What Game Developers Look for in a New Graduate: Interviews and Surveys at One Game Company

Michael Hewner
Georgia Institute of Technology
School of Interactive Computing
85 5th St. NW
Atlanta, Georgia
hewner@gatech.edu

Mark Guzdial
Georgia Institute of Technology
School of Interactive Computing
85 5th St. NW
Atlanta, Georgia
guzdial@cc.gatech.edu

ABSTRACT

Video game development is an attractive career objective for many computer science students. Colleges are starting degree programs and specializations to serve this interest, but faculty may not have an informed idea of what game programming is like or how to advise students interested in the field. This paper describes the results of interviews with developers, managers, and artists at one company to determine what qualifications were most significant when evaluating college hires for jobs in game development. The qualifications we elicited formed the basis of a company-wide survey.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education—*Curriculum*; K.8.0 [**Personal Computing**]: General—*Games*

General Terms

Human Factors

Keywords

Game curriculum, Game concentration

1. INTRODUCTION

If you want to know how to 'break into' video game programming, you can take your pick of popular press books on the topic. These books advise aspiring game programmers to "explore schools with strong computer science departments" and check out game programming books and web tutorials [3]. For educators in computer science departments looking to advise their students, the question is more difficult—there is very little research into what qualifications the video game industry considers important. As many schools establish video game specializations and degree programs to attract new students [9], this lack of research is becoming more of a problem. Though the fundamentals of computer science is a good starting point, we can better inform curricula for these video game programs if we know what game companies are really looking for.

This paper describes the results of a series of interviews and a survey at one game company. The game company (which asked not to be identified) employs over 100 people and produces primarily 1st-person shooter games for mainstream consoles. The goal of the research was to understand what qualifications game developers look for when evaluating college hires.

2. RELATED WORK

The most commonly referenced resource in designing video game curricula is the International Game Developers Association (IGDA) Curriculum Framework [5] (see [2] for an example of a program that used this). This document was produced by a team of educators and game developers in workshops and other venues. The framework lays out 16 broad topics in their "Game Programming" section. Included in this list are mainstream CS topics like Artificial Intelligence and Networks as well as more game specific areas like Game Engine Design and Design/Technology synthesis. While the curriculum framework is a good start, it is difficult to guess the relative importance of its items.

Monica McGill has published several papers [7][8] on game developer qualifications. Her work is based on content analysis of game developer job postings. This research provides detailed breakdowns about the use of languages and tools in game industry. She also reports that game development companies frequently list communication and interpersonal as important in their job postings.

In the broader area of IT careers, a variety of different studies have attempted to understand the needs of industry (e.g. [6], [10]). These studies employ a similar method: a set of focus groups with industry professionals to identify categories that are used to build a larger scale survey instrument. These studies have found a significant difference between the expectations of academics and industry, with a particular lack of education in project management and coordination activities. Begel and Simon [1] provide a more detailed analysis of this mismatch in their observations of newly hired engineers at Microsoft. They found new developers had the technical skills to succeed, but had difficulty using the resources of their team and integrating into the company culture.

3. METHOD

Our method consisted of two rounds of interviews followed by a larger scale survey. We interviewed a cross-section of the company: developers of varying seniority and department, developer managers and artists. Altogether nine employees participated in the interview process.

The first round of interviews began with the prompt "Say you were going to interview some new college hires and you decided to put together a document about what was important to look for in a new hire. What would be in that document?" The interview would proceed with the interviewer occasionally asking the participant to clarify something that was unclear or asking the participant to elaborate on a general category they had identified. In general participants had a lot of opinions and little prompting was necessary. Near the end of the interview, the participant was invited to look over the interviewer's notes and identify approximately five items they thought were most important in the various characteristics they had identified.

After the first round of interviews we extracted a list of qualifications identified by our participants. We combined commonly mentioned items into single items, but generally included even qualifications just mentioned in passing in our final list. Where possible we used the wording of our participants rather than our own descriptive categories. The result of this was a list of short phrases we felt corresponded to the qualifications that were mentioned in the first round.

