SHAPES, MARBLES AND PEBBLES: TEMPLATE-BASED CONTENT CREATION FOR LOCATION-BASED GAMES

Richard Wetzel, Lisa Blum, Audrius Jurgelionis[1], Leif Oppermann
Fraunhofer FIT
Schloss Birlinghoven, 53754 Sankt Augustin, Germany

ABSTRACT

This paper presents the TOTEM framework for supporting the content creation and structuring for location-based games. Shapes (templates), Marbles (instances) and Pebbles (raw data) build the conceptual background while two integrated tools support the overall workflow: TOTEM.Designer, a web-application that allows for desktop-based authoring and TOTEM.Scout, a mobile app that allows for in-situ authoring. The TOTEM framework supports different user roles and allows inexperienced users to collect and create content. All data can be exported and easily added to external games.

KEYWORDS

Location-based, in-situ, web-based, game, authoring, data structure, model, template, markup

1. INTRODUCTION

Powerful mobile devices like smartphones and tablets have become ubiquitous and enable designers and developers to create mobile games that take the physical location of the players into account. These location-based games can range from technically rather simple map-centric interfaces based on GPS to more complex augmented reality games. While many of these games place the game objects with no relation to the real environment, others go a step further and closely couple the digital game content with real world locations. Creating such location-dependent mobile games is a demanding task as it requires scouting out the environment, collecting game content and attaching digital content to real world locations. Although several attempts have been made at easing the overall creation and data structuring process, there still exists no generic framework that caters for all peculiarities and supports different experience levels of users, separates between types of users and spans mobile and desktop environments. Typically, the creation of the game logic and the corresponding data structures can only be performed by experienced users. The actual act of content creation for and localization (i.e. transporting the game to a different city) of such a game could potentially be also handled by less experienced users. Such users on the other hand might even be more familiar with the creative aspect of it. In addition, the gaming content itself usually consists of a collection of data: texts, numeric values, images, GPS positions, etc. In scavenger hunt games or location-based quizzes (and also in other genres) these game objects typically appear over and over again in the game while keeping the structure and just changing the data itself. Once the structure is defined, creating new objects of this type becomes rather easy and can be done very comfortably in a desktop environment. The users have access to a large screen, a good keyboard, image processing tools, online maps, internet for research, and can easily take breaks during the creation process and wait for creative sparks before they continue working. Location-based games however profit from content that is tailored to real world locations. Therefore, it is naturally necessary for the game creators to leave their desktop workspace and go out in the real world. Here, they want to take notes about interesting places, snap photos to later use in the game (or just as inspiration) and take samples of location information. Due to a lack of a dedicated system, today this can only be done by combining several apps or even devices that cover various aspects of this process (e.g. using a camera and a notepad). However, making use of this data afterwards becomes a tedious process as the relevant bits need to be extracted and structured manually.

[1] This work was carried out during the tenure of an ERCIM ”Alain Bensoussan” Fellowship Programme. This Programme is supported by the Marie Curie Co-funding of Regional, National and International Programmes (COFUND) of the European Commission.

In this paper we present a template-based approach for defining and creating structured objects for location-based games that is implemented in two integrated tools within our TOTEM framework. In section 2, we take a look at the related work concerning authoring tools. Section 3 presents the analysis of two use cases in order to elicit the detailed requirements for our framework. The conceptual part of the framework is contained in section 4. Section 5 describes the implementation of our approach and presents two tools, the web-based TOTEM.Designer and the mobile app TOTEM.Scout that are used for game content authoring in desktop environments and in-situ, respectively. Section 6 discusses the framework and states the advantages of our approach. Section 7 concludes the paper with a summary and a look at potential future work.

2. RELATED WORK

Public perception of (computer supported) location-based games arguably started with the advent of Geocaching, a simple treasure hunt-game that was invented on the day that GPS became publicly available for civilians on 02 May 2000. The game-content consists of a pair of GPS coordinates and an accompanying mission name and description which are published on a web-site. Using this information, players decide which mission to play and then go outdoors with their GPS-device to find the secret cache. Authoring a new mission for this game requires little more than reading a coordinate from a GPS-device when hiding a cache and then publishing the coordinate and a description of the mission on a web-site. Due to the simplicity of this structure and the mastering of the utilized devices by its users, the game did not need further tool support for authoring and has grown very popular without it [O'Hara 2008]. However, follow-up designs quickly grew more complex and required dedicated tool support for the programmers and/or the domain experts.

