Entertainment Computing 6 (2015) 1–10

Contents lists available at ScienceDirect

Entertainment Computing

journal homepage: ees.elsevier.com/entcom

A case study on Service-Oriented Architecture for Serious Games ☆

Maira B. Carvalho [a,b,*], Francesco Bellotti [a], Riccardo Berta [a], Alessandro De Gloria [a], Giorgia Gazzarata [a], Jun Hu [b], Michael Kickmeier-Rust [c]

[a] *DITEN, University of Genoa, Via Opera Pia 11A, 16145 Genoa, Italy*
[b] *Industrial Design, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands*
[c] *Knowledge Technologies Institute, Graz University of Technology, Inffeldgasse 13/5th floor, 8010 Graz, Austria*

ARTICLE INFO

Article history:
Received 15 April 2014
Revised 21 October 2014
Accepted 12 November 2014
Available online 20 November 2014

Keywords:
Adaptive Serious Games
Games for learning
Serious Games
Serious Games development
Service Oriented Architecture
Architecture Tradeoff Analysis Method

ABSTRACT

Service-Oriented Architecture (SOA) is a set of practices for architectural design of software that exploits services as loosely coupled components orchestrated to deliver various functionalities. The SOA paradigm is not well established in the Serious Games (SG) domain, but it is expected to provide benefits, particularly in reducing the conceptual and technological complexity of the development. In this paper, we propose and study the application of a SOA approach to SG development. We have used the SOA approach to develop an adaptive Serious Game for teaching basic elements of probability to high school and entry-level university students, called *The Journey*. Details of the architecture implementation are offered, as well as the results of an evaluation of the system using the Architecture Tradeoff Analysis Method (ATAM). Based on our experience, we argue that the SOA approach can make SG development shorter, more flexible and more focused.

1. Introduction

Games are gaining increasing importance as educational and training tools. Serious Games (SGs) – as games used for purposes other than to simply entertain are often called [1] – have been shown to have a lot of potential in education [2–4], offering the possibilities of making learning more engaging and satisfying [5]. Among the benefits are their role in engaging and motivating learners [2] and their ability to expose learners to experiences that would be impossible, unsafe or at least impractical to reproduce in the real world [1,6,7].

However, there is still a long way to make SGs widely deployed, especially because of the high development costs. In this paper, we propose the application of a Service Oriented Architecture (SOA) approach to Serious Game development as a desirable and beneficial solution for the field, resulting in efficient development and high quality products. In SOA, software is built as a set of independent, loosely coupled components that provide self-contained functionalities (services) to other components and applications. By employing the core principles of SOA, such as modularization and compositionality, we expect to achieve flexibility in the development of Serious Games and to enable the reuse of software parts. The SOA principles are already widely and successfully employed in several areas of software engineering, but the examples in the SG domain are limited.

To illustrate the benefits of the SOA approach in SG development, we report a case study on the use of a web service based on the Competence-based Knowledge Space Theory (CbKST) [8] to develop an adaptive Serious Game for teaching basic elements of probability to high school and entry-level university students, called *The Journey*. The paper also presents the results of an evaluation of the system employing the Architecture Tradeoff Analysis Method (ATAM), using the conclusions of the evaluation to give an account of the constraints, benefits and changes in the programming paradigm that are relevant to SG development.

In short, the novelty presented here consists of a demonstration of how the SOA approach can improve the process of SG development through component reuse, and how it can enhance product quality by enabling the implementation of features that are still rare in SGs, such as adaptation techniques, learning analytics, social media integration, etc.

This paper is organized as follows. In the next section, we define Service Oriented Architectures and list its possible benefits and drawbacks in SG development. Section 3 presents a compilation of projects using SOA or similar approaches. In Section 4, we discuss the concept of adaptivity in learning environments and

☆ This paper has been recommended for acceptance by Letizia Jaccheri.

* Corresponding author at: Industrial Design Department, P.O. Box 513, 5600 MB EINDHOVEN, The Netherlands.

E-mail addresses: m.brandao.carvalho@tue.nl, maira.carvalho@elios.unige.it (M.B. Carvalho).

http://dx.doi.org/10.1016/j.entcom.2014.11.001
1875-9521/© 2014 Elsevier B.V. All rights reserved.

present the Competence-based Knowledge Space Theory (CbKST), explaining how it can be used in adaptive SGs. Subsequently, we describe the game *The Journey*, offering details of its architecture and implementation. Section 6 presents an ATAM evaluation of the system architecture and discusses the benefits and drawbacks of applying SOA in SG development. Finally, we present the discussion and conclusion of our work and pointers for future research.

2. Service Oriented Architectures

A Service-Oriented Architecture (SOA) is "a software architecture that implements business processes or services by using a set of loosely coupled, black-box components orchestrated to deliver a well-defined level of service" [9]. It is a set of ideas, recommendations, policies and practices for architectural design. One of its goals is to employ modularization and compositionality to achieve flexibility and to enable the reuse of software parts, in an attempt to manage the complexity of large systems [10,11].

