability, in particular for building information visualisation and collective awareness systems, is largely limited.

As a solution to this problem, we propose the notion of *CARE – Contextual Augmented Reality Environment* (Fig. 1). CARE is a new approach to building AR applications, which combines the advantages of both directional and natural augmentation systems. In CARE, the AR scenes that are presented to users are based on natural augmentation, but they are dynamically composed in the real time based on a number of data sources and the current context, which includes such elements as location, time, user privileges and preferences, device capabilities and environmental conditions. Contextual approach to AR application development is a necessity to enable access to a variety of data sources, guarantee scalability and provide for seamless operation. In CARE, the same AR application can be used to display elements coming from different content- and service-providers. Moreover, due to the use of semantic linking, content coming from different sources can be geometrically and functionally combined into integrated AR scenes.

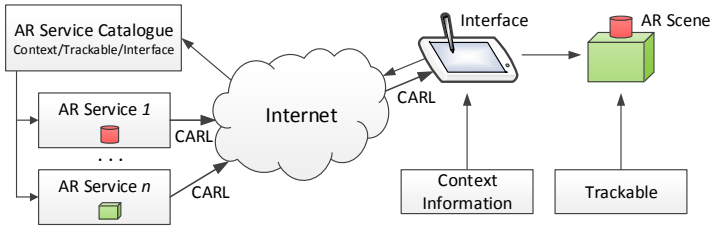

Fig. 1. Architecture of the CARE environment

To enable implementation of CARE environments, a language is required that would enable passing augmentation information from distributed service providers to the AR browser. Such a language must be designed to support dynamic composition of complex interactive AR scenes, and must enable passing information about different entities: *trackable markers*, *content objects* and *interfaces*. In practical applications, these aspects may be separated. For example, trackable markers may come from municipal services, content objects from AR service providers, while interfaces from application developers. Only in such heterogeneous environment, practical ubiquitous AR environments may be realised.

In the next section, we present a language, called CARL – Contextual Augmented Reality Language, designed to meet these criteria.

4 The CARL Language

CARL enables specification of three types of entities that form an AR presentation: *Trackables*, *Content Objects* and *Interfaces*. The independence of trackables, content objects and interfaces in connection with the dynamism of each of these elements requires the use of loose coupling between descriptions of these entities. The problem can be solved by the use of semantic modelling and semantic linking, which is however out of the scope of this paper. Below, the main elements of the CARL language are presented.

4.1 Trackables

The *Trackables* element, shown in Listing 1, groups *Trackable* elements that are used for describing all real-world objects that an application can detect and track in 3D space. Every *Trackable* element has its unique *URI*, which identifies associated binary resource data used in the tracking process. The *Trackable* element has three child elements: *Begin*, *Active* and *End*.

```
<Trackables>
   <Trackable uri="…">
      <Begin>
         <ContentObjectBegin id="1"/>
         <ContentObjectBegin id="2"/>
         <InterfaceMessage value="…"/>
      </Begin>
      <Active>
         <DistanceChanged change="0.1" distanceMin="2" distanceMax="5"
angleMin="30" …>
            <ContentObjectAction id="1" action="show"/>
            <ContentObjectAction id="1" action="playAnimation"/>
         </DistanceChanged>
      </Active>
      <End>
         <ContentObjectEnd id="1"/> <ContentObjectEnd id="2"/>
      </End>
   </Trackable>
<Trackable uri="…" />…
```

Listing 1. Specification of trackables in CARL

In the runtime, when an application detects a trackable object, the *Begin* section is executed. In the presented example, it initializes content objects using the *ContentObjectBegin* command and displays a message using the *InterfaceMessage* command. The *Active* section contains rules of user interaction when the trackable is visible. In the example, the *DistanceChanged* event is detected and appropriate action is performed. The *End* section is executed when the trackable object disappears from the camera view.

4.2 Content Objects

The second element of AR environment specification in CARL are *ContentObjects*, containing multimedia content to be shown to end-users while augmenting real-world objects (Listing 2a). Each *ContentObject* has an *id*, which is used while calling object actions. The *id* can have the form of a fixed literal or a semantic expression to enable loose coupling with other elements of the dynamically created AR scenes.

