Technical Report

A Survey on a State of the Practice in Video Game Development

Jürgen Musil[1]
Angelika Schweda[1]
Dietmar Winkler[2]
Stefan Biffl[2]

Institute of Software Technology and Interactive Systems
Favoritenstr. 9-11, A-1040 Vienna, Austria

[1]*{jmusil, angelika} @computer.org*
[2]*{dietmar.winkler, Stefan.biffl} @tuwien.ac.at*

Technical Report No. IFS-QSE 10/04
Issued: March 2010

A Survey on the State of the Practice in Video Game Software Development

Juergen Musil[1], Angelika Schweda[1], Dietmar Winkler[2], Stefan Biffl[2]
Institute of Software Technology and Interactive Systems
Vienna University of Technology
[1]{jmusil, angelika}@computer.org, [2]{Dietmar.Winkler, Stefan.Biffl}@tuwien.ac.at

Abstract

Video Game Software Development is a promising area of empirical research because our first observations in industry environment identified a lack of a systematic process and method support and rarely conducted/documented studies. Nevertheless, video games - specific types of software products - focus strongly on user interface and game design. Thus, engineering processes, methods for game construction and verification/validation, and best-practices, derived from traditional software engineering, might be applicable in context of video game development. We selected the Austrian games industry as a manageable and promising starting point for systematically capturing the state-of-the practice in Video game development. In this paper we present the survey design and report on the first results of a national survey in the Austrian games industry. The results of the survey showed that the Austrian games industry is organized in a set of small and young studios with the trend to ad-hoc and flexible development processes and limitations in systematic method support.

1 Introduction

Video game software development (VGSD) has maturated within only five decades from the hobby of talented computer scientists into a billion dollar industry that today easily outperforms the movies industry in gross revenue. Top game software projects have become artistic, technological and financial challenges with development costs ranging from few million up to 100 million dollars[1]. According to the recent PricewaterhouseCoopers media forecast[2], the video game business is predicted a constant annual growth through 2013, seeing video games on top in home entertainment spending. It seems that such results draw a promising future for digital games, the development and production practices of the games industry are far from the professional level that sales figures may suggest. Game developers have to face workflow integration problems, difficulties in requirement elicitation and workload fluctuations due to a lack of verified domain tailord processes and techniques. Although the video games industry seems rather turbulent, it presents itself as an interesting enviroment for empricial software engineering that varies significantly from traditional software domains. Video games are interactive entertaiment products of high system complexity that are built upon multiple computer science disciplines including artificial intelligence, computer graphics, distributed systems and human computer interaction. Their production is time critical and involves multiple heterogenious disciplines including experts from non-engineering domains (e.g. arts, design, animation, creative writing). Because games premier purpose is amusement, validation of their entertaining impact relies on qualitative and quantitvative methods[3]. A good example demonstrating that software engineering can indeed learn things from video game development is e.g. in the area of rapid development of new software products [25].

At the example of a state of the practice survey in the Austrian games industry, we therefore seek to investigate if empirical/evidence-based software engineering can be of assistance in providing solutions for problems in VGSD and in identifying similarities and differences between game and non-game software development. Furthermore, the results can (a) support video game producers in introducing lessons learn- ed from traditional soft-

[1]Digital Battle. http://www.digitalbattle.com/2010/02/20/top-10-most-expensive-video-games-budgets-ever (last visited 03/01/2010).

[2]PricewaterhouseCoopers. *Global entertainment and media outlook 2009-2013*. PricewaterhouseCoopers LLP. 2009.

[3]Microsoft Game Studios, Games User Research Group. http://mgsuserresearch.com/publications/ (last visited 03/01/2010).

ware engineering to increase process and product quality and (b) can be the starting point for a more detailed analysis of video game development practice in a larger context. We are interested if conflict areas in processes and techniques in VGSD could be found that would be possible candidates for techniques from existing best-practice software engineering. Also we seek to retrieve hands-on information about games industry-specific problems called "feature creep" and "crunch time" and their causes.
The remainder of this paper is structured as follows: Section 2 presents related work on video game development and best practice software engineering approaches. Section 3 illustrates the research issues with focus on best-practices in video game development. We described the study design of the survey in section 4 and provide initial results on the survey in section 5. Section 6 presents the discussion. Finally, section 7 concludes and identifies next steps for future work.

2 Related Prior Work

This section summarizes related work on VGSD and best-practices software engineering approaches with a particular focus on: (1) Examining the state of empirical studies in game software development processes. (2) Identifying major challenges in game software development and existing conflict areas. (3) Discussing the contemporary importance of best-practice software engineering.

2.1 Existing Empirical Studies

We have reviewed existing literature with particular attention on empirical studies of game development practices and processes. Unexpectedly, we have identified a significant lack of development process-focused publications and empirical studies, after investigating major computer science-related academic platforms (ACM, IEEE, Springer) as well as game industry/game studies related portals (DIGRA, Game Studies, Gamasutra). The only current study among game studios that is methodically comparable with our presented work is the *State of Game Development 2010 Survey*[4] from Think Services. The difference between our research and the Think Services survey is that our survey aims to cover the Austrian game studios and to recheck results from existing studies. Whereby the Think Services survey has been conducted among clients of Think Service products and questions aim predominantly demographics, hardware/software tool usage, project budgeting and purchasing behavior. A survey [16] dealing exclusively with game developer demographics has been conducted by the International Game Developers Association (IGDA) in 2005.
Since there is no widely validated body of knowledge in VGSD available to date, we therefore also propose to follow the five steps of evidence-based software engineering according to [10, 20]:

1. Converting the need for information into an answerable question.
2. Tracking down the best evidence with which to answer the question.
3. Critically appraising that evidence for its validity, impact, and applicability.
4. Integrating the critical appraisal with software engineering expertise and stakeholders' values.
5. Evaluating effectiveness and efficiency in executing steps 1-4 and seeking ways to improve these.