The second round of interviews began with each participant from the first round going through our qualifications list and ranking their importance. We instructed the participants to think aloud as they did this exercise, especially with regard to items that were confusing. Oftentimes as part of this process individuals would expand on areas that others had mentioned but had been absent from our first interview. Then we showed participants our notes from their previous interview and asked them to make sure the points they raised were adequately represented in the list. We used the feedback to revise and add to the list before presenting to the next interviewee.

A similar method has been used previously to elicit qualifications for generalized IT jobs (see [6], [10]). These studies generally elicited qualifications using focus groups rather than the two interview process we used. Similar to the focus group, participants had the opportunity to reflect on the written results of others (from the 1st round interviews) and revise their qualifications. Conducting the interviews privately allowed participants to express dissenting opinions more freely: the responses of their co-workers was anonymous, and therefore there was no social pressure to defer to more experienced peers or managers. This allowed us to focus on what might have been a touchy subject in a focus group: whether there was disagreement within the company about what qualifications were most important.

Using interviews instead of focus groups also allowed us to revise the qualification list more frequently during the second round. When one participant found a particular phrasing confusing and suggested a revision, we could revise the item. Then we would check in a later participants think-alouds that they were interpreting the new phrasing as intended.

Once the interviews finished, we used the qualification list we developed to create a survey. To keep the survey to a reasonable length, we could not include every item from our interviews. Qualifications that were ranked as highly important and provoked disagreement or interesting discussion in interviews were included. We removed qualifications that most of our interviewees ranked as less important. Survey respondents were recruited by a company-wide email message. The survey was conducted online. In the survey participants ranked each of elicited qualifications on a Likert-type scale ranging from "Not important" to "Essential, would not hire without good skills in this area". The qualifications were presented in a random order.

4. RESULTS

Table 1 provides a detailed breakdown of the survey results, including the complete list of qualifications we elicited. Thirty-two participants completed the online survey. Twenty-seven identified themselves in a software development role, four said they manage software developers, and one listed himself/herself as a technical artist. Respondents ranged between 2 and 18 years experience in the game industry, with the average being 8.3 years. Sixty-three percent said that they had participated in the interviewing/hiring of a software developer in the last two years. The following section provides more details on the qualifications as they were elicited in our developer interviews.

4.1 Programming

Skill with C++ programming was frequently the first skill participants would mention in their interviews. They emphasized that knowledge of every language feature was not necessary—simple code was considered more efficient and easier to maintain. Knowledge about the STL (C++ Standard Template Library) was not generally ranked as very important. Other languages such as perl, C#, and Lua were in use at the company but not mentioned by the participants in our interviews.

Ranked even higher than C++ in the survey results was data structures. This was something that a few interviewees mentioned in passing during the interviews but did not elaborate on. Unfortunately, it's not clear exactly what aspects of data structures were considered most useful for game programming.

An ambiguous qualification that frequently came up in interviews was sometimes called "problem solving" by participants. The company's interview process involved the interviewee working through tricky algorithm-oriented questions. Several participants said that getting the right solution was not as important as how an interviewee approached the problem:

> The important part is not if you know the answer to the question. It's how you attack a question, if you get frustrated with it. If you just stammer on and go off in some complete random direction. Or if you can just make...'hey this is kind of relevant, this is relevant, and that's what I know.' That's all anybody is really looking for.

Part of the evaluation of these questions focused on how well the interviewee interacted with the interviewer and took guidance. We revised the description of this qualification several times during the second round of interviews. The final version "being able to solve algorithmically challenging problems" was ranked highly in the online survey but does

Table 1: Rankings of each qualification's importance from online survey. [n = 32]

