Hindawi Publishing Corporation
International Journal of Computer Games Technology
Volume 2012, Article ID 103710, 20 pages
doi:10.1155/2012/103710

Review Article

A Guideline for Game Development-Based Learning: A Literature Review

Bian Wu and Alf Inge Wang

Norwegian University of Science and Technology, 7491 Trondheim, Norway

Correspondence should be addressed to Bian Wu, oscar.wubian@gmail.com

Received 17 July 2012; Revised 13 November 2012; Accepted 24 November 2012

Academic Editor: Zhigang Deng

This study aims at reviewing the published scientific literature on the topics of a game development-based learning (GDBL) method using game development frameworks (GDFs) with the perspective of (a) summarizing a guideline for using GDBL in a curriculum, (b) identifying relevant features of GDFs, and (c) presenting a synthesis of impact factors with empirical evidence on the educational effectiveness of the GDBL method. After systematically going through the available literature on the topic, 34 relevant articles were selected for the final study. We analyzed the articles from three perspectives: (1) pedagogical context and teaching process, (2) selection of GDFs, and (3) evaluation of the GDBL method. The findings from the 34 articles suggest that GDFs have many potential benefits as an aid to teach computer science, software engineering, art design, and other fields and that such GDFs combined with the motivation from games can improve the students' knowledge, skills, attitudes, and behaviors in contrast to the traditional classroom teaching. Furthermore, based on the results of the literature review, we extract a guideline of how to apply the GDBL method in education. The empirical evidence of current findings gives a positive overall picture and can provide a useful reference to educators, practitioners, and researchers in the area of game-based learning.

1. Introduction

Computer games and video games have become very popular in children and adolescents' life and play a prominent role in the culture of young people [1]. Games can now be played everywhere in technology-rich environments equipped with laptops, smart phones, game consoles (mobile and stationary), set-top, boxes and other digital devices. From this phenomenon, it is believed that the intrinsic motivation that young people shows towards games can be combined with educational content and objectives into what Prensky calls "digital game based learning" [2].

Besides of an abundant appearance of games in young students life, game development technology has matured and became more advanced than before [3]. Based on various existing game development software, the whole duty of game development process can be divided into several domains and roles such as game programmers, 3D model creators, game designers, musicians, animators, and play writers. Under this situation, some web resources and game engines can simplify the game development process. For instance, Microsoft's XNA game development kit provides the game loop function to draw and update the game contents, and it also provides convenient game development components to load the different format of graphics, audio, and videos. This makes it possible for students to modify existing games or develop their own new games with or without programming. They can design and implement their own game concepts with these game creation tools, learn the developing skills and relevant knowledge, and accumulate related practical experience.

In this context, not only can a game be used for learning, but also the game development tools be used for studying relevant topics within computer science, software engineering (SE), or game programming through motivating assignments. Generally, games can be integrated in education in three ways [4, 5]. First, games can be used instead of traditional exercises motivating students to put extra effort in doing the exercises and giving the teacher and/or teaching assistants an opportunity to monitor how the students

work with the exercises in real time, for example, [6, 7]. Second, games can be played within lectures to improve the participation and motivation of students, for example, [8, 9]. Third, the students are required to modify or develop a game as a part of a course using a game development framework (GDF) to learn skills within computer science and SE, for example, [10]. And we label this third as game development-based learning (GDBL). And the GDFs denote the toolkits that can be used to develop or modify games, for example, game engine, game editors, or game (simulation) platforms, or even any integrated development environment (IDE), like Visual C++, Eclipse, J2ME, and Android SDK since all of them can be used to build games. This paper focuses on using the GDBL method in education, where GDFs are used in student exercises to learn skills, extending the use of GDFs as a teaching aid. The motivation for teaching through game development is to utilize the students' enthusiasm for games. This GDBL method is not new. The earliest similar application of learning through programming in a game-like environment was in early 1970s. The logo [11], the turtle graphics, is one of the oldest libraries that was used to introduce computing concepts to beginners. The concept was based on a "turtle" that could be moved across a 2D screen with a pen, which could be positioned on or off the screen and, thus, may leave a trace of the turtle's movements. Programming the turtle to draw different patterns can be used to introduce general computing skill, such as procedural operations, iteration, and recursion. Further, in 1987, Micco presented the usage of writing a tic-tac-toe game for learning [12]. After several years of development, we believe that GDBL methods have been improved through the development of technology. Thus, we investigate how GDFs are being used in education through a literature survey and investigate how traditional lectures can become more dynamic, collaborative, and attractive to the students utilizing the current technology rich environment. However, this assertion needs to be further supported by relevant theory, application experiences, evaluation results, and empirical evidence. Nevertheless, to the best of the authors' knowledge, there does not exist any comprehensive literature reviews on the application of the GDBL method so far.

The aim of the study is to review the recently published literature on the use of GDFs in education to

(a) summarize a guideline for how to use GDBL in a curriculum,

(b) identify the features of GDFs related to GDBL,

(c) present a synthesis of impact factors with the empirical evidence on the educational effectiveness of the GDBL method.

The study is unique in that it presents an overview of the recently published literature on the use of GDFs in education, while taking into account both game engines and relevant toolkits to create/modify games or game-like systems (e.g., simulators). The study can provide useful guidance to teachers at different educational levels or areas, as well as to educators, practitioners, and researchers in the areas of game-based education.

The paper is organized as follows. Section 2 describes the method used for carrying out the systematic review of articles, Section 3 presents the results from the literature review, Section 4 extracts a guideline for GDBL according to the existing literature, and finally Section 5 concludes the paper.

2. Method

Informed by the established method of systematic review [13, 14], the review was undertaken in distinct stages: the development of review protocol, the identification of inclusion and exclusion criteria, a search for relevant studies, critical appraisal, data extraction, and synthesis.

2.1. Protocol Development. We developed a protocol for the systematic review by following the guidelines, procedures and policies of the Campbell Collaboration (http://www.campbellcollaboration.org/), the Cochrane Handbook for Systematic Reviews of Interventions [13], the University of York's Centre for Reviews and Dissemination's guidance for those carrying out or commissioning reviews [14], and also refer to reviews on serious game research [15, 16]. This protocol specified the research aim, search strategy, inclusion, exclusion criteria, data extraction, and methods of synthesis.

2.2. Data Source and Search Strategy. For the purpose of the study, a literature search was undertaken in December 2010 in the following international online bibliographic databases: (a) ACM portal, (b) IEEE Xplore, (c) Springer, (d) Science direct. The search string used was ("Game") AND ("Learning" OR "Teaching") AND ("Lecture" OR "Curriculum" OR "Lesson" OR "Course" OR "Exercise"). And "education" was not included in the keyword list since we considered that education was a quite general word and did not help minimize the searching scope. Searches were limited to titles and abstracts of articles published in journals and conference proceedings (some are book chapters), in English, from 2000 and onwards. The latter limitation was posed due to the rapid changes in ICT (Information and Communications Technology) in general, and in computer game development technologies in particular.

2.3. Data Extraction with Inclusion and Exclusion Criteria. Figure 1 shows the complete process of the data extraction. The first step was to identify relevant studies. A number of journal and proceedings articles about GDBL were located during searches in the afore-mentioned databases. The articles were examined and the search resulted in 1155 articles. In step 2, from abstracts of each article, we distinguished learning through game play or game development. And most of the excluded articles were using games directly in the classroom to motivate the students' interest and attendance rate and using game play instead of traditional exercises to study or review the course content. For instance, these were

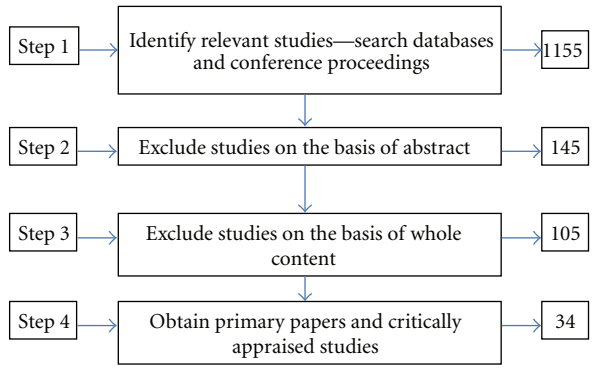

Figure 1: Steps of the study selection process.

articles generally addressing using virtual online multiplayer game environments to provide a collaborative learning style, for example, [17, 18], articles which referred to games used in the classroom to motivate attendance and to review the course knowledge, for example, [8]. In addition, the articles related to the economics terms "game theory" and "business game" used as business terms were also excluded from this category. Besides, we excluded articles that depicted novel game concepts that were not computer or video games but physical game activities without any technology support for the lecture. For instance, the article in [19] used a self-made table card game in SE education. Mainly based on these three criteria, a total of 1010 articles were excluded after this step.