2.2 Video game development

Peltoniemi [26] evaluated the maturation process of the game sector and concludes that the game development sector after decades is still less developed than the automobile industry in its most turbulent years of industry formation. One reason is that video game development has become increasingly challenging over the last decade [9, 12] with global competition of creative concepts and a strong increase in systems complexity [15]. Games industry veteran Jonathan Blow has identified the following obstacles in current video game development [9]: (a) workflow problems in programming, testing and content production (e.g. 3D modeling, game level design), (b) problems because of the concurrent development for multiple target platforms (e.g. PC, Playstation 3), (c) integration problems due to a lack of domain-specific software patterns and a dependence on third-party components/game engines in order to maintain project timeliness, (d) project risks from trade-offs between game design and product-value based considerations that Blow summarizes with the questions "how will this never-implemented feature feel to the end user? Is it going to be worth [...] to implement it?" [9].
Petrillo et al. [27] note that many problems of the software industry occur in the electronic games industry as well. In their postmortem analysis Petrillo et al. identified, besides technological problems that have been discussed by Blow [9], also

[4]Game Developer Research. *State of Game Development 2009-2010 Survey.* UBM Think Services. San Francisco, CA, 2010.

scheduling problems and problems of scope:
Scheduling problems occur in the form of classical project delays and crunch time, but are also often contributory caused by technological problems [27]. *Crunch time* is a video games industry term for phases of extensive overtime work lasting from one or two weeks up to some months in the worst case. Crunch time is regarded as a serious and enduring problem by developers[5] that has led to lawsuits against large game companies [2]. The most recent incident[6] has again confirmed Peltoniemi's theory about the video games industry's maturation level [26] and nurtures assumptions about deep-rooted production process difficulties.
Problems of scope occur in the form of unrealistically large project size, design problems, the cutting of features and a behavior that is called *feature creep* [15, 27]. Chandler [12] notes on feature creep that it "occurs if additional features are added without adjusting the other project variables (time, resources and quality) to accommodate the additional work". Frequent and late changing requirements hinder systemic approaches (V-Model XT) and foster a trend to agile game development (e.g. Scrum) instead.
Project scope problems have also been confirmed by Callele et al. [11] who identified the following areas where VGSD benefits from requirement engineering and project management best-practices: *"(1) communication between stakeholders of disparate background, (2) remaining focused on the goal and resisting feature creep, (3) influence of prior work (e.g. building a new game on top of an existing game), (4) media and technology interaction and integration, (5) the importance of non-functional requirements, and (6) gameplay requirements"* [11]. Callele et al. note that the domination of non-functional-requirements and especially the category of gameplay requirements are characteristic for the game software domain and that these aspects require further investigation due to a lack of foundation research [11].

Another inherent challenge is heterogeneity of disciplines (e.g. arts, engineering) that collaboratively practice game development and arising integration conflicts [15]. On the example of the game title *Dragon Age: Origins* we demonstrate the semantic ambiguity by the usage of the word *design*: software system design, user interface design, game design, art direction, animation and cinematic design, audio/voice-over direction. Although the previous list is not exhaustive it still gives the impression that it resembles more activities required for a hollywood movie, it is a fact that video games have arrived manufacturing complexity and budgeting of even such. The significant increase in production quality over the previous years becomes particularly apparent when comparing material of the movie *Final Fantasy: The Spirits Within* and the state-of-the-art game title *Final Fantasy XIII.* Peltoniemi [26] concludes that what makes the games industry different is the creation of non-utilitarian products and proposes the classification of the game sector in the category of cultural and creative industries (e.g. movies, music) that have the following key economic characteristics: monopolistic competition and horizontal differentiation, hits and misses, increasing returns, gatekeepers, taste formation and experience goods, skewed labor markets, majors and independents [26]. The circumstance that the games industry is derived from the software industry but located in another industry family that produces a different type of goods hinders direct adoption of proven, established software development approaches and is a main reason for many misunderstandings between these professional domains. Concluding, the key challenges for empirical research in video game development can be summarized as:

1. Integration problems in workflow due to development across heterogeneous disciplines demand tailored process support for arts and engineering domains.

2. Requirement elicitation problems because of an emphasis on non-functional requirements, "gameplay" requirements and a dominance of audio-visual, narrative and interactive aesthetics in the end product.

3. Identification of methods that address domain-specific risk assessment, value-based consideration and verification and validation.

4. Challenges for product family engineering, since current installments of video games series are concurrently developed for multiple (heterogeneous) target platforms and have hard deadlines (e.g. Christmas season).

Empirical software engineering can play an important role in identifying practices and processes that address these challenges.

[5]Grant, C. Epic's Mike Capps responds to accusations of 'exploitative' working conditions. In *Joystiq*. Weblogs Inc. Network. 22. April, 2009. Available at http://www.joystiq.com (last visited 03/01/2010).

