A Web-Based Service-Oriented Solution for a Mobile Digital Storytelling Application

Ensar Muratović*
Irfan Prazina†
Supervised by: Selma Rizvić‡

Faculty of Electrical Engineering Sarajevo
University of Sarajevo

Abstract

Today most of the users prefer mobile over desktop applications. The majority of current software solutions either do not use all of the benefits from the service-oriented architecture, or do not support mobile devices. Online stored content reduces the size of the application and makes possible to change content without changing the application. This paper presents a web based service oriented mobile solution for interactive digital storytelling with 3D models and videos. The concept is implemented on a case study of White Bastion, a medieval fortress located on hills above Sarajevo. The users can explore the models of the fortress from different time periods and play video stories by clicking on available markers. Marker data and a list of models are stored on Google Drive and accessed via Google web service APIs.

Keywords: Digital storytelling, 3D model presentation, Mobile application, Service-oriented architecture, Web-based, Google Drive, jMonkeyEngine

1 Introduction

Digital storytelling is the use of digital media platforms and interactivity for narrative purposes, either for fictional or for non-fiction stories[11]. It has applications in museums, teaching, tourism and many more. In the today's world where many cultures intersect there is a need for an efficient and effective way of presenting stories. With digital storytelling, people have an opportunity for producing and delivering stories to a large audience through online media.

Most of storytelling applications have a problem with separating the application and media. Often 3D models and markers in 3D scenes are built into the application. In this paper, we will describe a mobile application for presenting 3D models linked with stories using markers. Media used in the application is stored on the web, and the application is planned in a way that allows adding, deleting and modifying content without changing the application itself.

The application is web based and service oriented. Service oriented applications use services as fundamental elements. Web service is a way of exposing system functionality over the internet [1]. They are technology neutral and offer well-defined interfaces for clients. Through the application, a list of media content can be acquired from the web service. Web services can be used from different devices and platforms. The client in this solution is the Android mobile application. This way of separating media (data) and application allows future development of different applications which can use existing online stored content.

The paper is structured as follows: Section 2 gives an overview of current mobile storytelling applications, describes jMonkeyEngine [2] and service oriented architecture and presents the process of preparing 3D models for mobile application. Section 3 shows implementation details, Section 4 and Section 5 analyse performance of the application using 3D models with different number of triangles. In Section 6 we offer our final considerations.

Figure 1: White Bastion fortress, Sarajevo, present appearance

*emuratovic2@etf.unsa.ba
†iprazina1@etf.unsa.ba
‡srizvic@etf.unsa.ba

2 Background and Related work

There have been many approaches to the presentation of cultural heritage objects using various computer hardware and software. With the daily progression of hardware and software possibilities, there is a constant need for discovering a better and more useful solution for 3D scenes presentation problem. The importance of cultural heritage objects is evident but it is also very important to provide a more accessible way for the presentation of those objects. Some of the cultural heritage objects are partially or totally destroyed and by using advantages of today's software and hardware there are limitless possibilities to "revive them", at least in a virtual sense. The number of smartphone users is in constant growth and keeping that in mind, there is always a strong call for developing various types of applications for this market. Furthermore, advantages of the internet and service-oriented architectures allows developers to focus their applications to larger society. In the next Section, we present some of the related approaches to digital storytelling applications.

2.1 Digital Storytelling Applications

Dušan Tatić et al. [14] worked on application wihich helps visitors of various exhibitions to browse multimedia information content with mobile devices. The system consists of several independent programming modules that can be combined in different ways to create various mobile applications for presenting cultural heritage. In this way, authors ensure the universality of the guide and its applicability to different exhibition scenarios. A special module for augmented reality technology is implemented to provide a possibility for the presentation of 3D computer-generated reconstructions of objects at historical sites using either the location based or vision-based techniques. Assets used in this projects are managed during the development phase and they are included in generated Android application package (APK). Therefore, using larger 3D models and various different types of assets (audio, video and text) leads to increased internal memory consumption of the Android application. Our approach solves this problem by dividing application logic and assets, using a server as a storage for keeping 3D model files.