In step 3, the whole content of the articles was checked. The inclusion criteria were further limited to the scope: a case study or several case studies in the article to describe GDBL. In particular, it required (a) a relatively detailed description of the lecture design process. The articles without a detailed description of their teaching design or exercise process made it impossible to validate their process of how to integrate GDFs in lectures or exercises. According to this requirement, posters, tutorial presentations, and some short papers without detailed description on teaching process were excluded since they could not provide valuable data for our research aim and made it impossible to validate the effectiveness of the method, for example, [20–25]. This was also a measure to ensure the inclusion of the high quality literature in the review. (b) Articles using development toolkits in the curriculums but did not aim to develop games were also excluded, for example, [26]. (c) Articles emphasizing on other aspects apart from GDBL were excluded as it was difficult to validate how game development was integrated in class, for example, learning in an interactive e-lab [27]. Similarly, articles that presented the development of an educational game framework but did not mention how it was integrated in a specific curriculum were excluded, for example, [28–31]. (d) Articles, which focused on changing the controller of the software or hardware, but without elements of computer game development, were also excluded, for example, [32, 33]. Most of them focus on creating a robot controller to learn algorithms or changing some component of a robot to learn artificial intelligence (AI). In contrast, we included learning from modifying parts of a simulator to create the game elements or a game-like system, for example, [34, 35]. Finally, a total of 105 articles were remaining after this step.

In step 4, we carefully looked through the remaining articles and compared their topics, methods, teaching process, and evaluation quality from the presentation of their concepts. After the comparison, the following studies criteria were included: (1) articles that had collected data from assignments or scores after using the GDBL method; (2) articles that had questionnaires with quantitative data and interviews or feedback with qualitative data; and (3) detailed discussion of the collected data and conclusion. In addition, diverse and innovative articles were not neglected, in order to show the various ways to integrate GDFs in education. However, articles reporting on the use of hardware tools to create game or game-like system, such as real robot hand [34], Wii remote [36], Microsoft surface [37], and a projector-camera system [38] to support teaching or learning environment were not included. Finally, a total of 34 articles were included in the review. And we believe these articles were sufficient to get a complete guideline to explain how to integrate the GDBL method in the curriculums.

2.4. Synthesis of Findings. A typology to categorize the 34 articles has to be devised. The classification scheme proposed by [39] in their review of the general instructional gaming literature was adopted for the needs of the present study. This scheme, which was also used in [40], defines the following five categories [39]: (a) research (systematic approaches in the study of gaming targeted at explaining, predicting or controlling particular phenomena or variables), (b) theory (articles explaining the basic concepts or aspects or derived outcomes of gaming), (c) reviews (syntheses of articles concerning general or specific aspects of gaming), (d) discussion (articles stating or describing experiences or opinions with no empirical or systematically presented evidence), and (e) development (articles discussing the design or development of games or projects involving gaming).

Specifically, for the categorization of the articles, the following criteria were applied in this study. Articles comprising empirical research related to GDBL were assigned to the "Research" category. Articles comprising theoretical analyses of concepts, aspects, or outcomes of GDBL were placed in the "Theory" category. Articles presenting syntheses of articles concerning GDBL conducted according to explicit methodology were placed in the "Review" category. Articles reporting on opinions and experiences regarding GDFs used in teaching, with no empirical or systematically presented evidence, were assigned to the "Discussion" category. Finally, articles mainly reporting on the design or development of GDFs used in the GDBL method were assigned to the "Development" category. The articles were grouped into these five categories according to their primary focus. Of the 34 articles found after step 4, 20 were placed in the "Research" category, 1 in the "Theory" category, 7 in the "Discussion" category, and 6 in the "Development" category, whereas no articles fit the "Review" category, which highlights the usefulness and originality of the present study.

Like results from other literature reviews on instructional games [40, 41], in this study there were fewer articles in the "Theory" categories than in the "Research," "Discussion," and "Development" categories. This can be explained by the fact that instructional games, including GDBL are a relatively new domain of educational technology.

3. Results

This section presents an overview of the studies of the GDBL method based on the results after step 3 and step 4 in Figure 1.

3.1. Overview of the Study after Step 3. In order to have a complete overview of GDBL, we chose the results from step 3 mainly due to the following: (a) they covered a more complete variation of types of GDFs and contained more information than the 34 articles from step 4. (b) They provided more cases in the diversity of GDFs methods used in teaching, which also presents the potential advantages of using GDF in education. (c) They showed the development tendency of GDF related to other factors (e.g., times and technology). We had a study of 105 articles from step 3 representing the use of GDBL method spanning over 11 years. Figure 2 presents the distribution of these articles related to the publishing year after step 3. The result after step 4 is also presented for reference.

The types of GDF are classified as (a) Game engines: it mainly covers the commercial game engines and mature and well-known toolkits mainly to create games. (b) Self-made GDF: it mainly includes the game development frameworks that were made by the authors of the articles for usage in a specific course. (c) Games or game editors: it mainly contains editors or platforms that can be used to modify games. (d) Simulation platform: it mainly includes controllers to create a game-like system for robots or other simulation platforms. (e) Hardware platform: it mainly includes both game hardware and related software to build games (laptops and computers are excluded), like Wii remotes, windows surface with XNA, robotic hand. (f) Others are general IDEs, like Visual C++, J2ME, or unspecified game creation toolkits with no specific requirement for learning. For some articles that covers more than one attribute like self-made GDF and simulation platform, we choose priority adhering the following sequence: game engine, self-made GDF, game editor, simulation platform, hardware, and others. Figure 3 shows distribution of types of GDFs applied in GDBL articles in percentage. Further, the top five in game engine subcategory are XNA (9 articles), First Person Shooter (FPS) game engines (unreal: 2 articles, Torque: 2 articles, half-life: 1 article), Flash (4 articles), Alice (4 articles), and Scratch (3 articles).

From statistics shown in Figures 2 and 3, we discovered the following clues.

(1) Tendency of Popularity. Figures 2(a) and 2(b) present the tendency of increasing the number of publications of GDBL articles from 2000, especially from 2006. Between 2006 and 2009, the number of GDBL publications grew with 3–7 articles per year, up to max of 25 articles in Figure 2(a). Figure 3 shows the distribution of the types of GDFs. From the statistics, game engines are most frequently used in GDBL method. We can infer that the continuous development and improvement of game engines will drive the GDBL's development further in the near future.

(2) Technology Changes the Ways of Learning. After 2006, there was a rapid increase in the number of GDBL articles published. We have analyzed possible reasons concerning this phenomenon from three perspectives: (a) frequent release of new commercial GDFs free of charge, like XNA (2007), Android SDK (2008), and evolution of software development environments, like Flash (acquired by Adobe in 2007) made game development easier than before. Technology changes or enriches the ways of learning and teaching. (b) Cross-disciplinary curriculum started to be used after 2006, for example, [42, 43]. It provides the possibility to use game development in these topics. (c) The upgrowing generation of students is a part of a game accepting culture where the public has an open mind towards games. This culture does not only focus on negative effects of video games such as violence and sex but embraces the positive aspects of games such as social integration, various improved skills, and usage of games for educational purposes, such as Sim-city and Civilization. Furthermore, students that grew up with games have become teachers in schools and may use games in their teaching. They show how technology changes the learning style. Whether it has positive and negative impact on learning depends on how we adopt the technology (game) and how it is used in teaching and learning.

3.2. Overview of the Study after Step 4. In terms of the classification method used in e-learning literature [44], a subcategory was iteratively developed based on the thematic topics found in the articles. Each subcategory was labeled with the disciplinary area-programming, SE, art, and other topic areas. As already mentioned in the introduction, the intended target audiences of the present study are educators, practitioners, researchers, and game designers who use GDFs in learning. The thematic subcategories should help the readers review teaching design, benefits, empirical findings, and future research topics in their own topic of interest. A similar thematic subcategorization of research articles was also performed in review of the general instructional games literature [41]. The overview of 34 articles after step 4 is shown in Table 1 grouped in four categories and labeled with course topics.

These articles present various GDFs used in GDBL and the covered course topics are summarized in Figure 4. The article T28 in Table 1 presents a study for using mobile game development as a motivational tool and a learning context in the computing curriculums. From their survey, the game development process can be used in the study of AI, database, computer networks, SE, human-computer interaction, computer graphics, algorithms, programming, computer architectures, and operating systems.

Table 1: Overview of articles.

