GDD as a communication medium

Kim Nevelsteen
Mobile Life
Interactive Institute
Box 1197, SE 164 26 Kista
Email: kim@mobilelifecentre.org

Sergio Gayoso
Mobile Life
Box 1197, SE 164 26 Kista
Email: gayoso@kth.se

***Abstract*—Inquiry into the current development methodologies used by the major players in the gaming industry of Sweden has uncovered many abandoning the Game Design Document(GDD) paradigm. We speculate that the move is primarily because of the long unaddressed shortcomings of the GDD in the rapid paced game industry. We set out to design a new GDD medium, especially designed to expedite communication between different teams of a game production.**

Through published criticisms, post-mortem reports and in combination with our own experiences, we have distilled a set of preliminary general requirements for a new GDD medium. The complete design of this medium will take place in three distinct phases. Aside from the general requirements, this article reports on the first structuring phase, substantiating the general results. The derived structure was tested for its ability to bind pertinent GDD information and support communication between the different production teams.

I. INTRODUCTION

The Game Design Document(GDD) has long been said to be the development paradigm of the gaming industry. The GDD is simultaneously a development methodology and a medium for the design of a game production, most often a video game. The GDD is often initially created by the Lead Designer(s). It can then serve as a written contract between parties as to what shall be implemented. Designers, artists and programmers of a development team then reference it or update it during the development progress. Continual updates during production put the GDD in an ever changing state, which is why it is often referred to as a *'living document'*[1][2][3]. After development it can serve as documentation to what has been implemented.

There isn't a consensus on what exactly constitutes a valid GDD. Each company might utilize the GDD to their own liking. The Game Design Document can hold any number of (sub-)documents within it[4], each with a different purpose or audience; examples include the High Concept Document, Game Treatment Document, Character Design Document, World Design Document, Level Document, Game Script Document, Flowboard and/or Story. These documents might be authored or consumed by people or groups of people with the role of Lead Designer, Game Designer, Level Designer, UI Designer, Writer, Art Director and/or Audio Director.[5]

One of the largest complaints of the GDD is that it can become bloated.[6] A single lead designer can already produce a design document that is quite lengthy. With the production sizes of today, it can be argued that the GDDs have become so large that they are "write only", never read. In addition to this, each individual of the production team keeps a record of their own development progress in the GDD as well. It used to be that the Lead Designer would have written the design in one long flat file. Other mediums have come into existence that can also serve as medium for the GDD, but none seem to satisfy the needs of the game industry. An inquiry into the current development methodologies used by the major players in the gaming industry of Sweden[7] has uncovered many other methodologies in use, including audio/visual design techniques, mnemonics, themes and catch phrases.[7] It would seem that the GDD is being abandoned.

We speculate that a key factor in how the GDD is used is dependent on the company or team size. Smaller core teams will rely more on direct communication using the GDD as more of a documentation tool. When the production is large and/or distributed over teams in different locations, good communication becomes imperative and so participants rely on the GDD more for communication.

Recognizing the short comings of modern mediums to serve as GDD, we set out to design a new medium; the only medium especially designed to serve as GDD and expedite communication between different teams of a game production. The complete design and development will take place in three distinct phases: (1) devise a structure that will hold the GDD and communication data; (2) design the user interface and interaction model to further facilitate communication and effectively visualize the information; and (3) finally build a prototype implementing the results of the previous two studies. Research was first done into the design of the GDD medium at a high conceptual level, which gave us a set of general design requirements and an educated notion of how to build an effective GDD medium. These general requirements were then used in the first structure phase and will be used in the subsequent two phases. Throughout the phases, we will employ user studies to guide the design iterations and allow industry partners to validate our results.

This article will report on the findings of the overall study into the GDD and present them in the form of a list of general requirements. We also report the results from the structuring phase; how this phase implemented the general requirements in an iterative process to obtain a structure for the GDD medium and a structure specific set of requirements.