Content objects support states. The initial object state is specified by the *initialState* attribute. The state can be changed by the use of the *ContentObjectState* command in actions. The *Resources* element contains information about particular *Components* and *Locations* of component resources. Several locations can provide alternative URIs for components meeting particular criteria, e.g., regarding the model complexity or language.

The *Actions* element specifies actions that can be called on this content object. An action is declared within the *Action* element, which has a mandatory *name* attribute and an optional *state* attribute. If the *state* is specified, the action (if called) will be executed only when the object is in the given state. Within the *Action* element, commands based on the VR-BML [18] language are used. These commands can be used to load, manipulate and animate object components, change object state, execute interface operations and control other objects.

```
<ContentObjects>
 <ContentObject id="1" initialState="hidden">
  <Resources>
   <Component id="c1">
    <Location details="low" uri="..." />
    <Location details="high" uri="..." />
   </Component>
  </Resources>
  <Begin/>
   <Actions>
    <Action name="show" state="hidden">
     <SetPosition component="c1" value="…"/>
     <Activate component="c1" active="true">
     <ContentObjectState value="shown"/>
    </Action>
   </Actions>
  <End/>
</ContentObject>
…
```

a)

```
<Interface>
 <TouchSingle target="…">
  <ContentObjectAction
action="highlight">
 </TouchSingle>
 <TouchSingle target="…">
  <ContentObjectAction id="1"
action="hide">
  <ContentObjectAction id="2"
action="show">
 </TouchSingle>
 <Pan target="…">
  <ContentObjectRotate>
 </Pan>
 <Pinch target="…">
  <ContentObjectScale>
 </Pinch>
</Interface>
```

b)

Listing 2. Specification of content objects (a) and interfaces (b) in CARL

4.3 Interfaces

The third main part of a CARL environment specification are interfaces (Listing 2b). An *Interface* element contains specification of possible interactions on an end-user device. Listing 2b shows an example of an interface specification, in which *touch*, *pan* and *pinch* gestures can trigger actions. An interface can recognize interactions with objects and interactions with trackables. Interaction with objects enables activation and manipulation of objects. Interaction with trackables can be used, e.g., to display

objects associated with the tracked markers (e.g., information about a known product).

5 Implementation

CARL has been implemented within the *Mobile Augmented Reality Authoring Tool* (*MARAT*) for virtual museum exhibitions [23]. Currently the application enables only playback of CARL presentations, but the intention is to implement also content authoring directly on a mobile device within MARAT.

The following figures show examples of CARL usage in the MARAT application. Figure 2a presents an example when an end-user moves his/her finger on a device screen. A *Pan* event triggers an action responsible for object rotation (*ContentObjectRotate* – c.f. Listing 2b). In turn, Figure 2b shows example of using a *Pinch* gesture, which triggers an action responsible for object scaling (*ContentObjectScale*).

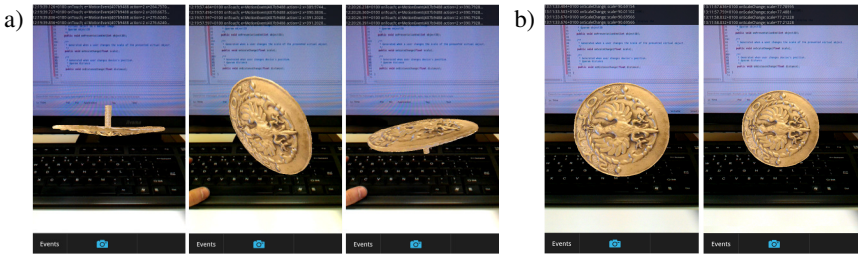

Fig. 2. Triggering the *ContentObjectRotate* action on a *Pan* event (a) and the *ContentObjectScale* action on a *Pinch* event (b)

6 Conclusions

In this paper, the overall concept of Contextual Augmented Reality Environments and fundamentals of the CARL language, designed to support creation of such environments, have been presented. CARL enables modelling ubiquitous, contextual and interactive AR environments. The language is highly componentised with regards to the structure of the AR scenes as well as the presented content, enabling dynamic composition of complex AR environments. The CARL language can support building ubiquitous AR-based data visualisation and collective awareness systems.

Future research will incorporate several facets. First, the method of semantic coupling of elements of dynamic AR environments will be elaborated. Second, techniques for combining CARL specifications from independent sources will be investigated, with particular focus on security. Finally, methods of automatic creation of CARL specifications through WYSIWYG actions on mobile devices, to enable direct on-site authoring of AR environments, will be designed.