Category	Item	Article	Major topics	Course topic
Research	R1	[55]	Students develop games on Torque game engine to learn game development	Game development
	R2	[56]	Undergraduate and graduate build games by adding code in Spacewar simulator to learn artificial intelligence	AI
	R3	[57]	Undergraduates develop games on XNACS1Lib framework to learn programming	Programming
	R4	[58]	Students develop games on Scratch to learn basic programming	Programming
	R5	[59]	Students develop games on Game Maker platform to learn software engineering	SE
	R6	[60]	Students develop games using Greenfoot to learn programming	Programming
	R7	[61]	Students build games by adding code in Wu's Castle to learn programming	Programming
	R8	[42]	Students build 3D movies on First Person Shooting game engine, Maya, and Photoshop to learn digital character production and machinima	Art
	R9	[10]	Students develop or modify Warcraft3 game editor, unreal game engine, and so forth, to learn software development, programming, project management, artistic concepts, and so forth	Mixed topics
	R10	[43]	Undergraduates develop games to learn outsourcing and software engineering	SE
	R11	[62]	Students develop games on self-made toolsets to learn programming	Programming
	R12	[63]	Students develop games on GameMaker to learn programming	Programming
	R13	[64]	Undergraduates develop Critical Mass board game on the web-based platform to learn data structure	Data structure
	R14	[65]	Undergraduates develop games to learn programming	Programming
	R15	[5]	Undergraduates develop minigames on XNA to learn programming	Programming
	R16	[66]	Graduates develop games on XNA to learn software architecture	SE
	R17	[67]	Students build games on Scratch to learn the Boolean logic	Boolean logic
	R18	[68]	Pupils build games by adding quiz to a web-based game shell platform to learn literacy	Literacy
	R19	[69]	Students build games by adding a code to a board game RoboRally, to learn artificial intelligence	AI
	R20	[70]	Middle school students build games on Storytelling Alice to learn information technology	Mixed topics
Discussion	D21	[4]	Graduate Students develop games on XNA to learn software architecture	SE
	D22	[71]	Middle school students build games on adding code in StarLogo TNG to learn 3D programming	3D programming
	D23	[72]	Art design students develop games on Flash to learn programming	Programming
	D24	[73]	Electronics design field students build a game-like system to learn programming, distributed system, and so forth	Mixed topics
	D25	[74]	Undergraduate students develop games to learn programming	Programming
	D26	[75]	Pupils develop games on NeverWinter Night toolsets to learn basic ICT curriculum	Mixed topics
	D27	[76]	Students build games by adding code to Bomberman game to learn programming	Programming
Theory	T28	[77]	Survey of mobile game development for different learning purposes	Mixed topics
Development	Dev29	[78]	Develop MUPPETS that students could use for game development to learn programming	Programming
	Dev30	[79]	Develop XQUEST based on XNA that graduate could use for game development to learning software architecture	SE
	Dev31	[80]	Develop Sheep based on Android that graduate could use for game development to learn software architecture	SE
	Dev32	[81]	Design and develop SIMPLE framework that students could use for game development to learn programming	Programming
	Dev33	[82]	Develop BiMIP framework that undergraduate could use for game development to learn programming	Programming
	Dev34	[83]	Develop JGOMAS framework that undergraduate could use for game development to learn artificial intelligence	AI

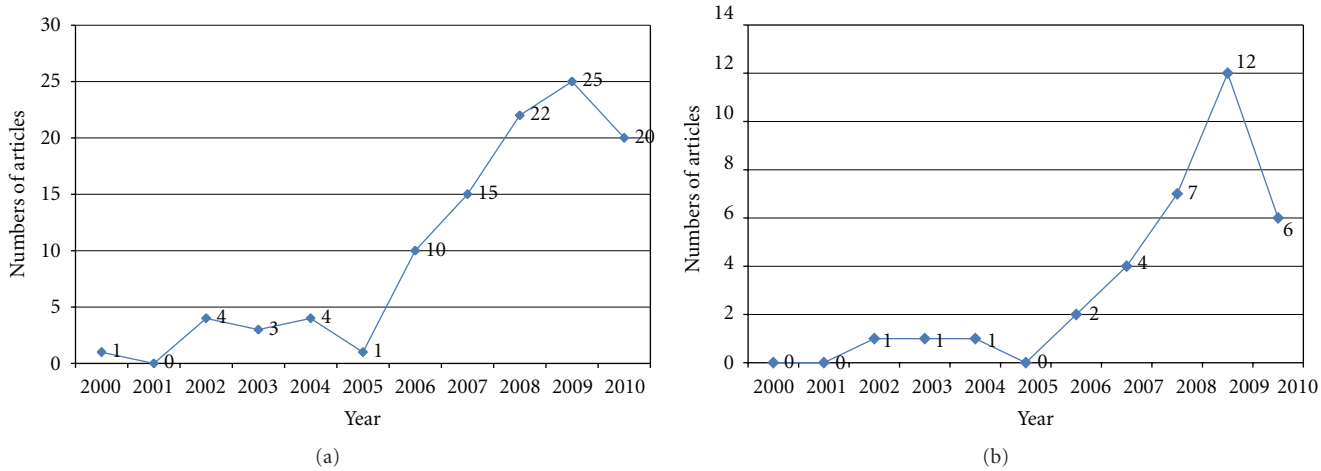

Figure 2: (a) Study of each year on using GDBL method (step 3) and (b) study of each year on using GDBL method (step 4).

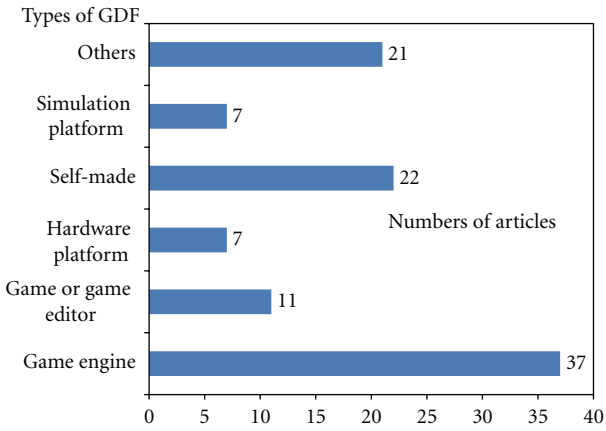

Figure 3: Study about types of GDF.

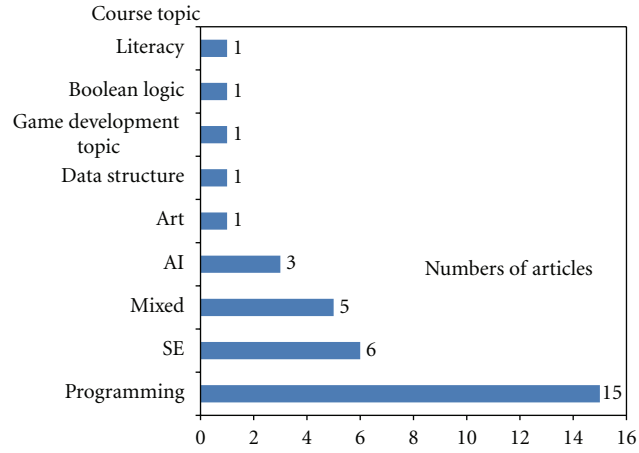

Figure 4: Distribution of the course topic.

Both the data from Figure 4 and article T28 can validate that the GDBL method can be used to teach various topics. Most applications are in the field of computer science, electronic, and basic IT learning. However, there are some innovative examples of other applications as well. Article R18 presents how a web-based game-shell platform is used to create a quiz game to teach pupils literacy with no programming requirement. Article R8 presents how Maya and Photoshop are used to create the digital character and movies that could be used as a video inside of a game.

From Table 1, it also shows that GDBL not only can be used in higher education, but also for basic IT education for kids in middle schools. The article D26 presents how pupils are taught basic ICT (Information and Communications Technology) curriculum by creating games. And the articles R20 and D22 describe how middle school students are taught IT and basic 3D programming by building games. The common GDFs used in the primary and middle schools are some GDFs that do not require much programming experiences for pupils, for example, the game editor. This will be further discussed in Section 4.

4. Findings

The articles collected after step 4 are further discussed in this section to serve the purpose of helping to identify and extract the significant elements to meet our aims, like elements to be used to guide the teaching design process when using GDFs in education. Findings are further presented as three aspects: (1) pedagogical context and teaching process, (2) technical aspects, and (3) evaluation results in relation to the aims of this study.

4.1. Pedagogical Context and Teaching Process. This section focuses on the current design process of integrating GDFs in courses or exercises to make the traditional teaching style become more engaging and diversified. This section also provides the detailed steps of how pedagogical theory can be used to guide the teaching design as well as strategies to aid the teaching.