The benefits of using a SOA approach are many. Unlike the case of traditional library reuse, which requires replication of code, SOA supports reuse of the services themselves, which provides a significant benefit in terms of having up-to-date components without concerns about maintenance of the code. In addition, it supports such a level of abstraction that multiple services can offer the same functionalities, potentially giving the developer a wider choice of providers from which to obtain the service needed. Furthermore, SOA establishes standardized contracts between endpoints, placing formal obligations between consumer and provider and largely increasing reusability and interoperability. An implementation that complies to known web service standards (e.g. REST or SOAP) has additional benefits, such as standardization, technology/platform neutrality and automatic discovery and use [10]. The automatic binding of services removes compile-time dependencies; the interface definition happens in runtime, removing the need to alter the code every time when there is a change in the service provider. This provides flexibility in the development and improves maintainability [12,13].

In the specific case of game development, a SOA approach can bring the potential benefit of decreased interdependencies and usage dependent payment models [14]. Furthermore, it facilitates dealing with scalability issues, which is particularly relevant to online games in which several thousands of players interact in a common platform, as the increased load in the servers may bring performance concerns [14]. SOA also makes it possible to access games from simple devices, eliminating the dependency on the quality of gaming hardware. In addition, providing pervasive gaming experiences becomes easier, as support for different platforms is highly simplified if the core of the gaming experience is provided via a service in a centralized server [15].

Educational applications in general – not only SGs – can also benefit from the application of SOA [16,17]. In addition to the points listed above, the most relevant advantage is the possibility to reuse educational components and domain-independent features (e.g. shared user profiles, knowledge databases on learning topics, natural language processing dialog services), which could potentially be deployed as web services that could be composed and invoked by a learning application or game when needed.

There are, nevertheless, challenges in adopting a Service-Oriented Architecture. Quality assurance and testing module integration tend to be more difficult when developing SOA applications [9]. In addition, a service can be practically unusable if its interfaces lack clarity or are badly documented. Finally, extra attention has to be given to service descriptions, as they are the way to advertise the capabilities, interfaces, behavior and quality of a service, providing the required information for discovery, selection, binding and composition with other components [18].

3. Related work

Service-based architectures are already widely and successfully employed in several areas of software engineering, including game development. There is an increasing availability of service-based tools for game development, such as cloud-based infrastructure for building, deployment and distribution [19], platforms providing social connectivity to games [20] and services that provide generic gaming features such as achievements, leaderboards and cloud saving [21,22].

Although there are clear benefits in employing service-based architectures to SG development, the examples of deployments of SOA-based SGs are limited. This is true even if there are several gaming-related services already available and a large number of (non-educational) digital games already utilizing those services.

While not necessarily SOA-based, decoupling the content of the SG from the underlying gaming software is a way of facilitating the extensibility of SGs and to support domain experts in the creation of content, which can then happen independently of the development of the game itself [23]. The project *Travel in Europe (TiE)*, for example, proposes an architecture style that supports both code reuse and consistent interaction modalities across games [24–26]. The *MetaVals Serious Game*, a game for practicing basic finance concepts, consists of a modular database and an independent graphic interface, with a management interface that facilitates configuring the game to different contexts [27].

Authoring platforms also aim to reduce the complexity of game development. The *eAdventure* game platform serves as an authoring platform for educational point-and-click adventure games, executing games defined in a specialized markup language [28,29]. The authoring tool *Puzzle-it* divides the process of developing games into content authoring and core engine development, making it possible for instructors to create content for the games via the authoring tool without needing to be concerned about engine behind the games [30].

When it comes to the actual usage of SOA in SGs, examples available are very few.

While the game itself has not been developed, a Service-Oriented Architecture was the approach of choice for an envisioned gaming platform based on mobile augmented reality, called *MARL*. In this system, on-demand location-based instruction would be delivered through a head-mount display by a virtual instructor. The complete *MARL* game service would be composed of subsystems that would provide visual, human computer interface, and training services, allowing for the lower level objects to be encapsulated by the higher level interfaces, making it easier for improvements in the algorithms to be incorporated into the service [31].

The *Rashi Intelligent Tutoring System* teaches human anatomy through a problem-based environment. *Rashi* is built as a web service architecture that supports on-demand requests for small chunks of specific knowledge, instead of requests for an entire case specification at once, giving developers flexibility to develop lightweight inquiry tutors that run efficiently over the web [32]. On top of the same existing service structure for the original (2D) inquiry system, the researchers built a 3D game in which the student is a doctor who must diagnose a patient in a virtual hospital. Despite being limited to a specific type of learning (i.e. problem-based inquiry), *Rashi* constitutes one of the very few examples of service-based architecture for game-based education, demonstrating well the benefits of the SOA approach for SG development.

The Serious Games Society has developed the *Serious Games Web Services Catalog* [33], a repository of web services with the