II. Input for the design requirements

A. Published criticisms and post-mortems

Many mediums have been tested for the GDD; each with its own set of advantages and disadvantages. Cook[8] and Lang[9] have collected lists of pros and cons for different possible mediums, which ranged from a flat file or document to a blog or wiki. Much of the information that we used to specify the design requirements came from these surveys.

One critical aspect was the demand for linearity, or in other words the demand for "one voice" narrating. A lead designer can sit down and produce a nearly complete set of requirements in the same narrative style and can choose to remain in complete control of the text. But, if we have a common medium and allow multiple editors, we undoubtably lose the one narrative voice. According to Danc[8], blocks of loosely linked text written by multiple users are unconvincing when one must sell the design, to publishers for example. The purpose of the document called *"The Pitch"* is exactly that, sell the game design to publishers, and, it is usually placed in the GDD. One way to work around the loss of narrative voice is to devote one single person to collect texts and edit them into one presentable narrative.[1] But, this is provided that the company has the resources for this. Some mediums implement editor roles assigning different levels of write permissions to different authors. With these editor roles it is possible to require changes to a document to be signed-off by a lead editor, mimicking that one single person remains in control of the document.[10]

Another aspect that was complained about was the lack of support for importing Excel sheets (XLS).[9] We take this one step further by recognizing that most mediums lack support for a number of media types. We will expound on this later.

Along side the search for criticisms and surveys of mediums, we also studied the post-mortem's of gaming companies that failed. It was obvious that two of the prevailing reasons for a company failing were due to problems with communication and documentation.[11][12]

B. Survey of existing technologies as potential mediums

At the time of this writing, technologies such as blogs, wikis and flat files are readily available and can easily be put to use by a production team. The criticisms of these technologies as medium for the GDD have been well covered in published material, so we shall not go into individual descriptions of their characteristics. Instead we shall discuss a technology that has not widely been considered as medium for the GDD, but which proves interesting.

Google has developed a number of technologies for collaborative editing, including an "office suite" that is essential the same as the single user versions, but made collaborative. One of these is Google Docs which can be described as a flat file, but online and readily available to multiple users simultaneously. Google Wave gives users a tree structured editing platform where updates to a common *wave* are reflected immediately towards other users. Users are allowed to break in at any point in a running text block and start a new conversational branch. The advantage is that users can visually see where all conversational branches started, because each comment is bound directly to its relative text. Disadvantage being that the original text becomes severely mutilated with ongoing conversations, making it hard to read. What is missing from the technology is the ability to reorder blocks of conversation into different views. A more specific example being, gathering those blocks of text which are important conclusions of conversation branches.

SWC Technology Partners presented the Pivot Browser at the TED conference in March 2010. The appearance of this technology is of particular interest, because it fills exactly the gap we mentioned was present in Google Wave. The Pivot Browser is especially designed to reorder information.[13] It does not use a tree hierarchy, so it can therefore reorganize data on the fly according to user selected criteria.

Before we leave this section on technology, it is important to mention the significance of *mind mapping*.[2] The concept and term has become almost mainstream in modern day and we see a particular similarity between how a mind map links relevant data and what we must do to track information in the GDD.

C. Design aspects from our own experiences

In order to design a new medium for the GDD, we have to go beyond what has already been done. In this section we shall pinpoint two major design requirements.

We have already stated that if we want the GDD to be a communication tool, we must make it a collaborative. But, if the tool is accessed and updated by a large amount of people, we must resolve the issue of the GDD becoming bloated, reducing the possibility of users finding pertinent information efficiently. Users are only interested in a subset of the GDD. If we break up the monolithic GDD into smaller blocks, we must only present each user with those blocks that are of interest. The reorganizing of data was what we recognized as missing from Google Wave. To exemplify, it should be possible for the marketing team to gather unique selling points (USPs) into The Pitch in order to sell the game to a publisher.