Culture Shock Story [3] is a project which is trying to encourage people to explore and share their heritage. It helps museums to present their collections and allows creation of new stories. It consists of three different types of media (digital stories, interviews and documentary films) combined in one web page. The web page is made in WordPress with open-source templates. This project's focus is on the media and in the result it is a simple web page. The project is built for users that do not necessarily have advanced knowledge in the field of computer technologies, therefore they are encouraged to give they contribution for community heritage (museums, schools and libraries). With Culture Shock digital storytelling, people have the opportunity to learn how to produce their own story, therefore it is about giving people access to the basic human instinct of storytelling. The main disadvantage of this project is limited interactivity with created story in form of video file. Also, it is not possible to make an interactive presentation of 3D objects.

S. Rizvić and I. Prazina [12] worked on application which combines the serious game with digital storytelling. In this application hyperlinked video stories allow the user to choose the way how a story will be told. This application is made using Unity 3D engine. It consists of multiple video stories presented in nonlinear way. Users watch the main video story and in the moments when a certain detail is mentioned they can choose to watch a sub-story. This way of storytelling gives users the freedom to watch stories with sidestories or only a main story if they are not interested in details. This type of storytelling can be confusing for some users.

Min Lu et al. [10] worked on project which presents a way of integrating multimedia and storytelling in a walking tour mobile application. Tour mobile applications use additional multimedia content to make tour maps interesting, informative and friendly to use. Stories and media is delivered with the application and adding or changing media is difficult. Users have no possibility to create or publish their content.

Another comparable work is a use case study of application for children to share, edit and create their stories. This application [9] is made for iOS devices and stories are stored in a device memory. Story content can be managed only using this application on a mobile device. Also, stories are saved on StoryKit web browser but they are only accessible through StoryKit mobile application. It would be much better if the user can manage his content on the server and include it when needed in one or multiple stories.

S.Rizvic et al. [13] worked on a 4D virtual presentation web application of the fortress in Sarajevo. Combining digital storytelling with interactive 3D models, this interactive storytelling application has the purpose of providing more information to the world about the importance of the fortress. The application is implemented for access through web browser and it is not intended for mobile devices, which is main disadvantage of this approach. Also, any additional changes to 3D scenes used in this project requires application code to be changed. Models and stories from this application are ingested in the new mobile application presented in this paper.

2.2 jMonkeyEngine

jMonkeyEngine is a free integrated development environment for developing 3D applications for multiple platforms. The application described in the paper uses jMonkeyEngine libraries imported in Android studio [4]. The Android device needs to support OpenGL ES 2.0 for the application to work. It offers a list of available 3D objects

which will be downloaded and displayed on a mobile device.

2.3 Service-oriented Architecture

A service-oriented architecture encapsulates system functionalities and offers communication with clients via services. Web service is identified by URI (uniform resource identifier [5]). Google Drive API, integrated in the application, provides a list of web services used for access to online 3D models and markers. Google Drive offers free API [6] for accessing cloud-based storage service. Each model is stored on the user's Google drive account and has its URI. Web service which returns a list of 3D objects also returns URI of an object on Google Drive which can be used to download and display that object in jMonkeyEngine.

3 Android Application Implementation

Due to the high popularity and presence of the Android operating system on mobile devices, we have decided to implement Android application as a practical example of using web services in interactive 3D content visualisation. In this section, we will describe prerequisites, implementation details and final results.

3.1 Prerequisites

The complete solution is based on free and open source technologies, which are available to everyone. As a development environment for Android application, Android studio is used, which is free for download from official page [4]. Although there are many popular development environments like Eclipse, Corona, IntelliJ or Xamarin Studio, we have chosen Android studio because it is designed particularly for Android development with support for Windows, Mac OS X and Linux. In order to develop and run Java applications, it is required to configure Java Development kit (JDK), as for any other case where Java programming language is used. Android studio relies on Gradle, open source build automation system, which is used for simple management of dependencies used in this solution