[6]Rockstar Spouse. Wives of Rockstar San Diego employees have collected themselves. In *Gamasutra*. United Business Media LLC. 7. January, 2010. Available at http://www.gamasutra.com (last visited 03/01/2010).

2.3 Best-practice software engineering

The main goal of business IT software development, e.g. database driven systems and web applications, is the construction of high-quality software products [1, 29]. In contrast to game development, business IT development focuses on databases, business logic, and user interfaces enabling the interaction with underlying systems. Nevertheless, the user interface is predominantly limited to a functional behavior.
Traditional software engineering best-practices include (a) software processes, (b) constructive approaches, and (c) analytical approaches for verification and validation purposes. Software processes provide sequences of steps for project planning, monitoring and control. Traditional software engineering processes, e.g. V-Model[7] and Rational Unified Process [24] focus on separated sequences of steps for project planning with limitations in addressing frequent changing requirements. Flexible and agile approaches, e.g. eXtreme programming and Scrum, support frequent changing requirements by providing a flexible structure by providing small and high-efficient teams [5, 28].
Nevertheless, methods are necessary to (a) construct valuable products and (b) to verify and validate deliverables. Products can be based on models [7] and tests [4] within an integrated development environment. Early definition and execution of test cases based on models enables continuous integration strategies (CI&T) [14] and can enable frequent test runs, early availability of components and systems functions and foster frequent changing requirements during project duration. Thus, CI&T approaches are promising approaches for VGSD because of high flexible process and method support. Additionally, introducing best-practice software engineering approaches can help improving game development practices. Our research showed that there is little research regarding game development practices.

As a reaction on the absence of related empirical work and to find out if empirical software engineering can be helpful in investigating the rather unknown domain of VGSD, we consider it necessary to extend the existing knowledge by collecting hands-on information about the state of the practice at the example of the Austrian games industry.

3 Research Issues

Little research has done to capture game development practices. Thus we see the need to capture best-practices in the video game development practices in an systematic survey in Austria. Based on the related work, we have set our focus on workflow integration problems and identified a set of research questions with focus on the state of the practice in the Austrian games industry.

RQ1 - Studio distribution. Since there is no existing information about the distribution of game studios available, we need basic data about studio demographics. Depending if the Austrian games industry is dominated by major or independent studios [26] or a mixture of both, different further research steps would need to be taken.

RQ2 - Process Support and Method Application We are interested to see how far development processes are matured and to what extend agile processes and techniques like automated testing or continuos integration one applied. Berner et al [6] show that test automation is still inappropriately handled in traditional software industry and we expect a worse situation in the games industry. We are also interested to see how far automation is used with respect to development phases and overall studio size. We further assume that user-interaction-focused techniques like interaction sketching tooling and tailored scripting languages could be of particular interest for the studios.

RQ3 - Feature creep and crunch time affliction. Petrillo's post-mortem study [27] sets crunch time affliction with 45% of the examined (major) studios (average team size of 22). We assume a higher percentage of studios suffering under low to mediocre pressure of crunch time, since: (a) the majority of examined studios are seasoned and rely on established, streamlined production processes, and (b) depending on the game type we expect different affliction rates.

4 Study Design

This section describes the basic setting of an online survey to capture the state of the practice in the Austrian games industry. Our study design is based on the study guidelines of Kitchenham et al. [23], Wohlin et al. [30] and the reporting guidelines of Jedlitschka et al. [17].

4.1 Survey process

The applied survey process is sequential and separated into (1) preparation phase, (2) execution phase and (3) data analysis and evaluation phase. A key requirement that has to be guaranteed during the whole process is participants' anonymity and confidentiality of answers.

[7]http://www.v-model-xt.de (last visited 3/3/2010).

1. Preparation phase
The questionnaire design is based on the Goal-Question-Metric approach proposed by Basili et al. [3]. Figure 1 shows the study design where the survey metrics are derived from the research questions. The concrete survey questions are structured with regards to the developed metrics. After reviews by game designers and after redesign, the questionnaire is published via Google Docs.

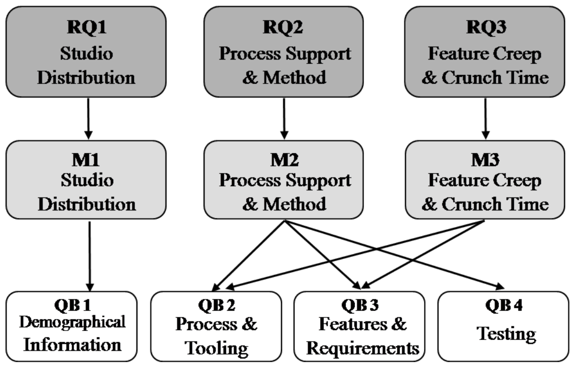

Figure 1: Survey design based on GQM-approach.

2. Execution phase
On October 1st 2009 the participating game studios have been invited via e-mail to the survey and informed about its purpose and benefits. Reminder e-mails are sent to the studios to invite them to complete the survey, two weeks after the initial e-mail and three days before the closing at the end of October.

3. Data analysis and evaluation
After closing the survey we start the analysis phase and evaluate the responses. In a first step, the key characteristics are identified using mean and standard deviation. In a second step we analyze the collected data on the basis of the defined metrics and research questions.