Through our survey of technologies and own development techniques, we recognize an extreme lack of support by the GDD medium for a multitude of media types. We have already mentioned the complaints that others have had about the lack of support for XLS sheets in the GDD. This is just one example of an unsupported media type. To give another, say two game designers happen to brainstorm in a cafe and come up with the famed *'napkin design'*, then it should be possible to import that napkin in the GDD somehow. The same applies to whiteboard notes in a design meeting. The only way to

[1]It might be interesting to note that Danc has also analyzed the use of a blog as GDD[8], which neatly allows one person control over the editing by collecting comments made by users and re-posting the changes to the blog.

[2]The Brain was the first mind mapping tool that influenced this work. http://www.thebrain.com

easily capture the drawn up notes without tediously copying them (possibly incorrectly), is to take a snap shot of the notes with a camera. The GDD medium should support the import of those photos. Images of an entire whiteboard or an entire design workshop are monolithic. Not only does this information need to be in the GDD, but users need to be given the tools to operate on the information *i.e.*, adding notes or dividing up the information making it more accessible. If one person speaks to another in a long recorded audio session or a design workshop is recorded, those recordings are vital elements in the design stage and should be added to the GDD.

It should be obvious that there are many multimedia types that need to be supported by the GDD. We want support for images, sound, video and even project prototypes in the GDD. And if all else fails, we should at least be able to link to the data.

III. Design Requirements/Ideologies

After having considered the different aspects we uncovered, we compiled an initial list of general design requirements. It should be noted that some of these requirements might have contradicting aims, so a compromise will have to be sought. We present a short list and then go into detail below.

Communication based requirements for the medium include

- collaborative user editing with enabling communication mechanisms;
- being readily available at all times to all users *e.g.,* web based;
- ensuring changes are communicated to the users, with differentials; and
- support for a variety of different discussion channels *e.g.,* real-time and non-real-time based on video, audio or text.

The medium must also support

- a mechanism to allow for narrative linearity and linear printing;
- editor roles with an option to force edits to be approved by a lead editor[10];
- a familiar user-interface and intuitive interaction model;
- quick updating, with fast access and editing;
- many media/file types *e.g.,* audio, video, images, spreadsheets, et cetera;
- the ability to link relative information, with auto-linking;
- and revision control tracking and a backup system.

In addition to the requirements, we stress the importance of communication and visualization in the design, while targeting as audience either large companies and/or those which are highly distributed.

A. Communication based requirements

In order for the GDD to be an effective communication tool, the most basic requirement is that the medium must be readily available to many users simultaneously. Web-based is usually the most straight forward approach. Not just that the Internet is always available, but that most everyone has an Internet browser installed, so it takes very little time or effort to start the browser and surf to the right page. Both wikis and Google Docs are examples of this.

Once the medium has been accessed by the user, the changes in the information there serves as a form of communication between parties because it is a contract of what shall be developed. We will discuss revision control in detail later, but here revision control is particularly interesting because it shows, who changed exactly what information, at what time. This is usually shown with differentials so that the user can see exactly what parts changed and what the original information was. Note that revision control is also relative to non-text media as well. Images and video can also have edits.

Notifications are another form of communication, in which alerts are presented to the user that particular parts of the GDD have been updated or to other events that require attention. Websites use RSS as their form of notifications and Wikipedia has a very extensive "watchlist" feature which allows users to keep track of changes to pages of interest.

Aside from a user being able to directly edit the information in the GDD, it is important to point out that the user needs a number of ways to **discuss** what has been written. The spectrum of communication types stretches from real-time to non-real-time communication. On one end of the spectrum is chat and voice, while at the other end is perhaps email and blog comments. These channels often have beneficial additional features, such as being able to see who is online and addressing multiple people simultaneously. In this spectrum of channels we also have different media types that can be used *e.g.* video, audio or text. We strive to support as many different communication types as possible in the GDD, but we do not want to stand in the way of users using third party communication applications. Ideally we would like to allow all media types to be added or linked to the GDD.

B. Additional requirements

In addition to the communication based requirements, we have a number of requirements that guarantee the usability and efficiency of the GDD. The first requirement we shall discuss is the important concept of narrative linearity and the one voice. With the advent of wikis, we have support for multi-user editing and a rather unstructured way of organizing text, by linking different blocks of text together. There is no one set path through the web of inter-linked texts unless some structure is explicitly imposed. This is an example of how we lose narrative linearity. Similarly, we lose the one voice because merely stating the medium is multi-user suggests the contrary. This is also effects printing, since one might want to print the entire GDD, linearly.

