What do Game Developers Expect from Development and Design Tools?

Jussi Kasurinen  
Software Engineering Laboratory  
Lappeenranta University of Technology  
P.O. Box 20  
FI-53851 Lappeenranta  
+358 400 213 864  
jussi.kasurinen@lut.fi

Jukka-Pekka Strandén  
Software Engineering Laboratory  
Lappeenranta University of Technology  
P.O. Box 20  
FI-53851 Lappeenranta  
jukka.stranden@gmail.com

Kari Smolander  
Software Engineering Laboratory  
Lappeenranta University of Technology  
P.O. Box 20  
FI-53851 Lappeenranta  
+358 40 546 3493  
kari.smolander@lut.fi

ABSTRACT
Context: Development of software is a complex process with several stakeholders and their varying practices. The game industry has also additional requirements such as artistic presentation, an interesting story and high quality sound effects that further complicate the process.

Objective: The objective of this paper is to understand what kind of requirements and expectations are attached to the technical infrastructure of game development in practice.

Method: This study observed and qualitatively analyzed seven game-developing organizations, from recent startups to established organizations to allow comparison of their use of technical infrastructure.

Results: Based on our study, the game organizations, regardless of their size, are generally pleased with the tools they apply. The selections of the applied tools are based on their ability to test concepts and build prototypes to help design.

Conclusion: In general, the game development organizations expect their tools to allow adaptability to changes during the development process. The case companies did not seem to have many problems with their current technical infrastructure, and ability to test different solutions was considered very important feature.

Categories and Subject Descriptors
D.2.1 [Software Engineering]: Requirements/Specifications – tools, elicitation methods

General Terms
Management, Design, Human Factors.

Keywords
Technical infrastructure, Software Development, Game industry, Development Tools.

1. INTRODUCTION
There is a reported estimate [5] that in year 2010, in United States alone the consumers spent 25.1 billion dollars in games, purchasing 257.2 million games over different publishing formats including mobile games, game consoles and computer systems. Based on these figures it is no surprise that one current area of interest in software engineering and information technology in general seems to be in game development and game industry.

This work studies the design and development tools used by the game-developing organizations, focusing on a number of new business startups and small established game developers. Rather than naming specific tools, this work discusses them in types; what kind of tools are used by the organizations, what are the common problems and needed enhancements for the tools, what features and attributes are considered most important in the game development. There are studies which mention problems with the game industry-related tools [2, 10, 13], so our research question is what sort of expectations and requirements the game organizations have on their design and development tools?

The rest of the paper is constructed as follows; Section 2 provides an overview on the related research in game development from the viewpoint of software engineering. Section 3 introduces the observations made on the technical infrastructure and the collected statistics, while Section 4 discusses these observations and their implications. Finally, Section 5 closes the paper with conclusions.

2. RELATED RESEARCH
In traditional software engineering, the term “development” is associated with the phase where the application is designed and implemented and “production” is often the phase where the software is in use and running [1]. However, in the game development industry, the development phase is often called “production” and the production phase is called post-production or shipping phase. Therefore this work applies the game development terminology [1,2,9] when appropriate or when discussing the concepts presented in the referenced literature.

The interest to study game development from the viewpoint of software engineering is currently quite active, and there are several studies discussing the processes of game developing organizations. For example a study by Kultima [9] discusses the game design process and the innovation phase of the game design in more detail. In many occasions, defined techniques do not work or lead to unsatisfactory results. Based on the results, the most
common method of developing game ideas is by deciding on a theme, designing the initial concept alone, and then “bouncing” the idea in a peer group until a mature and generally acceptable design is completed.

On application of software tools, Blow [2] has discussed the current state and problems of game development. The paper discusses many hardships and problems in game development, and the general difficulty of game development when compared to other areas of software development. Generally the games are complex constructs, and as the technology advances the requirements for versatility in many different areas of the game development like the game engine, network code and artificial intelligence keep growing. To address these problems, Blow also identifies pitfalls and problems which should be addresses in the development of development environments for game industry [2]. Like Kultima and Alha [10], Blow also concludes that the current development tools do not support the game industry well enough. The most common problem with the game development tools usually is that they are not designed for game industry, but are rather used as nothing more appropriate is available. As the game industry develops products to the specific hardware systems such as game consoles, the system-specific development tools tend to become old-fashioned as the system becomes older, and obsolete when the console generation changes. Blow also criticizes the game industry for not applying enough project management tools and techniques to control the spiraling complexity of the development projects and manage the development cycles. The challenges of game development have been studied by Kanode and Haddad [8]. The most prominent observation of their study is that the game industry needs to adopt more rigorous software engineering strategies to address challenges such as managing diverse assets, application of new technologies, and team management. Additionally, Petrillo et al. [13] have conducted a survey on general problems of computer game development. Their survey on game software development lists team management. Additionally, Petrillo et al. [13] have conducted a survey on general problems of computer game development. Their survey on game software development lists six main problems, which were unrealistic original scope of the game design (75% of projects), feature creeps (75%), cutting features (70%), general design problems (65%) delays or optimistic schedules (65%) and technological problems (60%).