The articles collected in this section are mainly from "Discussion" part in Table 1, and the rest is from the

"Research" and "Development" categories. The "Discussion" articles usually have a more complete description than the articles of other categories and include students background, GDF analysis, course setting and background, and teaching design with strategies. We are also concerned with the diversity and flexibility of using GDBL. The diversity shows not only that standard game engines or game frameworks are used in teaching, for example, XNA, but also that GDFs that are adapted or extended for teaching, for example, in article Dev31 they developed an extended library for the Android platform as a GDF for a specific course. Flexibility shows that (a) the same GDF can be used in different situations, for example, article D22 use XNA to teach software architecture and article R15 use XNA to teach programming, (b) the teaching process can be flexible to include other strategies than just integrating GDFs in the learning. For instance, article R13 adds the competition in game development for the assignments.

4.1.1. Pedagogical Context. Integrating game developments in a course study can provide increased motivation and attractiveness for the students. What is behind this motivation and can any theoretical context explain why GDBL can support learning? We investigated this question in the literature review, mainly focusing on (a) why apply the GDBL method in education and (b) how to apply it in a course in the first place. We found it was common to present the teaching design using a GDF in articles from the perspective of a teacher's experiences from the course, not thinking this process from a learning theory perspective.

Apart from the fact that games motivate for learning, we do not have strong evidence from the pedagogical theory to explain why it is a good idea to apply game development in education yet. However, there exists literature that explains game development, opposed to game play, as a pedagogical activity in the classroom. El-Nasr and Smith mentioned that Seymour Papert presented a relevant conclusion that programming is one example of the constructionism learning theory [10]. Constructionism involves two activities [45]. The *first* is the mental construction of knowledge that occurs with world experiences, a view borrowed from Jean Piaget's constructivist theories of learning and development. The *second* is a more controversial belief that new knowledge can be constructed with particular effectiveness when people engage in constructing products that are personally meaningful. The important issue is that the design and implementation of products are meaningful to those creating them and that learning becomes active and self-directed through the construction of artifacts. In the GDBL method, creating games with GDFs could be this artifact. This could be the fundamental concept to explain the pedagogical context of the GDBL. We can find support for this view from the articles in Table 1. For instance, article R9 considers the learning activity—modifying/creating a game using GDFs as a design activity that has educational benefits such as learning content, skills, and strategies [46]. Design activities are meaningful and engaging to students for exploring skills (analysis, synthesis, evaluation, revision, planning, and monitoring) and concepts to understand how they can be applied in the real world. Further, GDBL can be considered as a variant of several available construction activities. Similarily, for "learning by design," the article D23 presents using Flash for students from aspect of "learning by doing"—Dewey's theory [47, 48]. The article D26 uses "learning by making" to learn basic ICT (Information and Communications Technology) knowledge by making games, and it describes that "game making" has the potential to be a powerful learning environment according to attributes identified by Smeets [49]. These contexts are the evidence to explain the GDBL method as a constructionism activity from a theoretical aspect.

Based on Seymour Papert's opinion, another question pops up: how to use the pedagogical theory to support the design? A positive response is the article Dev31. It presents a case study on the use of double stimulation [50] to guide the exercise design. It considers that using a GDF in education could be a knowledge construction process and describes how to use double stimulus to guide a teaching activity. In schools, learners face a challenge, a problem, or a task that has been designed for a particular pedagogical purpose or they face situations that are likely to appear in work and public life. In both cases, the purpose of exploiting tools is for the learners to respond to such challenges. Based on constructionism, it constructs the relationship between the educational tasks and the material artifacts. This relationship is at the heart of Vygotskij's notion of double stimulation [50], a method for studying cognitive processes and not just results. In a school setting, typically the first stimulus would be the problem or challenge to which learners are expected to respond to. The second stimulus would be the available mediating tools, like GDFs. Similarily, other pedagogical strategies are also found to support for the GDBL's teaching design. Problem-based learning (PBL) presented in the articles R6, R14, D25, and Dev33 are also considered as theoretical reference when using GDBL methods. PBL is a pedagogical model that emphasizes the role of a real-life problem and a collaborative discovery process in learning [51]. Within a typical PBL setting, students are first given a challenging but realistic problem of significant size, relevant to the learning objectives of a given course. They are then encouraged to solve the problem in a group throughout the semester as independently as possible with minimum help from the instructor of the course. Even further, article D25 classified the process into the inception phase of PBL by giving game development requirements; the elaboration phase of PBL by building a rapid game prototype; the construction phase of PBL by implementing a game in a project; and the transition phase of PBL by a results evaluation. Apart from the traditional lecture-oriented teaching approach, PBL puts more emphasis on the instructor's role as a facilitator to prepare meaningful and interesting problems and to create and organize course materials in a manner that students have a just right dose of information in each class to incrementally develop a final solution based on a GDF to the primary problem of the semester. In addition, the articles R12 and Dev29 proposed to use collaborative learning together with the

game creation process, and article D24 proposed using "old model of Aristotle" [52] in the teaching design. All of them are of helpful support for the understanding of the teaching process.

The collections of the above results explain the validity of using a GDF in education from a pedagogical angle. Basically, it explains that applying the course content on GDFs by creating games fits well into a knowledge construction process, and it can be integrated with the pedagogical theory supports, like double stimulus or PBL to achieve an improved learning process and outcome. For instance, when we choose double stimulus as a pedagogical theory support, the learning design can be decomposed into two main elements: one is a problem, task, or goal that is designed by the teacher and the other is a responding learning activity that is implemented by students. From the double stimulus perspective, the first stimulus is tasks or assignments and second stimulus can be chosen as a corresponding tool based on the first stimulus. Its outcome depends on teachers' capacity to keep the two elements match each other. A good task (first stimulus) with inappropriate GDF (second stimulus) will not optimize the output. With this double stimulus support in mind, teachers should find an appropriate match between tasks and GDFs instead of just focusing on one aspect more than the other, like over focus on the design of task but neglect the effort of selecting the GDF. This is not a correct way for applying double stimulus. Further, if the selected GDF always conflicts with the tasks, we should reconsider changing the tasks or GDFs, or even apply a nongame tool. It implies that double stimulus can support learning activity for both GDBL and non-GDBL methods. The teachers should realize it and analyze which tool is better for the course aim and for the students when they apply double stimulus in teaching.

The number of case studies shows that only 30% of 34 articles include both pedagogical and technological design when applying GDBL. This phenomenon reminds us to improve the teaching process with relevant theoretical support. We believe our analysis points to the necessity for further pedagogical and technological codesign to better facilitate awareness of GDBL, thus better conduct the teaching process.

4.1.2. Teaching Process. How to integrate GDFs in teaching and exercises is a very important process when applying GDBL. This section analyzes the teaching process and exercise designs on various GDFs to achieve learning by implementing/modifying a game using GDFs. From our survey, we found necessary and common steps for the integration of GDFs in a course from the selected articles.

The *first step* is to identify explicit course aims. Figures 3 and 4 show relationships between GDBL and other fields and provides the case studies of how to integrate GDBL into different courses. After the course aim is clear, a common way to integrate GDBL in the course is that the teacher can design an assignment asking to develop a game. The students should then find a solution to this assignment that is in alignment with the course content. When facing such situation, the teacher should find an entry point in how to integrate GDBL with the course and exercises. If this is not possible, we recommend reading articles about similar courses from the selected examples in Table 1 and getting some inspiration. The *second step* is the exercise design and selection of GDFs. When applying a GDF in a certain course, the selection usually depends on the course content and exercise types, and so forth. We have recognized three types of exercises: one type is to modify the game or adding component to game platform or simulation platform to achieve a complete game, like in the articles D26 or Dev29. The second type is to create a simple game as an exercise to study or practice one or two concepts from the course content, like in the article R9. The third type is to do a complete game development project applying all concepts from the course. Usually, the first and second types can be used in the beginning of a course as a transition period when students are not familiar with the GDF environment, while the third type exercise can be used as a final exercise. However, there are other special cases, like in article R2 where only one type exercise was selected and applied in the whole process. The main driver of exercise design depends on the course aim and students' background. Selection of GDFs is separately discussed in Section 4.2. The *third step* is to do a tutorial lecture where the GDF is introduced to the students. The *fourth step* is to run an initial exercise, which should be easy to do and let the students get familiar with the development environment. The *fifth* and *final step* is to do exercises that include implementation of a game. Usually, it is accompanied with some suggestions that were applied in most of the literature: (a) collaborative learning: the student groups range from 2 to 6 students in our statistics; further article R12 has some discussions about how to locate student members in groups such as regular meetings with instructors and flexible meetings among group members. It is important to keep instant communication with the exercise requirements, which would be positive to the students' learning towards the GDBL method. For each group member, it would be a tradeoff between cooperative and individual work during the work duty allocation. Further, a workshop is suggested to be held at the end of the course. (b) Support: technical support to help students overcome the technical difficulties they face. It is helpful to give examples in the beginning such as to provide optional examples codes and exercise examples to explain the exercise's complexity. Also there are other strategies like conducting a pilot study before the formal application of GDBL. This approach only appeared in two articles. After the whole teaching process is completed, usually a survey to evaluate both the teaching process and the used strategies is conducted and a more detailed analysis is performed considering the impact factors described in Section 4.3 based on the evaluation from the literature.