One possible solution is to explicitly impose structure by using *transclusion*; creating a single page that contains only page "includes" of other pages, so that the result is one long ordered list of blocks of text.[3] Using transclusion alone runs

[3]a technique similar to what we define as a "view" in our design.

the risk that the text becomes hard to maintain. Most wikis that support transclusion have it only implemented for an alternate purpose, such as creating templates.

A second work-a-round option is to require all edits on certain pages to be validated by a lead editor.[10] In order to preserve narrative voice in the GDD, one person can be responsible for its moderation and modification. It should be immediately obvious that we lose some of the multi-user collaborative aspects by doing this. We can implement this feature by using editor roles, but we should be careful not to enforce them by default. Not all adopters of the GDD will want to go this route. This type of security is not at all new and already implemented in many fora and wiki.

We have mentioned wikis quite a lot. But, if you look closer at a wiki, there is a strange syntax which must be learned in order to be efficient in editing. What we want to state is that we require the user interface and the interaction model be familiar to users. But, this is quite a paradox. In order for an system to be familiar to users, it must be popular, but it can't be popular if the system is strange and new to users. We get around this by building upon already widely popular constructs to ease the learning curve.

Besides intuitiveness, we also require a fast edit loop. Namely, users must be able to access the medium fast, edit and commit with minimal delay. The reason for this being that the GDD is often neglected leaving documentation outdated. A fast edit loop lowers the burden of editing in hopes of keeping the production crew updating the GDD.

Here again we would like to stress the design ideology that the GDD should encompass all information relative to the production. The gaming industry is loaded with multimedia, so it is highly needed that as much of this information as possible is included. Also, an important distinction needs to be noted here. The GDD needs to be accessible by all members of the team. Depending on the implemented architecture of the GDD medium, how the information is saved in the GDD can be tricky. If the GDD is centrally located, then information must be moved to the central location to be made available to all users of the GDD. Except, of course, if we have a link from the GDD to an external source. In this case the external source needs to be available to all users of the GDD. We can also have a distributed model as well. To keep it simple, we state the requirement to be to incorporate the media directly into the GDD and if this is not possible, then attempt to link to it. The GDD is relinquished of the responsibility of how to access the linked information. This type of link we refer to as an external link **out** of the GDD. Similarly, third party applications should also be able to link to information in our system through an external link **in** to the GDD.

In addition to this, we have two types of internal links as well. We require a way to link **in** and **out** of an information block in the GDD. We justify this with an example, referring again to a very long audio recording of a design meeting. The likelihood of someone listening to the entire audio file more than once is about as small as someone reading a very long monolithic GDD more than once, in other words, next to none. If the GDD medium is able to link into the audio effectively creating chunks of audio relative to designated subjects, then the chunks of audio become more readily accessible. We could also have internal out links from the audio that lead to different sections elsewhere in the GDD.

With the internal/external in/out links we specified, we can implement *auto-linking* and *autocompletion* with little effort. A game usually gains its own lingo during design and development. Auto-linking is a feature which allows users to predefine a set of terms which always link to their respective definitions. This saves the user from having to redefine common terms over and over. If the definition were to change, all references would remain up-to-date. Autocompletion just saves the user some typing by having the system offer the user the common definitions while typing. Both auto-linking and autocompletion are quite superficial.

If the GDD is to hold all the data mentioned, then it has a responsibility to maintain it. We can use revision control to grant certain desired features. During the development process, it is often desirable to revert to a previous version of your work; this is called a *rollback*. In addition to this, branching the version tree should also be possible. This means that the revision control system is maintaining two differing copies of the data with a common ancestor, allowing for a branch merge, if necessary. Having revision control on a body of text is a common tactic and is employed by some wikis; on the GDD it is mandatory.