		Not useful	Sometimes useful but not required or evaluated in interviews	Important, has an impact on the hiring decision	Very Important, has a large impact on a hiring decision	Essential, would not hire without good skills in this area
Programming	Proficiency with the C++ language including basic knowledge of features like templating	0.0%	0.0%	19.4%	**51.6%**	29.0%
	Knowledge about data structures	0.0%	3.1%	15.6%	**43.8%**	37.5%
	Being able to solve algorithmically challenging problems	0.0%	0.0%	31.3%	**53.1%**	15.6%
	Debugging and familiarity with debugging tools	0.0%	15.6%	25.0%	**43.8%**	15.6%
	Professional programming experience of any sort (e.g. an internship)	0.0%	12.5%	34.4%	**46.9%**	6.3%
Optimize	Understanding the performance implications of particular language constructs and hardware platforms; familiarity with how to optimize code	0.0%	6.3%	31.3%	**53.1%**	9.4%
	Algorithm analysis using big O (e.g. determining an algorithm is O(n log n))	3.1%	25.0%	**40.6%**	25.0%	6.3%
Design	Ability to build a good object design for a large system and understand the implications	0.0%	6.3%	37.5%	**46.9%**	9.4%
	Willingness to write a "good enough" solution, rather than spending a long time engineering an elegant solution	0.0%	12.5%	**59.4%**	25.0%	3.1%
	Writing clean code	0.0%	0.0%	15.6%	**65.6%**	18.8%
Specializations	Basic familiarity with the implementation of renderers and the graphics pipeline	0.0%	34.4%	**46.9%**	9.4%	9.4
	Linear Algebra	0.0%	12.5%	**40.6%**	25.0%	21.9%
	Understanding how a compiler works and its limitations	0.0%	**43.8%**	**43.8%**	9.4%	3.1%
	Multithreaded Programming	0.0%	**34.4%**	**34.4%**	31.3%	0.0%
	Newtonian physics and how to simulate it	9.4%	**50.0%**	40.6%	0.0%	0.0%
	Assembly language programming	15.6%	**62.5%**	18.8%	3.1%	0.0%
	Network programming	3.1%	**56.3%**	37.5%	3.1%	0.0%
	Deep knowledge in a particular specialization (e.g. AI, Audio)	0.0%	31.3%	**53.1%**	15.6%	0.0%
	Flexibility to work on any part of a game project	0.0%	21.9%	**50.0%**	28.1%	0.0%
People Skills	Ability to work with others and check your ego at the door	0.0%	0.0%	15.6%	9.4%	**75.0%**
	Ability to work with someone in a different part of the organization and understand their requirements	0.0%	6.3%	31.3%	**40.6%**	21.9%
	Being able to communicate clearly to both technical and nontechnical audiences	0.0%	15.6%	31.3%	**37.4%**	15.6%
Game Industry	Enthusiasm for building video games	0.0%	6.3%	28.1%	**37.5%**	28.1%
	Modding games as a hobbyist or other extracurricular game projects	21.9%	**53.1%**	15.6%	9.4%	0.0%
	Willingness to put in extra hours to complete a feature on time	0.0%	0.0%	37.5%	**43.8%**	18.8%
	Knowledge about the game industry	6.3%	**56.3%**	34.4%	3.1%	0.0%
	Having contacts within the gaming industry	34.4%	**56.3%**	3.1%	6.3%	0.0%
	A bachelor's degree in computer science	3.1%	25.0%	**43.8%**	18.8%	9.4%

not express the complex way these questions were evaluated by interviewers.

4.2 Optimization

"Performance doesn't just apply to making a computer program fast...You wouldn't even need to specifically teach performance if you taught people to understand the implications of what they are doing. If people really understood that putting one little include brings in a whole bunch of code..."

Optimization techniques were mentioned nearly as often in our interviews as C++, though they were ranked less highly. A frequent complaint was that students did not understand how the compiler might produce poorly optimized code. The performance optimization the participants talked about generally focused on improving the speed of frequently called functions; this often required a understanding of the hardware, operating system or the compiler. Distributing work correctly across multiple processors also was mentioned as a technique that was becoming more important. Memory optimization was also discussed: optimizations focused on reducing the size of frequently allocated objects and finding ways to limit the number of objects. Being able to optimize C++ code was considered a very important skill although participants varied in how much they expected students out of college to be able to do this.

One optimization technique that received varying response was big-O. This optimization technique did not often come up in interviews. When prompted participants frequently said that while big-O might be useful under certain circumstances they didn't think about it frequently. In the online survey, big-O was one of two qualifications that was rated as both "not useful" and "essential". This may reflect a divide in the game industry as to the usefulness of academic computer science—the other qualifications that similarly spanned all rankings was "a bachelor's degree in Computer Science".

4.3 Design

Being able to develop an object design to solve a particular problem was another commonly mentioned skill. Many interview participants emphasized that a good designer was willing to compromise and build a "good enough" solution rather than wasting time overengineering things the "right" way. A few mentioned that they were particularly wary of overengineering in applicants with advanced CS degrees.

