A Method for Risk Mitigation During the Requirements Phase for Multimedia Software Systems

Kohl Witmer
Department of Computer Sciences
Florida Institute of Technology
kwitmer@fit.edu

Rhoda Baggs Koss, Ph.D.
Department of Computer Sciences
Florida Institute of Technology
rkoss@fit.edu

Tamas Kasza, Ph.D.
Department of Electrical and Computer Engineering
Florida Institute of Technology
tkasza@fit.edu

ABSTRACT

The Multimedia Requirements Method for Easy Risk Mitigation (MURMER) is proposed. This method represents requirements for multimedia document types containing audio, video, image or textual data. MURMER aims to increase the overall transparency and reliability of representations as well as provide a straightforward and clear understanding of the whole requirement pane of the final product. It also aims to decrease risk levels associated to the requirements phase and improves overall completeness. The feasibility of the proposed method is shown in a *Composting 101* tutorial example created in Macromedia Flash and corresponding MS Powerpoint documents. MURMER investigation is compared to other UML-driven schemes. Its efficiency is based on simplicity and completeness. It is also effective as a risk mitigator.

Categories and Subject Descriptors

D.2.1 [**Software Engineering**]: Requirements/Specifications, Methodologies

General Terms

Design, Documentation, Standardization

Keywords

Risk Mitigation, Multimedia Requirements

1. INTRODUCTION

The term 'multimedia' has many definitions. The first computer-based multimedia occurred in the early 1960's when text and images were combined in a document, with the incorporation of continuous media such as video and audio soon thereafter [1]. Within the context of this research, the term 'multimedia' will refer to the encapsulation of text, images, audio, and video in a single document.

Many experts in the field admit the explosion of capabilities in recent years towards the goal of creating worthwhile and creative multimedia documents [2]. As it has often been the case in the past, the creators of these documents often ignore sound design principles that are used with other traditional software processes, and systems are often hacked or thrown together without any design documentation at all. Software engineers in other areas have been adopting UML as a means to force sound, supportive documentation. However, many engineers cringe at the thought of adopting the UML design methodology for multimedia documents because of the cumbersome nature of UML.

This research proposes a simple method for representing requirements for multimedia documents which contain audio, video, images, and/or text. The primary goals for implementing and using this method include risk mitigation, simplicity, completeness, and a straightforward representation of the final product. The requirements document methodology proposed would also work well in a customer-driven enterprise environment, where customers insist upon the early risk mitigation of costly media components such as video.

As a proof of concept, the researchers produced a PowerPoint Document for the requirements method and the final product is a simple tutorial created using Macromedia's Flash software. In this research investigation, since content is rather irrelevant, the researchers chose to produce a *Composting 101* tutorial for school-age children, incorporating all original video with text, audio, and images.

2. MULTIMEDIA REQUIREMENTS RESEARCH BACKGROUND

Multimedia research through the middle 1990's focused on the development of infrastructure to support the capture, storage, transmission, and presentation of multimedia data [1]. Now that products have matured and developers are getting more formal about the design and documentation of such systems, it is natural to employ some worthwhile methods for documenting sound engineering principles.

For example, as B. Adams, et al. point out in [2], a typical viewer of a video is expecting to watch a movie, i.e. to be entertained or educated somehow. However, without planning, a lot of video that is produced is not planned out or thought out, and therefore lacks the necessary elements to make the content interesting or informative.

If a multimedia document designer uses the traditional three-part creation process used by professional filmmakers: namely, preproduction (storyboarding and scripting), production (principle photography), and postproduction (editing) [2] then a typical requirements document could be represented as a storyboard plus some other elements representing flow, behavior, placement, and content.

R. M. Newman has also proposed a hypermedia product using storyboarding as a means of representing linear sequences that are "industrial strength", or best suited for formal verification and validation of processes and procedures in industry. The method Newman proposed is somewhat heavyweight, whereas the MURMER method is meant to be used for basic tutorials and is simpler and less formal and represents a more lightweight process. If risk mitigation is taken as a primary goal, the MURMER method is simple yet also capable of showing a potential customer the general proposed layout of the multimedia presentation without producing the more expensive elements such as video, audio, or imagery. This capability is key since many customers might be reluctant to authorize contracting of a multimedia project if the video will be costly or somehow otherwise *high risk*. Most customers will first want to get an idea of what to expect of the final product. The proposed method acts as a risk mitigator (cost) and a means by which a developer can easily and quickly present ideas at the requirements stage.

3. A PROPOSED METHOD FOR RISK MITIGATION IN MULTIMEDIA SYSTEMS: THE MULTIMEDIA REQUIREMENTS METHOD FOR EASY RISK MITIGATION (MURMER)

The design principles of MURMER include simplicity, completeness, and risk mitigation. Risk mitigation means that the costliest components, although represented and determined at the requirements stage, are not implemented or even prototyped at this early stage.