Many revision control systems often double as backup *i.e.* committing changes to "off location" data centers with redundancy mechanisms. We want to explicitly state backup as a requirement. Source files must be kept available during production, contrary to lost or deleted.

IV. The structure phase

Given these problematic aspects and others, we attempt to design a medium specifically tailored to the chaotic and creative development process found in the gaming industry. Starting from the requirements stated above we have refined and augmented them to obtain what we believe to be a list of design requirements that encompass what is needed to implement a structure capable of holding the data for the new GDD medium.

Dominate structure design elements include …

- a node graph structure similar to that found as the basis of a wiki;
- collaborative editing of each node with revision control tracking changes;
- each node supporting different media/file types;
- each node having one or more links to other nodes;
- an option to save nodes into user customized views;
- one more ways a user can monitor or be notified of pertinent changes; and a
- wide variety of operations allowing for communication in and about the GDD.

A. Method for structuring phase

In order to test our design ideas we utilized an iterative process; three iterations and ending with a final workshop. Each iteration consisted of designing a prototype, testing it through a workshop, analyzing the user experience and refining the prototype for the next cycle. The final workshop included all feature tests from the previous iterations.

In each iteration we organized a design workshop, three in total. Each consisted of a one hour session with two or three people participating and a final workshop consisting of two sessions, of one and a half hour, with five and six people participating. Users were introduced to the stages of game design[14] and given a chance to figure out different ways to document and store the information using the paper prototype. Our first objective, was to test the validity of the structure proposed. Our second, was to simulate problematic situations that could take place during game development due to misunderstandings or lack of communication. The users were expected to solve situations using only the GDD for communication and documentation.

B. First version of the structure

We required a structure to store the GDD satisfying the requirements. We started with a basic structure similar to that of a wiki; blocks of text with links to other blocks. The GDD, therefore, is the collection of all these blocks of text. We refer to a linkable block of text as a node. User were able to interact with the nodes through a list of simple operations, which were: define a new node, define a new link, delete a node or split a node into two nodes. In addition to the operations, users were given a collection of communication options, which were: chat, email, VoIP and collaborative editing.

We chose paper prototypes for the design process, because we considered them flexible, allowing for modifications during design. Even during workshops, the users were able to propose small changes to the design, which could be tested on the fly. The paper prototype consisted of small pieces of paper notes with a template printed on them representing the nodes of the GDD. The fields of the template contained: title of the node, parent of the node, list of links to other nodes and summarized content of the node. Users were also handed a sheet of paper which listed all operations or communications a user could perform on the nodes. An empty desk was used as the workspace and users were allowed to physically arrange and organize the notes in a way they felt most comfortable with. The users interacted with the nodes in the workspace by choosing an operation or communication action listed and then updating the GDD as the action specified. If communication was text based, paper notes were used for to record it. If communication was audio or video based it was direct conversation between users.

C. Iteration 1

The results from the first workshop were quite positive. A first impression was that the position of the notes on the workspace facilitated the understanding of the content to the users. Because the task was to generate a lot of content in a short period of time, the users were forced to organize the notes as well as they could, to keep track of the growth of the documentation. The prototype also allowed users to easily sort and order the notes to create new documentation based on previous contents. However, the users got confused when the number of notes grew quickly. It was complicated for them to keep many under control simultaneously. Users communicated using only the communication mediums allowed, which was mostly real-time mediums such as VoIP and chat.

During a period of reevaluation we focused our efforts on solving the problem of the GDD growing severely fast. To promote order, we included a new feature called a *"view"*. We defined a view as a subset of nodes defined by users. Views do not affect the node structure of the GDD or the other views, it is just a way to visualize and sort the GDD information. The objective was to keep the workspace manageable and to facilitate work by reducing the amount of simultaneous content. The workspace was now defined to contained one or more views. Another small change was to remove the parent link from the nodes. The paper prototype was altered to reflect the new features and some of the fields of the notes were removed to make note creation faster during workshops.