Metrics
This section defines the metrics that we have developed from the research questions and that are used in the current study.

M1 - Studio distribution. Since game studios are part of creative industries [26] we distinguish between a major and independent studio type and therefore treat their data separately. Independent studios are categorized as small and self-funded whereby major studios have a larger staff size and are mainly publisher-dependent. As key studio characteristics we have defined staff size, average development time, the number of released original games and the target platforms.

M2 - Process Support and Method Application. Processes are grouped into a flexible (e.g. Scrum), traditional (e.g. PSP) and unstructured (e.g. greedy approach) type. Tooling is composed from the domains of software engineering and interaction design. Correlations between processes and tools provide an estimate about the similarity of software development in VGSD and traditional software engineering.

M3 - Feature creep and crunch time affliction. The incidence of feature creep and crunch time shows how strong the industry is affected. Correlation of both issues could be an indication for additional causes. In this case the data should be analyzed with respect to characteristic patterns.

4.2 Survey Material

The survey is executed in form of an online-questionnaire which is hosted via Google. Participants can complete and submit the questionnaire directly in their web browser and submissions are stored anonymously in an online spread sheet. The questionnaire is arranged in a way, so that question formulations and ordering do not influence the respondent's answers [18]. Once the questionnaire was finished, the draft has been provided for review to a group of industry experts. Moreover the final draft has been reviewed by two Austrian game designers. The questionnaire is divided into four question blocks: (1) demographical information, (2) processes & tooling, (3) features, requirements, and (4) testing.

Section	Questions	Share
Demographical Information	12	48%
Processes & Tooling	6	24%
Features, Requirements	4	16%
Testing	3	12%
Total:	25	100%

Table 1: Distribution of topics and survey questions.

Kitchenham et al. recommend standardized response formats [22], since open questions are more difficult to analyze. We completely omit open questions and only use closed questions instead. An exception is the "Other" field at some questions where the respondent can provide an individual answer. The questionnaire consists of a mixture of three different types of questions. We use Likert scale of an ordinal scale between 1-6 to measure the level of agreement or disagreement to a statement as well as multiple/single choice answers.

4.3 Participants

Survey participants have been selected among all game studios in Austria. To make sure that as many studios as possible are included, a list of prospective survey candidates has been provided for review during a regular game developer meeting in Vienna. 20 studios could be identified and have been invited to participate in the survey. The studios have been informed via e-mail to the studio's technical lead explaining purpose and benefit of the survey. The e-mail included a direct link to the questionnaire.

4.4 Data analysis process

Before starting with the analysis process it is necessary to check data validity. Kitchenham et al. [19] recommend first to vet the responses for consistency and completeness including checks if all questions are answered correctly so that they can be analyzed with regards to the defined metrics. To be able to interpret the collected data, the following steps are defined:

1. A basic evaluation is performed using descriptive statistics. For a brief overview, we present the most interesting aspects of the data set and show central tendencies and dispersions using statistics for numerical values like total number of respondents (N), mean and standard deviation.

2. The data are analyzed with respect to the research issues. We use histograms to visualize distribution density of variables and scatter plots for assessing dependencies between variables. With scatter plots clusters should be identified and possible correlations observed.

4.5 Threats to validity

This section discusses potential validity threats to this study and how they will be mitigated [23, 30].

Internal Validity. To ensure internal validity the following measures have been taken:

- To guarantee objective evaluation we reduce bias by applying an appropriate level of blinding [23] and by using blind analysis, so that researchers do not adulterate analysis by anticipating results. In order to prevent invalid results because of mixing major and independent studio data [12] it is important to analyze these types separated.
- The design of the demographic-related questions is based on the experiences from the IGDA game developer demographics survey [16].
- To ensure industrial relevance of questions we have conducted reviews [18] of the survey's content by experts from the national games industry. The expert feedback was used to counter-check the consistency and integrity of the questionnaire.

External Validity. To ensure external validity the following measures have been taken:

- Participants were selected among all known game studios that we collect in a list which is extended by reviews by the national community.
- The questionnaire is engineering-focused and targets the studio's lead programmer/developer.
- The results might suffer from the survey's nonresponse rate [21]. We minimize the possibility of a high non-response rate by sending reminder e-mails and by particularly promoting the questionnaire in the regional community.

5 Results

After closing the questionnaire the data are checked for consistency and completeness [19]. The survey's response rate is 65% that is 13 (valid N) of 20 invited studios. Table 2 summarizes the main features of the collected data as a basic evaluation. Based on the mean the dominant studio type can be identified as independent with a staff size of 1-4. The average duration of a project iteration is 2-4 weeks. The highest standard deviation has the average development time with a mean of 1 1/2 years. On a scale of 1 (never) to 6 (every time): (a) there is a mean of 3.5 to completion on time with high standard deviation, (b) there is a mean of 3.2 to crunch time affliction with some standard deviation and (c) there is a mean of 3.5 to feature creep occurrence with a little bit standard deviation. The results in the following are ordered and grouped according to the previously presented metrics.

Variable	**Mean**	**STDV**
studio type	1.9	0.38
staff size	1.6	0.87
feature creep	3.5	1.05
project iteration length	2.0	1.17
crunch time	3.2	1.17
completed on time	3.5	1.45
average development time	3.0	1.96

Table 2: Descriptive statistics of key variables.