Generic authoring tools like the ones described by [Tutzschke et al. 2009], [Fetter et al. 2007], [Linner et al. 2005], [Magerkurth et al. 2003], [Wetzel 2008] and [Hull et al. 2004] are too complex to use as they try to provide a completely open system including (and focusing) on the possibility to implement the game logic itself. The MARS authoring tool on the other hand provided journalism students with a domain-specific authoring tool for authoring content for a campus-tour for an existing early mobile augmented reality system [Güven and Feiner 2003]. The tool could only be used for that purpose and with a bespoke custom hardware setup. Similarly, but working with off-the-shelf mobile-devices, the CAERUS system provided a desktop-based editor and a mobile player that allowed preparing content by filling a given structure using the editor at the desktop and then testing it outside using the player [Naismith et al. 2005]. Testing outside is a necessity for this type of games. Even more so, [Weal et al. 2006] highlighted the importance of being in-situ for triggering the ideas for location-based content and later revising the content back at the desktop. In addition to this [Oppermann et al. 2008] proposed to further integrate the mobile and stationary workflows to support an iterative development that alternates between these two modes of interaction and which helps in better understanding the characteristics of the material, which is the underlying wireless positioning infrastructure.

The need for an easy to use content creation interface has been addressed in numerous publications [Sauer et al. 2006], [Cheong et al. 2008], [Meddler et al. 2006] and [Bäckström et al. 2009]. They are mostly focusing on interfaces for interactive story creation and authoring. The main goals have been to reduce the effort and enabling non-programmers to create elaborated contents [Sauer et al. 2006]. Similarly, the authors of [Bäckström et al. 2009] introduced an authoring tool for the Backseat Playground pervasive game. The mentioned authoring tools are limited to their application domain, which is very narrow since all of them are suited only for a particular type of content. [Wang et al. 2011] focus on a wider range of pervasive games proposing a general gameplay model and the design of a platform independent authoring tool.

3. REQUIREMENT ANALYSIS

When looking at the creation of content for location-based games, two main scenarios come to mind. On the one hand there could already be a definite game idea and the users want to create content of a certain structure for a specific game (use case 1).

Alternatively, users might just want to collect data without yet knowing for what game they might want to use the content later on. This could be seen similar to a brainstorming while getting inspired by the environment (use case 2).

3.1. Use Case 1: Creating content for a specific game

Tidy City, described in [Wetzel et al. 2011] and [Wetzel et al. 2012], is a typical location-based scavenger hunt game. Players walk around the city where they can find riddles consisting of an image and a hint. The riddles point them to another location that they need to visit in order to progress in the game. The game uses very simple game mechanics and in order to really shine requires creative riddles that are closely connected to the real world where they are placed. This makes it impossible to auto-generate or easily port riddles to another city. Instead the necessary data for a riddle needs to be created manually. The structure of a riddle is always the same and consists of a title, a difficulty, a category, a pick-up location, a textual hint, a photographic hint, the correct destination, a reward text and a reward photo. Creating a new riddle does not require any programming skills as content creation is completely separated from any game logic and can thus be done by inexperienced users who can therefore create their own missions easily. For these users it is useful (or perhaps even necessary) to walk around the real environment to scout for potential riddles. On the one hand, photos need to be taken and GPS positions are also required. Walking around in a city has also proven as extremely useful to gather ideas for potential riddles. Checking whether a location is actually useable in the game is also necessary as construction sites or bad GPS reception might thwart any attempts to include the location in the game. On the other hand, creating the textual hints and reward texts is preferably done at a comfortable position at a PC with a good keyboard as typing long texts on a mobile phone is a tedious task and creativity is needed to create an entertaining hint. When users have created a set of new riddles, they can publish them as a new mission, so that other players are able to download the new content from within the game.

3.2. Use Case 2: Creating content for later usage

More experienced users might be creating several location-based games over time. When walking around a city they often come across interesting places and locations they would want to use in a later game. However, at this stage they are only interested in taking quick notes and not yet decide in which game they might want to use the data and in what form. In such a typical in-situ situation it is important to be able take textual notes, but also to record images, audio and video as well as location samples. The data should then be stored in a more accessible system where the users can sort, filter, edit and remix the data they have collected. At this stage it is important that the users are not limited in what they can record and how much of it. Similarly, users might go on a “hunt for content” when they already have a rough game idea and are just looking for potentially interesting locations that might get used later when the game is more developed.