3.2 Implementation Details

The fortification known as "White Bastion" is one of the most impressive and important historical sites in Sarajevo. It is located on the south-east outskirts of the city, with an overview on the city valley. Through history it had a very significant and strategic position. The fortification is a part of the dominant defense walls that were surrounding the old city of "Vratnik". The value of the historical site presents the various strata, starting from medieval until the present time. During the archaeological excavations there were found the remains from medieval fortification from 14th century, Ottoman period (17th century) when the fortification was expanded and some new objects were built. During Austro-Hungarian rule the part of the fortification and the object inside the walls were demolished and destroyed and a new group of objects was built. During the early excavation, a significant number of artefacts was found, registered and conserved for the purpose of the exhibition hosted in Museum of Sarajevo.
4D virtual presentation of White bastion aims to present the historical development of this cultural heritage object using virtual reality and interactive digital storytelling, in order to support archaeological research of the site and raise awareness of general public. This idea was proposed by the leading archaeologist of the White bastion excavation in order to visualize their hypotheses on the appearance of the object in different historical periods. There was also a need to inform the Sarajevo citizens and tourists about the cultural importance of the fortress and facts from its history. Benefits of this work are enjoyed by the archaeological scientists, museum professionals and visitors of the site, as well as internet visitors of the project. BH Radio Television, a co-producer of the project, broadcasted a documentary about White bastion created from digital stories within the project, which will benefit their viewers.
Implementation of 4D virtual presentation of White Bastion consisted of following steps: materials collection, design of the application and interactive digital storytelling concept, digitisation and virtual reconstruction of archaeological findings, 3D modelling, digital stories production, interactive virtual environments creation and web implementation.
All digital stories, interactive environments and virtual reconstructions are accessible through the project web site. Project contains 10 digital stories (Intro, Medieval period, Ottoman period, Austro-Hungarian period, the mosque, military music, Hećimoglu Ali pasha, gunpowder magazine, reconstruction in 1889, the end), 6 interactive virtual models of the fortress and presentations of 22 selected digitized and virtually reconstructed archaeological findings (Figure 2).

In this paper we describe the development of the mobile friendly version of the White Bastion project. The proposed solution can be used in any case of using cloud storage for keeping 3D model that is downloaded once upon user request through REST web service. As the use case of our solution, we implemented Android application with access to Google Drive storage through its API. User selects one or more 3D models for display on the device. Besides .obj, .mtl and texture files, storage folder contains .json file with information about marker location in 3D space and URL mapped on the marker. The marker is used to add additional information to 3D model on the scene. For a better understanding of architecture and communication between the components, Figure 3 shows the