D. Iteration 2

Views were successfully used to represent the sub-documents from the game development process (for example The Pitch) through customized subsets of the GDD. This iteration's workshop also had a specific section focusing on the links and the relations between the nodes. But, users were confused during this section, due to the high number of links and the disorder they created on the workspace. In our opinion the links between the nodes are an important feature of the design, but due to the limitations of the paper prototype we were not able to get any conclusion. As for communication through the GDD, it fomented debate and discussion between users pertaining to the game design features. The users reached several conclusions from the debates that were important for the development of the game.

It occurred to us that the discussions and conclusions that took place between workshop participants should be integrated into the GDD as well. Instead of just incorporating text communication into the GDD, we generalized GDD nodes from a text block to any kind of media. Again the prototype was updated, this time with audio and video nodes allowing audio and video communication to be recorded.

E. Iteration 3

During the last workshop the users were allowed to “share” views to exchange information. One example of this, was a user creating a “starter packet”[7] as introduction to their game in order to facilitate the integration of a fictive new designer into their production team. Due to lack of a context, orphaned nodes (those without any links) were the cause of user confusion, this iteration. Users also had problems selecting nodes which could be important for them to add to their personal view, effectively tracking their updates. Some users added conversations, which happened during the workshop, to the GDD.

To the list of operations and communication actions, we added the option to “share” a view along with a user defined description. In the previous iteration, nodes were defined to allow for audio and video and because of this we want users to be able to operate on those nodes also. Either the GDD needs to build in tools to manipulate those nodes or users should be able to use third party applications to operate on the nodes, with the results being incorporated into the GDD. With such tools in place, users should be able to split audio/video nodes or link in to or out of them. A notification system was added to views, in order for a user to better track changes to nodes they are working with. If any new nodes were created related to the users view due to links, a notification was generated for these as well. Nodes were required to have at least one link to another node to avoid the lack of the context *i.e.,* no orphans were allowed.

F. Final workshop

The final workshop was intended to test all the new features which were added to the design during all of the iterations. The main difference in this workshop was the presence of three designer teams, two of them working in the same room simultaneously and a third working in a entirely different room, preventing direct communication and awareness of the ongoings of the other two teams. The objective of this workshop was to validate the real-time and non-real-time communication mediums. The workshop was divided into three parts (one for each of the main features of the design): (1) to introduce proposed GDD structure to the users and check its capabilities, (2) scrutinize the communication mediums, and (3) see how well the information generated during the communication integrates into the GDD.

G. Results of the structure phase

To recapitulate, the final GDD structure design consisted of a interconnected node based structure and customizable views which reflect a subset of the GDD. A node contains any kind of media type, including but not limited to text, audio, video, images or XLS. The GDD design supports communication and a notification system is in place to alert users to changes in the system. Through the structuring phase, we have been able to reaffirm some of the general requirements and also distill requirements specific to the GDD structure. We shall discuss the, node structure, views and communication in detail now.

The node structure we devised satisfies the definition of directed graph, with each link equal to a directed edge between nodes. We have required each node to have at least one link in order to exclude orphans and guarantee each node has a context.

Introducing the concept of views was a turning point in the design. Views make it possible for designers to define their area of interest in the GDD, without interfering with another designer. Designers can customize a recognizable version of the GDD on their workspace through a collection of views, with notifications to keep them informed about events and changes relative to their workspace. Views can define the sub-documents of the GDD.

Dialogs and debates between workshop participants generated important conclusions and decisions for the design. The possibility to document all communication in the GDD improves the quality of the documentation.

V. Verify with companies/game designers

The final step in the design process was the validation of our work through a major company in the gaming industry. We chose Digital Illusions Creative Entertainment, DICE,[4] in Stockholm, Sweden as a creditable representative to gauge our work, particularly because they are owned by Electronic Arts, EA, in California, USA and are currently the largest gaming company in Sweden. We were counting on their company being highly distributed internally and also externally through outsourced work. Upon contacting them they presented us with two Senior Designers with opposing, for and against, views of GDD. Upon meeting the designers, they confirmed the distributed nature of the company. Pertaining to the GDD, they confirmed or added (1) the lead designer is **far** too busy to be a responsible document editor, (2) no documenting just for documentations sake, (3) the GDD should be a contract of specifications between parties, but it isn't used that way, (4) lots of design materials (images, audio, text, ...) are potentially lost in the fray, but room walls remain as the most influential design reference, in combination with the "vertical slice", (5) and, the layering of information is very important *i.e.,* those individuals that need to can dig deeper into the GDD.