3.3. Derived Requirements

Based on these two use cases a framework for supporting the content creation for location-based games must support the user in:

1) Defining a structure for game objects allowing a variety of attributes like texts, images and GPS positions
2) Creating game object instances from this structure
3) Taking notes and recording thoughts, media assets and location data while on-site
4) Creating and editing game object data while in a desktop environment
5) Creating and editing game object data while on-site with a mobile device
6) Exporting completed game objects so they can be easily parsed by other systems (i.e. games)
7) Synchronizing data between different tools (if separate tools are developed)

The first three requirements have to be dealt with by the conceptual framework, while the latter ones are implementation specific.

4. CONCEPTUAL FRAMEWORK

The TOTEM framework introduces three different entities to satisfy the first three requirements: Shapes, Marbles and Pebbles. Shapes are templates that define the structure of Marbles (which are the actual game objects). Marbles can be filled with data directly from the users. Pebbles are unstructured collections of raw data, typically created in a mobile context. Their data can later be added to Marbles. An overview can be seen in figure 1. The different components are intentionally not named after established programming paradigms or similar in order to make them more accessible to non-experienced users. The main idea is that Pebbles are collected in the wild. They are rough and unpolished. A Marble on the other hand could be seen as a Pebble that has been polished and pressed into a certain Shape. Furthermore, marbles in general are intended to be used for gaming purposes, which is also true for the Marbles in our framework. Marbles are the essential game objects in the TOTEM framework.

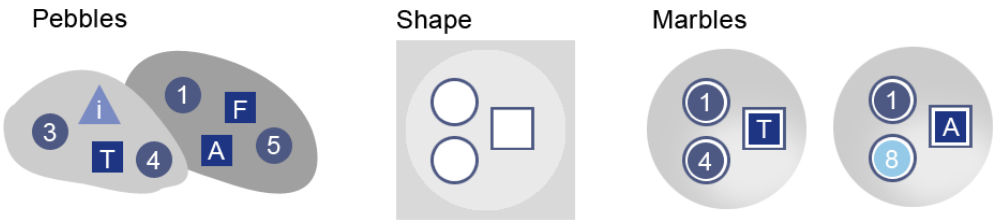

Figure 1. Two Pebbles with unstructured data, a Shape and two Marbles filled with data from the Pebbles.

4.1. Shapes

Using a well-defined structure for game objects and then instantiating an object from this structure bears clear similarities to object oriented programming (where the class defines and object) or modeling a database and then letting the users fill-in the data (e.g. scaffolding). The big advantage of such templates is the fact that the (conceptually) complex creation process of the template is followed by easy instantiation. In the TOTEM framework, these templates are called Shapes. Each Shape consists of a collection of different attributes that can be selected by the users. We distinguish between:

- properties (of type string, text, integer, float, ..)
- media assets (image, audio, video, 3d model, ..)
- locations (GPS position, GPS polygon, NFC tag, Wi-Fi signature, ..)

They can be combined in any way desired and can appear several times in a single Shape or not at all. It is certainly possible to come up with other attributes that might be needed by some games – in this case another type of attribute can easily be added to the concept. A very important aspect in the presented work is however that all this information is not only contained in the data structures on a software engineering level, but also reflected in the respective user interfaces of the two integrated tools.

For some cases, it might be useful to be able to assign a default value for an attribute and even make this value a constant so that all instances share the same value. All attributes that have been chosen are however mandatory and cannot be left empty when instantiated. Also, if the Shape creators want to enable e.g. the use of two images in a game object, they need to create two different image attributes for the template. A sample Shape for Tidy City is shown in figure 2 (left).

4.2. Marbles

When the Shape has been defined, it is trivial to create instances from it. These instances are called Marbles. Other users can now fill the empty structure provided by the Shape with data as they know exactly what is missing to complete a Marble. As the Shape and Marble creation are separated from each other, it allows expert users to create Shapes for their games (as they know what is required by their game design and the game implementation). Less experienced users (or alternatively: content experts) can then create new Marbles and populate the game.

4.3. Pebbles

For collecting unstructured raw data while on location (and fulfilling the third requirement), Shapes and Marbles are not sufficient as the users do not know beforehand what they want to collect and should not be limited by a rigid Shape definition. To solve this issue, we introduce Pebbles, which are a new type of collection object that let the user store textual notes as well as one location and an unlimited amount of media assets. Data from a Pebble can later be used in a Marble and thus be included in the game.