4.2. Technical Solution. The technical aspect of the GDBL method is mainly about GDFs' features described in the 34 articles. And this section will not go into technical details of development of GDFs due to that it is out of the scope of this paper. On contrary, we mainly analyze the GFDs features in the context of GDBL based on our aim of this study.

4.2.1. GDFs Survey. In order to provide a guide to choose a GDF for GDBL, we classify GDFs into two categories: GDFs for novices and GDFs for developers. The main focus of GDFs for novices, including nonprogrammers, is to provide visual methods for customizing game templates and to allow creating or designing games with little or no programming skills. The main focus of GDFs for developers is to offer toolkits that support development of high quality 2D/3D rendering, special effects, physics, animations, sound playback, and network communication in common programming languages, such as C++, C#, and Java. The 34 articles are classified into Tables 2 and 3 according to the GDFs used in the study. The unspecified GDFs or general SDK have been excluded, for example, the articles R10, R11, R14, D24, D25, and T28.

In addition, one mature GDF selected from step 3 in Figure 1 could be a backup for novices—CeeBot Series (http://www.ceebot.com/ceebot/family-e.php) [35]. The programming language in CeeBot is very similar to Java, C++, and C#. It has been developed especially to make learning programming easier. "CeeBots4 School" is a programming course for middle and high school.

4.2.2. Criteria for Selection of Suitable GDFs. Choosing a GDF is considered to be an important procedure during the preparation work for teaching. This process can be described by the following steps: (a) finding various GDF candidates, (b) analyze each GDF's features, (c) make criteria to filter GDF candidates, and choose one or more GDFs that fit best with the course content. Although our literature survey shows that different course aims have different requirements for the selection of the GDFs, there are still some common points to share. The article D21 presents a general criteria to choose a suitable GDF for the education in terms of theory—"What makes learning to be fun" by Thomas [53]: for example, easy to learn, allow rapid development, and provide an open development environment to attract students' curiosity. R1 presents that the GDFs should be chosen based on its cost and license, quality, difficulty, textbooks for guidance, and its main functionality. The article D26 explains that their students were not to become experts in programming, and thus they chose GDFs for novices. The article D27 introduces their self-made GDF and assess their own GDF by comparing it with other GDFs in terms of interactive, amusement, easy to use, using official program language, combine with teaching materials, evolutionary learning mode, census analysis, and storylines. The article Dev31 chose the GDF based on analysis of development environment, tutorial documents, emulator, programming language requirements, test devices, interface of the GDF, and possible ways to share games. Further both articles R3 and Dev31 developed a library for the GDF to make it more suitable for the course context. If we face the condition of only one choice, article D22 presents their effort to improve the only GDF. The article D23 presents how they compared different versions of same GDF and made a choice between the newest version with powerful functions or old version but more stable.

To summarize, there are common and essential guidelines when selecting the GDFs: (a) *technical environment and inexpensive (low costs) to use and acquire*: the technical environment requirements include required operating system and hardware, what tools are provided, are third-party tools supported, and how difficult it is to install GDF. A typical problem can be, for example, that XNA runs only on Windows and many students now have PCs running Linux or Mac OS X. The technical requirements might also be an economical issue, as the choice of GDF might force hardware upgrades or paying for licenses. (b) *Sufficient documentation to guide the usage of the GDF*. Students need to explore the GDF as an extra task before they start game development on the GDF. If the resources and materials are sufficient and easy to acquire for beginners, it will help them shorten the time spent on learning the technical environment. Time is an important factor during the whole teaching process, which will be further discussed in Section 4.3. (c) *Meets the students programming technique contexts*. The GDF must be easy to learn and allow rapid development. This issue is also driven by time constraints. Usually, if learning the GDF is not the major educational goal in the course and only an aid to learn something else, learning a new GDF will steal time from the course schedule. An easy and friendly environment is welcomed in order to save time for the students and to keep the focus on the course content and less on the GDF. (d) *Not in conflict with the educational goals of the course, flexibility to combine a GDF with teaching materials, and possible to add/change libraries that can be used within the GDF*. All GDFs have constraints related to the course content in how they have been designed or how they are released. One example is in SE education where open source GDFs make it possible to do white-box testing on the GDF, while the source code for other GDFs might not be available. Further, some GDFs might have constraints on how you can design your games, what design and architectural patterns you can use, how event-handling must be managed, the freedom of expanding the GDFs functionality, and more. These constraints must be integrated in the SE teaching to introduce the students to the real world where software rarely is built from scratch. In addition, if GDFs are not easy to use, and not strongly relevant to the course content, we can add/change a library with a user guide to apply course content in the GDF. (e) *Using an official programming language*. Conditionally, it applies to the types of GDFs for developers using a commercial game engine with widely known programming languages, like C#, Java, and C++, which are familiar to the students. But for the types of GDF for novices, if the course just lets students know the data structure, an official language is not really needed. But special programming languages are not widely accepted and as useful as official programming languages in a long run if the students will do more software programming in the future. (f) *Amusement and interactive*. The GDF should provide a visual and stable development environment to attract the students' curiosity and engagement. A game development assignment in a user-friendly game development environment could be a good motivation for the students compared to traditional

Table 2: Study of GDFs for novices.

GDFs	Features description	Origin
Alice (http://alice.org/)	Alice provides a point-and-click programming interface allowing the creation of simple 3D games and animations. It is a tool for teaching object-oriented programming through creating simple games or animations	R20
Scratch (http://scratch.mit.edu/)	Scratch provides a point-and-click programming interface to create media-rich games, animations, and applications for the Web. Scratch is suitable for teaching children basic programming (variables, arrays, logic, and user interface), and for creating simple 2D quick-and-dirty applications	R4, R17
Greenfoot (http://www.greenfoot.org/)	Greenfoot is a solid tool that provides many of the needed constructs for creating 2D computer games at a level that is especially appropriate and fun for novice programmers	R6
Maya/Photoshop/Flash	They are mainly used for art design to create digital characters and animations for games. Flash could also create Flash games	R8, D23
Game maker (http://www.yoyogames.com/)	Game Maker is a rapid-application development tool for young people at home and in schools to create two-dimensional and isometric games	R5, R12
StarLogo TNG	StarLogo TNG is designed upon the basic framework of Logo. The programming is done with programming blocks instead of text commands and moved programming from abstract to visual	D22
Game editor: Warcraft3 Editors/NeverWinter Night toolsets	The editor provides a simple GUI for customizing game templates and requires little or no programming skills to create interesting game designs. The editors are implemented as visual programming tools that allow users to visually customize game behavior, including character behavior, game map, and game play	R9, D26
Game platforms: Bomberman/Wu's Castle/Critical Mass board game/quiz-based web game shell	These are concrete games, but provide visual interface for the users to modify or add a basic code to change the game scenarios	R7, R13, R18, D27

Table 3: Study of GDFs for developers.

GDFs	Features description	Origin
FPS game engine: Torque game engine/Unreal Engine	These are original commercial game engines and already have applied in commercial and popular games. They are usually not free and provide with some edit tools. And more complex than a concrete game editor	R1, R8, R9,
XNA (http://www.xna.com/)/ XNACS1Lib framework/XQUEST/BiMIP	These are game development tools based on MFC and DirectX from windows platform and have the same structure on game loop concept. BiMIP is a self-Made similar to XNA. And XNA is a GDF to develop cross-platform games for the Windows PC, Windows mobile phone, XBOX, and the Zune platform using the C#. XNA features a set of high-level APIs targeted for 2D and 3D games. It consists of an integrated development environment (IDE) along with several tools for managing audio and graphics. XQUEST and XNACS1Lib are game library for XNA that contains convenient game components	R3, R15, R16, D21, Dev30, Dev33
Android/Sheep (http://www.android.com/)	The Android mobile platform is a mobile application development platform issued by Google. And Sheep framework is an extended game library for Android	Dev31
Simulation platforms: Spacewar simulator/RoboRally/JGOMAS MUPPETS/SIMPLE framework	There are self-made simulation game or simulator that provides the controller for the users to modify the parameters and control the avatar in these simulation platforms; they are usually to teach the programming and AI field	R2, R19, Dev29, Dev32, Dev34

assignments. For example, most students think it is more interesting to work on a game project than, for example, a system for a bank. (g) *Ability to develop games in a cross-platform environment.* Conditionally, it applies to the types of GDFs for developers. One good example is XNA where the students can choose developing their games either in PC, mobile (Windows Phone 7), and/or console (Xbox360). Other game engines such as Unity3D also allow developing the game in multiplatform. The advantages are the following. (1) Provide students degrees of freedom in developing their games for the platform of their choice and (2) learn about the strengths and constraints of different platforms (e.g., user interface, viewing screen size, resolutions, resources such as memory and processor power, storage for saving/loading the game, and, etc.) in game development.