Overall, it seems that the game industry has some restrictions with the solutions available to them. The most common problems related to the software tools, as Blow [2] and Kultima and Alha [10] observe, are that the development tools have problems enabling game organizations to do design and innovation, and that the project management aspects are not implemented to a satisfactory degree. This can be observed in the study by Petrillo et al. [13] from the fact that 60% of the interviewed game development projects suffered from technical problems, and almost three out of four projects had problems with some aspects of the game design or management.

3. RESEARCH METHOD

The software process including the design, development and testing of a commercial product is a complex phenomenon to observe and study. Acknowledging this, we decided to pursue empirical qualitative analysis by applying the grounded theory method [6,15]. We considered Grounded theory suitable for discovering and analyzing the technical infrastructure, as it observes and describes real-life phenomena within their social and organizational context. Our approach was in accordance with the Strauss and Corbin [15] approach that further extends the Grounded Theory with the application of the systematic codification and data collection methodologies. In the process of building a theory and interpretation of the results, we followed guidelines by Eisenhardt [4] and Klein and Myers [11].

3.1 Data Collection

To observe such a diverse and large group of organizations such as game industry, our earlier approach of applying polar examples [5] of different operating domains and company sizes along with different viewpoints into the game development was used to gain wide perspective on the industry as a whole. We carried out four interview rounds in our study (Table 1) with four different interviewee groups; project managers, game developers, upper management and game designers. The sample of the interview rounds consisted of seven game development organizations selected from our research partners and supplemented with additional volunteering organizations to achieve a heterogeneous group.

The seven organizations in the study were small to medium-sized [3] professional game developers from South-Eastern Finland. Organizations were either recent business startups or new companies (less than 5 published products) with two more experienced organizations with more than ten published titles. The selection of the cases was based on the polar type selection [4] to cover differences between organizations; the cases included different target platforms and different sizes of development projects. The organizations varied (Table 2) from newly started mobile game developers to browser-based games, PC games offered through digital distribution and even included an established developer with products in the retail stores. The smallest organization in the focus group was a startup with five persons; the largest organization included several hundred people that contributed to the product development.

The objective of this approach was to gain a broader understanding of the practice of and to identify the general factors that affect the technical infrastructure these organizations apply. To achieve this, our research team developed four questionnaires that included questions on themes such as development methods, test processes, quality requirements, test phases, applied tools, outsourcing and design. The complete questionnaires are available at http://www2.it.lut.fi/project/SOCES/. A reference list of the

<table>
<thead>
<tr>
<th>Interviewee role</th>
<th>Description</th>
<th>Main themes of the interviews</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team leader or project manager</td>
<td>The interviewee is responsible for the management of the development of one product, or one phase of development for all products.</td>
<td>Development process, test process, quality, outsourcing, development tools, organizational aspects.</td>
</tr>
<tr>
<td>Developer or tester</td>
<td>The interviewee was responsible for the development tasks, preferably also with the responsibilities of software testing activities.</td>
<td>Development process, test process, development tools, development methods, quality.</td>
</tr>
<tr>
<td>Upper management or owner</td>
<td>The interviewee was from the upper management, or a business owner with an active role in the organization.</td>
<td>Organization, quality, marketing, innovation and design process, development process.</td>
</tr>
<tr>
<td>Lead designer or Art designer</td>
<td>The interviewee was a game designer, or managerial level person with the ability to affect the product design.</td>
<td>Development process, design and innovation, testing, quality.</td>
</tr>
</tbody>
</table>
main themes in different interviews is also available in Table 1.

The interviews contained semi-structured questions, and the whole sessions were tape-recorded for qualitative analysis. We applied semi-structured approach to encourage in-depth discussion over the main themes. Typically, an interview lasted for approximately one hour and they were arranged as face-to-face interviews with one or two organization participant and one or two researchers. During the research, we collected approximately 1400 minutes of interview data for further qualitative analysis. In one organization we were only able to arrange three interview sessions due to the data collection schedules. With this organization the second-round specific themes were discussed with the other interviewees on other interview rounds. Overall, 27 interview sessions were held during the spring, summer and fall of 2012 by seven researchers from two research laboratories.