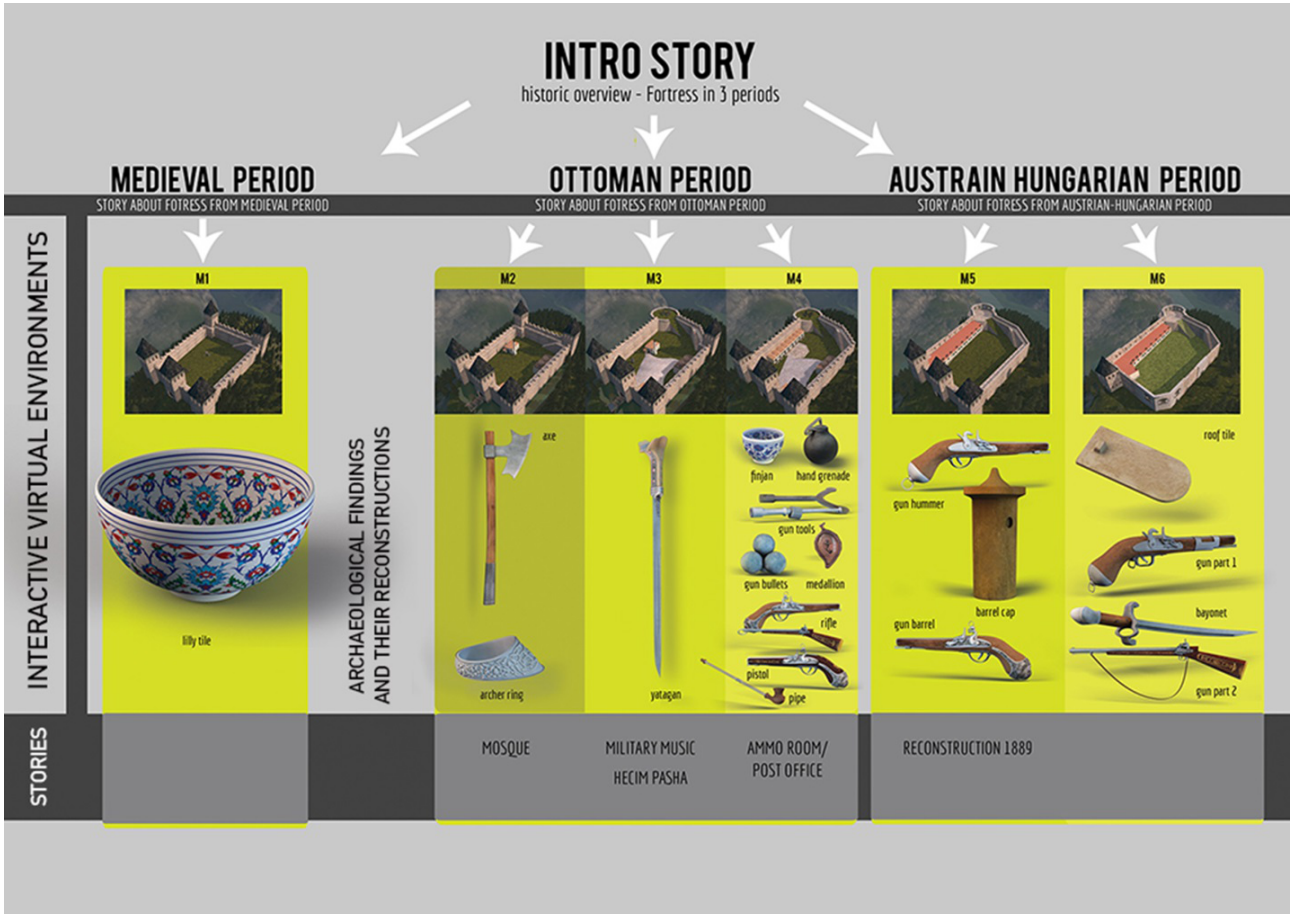

Figure 2: White Bastion project structure

logical view and common scenario of our solution. In next sections, it will be described in a more detailed way.

3.3 Model Loading

Most of the present 3D applications use internal memory of a device for keeping 3D model's data, which increases requirements for memory resources. Our concept is based on using cloud storage for purpose of reducing application's memory consumption. The basic idea is that the content creator uploads the 3D model on Google Drive cloud storage using web application interface provided by Google, and the user downloads it on demand. After completed download, the model is ready for presentation on the device.

After the start of the application, the user needs to choose one of his Google accounts that will be used in the application. Then the application asks from user to grant permissions for access to Google Drive storage. It is necessary to allow access in order to enable integration of Google Drive services and initialization of user credentials for communication with API. Drive API offers different scopes with different accessing restrictions [7]. Java Drive service, used in communication with the API of Drive server, in every request adds user credentials for identification on the server side. Drive server receives request and checks if the request is authenticated and authorised.

During first API call, only information about folders is fetched. The user enters a string in the search field and the server returns folders containing that string in the form of JSON array. The response contains only information about the requested folder, so we need to specify in the request which information we need. Example of response JSON from the server is bellow:

```
{
"fileId": "id",
"title": "someFolder",
"mimeType": "folder",
"link": FILE_URI,
"parentFolder": FOLDER_ID,
}
```

After fetching the information about folders, the list with download options is displayed. To download files from folders, they need to be shared and easiest way to do that is using Google Drive web application. When the particular folder is selected for download, a new request is sent for information about files that folder contains. The server generates JSON response containing information about