5. IMPLEMENTATION

The first decision that had to be made in order to implement a prototype was whether there should be a joint tool both for in-situ as well as for desktop-based authoring. While a single tool might initially sound tempting to save programming effort, we finally decided to develop two separate tools. The reason for this being the very different affordances that each authoring situation requires. For a mobile tool, the interface needs to be completely different and optimized for the mobile context in comparison to a desktop environment. Furthermore, the only possible technical solution for a single system would have been to make it completely web-based so that it could be accessed from the browser on the mobile device. While HTML5 offers many improvements, it is still lacking in regards to the recording of sensor data (GPS is possible, but cell id and Wi-Fi signatures are not) and media. In addition, one should never assume full connectivity in a mobile context, therefore relying on a website for the authoring is not an option for the mobile system.

The aforementioned reasons led us to the development of two separate tools: TOTEM.Designer for web-based desktop editing and the mobile app TOTEM.Scout for in-situ authoring. By having these two tools it is now possible to satisfy the requirements for comfortable editing in both contexts (requirements 4 and 5).

5.1. TOTEM.Designer

The desktop oriented part of the authoring suite is called TOTEM.Designer and is being developed as a web application utilizing Django[2], a Python-based web framework. In a desktop setting we can assume constant connectivity and therefore such a web-based approach seemed most promising as it does not require the users to install any software. Therefore the system can be run on any computer with a modern browser.

Django was chosen because it provides a powerful Model-View-Controller approach to projects as well as Object Relation Mapping and quick scaffolding properties which makes it well suited for rapid prototyping. The interface itself is built in HTML5 with JavaScript, JQuery and Ajax. After registering, the users have full access to the main areas of the tool: Shapes, Marbles, Pebbles and Export.

5.1.1. Shapes

When the users create a new Shape, they can assign a name and in addition also specify a color. The color is used to quickly distinguish visually between Shapes. The users can then add new attributes to their Shape. When the users add a new attribute, they also need to assign a name. A Shape can always be edited again at a later stage to add new attributes and rename or remove existing ones. The interface is shown in figure 2 (left).

5.1.2. Marbles

In the next step, the users are able to create new Marbles. As a Marble is always created from a Shape, the users only need to select the Shape the Marble should be based on. Afterwards, the users are presented with a form that includes all attributes of the Shape and their names. Properties can easily be filled in by entering text and media assets can be added with file upload dialogs. GPS positions appear on a Google Map. The users can modify the position by dragging and dropping a marker around the map or by entering latitude and longitude values directly in a text input field. In addition they can also specify the radius which marks the

[2] http://www.djangoproject.com

area covered by the GPS position. NFC tags and Wi-Fi signatures consist of unique strings representing the physical tag's serial number or the list of relevant access points respectively. It is important to note that the attributes themselves cannot be edited at this stage and only data can be entered to avoid an undesirable blurring between creating the content and defining the structure. If the users decide that they need a new attribute or want to change something about an existing one, they need to go back to the Shape definition and perform the changes there. This will then affect all Marbles that are derived from this Shape. A sample Marble as required by the Tidy City game is shown in figure 2 (right).