1. Studio distribution

Austria's games industry is still in its infancy and the results emphasize this situation, as the majority (84.6%) of the game sector consists of independent studios and only 15.4% are major studios. Most independent studios have a small staff size of 1-4

(72.7%) which we assume because of a small job market, recruitment difficulties and funding problems. The main platforms for which the independent studios develop are PC (54.5%) and iPhone (45.5%) and the majority has released 2 original games (36.4%). The average development time is uniformly distributed: 1/2 year (36.4%), 1 year (18.2%), > 2 years (18.2%). Short projects with minimal development time are mostly developed, due to low budgets and small studio staff size.

Due to the availability of more budget, major studios also develop for more expensive platforms like Xbox 360 (100%) and PC (100%) and one part of the studios has released 3-4 original games (50%), the other part has released > 10 games (50%). They produce more complex applications with longer development time so their staff size is > 15 (100%). The average development time per game of the major studios is 1-2 years (100%). Table 3 shows a short comparison of the demographic results depending on the studio type.

	Independent (84.6%)	**Major** (15.4%)
staff size	1-4	> 15
development time	1/2 year	1-2 years
released games	2	3-4, > 10
platforms	PC, iPhone	Xbox 360, PC

Table 3: Summary of studio results.

2. Process Support and Method Application

The 9 process methods that were available for selection in the questionnaire are separated into the groups of (a) flexible[8], (b) traditional[9], (c) unstructured[10] processes. Figure 2 shows that the majority of studios uses flexible processes and Scrum is the most popular process for developing games among independent and major studios (61.5%). The trend to flexible process methods also has an effect on project iterations which have a duration of < 2 weeks in independent studios (45.5%) and 2-4 weeks in major studios (100%).

Figure 3 shows an overview of the tools and technologies that are used to support the development process. All studios use instant messaging, e-mails and collaboration tools to improve their communication. Versioning tools are applied for change management by 76.9% of the Austrian studios. About half of the studios use white boards (61.5%) and interaction sketching tools (46.2%). Some studios also use the advantages of lightweight pro-

[8]Scrum, eXtreme Programming, agile/lean Techniques

[9]Rational Unified Process (RUP), Crystal Clear, Personal/Team Software Process (PSP/TSP), V-Model

[10]Other

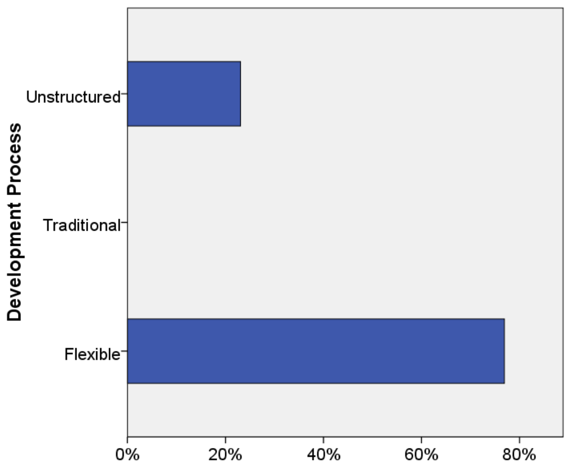

Figure 2: Distribution of development approaches.

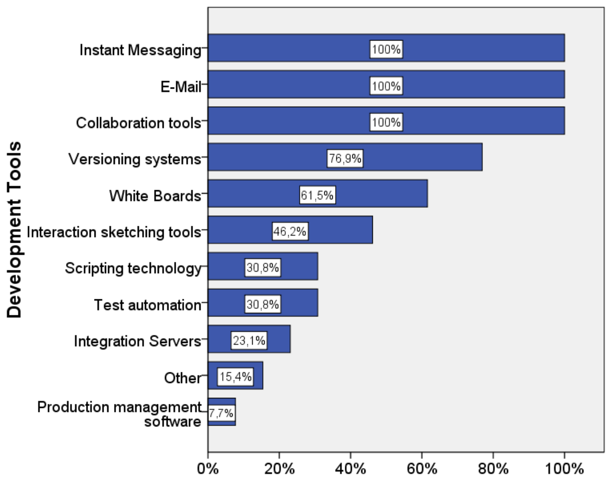

Figure 3: Overview of used tools and technologies.

gramming languages/scripting technologies (e.g. Lua) for rapid programming of game logic (30.8%). Object-oriented modeling tools are rarely used (54.6%) for modeling system design and better understanding the problem domain in the independent studios, and 27.3% never use such tools. On the other side all major studios use modeling tools (100%).

Test automation is used by only 30.8% of the respondent studios. The reasons that only few studios use test automation as well as modeling tools are debatable, e.g. due to development time, staff size or little knowledge about these tools.

All studios use in-house testing to handle testing. Besides all major studios also outsource testing and let their publisher take care of testing. Test phases are run in all studios multiple times during an iteration (53.8%) or at the end of every iteration (38.5%).