We interviewed people from the organizations to get several viewpoints into the game development processes. All of the interviewees were adult, professional full-time employees with varying educational backgrounds. The decision to interview project managers during the first round was based on our aim to gain a better understanding of the operational level in the study organization. During second round, we collected more data on technical details of the development process. In the third round, the focus of the interviews was to collect more generalized data on the company upper management, and during the last round, to understand the creative aspects of the game development by interviewing game designers. The interview rounds, interviewee roles in the organization and study structure are summarized in Table 1, and the participating organizational units are summarized in Table 2.

### 3.2 Data Analysis

The grounded theory method contains three data analysis steps: open coding, where categories and their related codes are extracted from the data; axial coding, where connections between the categories and codes are identified; and selective coding, where the core category is identified and described [15].

The objective of the open coding was to classify the data into categories and identify leads in the data. The process started with “seed categories” [14] that contained essential stakeholders and known phenomena based on the prior data such as literature. Our selection for the seed categories included general phases of the software processes such as design, development, testing and management, plus common terms and stakeholders such as financers, customers, personnel, software tools and quality.

In open coding, the classified observations can be also organized into larger categories. New categories appear and are merged because of new information that is discovered. For example, our initial concept of having tool-related problems as a separate category was revised as the interview data proved that the process problems were more related to personnel issues and management.

At the end of the open coding, we had 172 codified entries of 1574 individual excerpts from the interview transcripts.

The objective of the axial coding, which starts when the categories start to emerge and runs somewhat parallel with the open coding [15], is to further develop the categories by looking for causal conditions or any kind of connections between the categories. In this phase, the categories and their related observations were becoming fixed, allowing the analysis to focus on developing the relationships between larger concepts. In this phase, the categories formed groups in the sense that similar observations were connected to each other. For example, codes such as “Design process: refining designs”, “Development process: knowledge transfer” and “Problem: Documentation/knowledge transfer related to design” formed a chain of evidence of how the organization documented and refined their product designs and what problems they had with this approach. By following these types of leads in the data, the categories were coded and related issues were identified.

The third phase of grounded analysis, selective coding, is used to identify the core category [15] and relate it systematically to the other categories. As based on [15], the core category is sometimes one of the existing categories, and at other times no single category is broad or influential enough to cover the central phenomenon. In this study, the examination of the core category resulted to the category “suitability of applied tools to game development”, which we related to a set of tool and process-related concepts listing the issues and pervasive themes. This core category was created as an umbrella category to collect all tool selection-related concepts as none of the existing categories was considered influential enough to explain the entire phenomena.

For example, we observed the primary tool selection method in all of our organizations and were able to collect data on the suitability of these tools to the projects, but were unable to explain why the game developers were usually very satisfied with their existing technical infrastructure.

### 4. RESULTS

In this section we first introduce the categories formed in the analysis. Then we introduce the constructed observations on the technical infrastructure of the game organizations.

#### 4.1 Categories

The core category “suitability of technical frameworks to game software development” was compiled from the observations and categorizations made from the organizations. The codifications were categorized to six categories and the observations were then derived from these categorizations. Sorted Case-by-case observations are also summarized in the Table 3.

The category Source of the game engine describes the origin of the main development tool the organization applied. The game engine is the central component of the game product, which uses
The category **Tool selection principles** describe the decision principles on what tools to use or adopt in the future. Obviously the tools have to be compatible with the target development platform of the organization. This category describes the other attributes, which the organization considers important in addition of support for the target release platforms.

The category **Tool-related development needs** describe the most important development needs for the technical infrastructure in the organization. A development need may also origin from expected or needed changes in the near future, and it is not necessarily related to any of the existing solutions.

The category **Applied design methods** lists the most important approaches the organization has in the design of a new product or a feature for an existing product. In many organizations the tool selection is influenced by the possibility to do interactive design, rapid prototyping, with the selected tools.

The final category **Most used knowledge transfer methods** lists the most applied tools the organization transfers knowledge between the stakeholders. This category lists the methods the interviewees found most useful, either by providing task-critical information or enabling the development team to exchange ideas.

### 4.2 Observations

Observations are based on the codifications and the categorization of the strong and frequent codes. Generally these observations were selected to either explain the phenomena present in all organizations, or to explain the differences.

#### 4.2.1 Generally the organizations consider that their technical infrastructure is on an acceptable level.

In general the most important finding of the study was that most of the organizations were content with their technical infrastructure or at least considered it to be acceptable. This observation was in stark contrast to earlier studies such as [2, 12, 13] where many problems were reported. Overall, the organizations identified only a handful of possible development needs in their technical infrastructure.