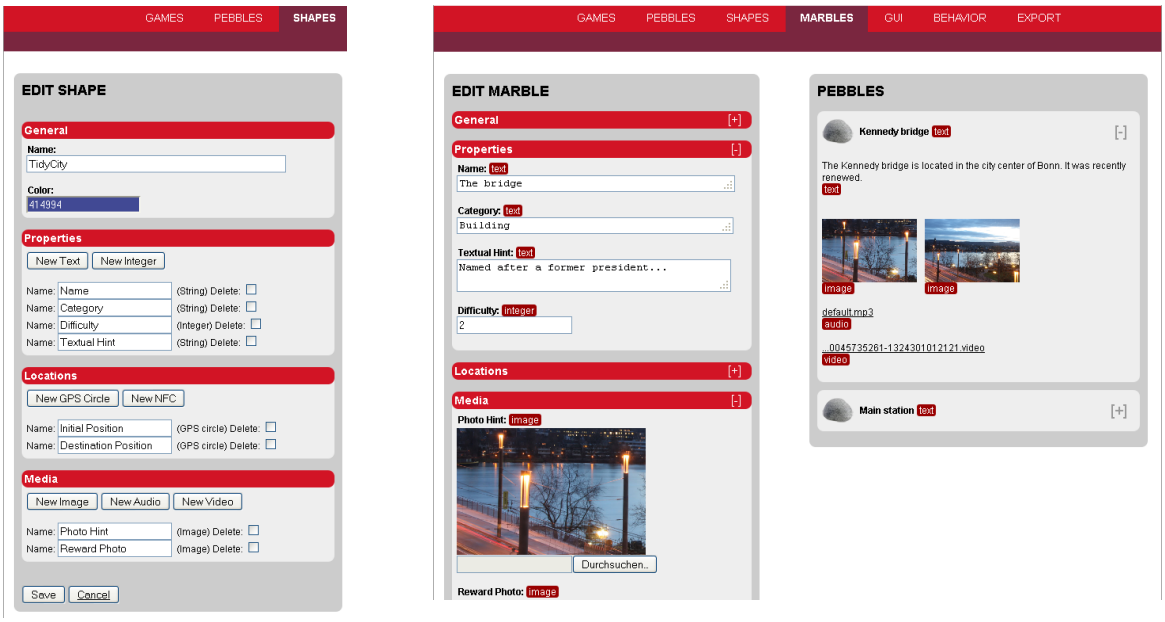

Figure 2. The Tidy City Shape (left) and a Tidy City Marble with a Pebble (right).

5.1.3. Pebbles

In addition to editing Shapes and Marbles, the users can also review and modify Pebbles that have been uploaded by the mobile tool. A Pebble has a name, a textual description, a location sample, and a variable amount of media assets.

In order to use data from a Pebble, the users have the option to display Pebbles alongside a Marble when editing the latter. They can now drag and drop values from a Pebble to a compatible field in the Marble, e.g. the description of a Pebble can be assigned to a text field, images to image fields and GPS positions to GPS position fields etc. This enables the users to also combine data from several Pebbles into one Marble. The data stored in a Pebble can also be used for several Marbles as the data is copied and not just moved.

5.1.4. Export

When the users are satisfied with the Marbles they have created, they can easily export them in various ways and standard formats via the Export menu. A textual representation of the data structure is currently available in JSON (with XML as the next planned option), which also include the Shape definitions. This enables the users to easily add the Marble data to their game as a resource which allows separation between the game logic and the game content. This means that a game can quickly be adjusted by creating new Marbles, exporting them and parsing them in the game during runtime. Finally, the data structure and its referenced media assets can also be downloaded as a fully self-contained zip file or accessed via a XML-RPC interface.

5.1.5. Collaboration

In order to foster collaboration between different types of users, the creator of the game can assign roles to other registered users of TOTEM.Designer. Co-authors have complete access to the Shapes, Marbles and Pebbles of a game, while Content Contributors can only create and edit Marbles and Pebbles. This way less experienced users and content experts can easily be included in the game development process without being overwhelmed by too many options while at the same time limiting access to sensitive areas.

5.2. TOTEM.Scout

For in-situ authoring a mobile tool, called TOTEM.Scout, was developed which runs on Android phones and tablets. The main interface of the tool consists of a map view that shows the position of the user as well as any Pebbles that have been created (figure 3, left). In addition to this, a list view is also provided.

5.2.1. Shapes

Shapes cannot be created with the mobile app, but Shape definitions are downloaded to allow the users to create Marbles.

5.2.2. Marbles

Marbles that have been created on the TOTEM.Designer can be downloaded and edited in the TOTEM.Scout. The main Marble view contains a list of games with their corresponding Marbles. In order to create a completely new Marble it is required to select a Shape from the available Shape list. The interface for creating/editing a Marble is similar to the one used in creating a Pebble, however what data exactly the users can enter is restricted by the Shape definition. For example if a Shape does not possess an audio attribute, the users will not be able to add an audio recording to it. If a Shape has two text attributes, the users will see two different text fields to enter information.

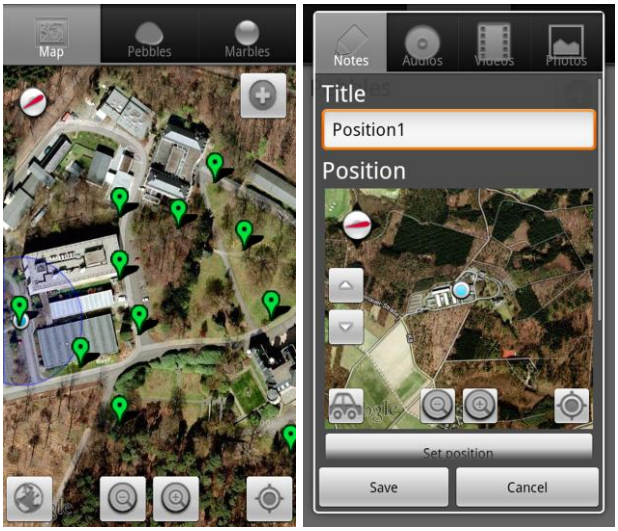

Figure 3. Pebbles on a map view (left) and creating a new one (right).