Development Tools	Development Process			
	Scrum	XP: Extreme Programming	Agile/Lean Techniques	Other
Instant Messaging	100.0%	100.0%	100.0%	100.0%
E-Mail	100.0%	100.0%	100.0%	100.0%
Collaboration Tools	100.0%	100.0%	100.0%	100.0%
Versioning Systems	87.5%	50.0%	100.0%	66.7%
White Boards	75.0%	100.0%	66.7%	33.3%
Interaction Sketching Tools	50.0%	50.0%	66.7%	66.7%
Scripting Technology	25.0%	50.0%	0.0%	33.3%
Test Automation	37.5%	0.0%	33.3%	33.3%
Integration Servers	37.5%	0.0%	33.3%	0.0%
Other	12.5%	50.0%	33.3%	33.3%
Production Management Software	12.5%	0.0%	0.0%	0.0%

Table 4: Distribution of development tools within process methods (% within development process).

Mostly the used tools depend on the applied process method, but the results show that this assumption is not always valid. Table 4 should amplify this situation. The results demonstrate the usage of tools in a certain process.
Although the majority of studios relies on flexible processes, integration severs (23.1%) and production management software (7.7%) is rarely used to support these processes.
As one result of the tooling-findings, problems such as feature creep, release delays and cutting features often exist. The advantages of the combination of good project organization and agile/lean techniques preserves direction on the one side and allows reacting on ad-hoc changes on the other side.

3. Feature creep and crunch time affliction
77% of the respondent studios are occasionally affected by feature creep, no studio answered that feature creep did not happen and 7.7% responded that feature creep happens every time. Teams are occasionally affected by crunch time (61.6%). Only 7.7% reply to crunch time "never" and nobody answers "every time" which constitutes a good result. Depending on the game type different crunch time affliction rates can be observed. The lowest frequency of crunch time is shown in the development of mobile games. In contrast, during the development of virtual worlds/MMOG crunch time occurs more frequently.
A correlation between crunch time and feature creep exists and is shown in figure 4 where a proportional clustering between both topics can be seen. As software development, VGSD is rather volatile, since only a minority of 30.8% of specified features remain unchanged during development. Results

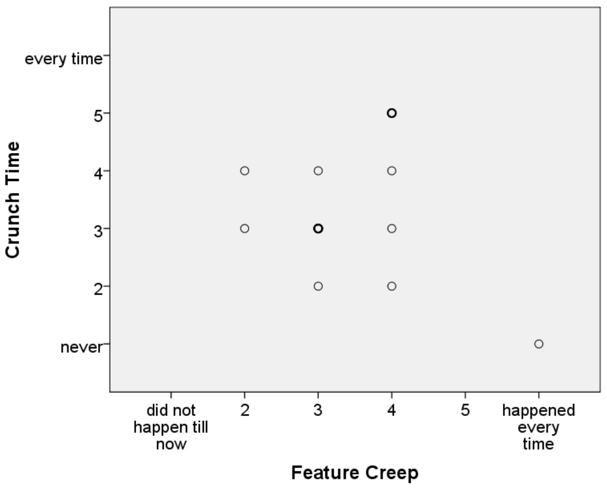

Figure 4: Characteristic grouping of feature creep and crunch time effects (the thickness of circles represents the frequency of response).

show that Austrian game development projects are sometimes completed on time (46.2%) and "every time" by only 7.7%. There are even studios which never complete their projects on time (15.4%).

6 Discussion

In this section we present the conclusions based on the collected data and the results we have found.

1. Studio distribution
Austria's games industry is dominated by independent studios founded within the last years with a small staff size between 1-4 and few games

released (∼2). This outcome is rather surprising, since it is widely spread that major studios would be predominant. With such characteristics, the Austrian games industry can be more compared with the industry of the Netherlands, having a strong independent games industry, than the highly developed major U.S. industry. This is especially problematic, since most game development literature comes from the United States, targeting complexity and problem dimensions that have not yet being reached by the Austrian industry. We assume key reasons for the current studio constellation in funding, recruitment and eduction. Low project budgets due to difficulties in initial project funding and acquisition of risk capital forces studios to maintain small staff sizes, to target platforms with low entry costs (PC, iPhone) and projects with short development time (1/2 - 1 year). Even if a studio would be able to acquire enough capital to develop a larger project, it would face the challenge of finding applicants for "talent"-oriented job positions with an artistic/creative focus (e.g. game designer, animator and graphic designer) on a national level. Our results also show a window of opportunity for game educators. Many current projects are developed within group-sized teams over a time of 1/2 - 1 year with low-cost technology, so such projects would be also well suited to be developed within lab classes or practical courses during a semester. Nevertheless, further research is required to identify if funding and recruitment are indeed the driving factors that slowdowns the prospering of Austria's games industry.

2. Process Support and Method Application

The dominant process methodologies are flexible (Figure 2) followed by an unstructured (greedy) approach. Scrum is used by the majority of the studios and by all major studios. There are several benefits from using this method, e.g. easy to learn, simple to use, clearly defined roles and easy handling of changes during the project. In contrast traditional process methods are not used by any studio. This fact demonstrates that agile development indeed is a key topic of interest for the games industry as well as the software industry. It has also shown that game studios using Scrum or agile/lean techniques trend to relying on test automation, integration servers and scripting technologies. Although the results indicate that game studios use flexible processes, we assume that these processes might be executed for games differently than for the development of non-game software, since games are developed by heterogeneous teams from arts and engineering disciplines [15]. An indication for this assumption can be found in our results that show a trend in favoring interaction sketching tools over the usage of object-oriented modeling tools. Further results about tool usage show that versioning systems are quite common, although a considerably low number uses test automation and integration servers. With regards to test driven development, the games industry shows the same bad situation described by Berner et al. [6] for software industry. In the current situation, we estimate low staff size and project complexity as delimiting factors for applying automation in a wide manner, which could also explain the low amount of production management software usage. In conjunction with Blow's description of development issues [9] we also see a potential for approaches like value-based software engineering [8] and software product lines [13].