We consider the above to be the most important criteria to guide the teachers in selecting one or more GDFs for their courses. And some criteria could be changed according to the specific context of the teaching environments. For instance, the target students are middle school pupils and the course

goal is to let students familiarize themselves with information technology, it is not necessary to choose (e). In principle, the course aims and students context are the two fundamental and prioritized attributes to decide the selection of GDFs.

4.3. Evaluation. Besides the pedagogical analysis and the GDFs' analysis in GDBL, this section summarizes the evaluation data from the articles mainly in the "Research" category. Furthermore, we hope to find empirical evidence to support the effectiveness of GDBL. Specifically, in order to approach the third aim of the study, the following information was drawn from each article (if provided in the article): (a) major empirical findings related to the actual effectiveness of GDFs used as an aid in teaching and (b) factors that impact the teaching outcome in terms of the experiment data from the articles are also posed. It is not a simple process to assess the effectiveness of GDBL and it covers at least two aspects: teachers' and students' satisfaction of using this method. The teachers' concerns are the researcher's understands of the course (not applied where the researchers and the teachers are the same), the GDFs' features, matching between the selected GDFs and the course content, and teachers' expertise on games. The students' concerns involve having interesting exercises and the difficulty of learning extra content—the GDF. Our literature review focuses on these aspects and Table 4 shows a summary of the evaluation process of GDBL in each article. Comparing the students' and teachers' satisfaction, students' satisfaction could be the most important result since it directly relates to teaching effectiveness. And the following results are extracted from the literature and used to validate the effectiveness of GDBL and the impact factors related to it. The results in Table 4 are mainly shown in three categories: (a) experiment data that describes the collected data and materials for the measurement of the effectiveness of the results, (b) conclusion of effectiveness of GDBL, and (c) impact factor that describes the elements that effect outcomes and is classified into positive, neutral, and negative categories based on the articles' data and conclusions.

From evaluation data in Table 4, the common expressions of measurements are (a) students' grade or score on the course exam, (b) project results, including analysis of project size and classes they used in game programming; obtaining certain requirements of exercises by percentage; length of codes; percentage completed of the projects and time spent on the projects or the GDF, and so forth, (c) questionnaire surveys to measure the following aspects: students' satisfaction about the exercise, course and GDF; students background; students' interest in game development topic; course, and exercise learned and open questions to get suggestions for the improvement of a course, and so forth, and (d) observation and feedback to perceive the fluency of the teaching process and interaction between students and the teacher.

From Table 4, the effectiveness of each article is collected. Generally, 22 of 23 articles have a positive conclusion about using game development in a course in most of aspects, for example, student motivation, engagement in lectures, and exercises. Only the article R5 presents that learning by game design did not have the expected outcome and that the time constraint was a critical issue. Students indicated that they needed more time than two weeks to write a satisfactory 2D game. And finally, it explained that they did not have an adequate number of participants to have an accurate picture about the effects of game design on students' motivation and attitudes.

Apart from validating the effectiveness of using GDBL methods, the impact factors that could cause a positive or negative outcome deserve to be analyzed. From Table 4, we have summarized what should be noticed when applying GDBL. The following items are the most common issues that appeared repeatedly in our survey.

(1) Communication between the Researcher and Teacher towards the Understanding of the Course Content. This item is not applied to the condition where the teacher and the researcher is the same person. If the researcher designs the method and the researcher invites the teachers to adopt it in schools, good communication and mutual trust between them are crucial to achieving the desired effect. The article R15 states that the teachers should become comfortable with using GDBL and spend a bit more time on it compared to traditional method in a certain course, otherwise it may cause a misunderstanding or bias against GDBL. Another aspect is that the researcher may worry about the teacher and not totally understands the game effectiveness in education, and how game motivation can be successfully used to improve the course design, which is mentioned in the articles R3, R6, R15, and R18. This indicates that the researchers should help teachers in gaining self-confidence and provide constant support while the decision is made to apply GDBL in the curriculum.

(2) Teamwork. This factor could have both positive and negative effects on the teaching results if students work in groups. First, the team size and working environment must be considered in advance. For instance, it was found in article R10 that a big team size could have positive impact on outsourcing course teaching, and article R1 claims that lab environment with teamwork could help improving the effectiveness of cooperative learning. On contrary, as the team gets larger, it becomes more difficult to set the time for general meetings and joint work hours. Further, it also means complex relations in a large group. A serious issue—bottleneck—could happen in the game development process. If one member of the team does not perform, then the entire game development process slows down. Second, the instant communication in a team has a significant impact. Article R2 mentions that the group work can help weaker students. Article R12 also agrees with this statement, but it describes that unexpected situations can occur during the teamwork to hinder the instant communication which the teacher should take care of. Third, the R14 article concludes that students need more experience in working effectively in teams. Most of the case studies found in the articles provide the evidence that teamwork can be used together with GDBL and the

Table 4: Evaluation data collection and impact factor.

Title	Sample	Comparison	Experiment data	Conclusion of effectiveness	Outcome of impact factor
R1	22 students	No	Quantitative data (1) Questionnaire of students background (2) Survey project results of students' game play preferences (3) Questionnaire on students satisfactory (4) Questionnaire on interest level in game development careers (5) Questionnaire on students assessment of gains (6) Questionnaire on helpful course elements (7) Students peer-evaluation	Students generally satisfied the elements in the course and resources (including teamwork)	Positive (1) Lab environment and teamwork helped to archive the effectiveness of cooperative learning (2) Teaching game development required a shift from teacher-centered to student-centered learning environment (3) GDFs provided an environment that students could integrate wide variety of skills and knowledge (4) Motivation factor: competition Negative (5) Poor textbook for GDF provided negative effect
R2	33 students (28 undergraduates and 5 graduate)	No	Quantitative data (1) General questionnaire Qualitative data (2) Students feedback about the course	Generally, students enjoyed the project and it fulfilled all of the criteria of a successful project outlined at the beginning plan	Positive (1) Flexible and interactive simulation platform (2) Providing examples for the difficulty part in the project that was out of the course aim (3) Group project and discussion helped weaker students Neutral (4) Difficulties at first year, but smoothed out by get more teaching experiences and previous evaluation for the improvement
R3	21 undergraduates	No	Quantitative data (1) General questionnaire Qualitative data (2) Video recording about course process (3) Faculty feedback	The GDF was an excellent catalyst, enabling faculty to begin exploring teaching with game topics and help students to be more engaged	Positive (1) Because of the immediate interactive graphical feedback, students were engaged and motivated to experiment with the programs Neutral (2) Instructor's attitude toward the interest in GDF Negative (3) Visual feedback, although a powerful learning tool, could also be a source of distraction for students (4) Time spending should not involve the reading of background material in class (better before class) (5) Limited classroom time was challenging for students
R4	35 female students from both preschool and university	No	Quantitative data (1) Questionnaire: students' opinions about GDF (2) Questionnaire: effect of students familiarization with Scratch in using of ICT education (3) Questionnaire (pre- and post-test) attitudes against Internet in education and application development	Scratch was user friendly and satisfied by the students, and it also has a rather positive effect on students' opinions and attitudes towards computer programming and ICT educational value in education	Positive Scratch helped to setup confidence of students in exploration of ICT in education.

Table 4: Continued.