5.2.3. Pebbles

Pebbles can easily be created on user input. A simple interface is displayed that allows users to assign a name, write down a description and record a GPS location (figure 3, right). Furthermore, users can take pictures as well as record audio and video files. Here, the users are not limited in the amount of images, audio and videos that they assign to a Pebble. The actual process of creating such a media asset file is handled by external apps installed on the Android system (via the Android specific intent mechanism). This simplifies the actual implementation and lets the users choose the apps that they are already familiar with for media asset creation (a proven concept for content creation by domain experts and designers).

5.2.4. Synchronization

Data needs to be exchanged between the mobile tool and the web-based tool (as per requirement 7). Communication between the two tools is implemented through a server-side XML-RPC interface. By calling the exposed methods, Shape definitions as well as already existing Marbles can be downloaded, while Pebbles and Marbles can be synchronized back to the TOTEM.Designer. For file size reasons, media assets are exchanged via standard HTTP requests.

6. DISCUSSION

The TOTEM framework with the presented tools has so far been used in the development of a small student game called Private Investigator. It is an adventure game in which the players have to visit several locations in the Old Town of Cologne on the hunt for an art thief. The students used TOTEM.Scout for finding interesting locations while on site, stored this information as Pebbles and then uploaded the Pebbles to TOTEM.Designer. Here, the students could then create the Shapes they needed in the game, create Marbles and update them with information from the Pebbles. The flexibility of the tools allowed the students to set-up test locations in their home town of Aachen and then quickly modify the content for a session in Cologne for the final presentation by exporting a merely updated set of Marbles. Two independent small projects are currently underway where external content experts use the tools in their own settings which will provide valuable feedback on the whole framework.

The scavenger hunt game Tidy City on the other hand served as a first test environment of the hybrid approach with two different tools. In Tidy City users could create new riddles by filling in a pre-defined set of game data, both in a mobile as well as in a web-based desktop environment. Here, access to the Shape is completely hidden from the end-users – they can only create new Marbles. Creation and editing with both systems worked together seamlessly and allowed users to focus on the actual content creation while being supported in mobile and desktop contexts. Small workshops in Waiblingen, Sankt Augustin and Erlangen with children aged 10 to 15 worked well with the kids being guided during the mission creation process. Furthermore, we had a former school intern creating and testing a full scenario as an outdoor learning station for her school's Day of Mathematics in only about three hours.

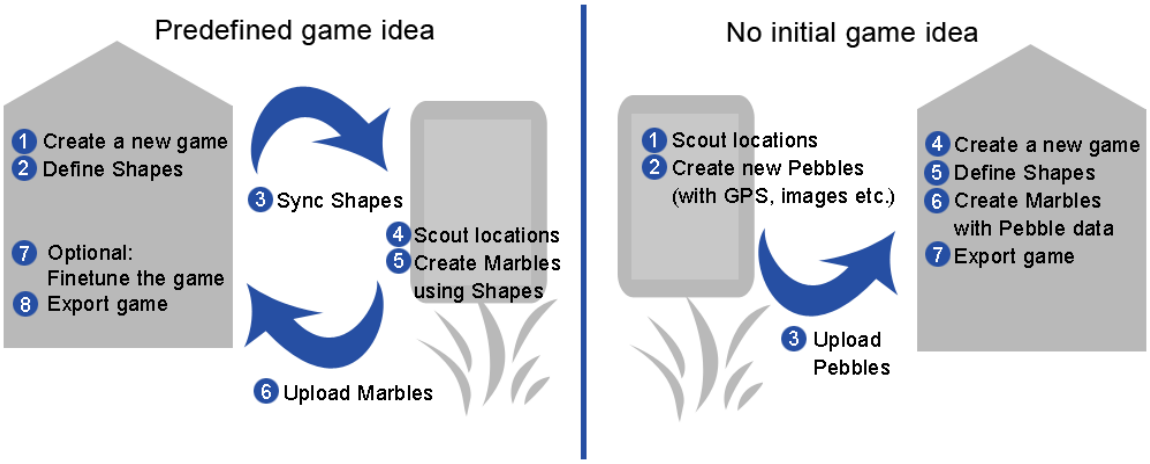

Figure 4. Showcasing two different ways of utilizing the TOTEM framework.

The TOTEM framework in its current state caters for two different ways of authoring as described in the use cases and also depicted in figure 4. A system overview is given in figure 5 showcasing the different user roles and authoring contexts. Here, the different activities provided by the TOTEM framework are split between the game designer and the content creators. When all content has been created, the Marbles can be exported and used in an external and independent game.