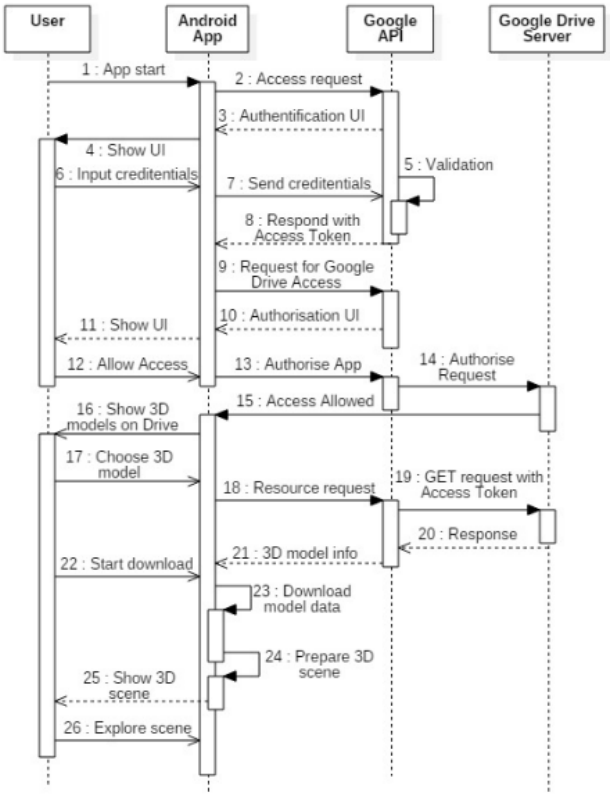

Figure 3: Sequence diagram that shows interaction and communication between the user, the application, Google API and Google Drive Server

files. The most important information is direct download link used for batch download of all files from the folder. After download of all files is finished, application stores the received information in SQL database for later use. It is always useful to store rarely changing information received from the server in order to reduce network traffic. Using this recommendation, it is possible to check if local 3D model is up to date with the online model. Redownload is necessary only if the online model is changed.

3.4 3D Model Preparation Process

Each model used in the application is simplified using Blender. Blender offers decimate modifier [8] which reduces the number of faces of an object. Simplification is the process of eliminating faces with minimal shape changes. Objects with fewer number of faces are rendered much faster which is important for application's frame rate. In mobile applications, 3D models are displayed on smaller screens and they can be simplified more without affecting render quality.

3.5 Displaying 3D Model

In the second part of the application, the downloaded model is displayed using JmonkeyEngine. Choosing the option of viewing 3D model, interactive model representation is displayed. It is possible to control the camera and start the content mapped with the 3D marker. Managing of the assets used in jMonkeyengine 3D scene is enabled through assetManager. AssetManager loads assets from "asset" folder on the Android device. Modification of this folder's contents is not possible during runtime but only during application development. To load data during application runtime, assetManager has the functionality of loading folders from custom paths. It is required to register location from which we want to load the model and display it in the 3D scene. Basic steps are given below:

```
assetManager.
        registerLocator(
                "folderPath",
                FileLocator.class);
model = assetManager.
        loadModel("name.obj");
rootNode.
        attachChild(model);
```

There is no need to separately load .mtl file for material definitions or texture files because assetManager loads it automatically. For correct display, the model needs to be prepared using instructions from Section 3.4. Jmonkeyengine uses Node data structure for connecting parts of 3D Scene and the main node is called Root Node. Only what is attached to the Root Node appears in the scene. Jmonkeyengine has two implementations of camera used for 3D scene viewing:

- Fly-By-Camera: default camera that is automatically enabled unless it is explicitly disabled and replaced with another. Center of rotation is its location in 3D scene
- Chase-Camera: it is implemented to be connected on another object in the 3D scene. Center of rotation is the centre of the connected object.

3.6 Interaction with the Model

The interaction with the 3D model is implemented in the form of a marker which is placed on some position in 3D scene, visible when the model is displayed. In this Section, general use of 3D marker will be explained, and later in Section 4, we will show the way of how we used it in our White Bastion project. Required information for marker placement, x, y, z coordinates and link which opens when user touches the marker, are kept in JSON file. For marker location definition we used Blender 3D modelling software. Clicking on any part of 3D scene it is easy to read position of 3D cursor and use that coordinates for defining marker location in our application. Marker data is stored in JSON file, alongside all other 3D model data on Google Drive storage. The example of marker.json file structure is given below:

```
{
```

```
"x": "10",
"y": "15",
"z": "10",
"link":"www.example.com"
}
```

This way, during download of 3D model data, JSON file is also downloaded and application can read marker information, create and place the marker in 3D space. jMonkeyEngine has a simple implementation of handling user interaction with 3D objects. When the user touches the marker, the application registers that event and opens the link specified in "link" field in JSON. The basic concept is described using one marker, but it is easy to expand implementation for support of multiple markers.