"For future projects, it would be nice to have a tool in which you could leave post-it notes to the level while walking there. It is really hard to understand from a spreadsheet that which one of the hundred palm trees in the level was misplaced or of wrong size." – Case A, Developer

In the organizations that named actual needs for their existing tools, the most common problem was bugs in the tools causing minor loss of work at random intervals, or simple usability issues. For example Case D considered their existing source code editor to be prone to unstable releases and missing some features.

"Sometimes there is a problem with bugs." “More elaborate editor would be nice.” – Case D, Developer

#### 4.2.2 Game engine and other complex systems are usually based on third-party solutions as they are considered difficult and expensive to develop independently.

In three case organizations (B, E, G) the organization used their own engine, or had at least until the previous project used their own engine. The need to acquire a third party system was evident based on the development needs and comments regarding the state of the product. For example, Case B had just recently acquired a third party game engine, and considered it to be a major improvement to their technical infrastructure.

"Well, [the new engine] is a considerable improvement over our previous solution... there still are some bugs, it might crash every now and then." – Case B, Project manager

Similarly, Case E had used their own engine in all of their previous products, and was going to start using a third party engine on their next project. Overall, in all organizations except in Case G there was a definite need for a third party game engine, as the amount of work and size of the engine that is capable of handling all of the requirements is rather large.

"A game engine is a huge piece of software. It’s really, I think we are talking, we’re near in the half millions line of code or something like this.” – Case F, Developer

#### 4.2.3 The tool selection principles are not based only on the applied methods or process requirements.

Obviously all of the organizations expected that the new tools should work with the target product platforms, but beyond that the organizations applied varying ways of selecting their tools. In most organizations personal preferences were given at least some credit. In Case F, the organization selected their new tools based on their prior experiences, allowing teams to use the tools they found most suitable for the task.

"Basically we select based on our own, and prior, experiences the ones we consider the most suitable for the task.” – Case F, Project Manager

Similarly, Case G, the only organization not applying a third party game engine in the near future, considered that the teams and people within teams could basically use whatever they wanted as long as it did not break the compatibility with the existing parts. Case E also mentioned that the programmers were allowed to choose their tools quite liberally, as long as they supported the applied infrastructure.

"Everyone can choose their own tools and it's totally fine as we do not compile the environment."– Case G, Project Manager

Besides personal preferences and compatibility, also some other priorities were mentioned. For example cases B and D mentioned that they based their infrastructure on portability. Cases A and G also mentioned that they selected tools in a way that would allow them to find suitable people more easily.

"When we start to need the effects and such, it will be the problem of the sound technicians to figure out what to use.” – Case G, Project manager

Interestingly, only Case B actually mentioned price when asked for the reason for their tool selections. Their selection of a new game engine was based largely on the costs related to adopting and using the engine in a commercial project.

#### 4.2.4 Some form of prototyping is commonly applied and it is important that the technical infrastructure supports fast prototyping.

In almost all organizations prototyping as the design approach was discussed. In the larger case organizations (A and C) the prototyping was actually a large part of the design work, even to the extent that the new game concepts were explored with actually working prototypes, constructed from the left-over or recycled assets from the earlier products. In Case C, this was also their...
main development concern; to enable easier and more fluent usability for their prototyping-tools. In Case D, the design work was done to a large degree with the same tools as the actual development work.

“We basically use the same tools to design than we use for development” – Case D, Developer

Similarly to prototyping, Case B and Case D also applied extensively screen mockups in their design. These organizations designed the user interface elements and crude usability by first creating “fake screenshots” of their games, in some cases extending the design to a playable prototype, similarly as in cases A or C.

“The user interface is mocked up with pen and paper and [graphics tools]… Prototype is made directly with the [game engine], it really is one of the fastest ways to do it.” – Case D, Designer

The most commonly applied design approach was to create design documents. In these documents the main functionalities, the basic story and the core properties are listed so that all of the stakeholders can understand the fundamentals of the project. In some cases, this document is extended during the project to cover more detailed designs, but in some organizations such as Case D the document was not really extended, simply updated to include the most relevant information.