Title	Sample	Comparison	Experiment data	Conclusion of effectiveness	Outcome of impact factor
R5	20 undergraduates	No	Quantitative data Questionnaire: Likert's scale (pre- and post-survey in game design course)	Game design had both positive and negative impact on students' attitudes about computer science, game design, and further development of programming skills	Positive (1) Students who had prior programming experience can express interest in game design Negative (2) Time constraints: assignment might be better received and increase students' interest if students were given more time and equal emphasis on other phases (3) Game design topic course had a negative impact on students' interest in pursuing a CS degree (4) Not adequate number of participants to have an accurate picture of true effects of game design on students' motivation and attitudes
R6	26 high school students and 8 teachers	Yes	Quantitative data (1) Questionnaire assignment survey (2) Questionnaire with pre- and post-survey: self-assessment on art and design (3) Questionnaire survey on teachers' attitude	It showed great promise for engaging high school students programming and increasing interest in computer related fields of study. Both teachers and students felt a significant improvement in computer programming and self-confidence	Positive (1) Researchers trained both students and teachers by applying GDBL Neutral (2) Teacher attitudes and self-confidence about GDBL's effect the teaching process
R7	26 students in experimental group. 29 in control group	Yes	Quantitative data (1) Each phase of study (2) Pre- and posttest score (3) Learning difference between groups and subgroups (4) Game statistics (5) Questionnaire survey of each task	Students in the game-first group felt they spent less time on assignments and all students preferred the learning game to the program	Positive "Wu castle" was more effective than a traditional programming assignment for learning and could help prepare students to create deeper, more robust understanding of computing concepts and improving their perceptions of homework
R8	—	No	Quantitative data (1) Questionnaire to survey students feedback (2) Compared with whole school average score	Students got higher score in this course than school's average score	Positive Assessing the GDF in the starting
R9	26 students	No	Quantitative data (1) Questionnaire to survey assignment difficulty with Likert's scale Qualitative data (2) Observation of students progress	Using game development motivated students to learn and allowing them to apply and visualize the utility and application of the concepts	Positive (1) GDBL could learn several subjects and concepts Neutral (2) Different game engines implicitly stressed the use and development of certain skills
R10	40 undergraduate students	No	Quantitative data (1) Pre- and posttest questionnaire to survey Changed perception of outsourcing concept (2) Questionnaire: SE outcomes Qualitative data (3) Observation: discoveries in communications	Students improved their understanding of outsourcing, developed better appreciation for the importance of SE techniques, and created ad hoc communication protocols between teams	Neutral Enlarging the teams' sizes to other universities to create an inclusive teaching environment, which had limitation that only applied in outsourcing teaching

Table 4: Continued.

Title	Sample	Comparison	Experiment data	Conclusion of effectiveness	Outcome of impact factor
R11	38 students (19 teams)	No	Quantitative data (1) Length of codes according to grade (2) Project size and classes (3) Methods used in programming (4) Weekly working hours (5) Proportion of work discussion, coding, thinking, graphics, and audio (6) Object-oriented skills applied in code	Positive experience had been gained in teaching the topic by using game framework	Neutral (1) To keep the students motivated and teachers tailored the course for each student (2) Using game development to achieve depth of objects and object interactions training
R12	124 students	No	Quantitative data (1) Grade (2) Questionnaire to survey students attitude	Learning by creating game was able to improve the student grades largely	Positive (1) Object-oriented programming concept became easier to understand after seeing object design visually in the GDF (2) Students felt happy with using cooperative learning system, games development, and visual design Negative (3) The group members' communication was hindered by the in front of computers (4) GDBL could help with the passing rate, but still have improving space for graduation aim
R13	55 students'	No	Quantitative data (1) User survey of game project percentage completed (2) Login times (3) Questionnaire with Likert's scale: student satisfaction (4) Questionnaire with Likert's scale: tournament features	Combination of game development and friendly student competition was a significant motivator for the increased student performance	Positive (1) Tournament could increase students' participation and motivation Negative (2) Students' common complaint of not having adequate time to complete the project
R14	—	No	Quantitative data (1) Individual and group creativity levels perceived by students (2) Students' perception of abilities developed at intermediate or high levels Qualitative data (3) Future career survey	Game project development with collaborative learning was manageable and effective for increasing students' teamwork capability and increase the employability confidence	Positive (1) Project (game project development) based learning motivated their team collaboration Negative (2) Teacher attitudes: initial resistance for problems that students' teams faced could be discouraging to faculty members who did not expect it (3) Teamwork: students were not born knowing how to work effectively in teams. A flawed team-based instructional model had negative effect
R15	CS1: 22 in GTA and 10 in Console CS2: 18 in GTA and 9 in Console	Yes	Quantitative data: (1) Success rate (Passing rate) (2) Assignment score (3) Self-reported time spent on assignment (4) Post Assignment Survey (5) Pre and Post course survey Qualitative data: (6) Feedback from faculty	Interactive graphical assignments could be a good tool for teaching CS1 students. The success of GDBL hinged on the instructor's expertise and enthusiasm	Positive (1) GDF feature: interactive graphical application supported experimentation and visualization Negative (2) Teacher's background and attitudes towards the games impacted the output of a lecture, faculty "dropped" GDBL in the end at first experiment, but became more comfortable later

Table 4: Continued.

Title	Sample	Comparison	Experiment data	Conclusion of effectiveness	Outcome of impact factor
R16	46 students	No	Quantitative data Questionnaire about learning process, tradeoff between technical and architecture problems, integration of game development and course, learning outcome	GDF was easy to use and did not conflict with course aim. A good GDF could save development time	Neutral GDF selection influenced learning process and extra technical issues, but students could learn a lot through a game project.
R17	27 in control group, 43 in experimental group	Yes	Quantitative data score of the pre- and posttest by a test sheet	Results showed that the proposed game development activity could have higher learning achievements compared to the traditional lecturing	Positive: (1) GDF issues: choosing modifying game according to course topic with simple scenario. And tutorials for GDF were prepared well. Understanding game topic could make engage learning Negative (2) Game was additive for students
R18	125 experimental students and 186 control group students	Yes	Quantitative data (1) GRADE test scores (pretest, posttest) Qualitative data (2) Interviews on teacher's feedback	Game development helped to improve the student content retention and so forth	Positive (1) Optimum amount of time to spend at a sitting on game development activities was about 45 min by observation Negative (2) Too little time allotted to the development of game and insufficient gaps between each game creation activities
R19	33 undergraduate	No	Quantitative data Questionnaire with Likert in general	Using GDBL indicated that the motivation of the students was higher and they understand complex problems easier and exercise could be done more rapidly	Positive GDF was searched and chose based on the requirements
R20	22 middle schoolers	No	Quantitative data (1) Questionnaire: pre- and post-surveys of participants information (2) Programs analysis Qualitative data (3) Daily log (4) Interviews on students	Findings suggested the middle school students could use Alice to make games to build information technology fluency	Neutral (1) To provide a proper challenge in class (2) Difficulty in using GDF to finish the assignment
T28	NA	No	Quantitative data (1) Survey of students background (2) Relevant application about Mobile GDBL	Mobile game development could be successfully integrated into computer science education	Positive Students' background: student lived in game environment and game development exercise could be a good motivation
Dev30	19 graduate	No	Quantitative data Questionnaire survey with Likert's scale and system usability scale survey	XQUEST enhanced XNA in suitability as a teaching aid in SE learning	Positive To design the XQUEST from the previous assessment experiences
Dev32	57 in group1, 45 in group2	Yes	Quantitative data (1) Questionnaire result of students' user experience (2) Score for pre/posttest	SIMPLE improved both learning motivation and programming skills for the students	Positive Use GQM approach in developing game metrics for students' exercise.

nature of teamwork is suitable for cooperative learning and teacher should take care of the issues that may happen during teamwork. However, most of articles did not mention the strategy of competitive learning in GDBL. Only the articles R1 and R13 apply both cooperative and competitive learning in the exercises with a positive feedback in both cases.

(3) GDF Relevance. The most mentioned aspects related to GDFs that impact the outcome are (a) the articles R2, R3, R12, and R15 present the advantages of using interactive graphical GDFs. It shows that visual graphics can provide instant feedback, making student engaged in programs, (b) the articles R3 and R4 describe how a GDF can improve students' confidence in programming tasks, and (c) the articles R1, R8, R9 R17, and R19 emphasize the need to analyze the GDF's features in the light the course content, and detailed GDF tutorials should be conducted before it is used in the later exercises.

(4) Students' Background. In the article T28 surveys, the students' background was that most of them had played games as they were growing up. This is a suitable prerequisite to apply GDBL. But a negative aspect is the addictiveness to games, as mentioned in the articles R16 and R17. Some students may focus too much on the game and game development thus losing focus on what they shall learn in the course. This means that the design of the course and the project must be carried out in such a way that the students are forced to learn and use course content. From the articles R5 and R11, it was also noticed that the diversity of student background causes some difficulty of using GDBL. For instance, the programming experience of the students strongly affects the choice of GDF between the ones for novices and the ones for developers. For instance, to use XNA/XQUEST or Android/Sheep from Table 3 for developers, the students must know object-oriented (OO) programming well and be familiar with OO design patterns and OO principles. And some other GDFs require learning a specialized and simplified programming language for game creation, which is more suitable for students without programming experiences.