4 Results

We applied previously described implementation process on the interactive presentation of White Bastion. For start, one 3D model is uploaded on Google Drive storage together with all textures and marker information in the form of JSON file. Application accesses to Google Drive, finds the corresponding folder and downloads all its content. After completed download, the model is available for display. It is loaded from .obj file and other model data is automatically loaded. Information about markers is loaded from marker.json file. After fetching of all information, the 3D scene is ready for display. Figure 4 shows a screenshot of generated 3D scene on the Android mobile device.

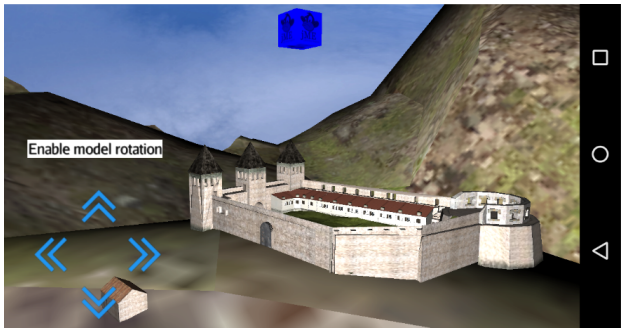

Figure 4: generated 3D scene. The blue marker above the White Bastion is used to provide additional information about the object in form of a digital story. Touch on the marker will generate a new event and Youtube video will be displayed.

jMonkeyEngine game engine used in this project is primary intended for development of 3D desktop applications. Android support is currently very limited which leaves a lot space for improvements in next versions.
Touch support is limited because all touch events are translated to mouse events. Currently, there is no support for real-time shadow rendering and shadow casting. A practical solution for this problem is using of light maps and static shadows which can be generated through texture baking in 3D modelling software.

5 Performance Evaluation

Presentation of 3D interactive content on mobile devices is hardware limited in comparison to desktop PCs. Obviously, reasons for limited hardware performances of mobile devices are requirements for small dimensions and low energy consumption. Good application performance is closely related with the satisfactory user experience. Because of that, there is a constant need for optimisation of 3D applications. Device screen resolution, OpenGL draw calls, scene lighting, texture quality and resolution have a huge impact on the frame rate of 3D application.
Loading models with different complexity and size have a direct influence on application performance. For performance evaluation we used Samsung Galaxy S4 mobile device with hardware specification: Quad-core 1.9 GHz CPU, Adreno 320 GPU and 2 GB of RAM memory. Obviously, on a device with better hardware configuration, load time and FPS from Figure 2. should scale proportionally.
Increasing number of triangles of the 3D model, loading time is increased, almost in linear dependence, as shown in Figure 5. The reason for this is parsing of .obj file and con-

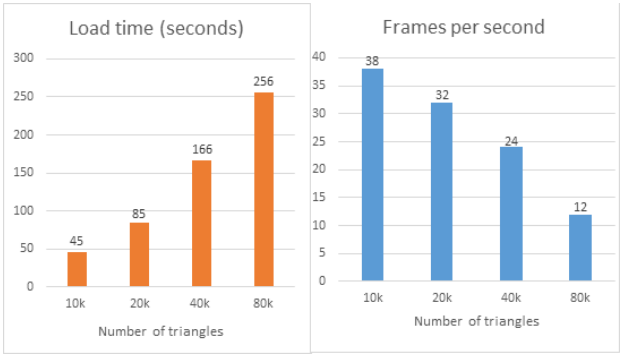

Figure 5: The chart shows load time (left) and average FPS (right) in relation to number of triangles

version in .j3o binary record that uses OpenGL. As a solution, the jMonkeyEngine development team has implemented conversion from the majority popular 3D formats in binary .j3o format. Conversion can be done through the code, although it is primarily done by using simplified user interface inside jMonkeyEngine IDE. Frequent conversion from textual 3D formats like Wavefront .obj in binary format can be optimised. After the model is parsed and loaded once, it is good practice to save generated a binary record for faster reloading of the same model.
The importance of this shows the fact that a model with 80k triangles loads from binary record within 10 seconds. This is a huge difference compared to 256 seconds needed for loading .obj format.