“The original design document had very much different things. It is quite loose definition for the game, and in fact has changed during this project in every way, even some core mechanics.” – Case D, Developer

5. DISCUSSION

The conclusion of all of the observations is that the studied organizations have quite suitable technical solutions for their game software development. In most organizations the technical problems and identified enhancement needs were relatively minor or related to the future needs. Obviously, several organizations in this study are startups or at least relatively new organizations, and thus it can be assumed that the technical framework is also relatively new, and originally designed for the first products the organization is currently developing. However, even in the larger established organizations no major technical improvement needs were observed. In addition, a list of the most important tools used and mentioned by the interviewees is available in Table 4. Several game organizations buy large amounts of assets to their products, keeping only the core competence such as design and main development inside the organization. In our study many organizations used this approach, as only one organization strove to do everything themselves. In many cases, the relatively small size of the organization was seen as positive factor: many interviewees were pleased with their informal and unstructured approach to the software development. The small organization size and unstructured process also affected the tool selection. The programmers could, especially in the smaller organizations, select their own tools. As for expectations and needs, it seems that game developing companies expect their tools to allow easy prototyping and ability to design while implementing; large parts of the design is done with same tools as the actual development work.

Overall, the organizations felt that the methods and tools they used worked well for them. Most common of the few mentioned issues were related to amount of bugs, or user interface design. Some organizations also considered that if they would have to increase the number of people in the project, they would probably have to adopt some sort of formalized project management system and possibly adopt more systematic development process model. Development support tools such as version control and file sharing services were used in all organizations to support working, but in smaller organizations the management tools were not used or they were abandoned after short experiments. Generally the mindset was that everyone knew what other team members were doing, so having an extra management system was unnecessary.

Table 3: Categories and codified observation case-by-case

<table>
<thead>
<tr>
<th>Activity</th>
<th>Preproduction, design phase</th>
<th>Production, implementation phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design, prototyping tools</td>
<td>Documentation tools, Doc. distribution, graphics software, development tools, game engine</td>
<td>Development tools, game engine, graphics software</td>
</tr>
<tr>
<td>Implementation, development tools</td>
<td>Graphics software, development tools, game engine, cloud services</td>
<td>Game engine, development tools, graphics software, audio software</td>
</tr>
<tr>
<td>Project support tools</td>
<td>Documentation, cloud services, file sharing service</td>
<td>Version control, cloud services, file sharing service, ticket systems</td>
</tr>
</tbody>
</table>
just added workload. In the larger organizations the used tools reflected more structured processes and the development tools were more unified. In the large organizations, the overall process was also more mature.

The case organizations did not apply self-made tools to a large degree. In fact, the case organizations were immigrating away from self-build tools towards third party systems. Based on results it seems that there are game industry-focused tools available, applicable even at the earliest startup level. The observation concerning the project management tools and techniques by Blow [2] is still valid; in most of the smaller case organizations the knowledge transfer and management still relied on personal communications instead of any management system.

There are threats that should be acknowledged when addressing the validity of this type of qualitative research [7]. For example, reliability and validity in qualitative research are not the same as in quantitative research, so they should be addressed and explained in more detail to put this study into a context [7]. In this study these risks have been taken into account when planning and implementing the study with several cautionary actions. For example, the study questionnaires were designed by seven researchers to avoid personal bias and the data collection interviews were conducted by six researchers working mostly in pairs and the analysis process for this paper was conducted by three researchers working separately on the collected data set.

All target organizations were small and medium sized game development organizations, located in Southeastern Finland. This can also be considered a threat, as some underlying phenomena might have been caused by the location. However, this bias was addressed by selecting the organizations to represent different release platforms, different business maturities (startup, recent startup, established developer) and different production sizes. However, in any case, qualitative study results are always context-sensitive. Outside the scope of the study they should be regarded as recommendations or considerations.

6. CONCLUSIONS

In this paper we presented the results of our qualitative study regarding seven game industry software developers and their development tool infrastructure. The objective of the study was to identify the expectations and requirements the game developing organizations have, the problems associated with design and development tools and the possible needs for improvement.

The results indicate that the game developing organizations are relatively pleased with their existing solutions. Most of the interviewed organizations use complex third party components such as game engines, allowing the development team to focus on the game core functionalities. The applied software tools are also an important part of the product design, as prototyping is the common design approach. Overall, the ability to make changes to the functionality of the developed product is an important factor in the tool selection. Quite surprisingly the price of the tools was not considered very important, even in the startup organizations.

The results of this study can be used to help new game developers to focus their resources on the important tools and find working solutions for their technical infrastructure. In the future research, further studies on how the tools are applied, how game developing organizations work and how satisfied the traditional software engineering startups are with their technical infrastructure would be useful for establishing context for the results presented in this paper.

7. ACKNOWLEDGEMENTS

This study was supported by the European Union Regional Development Fund project ”Kaakon Peliklusteri”, administered by the Council of Southern Karelia, Finland.

We would also like to thank all the interviewees and the project partners, Cursor and Innovire-research team at the LUT Kouvolan.

8. REFERENCES