The requirements document is an official statement or specification of the system requirements for customers, end-users and software developers. Stakeholders of the system are people who have a direct or indirect influence on the system requirements and may include end-users, outside customers, managers, as well as the software engineers themselves [4]. It is desirable to have a requirements document which is readable and easily understood by all stakeholders.

MURMER represents content and sequence timelines with three Formal Sets: 1) a Storyboard (PPT) of the proposed presentation, 2) a Content Filelist of audio files, video files, image files, and text files, and 3) one or more Finite State Machines showing the timeline of frames. Together, these three elements adequately represent at an abstract level all of the necessary elements for a multimedia presentation such as a simple tutorial. It is argued that at this early stage (requirements) a designer is only required to represent the essential components abstractly. Each of the three MURMER components are described below.

The slides of the *PowerPoint Storyboard* have a one-to-one correspondence to the frames of the final Flash presentation proposed. While developing MURMER, it became apparent that it would be productive to create a set of reusable Frame Templates to use at the storyboarding phase. The Frame Templates each represent a specific positioning of all types of content, including audio, images, video, and text. Then the developer chooses from the Frame Template set when designing the Powerpoint storyboard. This helps provide consistency, continuity, and reusability early on in the design process. A sample frame of the storyboard is shown in Figure 1.

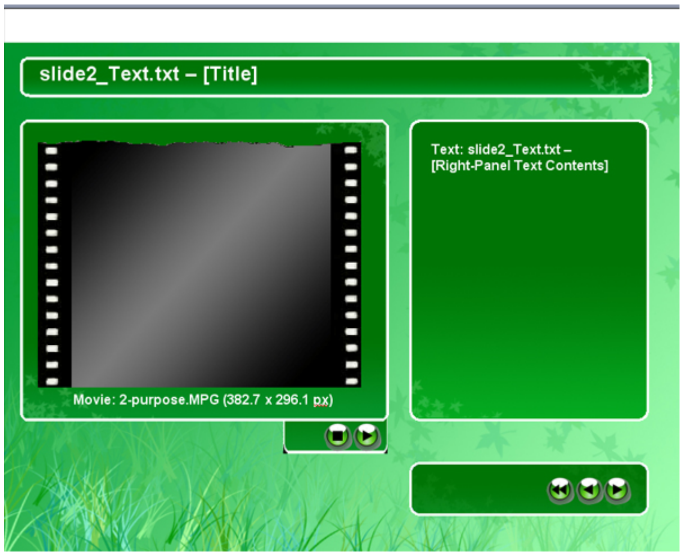
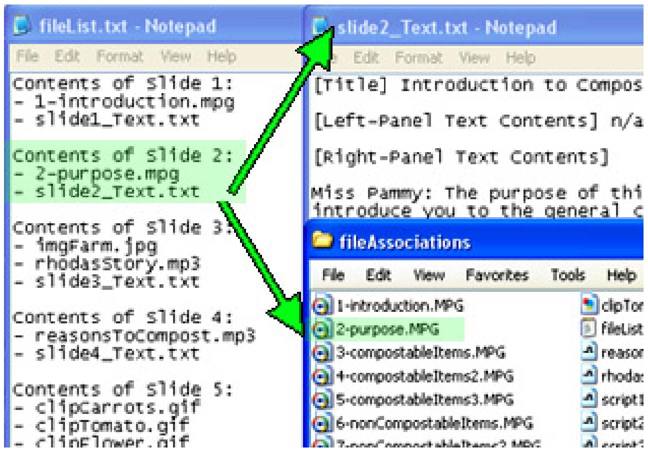

slide2_Text.txt – [Title]
Text: slide2_Text.txt – [Right-Panel Text Contents]
Movie: 2-purpose.MPG (382.7 x 296.1 px)

Figure 1: A Sample Frame Displaying the Layout of Multimedia in the Flash Document

Note the correspondence of each content item mentioned in the Frame and its corresponding content file in the Content File List. This is especially convenient if the customer initially provides content to be used such as images and video. For instance, if one is contracted to create a promotional Flash document for a company, it might be the case that the company will initially make available and provide images and video that can be placed and included at the requirements stage.

The *Content Filelist* represents all content and each file should follow a naming convention and directly correspond to those files used in the PowerPoint Storyboard described above. Each file may represent an existing piece of content or a proposed piece of content. There are four types of file lists in the Content Filelist: 1) text, 2) audio, 3) video, and 4) images. The naming conventions for the files should be used throughout the project.