3. Feature creep and crunch time affliction

Survey results confirm the existence of crunch time in the Austrian games industry and strengthen the theory of a correlation between feature creep and crunch time (figure 4) as described in literature [12, 27]. The crunch time rate lies in the midfield, which is unexpected, considering that most studios rely on flexible processes which should assist in dealing with workload irregularities. Possible causes for this contradiction could be an inappropriate application of the process or/and a missing domain specific tailoring of a flexible process to the needs of the games industry (heterogeneous teams, product focus on non-functional requirements, etc.). Concerning feature creep, we think that primary reasons could be found in poor requirement elicitation and missing prototyping in early project phases. Another particular serious problem that we have identified is a lack of project timeliness, as only some projects are completed on time. For an industry that makes most of its revenues during specific periods of the year (e.g. Christmas), there would be no more fatal risk than not to be able to make a title available during a certain season and would provide an explanation for the evident occurrence of crunch time. Depending on game type different crunch affliction rates are apparent, with the lowest frequency of crunch time in mobile games development and higher frequency in the development of complex systems like virtual worlds/MMOG.

Closer examination of our results demonstrates that feature creep and crunch time are not problems per se, but rather symptoms of substantial underlying workflow and integration issues. When game developers complain about crunch time, they indirectly question the company's production process. Although the first reactions on claims of excessive crunch time may suggest that the occurrence of this symptom could be primarily reduced on irrational decisions of business executives, the collected data supports the assumptions that a significant proportion goes also on misconceptions in the VGSD:

1. The statement that games are something completely different that cannot be compared with non-game software development is not true. Our results indicate that games are developed with process used in traditional software development and game developers will profit to a certain extend also from existing software engineering best-practices.

2. However, games *are* different in some aspects and their way of development demands domain specific tailoring. It is therefore not enough to just blindly apply the same proven techniques and best-practices from software engineering, but to countercheck if adaptations or complete redesign are necessary in order to serve game developers well.

We are confident that if more developers are aware of these two points, the developers' quality of life as well as the company's productivity can be improved concurrently and not, as it is today often the case, at the cost of each other.

7 Conclusion & Further Work

The presented study demonstrates that empirical and evid-ence-based software engineering can be of great assistance in identifying similarities and differences between the development of game and non-game software. Our results demonstrate that game developers rely primarily on flexible, iterative development processes like Scrum and other agile techniques and favor interaction-focused sketching and user testing instead of object-oriented modeling tools and automated testing. With regards to Austria it can be summarized that it is dominated by small and young independent studios developing small/short-scheduled projects for target platforms with low entry costs. Concerning the industry-specific problems of feature creep and crunch time, it can be concluded that our results indicate that these issues exist rather frequent, but also that they seem to be in fact symptoms of underlying process integration and workflow problems.
Although first results of this survey seem promising it is clear that further investigation is required in order to draw profound general conclusions. We see the following research steps as needed with regards to extending the body of knowledge in VGSD.

- Replication of this survey within the games industry of another country.
- Regular replication of this survey in Austria, as the games industry in general is rather short-cycled and volatile.
- Interviews with developers and studio executives with the goal of refinement and focussing of problem areas.
- Case studies of major and independent studios with respect to agile development processes and the elicitation and tracing of product requirements.

The question if empirical software engineering and the video game software industry could learn from each other can be confirmed with providing mutual benefits for both sides.
Due to the complexities of game systems and heterogeneous disciplines involved, VGSD subliminally relies on empirical approaches by means of empirical process control (e.g. Scrum and variations) and studies in order to improve product quality. Our survey results suggest that some serious workflow problems could probably even be solved by applying concepts from existing software engineering best-prac- tices (e.g. value-based software engineering and software product lines). We are curious to see how team productivity and product quality will improve by applying games industry tailored techniques and processes from empirical/evidence-based research.
On the other side, empirical software engineering research in the domain of video game/entertainment software has the opportunity to investigate large cross-disciplinary projects which rely to a big extend on disciplines outside the engineering domain (e.g. fine arts, film, product design) as well as to examining the behavior of existing techniques within a new industrial field. Investigating the creative software industry with its focus on end-user experience will also lead to new techniques (e.g. instrumenting the concept of game jams for new product development [25]) that will help software developers from other domains to be prepared for the complex projects of tomorrow and by supporting companies in providing better integrated solutions for their customers faster.

8 Acknowledgments

The authors would like to thank the Vienna chapter of the International Game Developers Association, Andrea Schmoll and all who helped reviewing the survey questionnaire.
Thomas Wagner was a member on the initial team

and helped planning, designing and executing the survey, and we thank him for his very valuable contributions.

References

[1] A. Abran, P. Bourque, R. Dupuis, and J. W. Moore. *Guide to the Software Engineering Body of Knowledge.* IEEE Press, Piscataway, NJ, USA, 2004.

[2] T. J. Allen. The passion of the developer: ea_spouse in the h_ouse!: a panel on labor relations and quality of life in the industry. In *Proceedings of the 2006 ACM SIGGRAPH symposium on Videogames*, pages 29–40, Boston, Massachusetts, 2006. ACM.