References

1. MacIntyre, B., Gandy, M., Dow, S.: DART: A Toolkit for Rapid Design Exploration of Augmented Reality Experiences. In: Proc. of ACM Symposium on User Interface Software and Technology (UIST), pp. 197--206 (2004)
2. ARToolKit, http://www.hitl.washington.edu/artoolkit
3. Anagnostou, K., Vlamos, P.: Square AR: Using Augmented Reality for urban planning. In: Proc. of the 3rd IC on Games and Virtual Worlds for Serious Appl., pp. 128--131 (2011)
4. Lee, G. A., Kim, G. J., Billinghurst, M.: Immersive authoring: What You eXperience Is What You Get (WYXIWYG). Communications of the ACM 48(7), pp. 76--81 (2005)
5. AAA Wang, M. J., Tseng, C. H., Shen, C. Y.: An Easy to Use Augmented Reality Authoring Tool for Use in Examination Purpose. IFIP Advances in Information and Communication Technology, pp. 285--288. Springer Boston (2010)
6. Seichter H., Looser, J., Billinghurst, M.: ComposAR: An Intuitive Tool for Authoring AR Applications. In: Proc. of the 7th IEEE/ACM Int. Symp. on Mixed and Augmented Reality (ISMAR), pp. 147--148 (2008)
7. Grimm, P., Haller, M., Paelke, V., Reinhold, S., Reimann, C., Zauner, R.: AMIRE - authoring mixed reality. In: Proc. of the First IEEE International Augmented Reality Toolkit Workshop (2002)
8. Zauner, J., Haller, M.: Authoring of Mixed Reality Applications Including Multi-Marker Calibration for Mobile Devices. In: Proc. of the 10th Eurographics Symposium Virtual Environments (EGVE), pp. 87--90 (2004)
9. Wang, Y., Langlotz, T., Billinghurst, M., Bell, T.: An Authoring Tool for Mobile Phone AR Environments. In: Proc. of the New Zealand Computer Science Research Student Conference, pp. 1--4 (2009)
10. Lee, J.Y, Seo, D.W.: A Context-Aware and Augmented Reality-Supported Service Framework in Ubiquitous Environments. In: Proc. of Embedded and Ubiquitous Computing – EUC 2005 Workshops, pp. 258--267. Springer (2005)
11. Stiktu, http://stiktu.com/
12. Aurasma, http://www.aurasma.com/
13. Layar, https://www.layar.com/
14. Junaio, http://dev.metaio.com/junaio/
15. Wikitude Developer -- Devzone, http://developer.wikitude.com/
16. Figueroa, P., Green, M., Hoover, H. J. InTml: A Description Language for VR Applications. In Proc. of Web3D 2002 Symposium, pp. 53--58. ACM New York (2002)
17. Okazaki, N., Aya, S., Saeyor, S., and Ishizuka, M.: A Multimodal Presentation Markup Language MPML-VR for a 3D Virtual Space. Workshop Proc. on Virtual Conversational Characters: Applications, Methods, and Research Challenges (in conj. with HF2002 and OZCHI2002), Melbourne, Australia (2002).
18. Walczak, K.: Beh-VR: Modeling Behavior of Dynamic Virtual Reality Contents, In: Interactive Technologies and Sociotechnical Systems. In: LNCS 4270, pp. 40--51. Springer-Verlag, Heidelberg (2006)
19. Ledermann, F., Schmalstieg, D.: APRIL: a high-level framework for creating augmented reality presentations. In: Proceedings of IEEE Virtual Reality 2005, pp. 187--194 (2005)
20. Vitzthum, A., Hussmann, H.: Modeling augmented reality user interfaces with SSIML/AR. J. Multim. 1(3), pp. 13--22 (2006)
21. AREL - Augmented Reality Experience Language, http://dev.metaio.com/arel/overview/
22. Walczak, K., Wojciechowski, R.: Dynamic Creation of Interactive Mixed Reality Presentations. Proc. of the ACM Symposium on Virtual Reality Software and Technology. pp. 167--176. ACM (2005)
23. Rumiński, D., Walczak, K.: Creation of Interactive AR Content on Mobile Devices. In: Lecture Notes in Business Inf. Processing LNBIP 160, pp. 258--269. Springer (2013)