See Figure 2. for a mapping of the Content Filelist from the Powerpoint slides to the actual files. This mapping provides completeness since it represents all content proposed for the presentation.



```
Contents of slide 1:
- 1-introduction.mpg
- slide1_Text.txt

Contents of slide 2:
- 2-purpose.mpg
- slide2_Text.txt

Contents of slide 3:
- imgFarm.jpg
- rhodasStory.mp3
- slide3_Text.txt

Contents of slide 4:
- reasonsToCompost.mp3
- slide4_Text.txt

Contents of slide 5:
- clipCarrots.gif
- clipTomato.gif
- clipFlower.gif
```

```
[Title] Introduction to Compos
[Left-Panel Text Contents] n/a
[Right-Panel Text Contents]
Miss Pammy: The purpose of thi
introduce you to the general c
```

```
1-introduction.MPG            clipTor
2-purpose.MPG                 fileList
3-compostableItems.MPG        reasor
4-compostableItems2.MPG       rhoda:
5-compostableItems3.MPG       script1
6-nonCompostableItems.MPG     script2
```

Figure 2: The Content File List on the Left-Hand Side Maps from Slide Number to Actual File Contents

The *Flash State Machine* is a finite state machine representing the flow of all of the frames of the PowerPoint Storyboard. It also serves to show the navigational components of each frame and should answer questions such as 'Can we access the previous frame from the current frame?'. The FSMs could be expanded to also answer such questions as 'What media can be initiated from this frame?' See Figure 3. for the FSM for the tutorial designed as proof of concept.

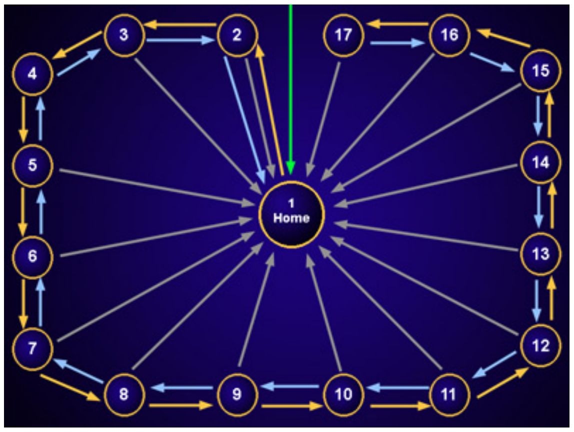
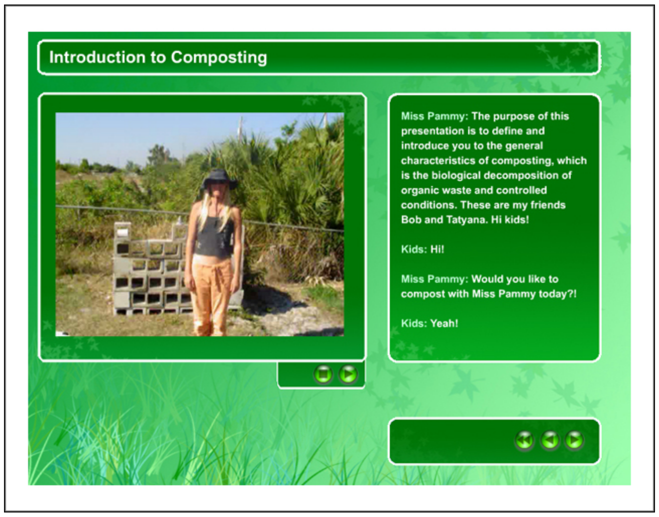


Figure 3: The Finite State Machine Model Displays Navigation Properties of Each Frame Within the Presentation.

4. FRAME TEMPLATES

The concept of Frame Templates is proposed to provide consistency, structure, and a general set of graphics, buttons, windows, or other graphical components for groups of frames in the final product. The actual Powerpoint Storyboard slides created as part of the requirements analysis may have a many-to-one correspondence to the Frame Templates. Template groups may be unique to a presentation or they may be unique to a group, company, or even just a single frame. There is no limit to the number of templates one can design. Note that the use of frame templates also provides reusable components for design.



Miss Pammy: The purpose of this presentation is to define and introduce you to the general characteristics of composting, which is the biological decomposition of organic waste and controlled conditions. These are my friends Bob and Tatyana. Hi kids!

Kids: Hi!

Miss Pammy: Would you like to compost with Miss Pammy today?!

Kids: Yeah!

Figure 4: The Final Version of a Frame from *Composting 101* Created From a Powerpoint Frame Template and Components From the File List.