Summarizing, the TOTEM framework provides the following advantages for users interested in creating content for their own location-based games:

- A mobile tool allowing for in-situ authoring of specific games as well as for generic content collection and structuring
- A desktop based web tool for comfortable editing and managing of gaming content and structures
- A clear separation between defining a game object and instantiating it to support different user roles
- Export functionality of all data so that everything can be used in games created by the users

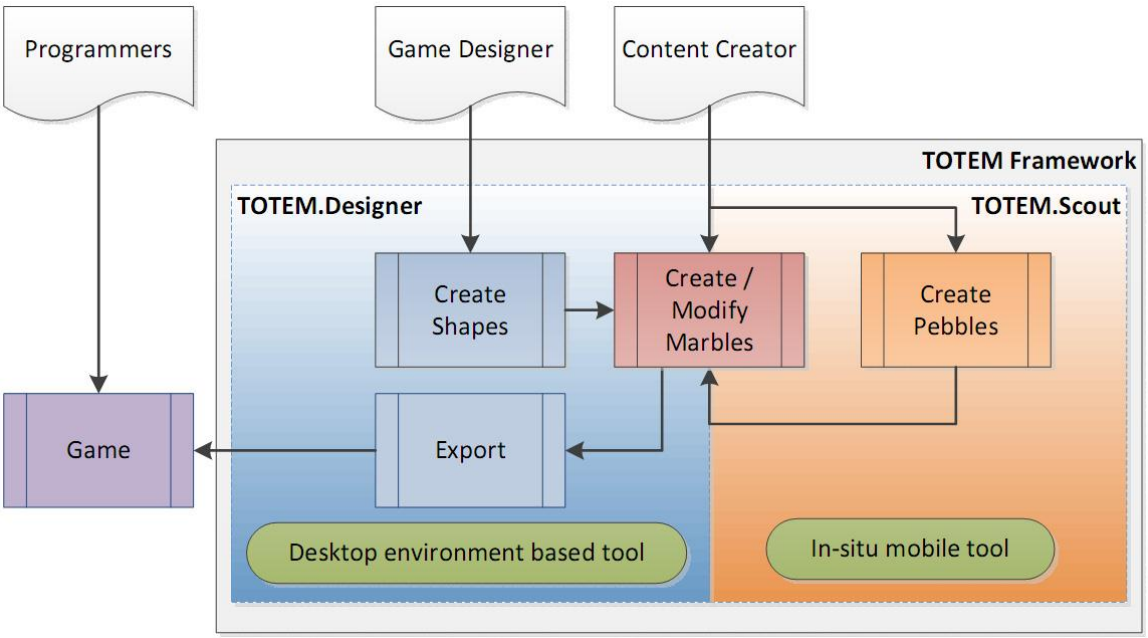

Figure 5. Different user roles and their responsibilities in the TOTEM framework

7. CONCLUSION

In this paper we presented the TOTEM framework for easing the content creation and structuring for location-dependent mobile games. It consists of a template-based approach that allows different type of users to participate in the overall content creation process. Shapes, Marbles and Pebbles have been implemented in two integrated tools that cater for the different affordances of users in different roles and support them in mobile and desktop contexts. TOTEM.Designer is aimed at comfortable editing of data in a desktop environment, while TOTEM.Scout allows for in-situ authoring and is optimized for the mobile context. Both tools work closely together and supplement each other.

There are a number of obvious technical enhancements to the system (e.g. addition of further Shape attributes, more complex synchronization between tools, allowing optional attributes) that will be addressed over time. Furthermore, it seems interesting to explore the possibilities offered by XML-based content validation and transformation tools which might prove to be increasingly useful as our data-structures continue to grow. But despite the technicalities, one particularly interesting avenue of research would be to further stress and evaluate the collaboration aspect that the tools offer by bringing together different users and their varying levels of expertise and interest. This will be addressed in our on-going and future work.

ACKNOWLEDGEMENTS

The work presented in this paper was performed as part of the German-French research project TOTEM - Theories and Tools for Distributed Authoring of Mobile Mixed Reality Games (http://www.totem-games.org). The project is funded by the Programme Inter Carnot Fraunhofer from BMBF and ANR (Grant 01SF0804). We would like to thank our project partners from Telecom & Management SudParis for their support and collaboration.

REFERENCES

Bäcksträm, A., and Danell, E.. 2009 Authoring tools for interactive narratives - an interface design of a script editor for the pervasive game backseat playground. *Master's thesis*, Umeå University, Department of Computing Science, March 2009.