Some interview participants stressed that they considered object oriented design questions to be more appropriate for more senior interviews. What they expected from college students was enough understanding of design to make sense of the codebase and the ability to write clean easy to understand code. The ability to write clean code was ranked more highly by our survey participants than object oriented design—85% ranked writing clean code as "very important" or "essential", compared to 56% for design.

4.4 Specializations

Computer graphics is often considered to be the computer science field closest to game programming, and interview participants did mention that some familiarity with the way the rendering pipeline works could be useful. However, participants generally also agreed that rendering in general was handled by a specialized team and not every programmer needed to be an expert. Experience with linear algebra was also frequently mentioned. Linear algebra was ranked much higher on the survey, with 47% ranking it as essential or very important, compared to 19% for rendering.

Other subfields of computer science were mentioned by interview participants. Compliers, multithreaded programming, physics, and assembly language were all brought up as skills that were useful. However, generally these skills were ranked much lower by survey respondents than programming, design, and optimization. Interview participants also did not agree on whether a deep knowledge of a particular field was useful, or if candidates who could move between all areas of game development were valuable. Participants agreed that when interviewing for a particular specialized team deep specialty knowledge was important—beyond that there seems to be disagreement.

4.5 People Skills

> "Many of us don't have the ability to walk into the room with a complete stranger and start up a conversation. We don't necessarily need that level of skillset. Just somebody who's not intimidated to really get in and figure out the problem rather than just implement what's on the sheet."

Several kinds of people skills where ranked extremely high by the participants—usually higher than C++ and other technical skills. Several participants mentioned that assessing culture fit was what they considered the primary goal of interviews, with technical qualifications being secondary. Technical skills could even be learned on the job.

Of the people skills mentioned in interviews, the skill that was consistently ranked highest was the ability to work on a team without excessive ego. This was the highest ranked skill in the online survey, with 75% of respondents ranking it as "essential" (by contrast, only 29% ranked C++ proficiency as essential). Participants individually emphasized that too much ego and unwillingness to take advice was against the company culture. Since the interviews were only at one company, it is difficult to know whether this concern with ego is shared across the game industry.

Another area frequently ranked as essential was the ability to communicate clearly with coworkers in other departments. Developers were frequently in the position of explaining game behavior or building tools for testers, artists, and game designers. These groups were often quite technically sophisticated themselves and would script or use complex in-house game development software. Participants frequently talked about how a good developer would design a solution that was technically feasible and satisfied the other stakeholders rather than simply coding from a bug report or specification document.

A question we asked most of the participants was "Do you think that most people at the company agree about the kind of game developer candidate that should be hired?" Participants who saw competing viewpoints often identified two camps—those who focused on technical skill and those who focused on culture fit. Those in the culture fit group often commented that the technical aspects of the job was learnable while social aspects of the job were not. They felt that their peers often attached excessive weight to success on algorithm-oriented questions and college degrees.

4.6 The Game Industry

Another highly ranked qualification was enthusiasm for building video games. Video game modding and other extracurricular projects were considered evidence of this enthusiasm—but not everyone agreed how important a consideration this was for an applicant. A few participants also mentioned a willingness to work extra hours when necessary and this was ranked highly in the survey. Long hours are part of the mythos of the gaming industry; at this company people worked longer days and weekend hours during crunch times. The IGDA conducted an industry-wide survey that focused on the topic of long hours [4]; their results showed that long hours during crunch times are widespread.

Two items that were not highly ranked were knowledge about the game industry and having game industry contacts. A few participants mentioned that they felt having contacts was given inappropriate weight in the hiring process, but this does not appear to be a widespread concern given the survey results. Some others commented that although contacts are not important for a college hire, because the game industry is small interviewers often call friends at other game companies to check on experienced candidates.

5. DISCUSSION

Although our goal was to discover results that were as generalizable as possible, our method has some limitations. One obvious limitation is that the interviews and survey occurred only at one game company. Although some of the results we found are similar to other, broader game developer studies [8], it is still difficult to extract what may be part of this particular company's culture from the game industry as a whole. Similar to this, our effort to get the viewpoints of those across the company may not reflect the real hiring process. Some viewpoints may carry more weight than others: this was outside the scope of this study.