The application performance is evaluated using the same 3D model of White Bastion but with a variable number of triangles. For the purpose of reducing the number of triangles, we have used the Blender decimate modifier. Results from the Figure 5 are satisfactory for a model up to 40k triangles on the device with hardware specification described before in this Section. Model size on a hard drive varies with the number of triangles and with a resolution of used textures. White Bastion 3D model with 40k triangles, including textures with a resolution of 512x512 pixels, is approximately 10 MB. Doubling this number to 80k, frame per second drops by half which is not the case for a lower number of triangles. The ability of smooth rendering relative to a number of triangles depends on particular device's performance. In this project, models are fetched through the network. Condition and bandwidth of a network define the time needed for a model to download.

6 Conclusions

In this paper, we presented a solution that shows the basic concept of using service oriented architecture in interactive digital storytelling. With the daily advancement of computer technologies comes the need for finding new ways of virtual environments presentation. Although it is easier to implement a more functional solution for regular computers, we have chosen to focus on mobile devices because of their prevalence. Also, almost every current software accesses and communicates through the network because of the benefits that come with it. We described the solution that integrates SOA concepts and presentation of virtual environments with storytelling on mobile devices. This solution shows advantages of using cloud storage for keeping 3D model data and additional data which is needed for interactive presentation. Keeping data on the server demands additional effort to implement even the simple application. However, it is inevitable to adapt existing outdated solutions to ever-changing technologies. jMonkeyEngine used in this solution is still limited in the matter of mobile 3D applications development.
Our research and proposed implementation can be extended to support more complex 3D environments and interaction, but that is left open for future work.

References

[1] https://simple.wikipedia.org/wiki/Web_services, 2017. Accessed: 10-February-2017.

[2] http://jmonkeyengine.org/, 2017. Accessed: 10-February-2017.

[3] http://www.cultureshockstory.com/digital-storytelling, 2017. Accessed: 10-February-2017.

[4] https://developer.android.com/studio/index.html, 2017. Accessed: 10-February-2017.

[5] https://en.wikipedia.org/wiki/Uniform_Resource_Identifier, 2017. Accessed: 10-February-2017.

[6] https://developers.google.com/drive/v3/web/about-sdk, 2017. Accessed: 10-February-2017.

[7] https://developers.google.com/drive/v3/web/about-auth, 2017. Accessed: 10-February-2017.

[8] https://docs.blender.org/manual, 2017. Accessed: 10-February-2017.

[9] Elizabeth Bonsignore, Alexander J. Quinn, Allison Druin, and Benjamin B. Bederson. Sharing stories "in the wild": A mobile storytelling case study using storykit. *ACM Trans. Comput.-Hum. Interact.*, 20(3):18:1–18:38, July 2013.

[10] Min Lu and Masatoshi Arikawa. *Map-Based Storytelling Tool for Real-World Walking Tour*, pages 435–451. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[11] Carolyn Miller. *Digital storytelling : a creator's guide to interactive entertainment*. Elsevier/Focal Press, Amsterdam Boston, 2014.

[12] S. Rizvic and I. Prazina. Taslihan virtual reconstruction - interactive digital story or a serious game. In *2015 7th International Conference on Games and Virtual Worlds for Serious Applications (VS-Games)*, pages 1–2, Sept 2015.

[13] Selma Rizvic, Vensada Okanovic, Irfan Prazina, and Aida Sadzak. 4d virtual reconstruction of white bastion fortress. pages 79–82, 2016.

[14] Dusan Tatic, Milos Stosic, Djordje Manoilov, and Radomir Stankovic. *UNIVERSAL MOBILE CULTURAL HERITAGE GUIDE BASED ON ANDROID TECHNOLOGY*. Nacionalni centar za digitalizaciju SANU, 2015.