Cheong, Y.-G., Kim, Y.-J., Min, W.-H., Shim, E.-S., and Kim, J.-Y. 2008. Prism: A framework for authoring interactive narratives. *In Interactive Storytelling, volume 5334 of Lecture Notes in Computer Science*, pages 297–308. Springer Berlin / Heidelberg, 2008.

Fetter, M., Etz, M., and Blechschmied, H.. 2007. Mobile chase - towards a framework for location-based gaming. In: Braz, José, Vázquez, Pere-Pau and Pereira, João Madeiras (eds.) GRAPP 2007 - *Proceedings of the Second International Conference on Computer Graphics Theory and Applications* Volume AS-IE March 8-11, 2007, Barcelona, Spain. pp. 98-105.

Güven, S., and Feiner, S. 2003. Authoring 3D Hypermedia for Wearable Augmented and Virtual Reality. *ISWC*. White Plains, New York.

Hull, R., Clayton, B., Melamed, T. 2004. Rapid Authoring of Mediascapes, *Ubicomp*, Nottingham, England, Springer.

Linner, D., Kirsch, F., I. Radusch, I., and S. Steglich, S.. 2005. Context-aware multimedia provisioning for pervasive games. *In Multimedia, Seventh IEEE International Symposium on*, page 9 pp., dec. 2005.

Magerkurth, C., Stenzel, R., Streitz, N., and Neuhold, E.. 2003. A multimodal interaction framework for pervasive game applications. *In Artificial Intelligence in Mobile System (AIMS)*, Fraunhofer IPSI, pages 1–8, 2003.

Medler, B., and Magerko, B. 2006. Scribe: A general tool for authoring interactive drama. *In 3rd International Conference on Technologies for Interactive Digital Storytelling and Entertainment*, TIDSE 2006, pages 139–150 pp., Darmstadt, Germany, December 2006.

Naismith, L., Sharples, M., and Ting, J. 2005. Evaluation of CAERUS: A Context Aware Mobile Guide. *mLearn*. Cape Town, South Africa.

O'Hara, K. 2008. Understanding Geocaching Practices and Motivations, *CHI*, Florence, Italy

Oppermann, L., Koleva, B., Benford, S., Jacobs, R., and Watkins, M. 2008. Fighting with Jelly: User Centered Development of a Wireless Infrastructure Visualization Tool for Authoring Location-Aware Experiences. *Advances in Computer Entertainment Technology (ACE)*. Yokohama, Japan

Sauer, S., Osswald, K., Wielemans, X., and Stifter, M.. 2006. U-create: Creative authoring tools for edutainment applications. *In Technologies for Interactive Digital Storytelling and Entertainment*, volume 4326 of Lecture Notes in Computer Science, pages 163–168. Springer Berlin / Heidelberg, 2006.

Tutzschke, J.-P., and Zukunft, O.. 2009. Frap: a framework for pervasive games. *In Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems*, EICS '09, pages 133–142, New York, NY, USA, 2009. ACM.

Wang, A.I., Jurgelionis, A., Guo, H., and Trætteberg, H. Designing Enhanced Authoring Tools for Pervasive Games. *3rd Workshop on Mobile Gaming (Moga 2011)* at ACE & DIMEA 2011, Lisbon, Portugal, 8-11 November, 2011

Weal, M. J., Hornecker, E., Cruickshank, D. G., Michaelides, D. T., Millard, D. E., Halloran, J., Roure, D. C. D., and Fitzpatrick, G. 2006. Requirements for In-Situ Authoring of Location Based Experiences. *MobileHCI*. Helsinki, Finland.

Wetzel, R., Waern, A., Jonsson, S., Lindt, I., Ljungstrand, P. and Åkesson, K. 2009. Boxed Pervasive Games: An Experience with User-Created Pervasive Games. *Pervasive '09 Proceedings of the 7th International Conference on Pervasive Computing*. Springer-Verlag Berlin, Heidelberg. 2009.

Wetzel, R., Blum, L., Feng, F., Oppermann, L., and Straeubig, M.2011. Tidy City: A Location-based Game for City Exploration Based on User created Content. *Proceedings of Mensch & Computer 2011*. Chemnitz, Germany, pp. 487-496.

Wetzel, R., Blum, L., and Oppermann, L. 2012 Tidy City – A location-based game supported by in-situ and web-based authoring tools to enable user-created content. *In Proceedings of Foundations in Digital Games 2012*, Raleigh, USA, tbp