Our overall impression was that we are definitely on the right track, but that perhaps we can aim even more toward multimedia types than we had anticipated. The meeting was immensely beneficial.

VI. Conclusion

There are three noteworthy conclusions drawn in this work. Through extensive research into the published criticisms and post-mortems, in combination with our own experiences, we have been able to distill a set of preliminary general requirements for a new GDD medium. This set of general requirements has been used throughout the first phase of the design and development and shall be used throughout the two subsequent phases as well. This set of general requirements in an important conclusion.

The next two conclusions we have drawn from the structure phase of the project. The next being the set of requirements specific to the structure of the new GDD medium. Some of these requirements overlap with the general requirements, but it is therefore obvious that they substantiate them. Others are specialized and refined to the functionality of the structure.

The design of the structure itself is the last of the conclusions we wish to bring attention to. We believe the derived structure to be encompassing enough to hold all pertinent information and flexible enough to allow for the needed interactions and communication mechanisms. There were of course some concepts that we could not test without a full working prototype. This reminds us that this is a work in progress and we can therefore expect the two additional project phases to further improve the design of a new medium. This is perhaps the only medium especially designed to serve as GDD and expedite communication between different teams of a game production.

Acknowledgment

We would like to sincerely thank Thomas Andersson and Sebastian Armonioso at DICE for making time in their busy schedule to meet with us. And, also Kjell Reutersward at DICE for his interest in our research.

References

[1] Wikipedia. Game design document. [Online]. Available: http://en.wikipedia.org/wiki/Game_design_document

[2] Wikipedia.org. Living document. [Online]. Available: http://en.wikipedia.org/wiki/Living_document

[3] D. Davies. (2000, Dec) Common methodologies for lead artists. [Online]. Available: http://www.gamasutra.com/view/feature/3115/common_methodologies_for_lead_.php

[4] R. Bartle, *Designing Virtual Worlds*. New Riders Games, 2003.

[5] E. Adams and A. Rollings, *Fundamentals of Game Design (Game Design and Development Series)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

[6] F. D. Laramee. (1999, Nov) The game design process. [Online]. Available: http://www.gamedev.net/reference/articles/article273.asp

[7] U. Hagen, "Designing for player experience," *Nordic DiGRA*, 2010. [Online]. Available: http://www.digra.org/dl/db/10343.03567.pdf

[8] D. Cook. (2005, June 2) Using a blog as a game design document. [Online]. Available: http://www.lostgarden.com/2005/06/using-blog-as-game-design-document.html

[9] T. Lang. (2009, May) Four ways to write your design docs. [Online]. Available: http://www.gamecareerguide.com/features/737/four_ways_to_write_your_design_.php

[10] T. Ryan. (2009, July) Learning the ways of the game development wiki. [Online]. Available: http://www.gamasutra.com/view/feature/4094/learning_the_ways_of_the_game_.php

[11] B. Sheffield. (2009, April) What went wrong? learning from past postmortems. [Online]. Available: http://www.gamasutra.com/view/feature/4001/what_went_wrong_learning_from_.php

[12] F. Petrillo, M. Pimenta, F. Trindade, and C. Dietrich, "What went wrong? a survey of problems in game development," *Comput. Entertain.*, vol. 7, pp. 13:1–13:22, February 2009. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1486508.1486521

[13] S. T. Partners. (2010, August) Swc pivot browser demo. [Online]. Available: http://www.youtube.com/watch?v=2s6AeT-KggE

[14] U. Hagen, "Where do game design ideas come from?" *Proceedings of DiGRA*, 2009.

[4]Digital Illusions Creative Entertainment, DICE http://www.dice.se