Our participants seemed to favor candidates with good C++ programming skills and some experience with object oriented design. Understanding how to write efficient code was also important, which requires abstract topics like data structures, compilers, and multithreaded programming. But in general our participants stressed that students interested in the game industry were better off building strong general coding skills rather than learning deep knowledge of specialized areas like rendering.

Another area almost all our participants agreed on was that students interested in the game industry need to be able to work on a team. Interpersonal skills were ranked as highly as technical skills by almost all our participants. The idea that developers need strong team skills is echoed in McGill's [8] game developer work and other studies on developer qualifications in general [6][10][1].

Of the game-specific qualifications that came up in our interviews, a basic familiarity with rendering and linear algebra was most consistently ranked high. Our participants also looked for candidates who could show they were excited about building games. But overall one of the most interesting results of this study is how similar the qualifications we elicited from our participants were to any industry-oriented computer science curriculum. Participants did not mention things like level design knowledge, experience with particular console platforms, or advanced knowledge of rendering and computer graphics. One reason for this may be because video game companies usually have to recruit from traditional CS programs and are used to having to teach game specific skills on the job. But this is good news for schools looking to recruit students pursing game programming careers: if professors focus on coding skills, efficiency, and team interactions their graduates should be attractive to the game industry.

6. CONCLUSION

There is a lot of mythology about game programmers. Some of it seems to be borne out by our interviews—game companies are looking for C++ programmers who can write efficient code. Some of the mythology seems to discredited—you don't need to have mastered every assembly language instruction on your video card's GPU, nor can you be a egotistical recluse who only communicates in code. Some seems to be half-true–game developers do favor pragmatic programming over theory, but many still value a college degree.

Although other studies have provided broad surveys of industry qualifications (e.g. [5] [8]) interviews provide more nuanced view of what game development companies are looking for and why. The qualifications elicited here also provide a starting point for research on a larger scale. One of the key results is that even at one company building one kind of game, there is disagreement about what qualifications are most important. To understand what skills are important across different kinds of games and different companies industry-wide research is necessary. Video game curricula is already being developed [9] and students who are attracted to game development specializations and degree programs expect their professors to prepare them. With larger scale research into understanding game developer qualifications, hopefully soon we can live up to our students' expectations.

7. ACKNOWLEDGMENTS

The authors sincerely thank our interview and survey participants for taking time to share their experiences with us.

8. REFERENCES

[1] A. Begel and B. Simon. Novice software developers, all over again. In *Proceeding of the fourth international workshop on Computing education research*, pages 3–14, 2008.

[2] L. Ficocelli and D. Gregg. B. sc. computer game development... why not? In *Changing Views*, pages 16–20. DiGRA (Digital Games Research Association), 2005.

[3] A. Gershenfeld, M. Loparco, and C. Barajas. *Game Plan*. St. Martin's Griffin, 1st edition, May 2003.

[4] International Game Developers Association. Quality of life in the game industry. `http://www.igda.org/qol/whitepaper.php\#source`, 2004.

[5] International Game Developers Association. Curriculum framework v. 3.2 beta. `http://www.igda.org/academia/curriculum\_framework.php`, 2008.

[6] D. Lee, E. M. Trauth, and D. Farwell. Critical skills and knowledge requirements of IS professionals. *MIS Quarterly*, 19(3):313–340, 1995.

[7] M. McGill. Critical skills for game developers. In *Proceedings of the 2008 Conference on Future Play*, pages 89–96, 2008.

[8] M. McGill. Weighted game developer qualifications for consideration in curriculum development. In *Proceedings of SIGCSE 2009*, pages 347–351, Chattanooga, TN, USA, 2009. ACM.

[9] B. B. Morrison and J. A. Preston. Engagement: gaming throughout the curriculum. In *Proceedings of SIGCSE 2009*, pages 342–346, Chattanooga, TN, USA, 2009. ACM.

[10] E. M. Trauth, D. W. Farwell, and D. Lee. The IS expectation gap: Industry expectations versus academic preparation. *MIS Quarterly*, pages 293–307, 1993.