[3] V. R. Basili, G. Caldiera, and H. D. Rombach. Goal question metric paradigm. In J. J. Marciniak, editor, *Encyclopedia of Software Engineering*, volume 1, pages 528–532. John Wiley & Sons, 1994.

[4] K. Beck. *Test Driven Development: By Example.* Addison-Wesley Professional, 2002.

[5] K. Beck and C. Andres. *Extreme Programming Explained: Embrace Change.* Addison-Wesley Professional, 2nd edition, 2004.

[6] S. Berner, R. Weber, and R. K. Keller. Observations and lessons learned from automated testing. In *Proceedings of the 27th International Conference on Software Engineering*, pages 571–579, St. Louis, MO, USA, 2005. ACM.

[7] S. Beydeda, M. Book, and V. Gruhn, editors. *Model-Driven Software Development.* Springer, 1st edition, 2005.

[8] S. Biffl, A. Aurum, B. Boehm, H. Erdogmus, and P. Grünbacher, editors. *Value-Based Software Engineering.* Springer, 2005.

[9] J. Blow. Game development: Harder than you think. *ACM Queue*, 1(10):28–37, 2004.

[10] D. Budgen, S. Charters, M. Turner, P. Brereton, B. Kitchenham, and S. Linkman. Investigating the applicability of the evidence-based paradigm to software engineering. In *Proceedings of the 2006 international Workshop on Interdisciplinary Software Engineering Research (WISER)*, pages 7–14, Shanghai, China, 2006. ACM.

[11] D. Callele, E. Neufeld, and K. Schneider. Requirements engineering and the creative process in the video game industry. In *Proceedings of the 13th IEEE International Conference on Requirements Engineering*, pages 240–252. IEEE Computer Society, 2005.

[12] H. Chandler. *The Game Production Handbook.* Charles River Media, Boston, 1 edition, 2006.

[13] P. Clements and L. Northrop. *Software Product Lines: Practices and Patterns.* Addison-Wesley Professional, 5th edition, 2007.

[14] P. M. Duvall, S. Matyas, and A. Glover. *Continuous Integration: Improving Software Quality and Reducing Risk.* Addison-Wesley Professional, 2007.

[15] J. P. Flynt and O. Salem. *Software Engineering for Game Developers.* Game Development Series. Thomson Course Technology PTR, Boston, 2005.

[16] IGDA. *Game Developer Demographics Report.* San Francisco, CA, 2005. Available at http://archives.igda.org/diversity/report.php (last visited 03/01/2010).

[17] A. Jedlitschka, M. Ciolkowski, and D. Pfahl. Reporting experiments in software engineering. In F. Shull, J. Singer, and D. I. K. Sjoberg, editors, *Guide to Advanced Empirical Software Engineering*, pages 201–228. Springer London, London, 2007.

[18] B. Kitchenham and S. L. Pfleeger. Principles of survey research part 4: questionnaire evaluation. *SIGSOFT Softw. Eng. Notes*, 27(3):20–23, 2002.

[19] B. Kitchenham and S. L. Pfleeger. Principles of survey research part 6: data analysis. *SIGSOFT Softw. Eng. Notes*, 28(2):24–27, 2003.

[20] B. A. Kitchenham, T. Dyba, and M. Jorgensen. Evidence-based software engineering. In *Proceedings of the 26th International Conference on Software Engineering*, pages 273–281. IEEE Computer Society, 2004.

[21] B. A. Kitchenham and S. L. Pfleeger. Principles of survey research part 2: designing a survey. *SIGSOFT Softw. Eng. Notes*, 27(1):18–20, 2002.

[22] B. A. Kitchenham and S. L. Pfleeger. Principles of survey research: part 3: constructing a survey instrument. *SIGSOFT Softw. Eng. Notes*, 27(2):20–24, 2002.

[23] B. A. Kitchenham, S. L. Pfleeger, L. M. Pickard, P. W. Jones, D. C. Hoaglin, K. E. Emam, and J. Rosenberg. *Preliminary Guidelines for Empirical Research in Software Engineering*. National Research Council Canada, 2001.

[24] P. Kruchten. *The Rational Unified Process: An Introduction*. Addison-Wesley Professional, 3 edition, 2003.

[25] J. Musil, A. Schweda, D. Winkler, and S. Biffl. Synthesized Essence: What Game Jams Teach About Prototyping of New Software Products. In *Proceedings of the 32nd International Conference on Software Engineering*, Cape Town, South Africa, 2010. ACM. (to appear).

[26] M. Peltoniemi. Life-cycle of the games industry: the specificities of creative industries. In *Proceedings of the 12th International Conference on Entertainment and Media in the Ubiquitous Era*, pages 54–58, Tampere, Finland, 2008. ACM.

[27] F. Petrillo, M. Pimenta, F. Trindade, and C. Dietrich. What went wrong? A survey of problems in game development. *Comput. Entertain.*, 7(1):1–22, 2009.

[28] K. Schwaber. *Agile Project Management With Scrum*. Microsoft Press, Redmond, WA, USA, 2004.

[29] I. Sommerville. *Software Engineering*. Addison Wesley, 8 edition, 2006.

[30] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in software engineering: an introduction*. Kluwer Academic Publishers, 2000.