5. A COMPARISON OF THIS METHOD TO A TRADITIONAL UML-BASED METHOD

The Unified Modeling Language (UML) is the de facto standard for graphically documenting modern software systems [5]. However, there is often a collective sigh of relief from the software developers when they hear that they will not be required to completely utilize UML, as it is complex and difficult to master. With a 10-20 minute tutorial to be designed and implemented, one might argue that the creation of the corresponding UML diagrams (there are thirteen of them) may take many times as long! Even the Object Management Group itself warns of the complexity associated with UML [5]. The drawback of the pure use of UML is that the standard is not complete and too generic and complex and therefore is ambiguous at several points.

The thirteen different UML diagrams have different representation rules and in a lot of cases, the use of UML does not help with risk mitigation or management at all. It is simply not designed for that purpose, however with extended guidelines one or more UML diagram types may be used in the proposed method. The six structure – including class, object, composite structure, package and deployment – diagrams can hardly be tailored to the proposed method's requirement-driven needs. The three behavior - use case, activity and state machine – diagrams however, are generally used by some methodologies during requirements gathering. It is especially true that the use case diagrams are of high importance and may be useful as an addition to MURMER. With use cases, one can easily set up scenarios for the different phases of frames and for the desired sequential order of different media. The four interaction – sequence, communication, timing and interaction overview – diagrams are all derived from the more general behavior diagrams and can therefore be utilized for representing design elements of MURMER. For example a sequence diagram can emphasize the timeline sequence of media items. Further work on MURMER should surely encourage these additions, especially if the tutorials proposed get more complex in layout, content, or behavior.

UML Profiles tailor the language to specific areas in software development and component engineering. These profiles are sponsored by the Object Management Group and current profiles exist for CORBA, Enterprise Application Integration, Enterprise Distributed Object Computing, QoS and Fault Tolerance, Schedulability, Performance, and Time, and Testing [10]. UML profiles provide guidelines for the use of different-purpose diagrams. Other profiles are currently in the works and in general profiles aim to encompass business models, certain types of systems, or particular technologies. So far, no profiles support directly the integration of the use of different media. Perhaps in the future a subset of the available UML techniques could make up a simple Audio and Video Media Profile. After all, the finite state machine is a Behavioral Diagramming technique defined by UML. Additionally, the authors believe as stated above, that a subset of other UML diagrams may be incorporated within MURMER.

6. CONCLUSIONS

The MURMER method represents a starting point for representing simple multi-media tutorials that merge audio, video, text, and images. Its primary function is that of a risk mitigator, so that risky or costly components such as video (or animation) can be postponed and designed around at a very early stage. Other goals with the method include simplicity and completeness when designing tutorial or presentation type media which incorporate text, images, audio, and video. The method is simple as it relies on well-known tools such as PowerPoint and state machines to represent the entire presentation. The method is complete as it covers all content and has a representation of the high risk components and where they would be placed in the overall design. MURMER also strives to be customer-oriented, since it is simple and accessible to customers, can be done at the early (requirements) phase, and can provide excellent risk mitigation at such early stages of development. Furthermore, there is great potential for expansion in the areas of template definition, UML Profiling, and usage with other types of presentations which merge different media.

REFERENCES

[1] Rowe, L., A., and Jain, R., "ACM SIGMM Retreat Report on Future Directions in Multimedia Research", ACM Transactions on Multimedia Computing, Communications and Applications, Vol. 1, No. 1, Pages 3-13, February 2005.

[2] Adams, B., Venkatesh, S., and Jain, R., "IMCE: Integrated Media Creation Environment", ACM Transactions on Multimedia Computing, Communications and Applications, Vol. 1, No. 3, Pages 211-247, August 2005.

[3] Kazman, R. and Kominek, "Information Organization in Multimedia Resources", ACM 11th SIGDOC Proceedings, Pages 149-162, 1993.

[4] Sommerville, I. and Sawyer, P., Requirements Engineering, A Good Practice Guide, Wiley, 1997.

[5] Tilley, S., Murphy, S., and Huang, S., "5th International Workshop on Graphical Documentation: Determining the Barriers to Adoption of UML Diagrams", Proceedings of SIGDOC '05, September 21-23, 2005.

[6] Newman, R., "Team Accessible Methods for Production of Safety Critical Hypermedia Documentation", Proc. IEEE IPCC/SIGDOC Technology & Teamwork , IEEE Computer Society Press, September 24-27, 2000.

[7] Bell, D., "UML Basics, Part II: The activity diagram", http://www.therationaledge.com/content/sep_03/f_umlbasics_db.jsp, Rational Software, 2003.

[8] Object Management Group (OMG), "Introduction to OMG's Unified Modeling Language", http://www/omg.org/gettingstarted/what_is_uml.htm, 2006.

[9] Hall, E., Managing Risk, Methods for Software Systems Development, SEI Series in Software Engineering, Addison-Wesley, 1998.

[10] Object Management Group (OMG), "UML Resource Page", http://www.omg.org/#UMLProfiles, 2006.