(5) Teachers' Requirements. Teachers' attitude of applying the GDBL method in the course is an essential aspect in a teaching process. The articles R3, R6, and R15 suggest that the faculty should have relevant technical background about the applied GDFs. The article R14 also mentions that they should prepare and solve the anticipated problems they may face during teaching. It is essential that the course staff have technical experience in the selected GDF to provide help for students and to avoid the focus shifting from the course content to technical matters.

(6) Time Constraints and Workload. This problem has been stressed repeatedly in several articles. Most of articles found that the time was limited. For instance, the article R5 mentions that time constraint caused to cut down the time in beginning phase. The article R13 reports that some students complain about insufficient time to complete the project. So there are some advices correspondingly, like the article R18 proposes some suggestions on the time consumption, and the article R3 suggests reading the background material better before the class in order to save class time for students. To help with the time management, a comprehensive time schedule should be prepared in advance for both the teacher and the students. Specifically, a series countermeasures can be the following: (1) make sure that the students learn, understand, and apply the GDBL-project process; (2) force students to set a mandatory rule for teams to create the schedule (strict milestones and deadlines); (3) get involved with the students early to make sure that they make a realistic goal; (4) teacher continuously monitor their progress and guide them to make adjustments, if needed, in order for them to complete their projects.

Other atypical factors could be found in Table 4. Further, Section 4.3 also provides a reference of how to assess the GDBL method. This indicates that future evaluation data of using GDBL is also beneficial, for example, [54]. As it does not only reveal the efficiency of using the framework along with how much the students actually learn from game projects, but also the social relationships' investigation of learner-learner, learner-teacher, and teacher-researcher.

5. Conclusions

From the above findings, we summarize a guideline for integrating a GDF in learning with teaching strategies. Figure 5 shows a simplified diagram that gives an overview of the design process of applying GDBL (adapted from article D21 and Section 4.1.2). It contains four elements (course aim, pedagogical theory support, GDF resource pool, and impact factor), two methods (learning by creating and learning by modifying games), and six steps in the teaching process and two subjects (students and teachers).

Basically, the course aim has the fundamental effects on the selection of GDF. And the pedagogical theory (Section 4.1.1) could support the teaching design. The GDF resource pool (Section 4.2.2) could be the reference for the selection of GDFs. Usually, during steps A to B in the teaching process in Figure 5, the pedagogical theory support and GDF resource pool play important roles in these two initial steps. Impact factors concern the whole process, but we suggest considering them at beginning as well. In terms of the course aim, pedagogical theory support and GDFs resource pool, the teaching process (Section 4.1.2) starts with designing the lectures and exercises with the selected GDF. After the lectures and tutorials, the course delivery starts and students begin the design and implementation of their projects. For the evaluation framework (Section 4.3), teachers/researchers are suggested to collect data using surveys. Based on the analysis of collected and teaching experiences, they can improve the teaching process framework. Here, we use a compact case to explain how each element in Figure 5 works in a certain course if the GDBL method is applied. The assumption is that the course aim is to teach basic programming rules for beginners. The choice could be made between "learning by

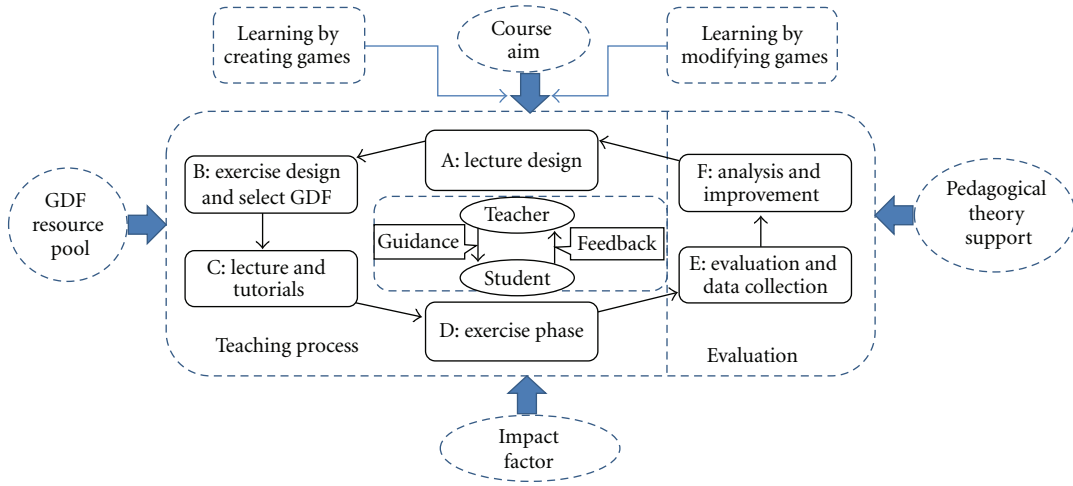

Figure 5: A guideline for technical and pedagogical codesign of GDBL.

modifying games" using a game editor with scripting, or "learning by creating games" using a GDF for novices. Then, we should consider the relationships between the problems and tools from the perspective of double stimulus or use other pedagogical theories to construct the learning process, for example, PBL. With this in mind and according to the criteria in Section 4.2.2, commonly used tools can be selected from the GDF resource pool—GDFs for novice in Table 2 or use another GDF if no suitable GDF is found in Table 2. After finishing steps of A to B in teaching process we start the lecture and the introduction of both exercises and GDFs. Later, students commence the implementation individually or in groups. During the whole teaching process from A to D, the impact factors are relevant but optional. For instance, we can choose a graphical interactive GDF and estimate time to be spent on lectures and exercises. Applying the impact factors in the teaching process depends on the courses' situations. That is why we have the evaluation and analysis steps E and F in Figure 5. The feedback data can help to validate the choice in each step—whether we choose a right task or a suitable GDF or focus on the most relevant impact factors in a course. In addition, since many elements interact in GDBL, which makes the real situation more complex to analyze and evaluate. Thus, an effective evaluation helps to validate the whole teaching process, and it is not only judged by teachers' own experiences, but also get opinions from students' aspect.

From the experience of accomplishing this paper, we still have the following limitation: (a) the scope of data search and collection from four scientific search engines is relatively limited; (b) due to the game research field is younger than other traditional research fields, the amount of articles with empirical data is still limited in our the survey, it maybe cause the pitfall of the evaluation results, for example, generalization; (c) some topics deserve further discussion. Cross-disciplinary courses, like game development course in article R1 covers programming and art design, and machinima course in article R8 have 3D animation and movie creation. Both of them could be further discussed since GDFs plays different roles—the main tool in article R1 and an innovative auxiliary in article R8.

This study has shown that GDBL do have the potential power to help students to learn different curriculums. We hope that the study will provide useful guidance to educators, practitioners, and researchers in the area of GDBL, as well as to GDF designers, and that it will inform their future professional practices and research.

References

[1] S. M. Dorman, "Video and computer games: effect on children and implications for health education," *Journal of School Health*, vol. 67, no. 4, pp. 133–138, 1997.

[2] M. Prensky, "Digital game-based learning," *Computers in Entertainment*, vol. 1, no. 1, pp. 21–24, 2003.

[3] J. Blow, "Game development: harder than you think," *Queue*, vol. 1, no. 10, pp. 28–37, 2004.

[4] A. I. Wang and B. Wu, "An application of a game development framework in higher education," *International Journal of Computer Games Technology*, vol. 2009, Article ID 693267, 12 pages, 2009.

[5] K. Sung, C. Hillyard, R. L. Angotti, M. W. Panitz, D. S. Goldstein, and J. Nordlinger, "Game-themed programming assignment modules: a pathway for gradual integration of gaming context into existing introductory programming courses," *IEEE Transactions on Education*, vol. 54, no. 3, pp. 416–427, 2010.

[6] G. Sindre, L. Natvig, and M. Jahre, "Experimental validation of the learning effect for a pedagogical game on computer fundamentals," *IEEE Transactions on Education*, vol. 52, no. 1, pp. 10–18, 2009.

[7] B. A. Foss and T. I. Eikaas, "Game play in engineering education—concept and experimental results," *International Journal of Engineering Education*, vol. 22, no. 5, pp. 1043–1052, 2006.

[8] A. I. Wang, T. Øfsdahl, and O. K. Mørch-Storstein, "Lecture quiz—a mobile game concept for lectures," in *Proceedings of the 11th IASTED International Conference on Software Engineering and Application (SEA '07)*, pp. 305–310, 2007.

[9] A. I. Wang, T. Øfsdahl, and O. K. Mørch-Storstein-Storstein, "An evaluation of a mobile game concept for lectures," in *Proceedings of the IEEE 21st Conference on Software Engineering Education and Training*, pp. 197–204, 2008.

[10] M. S. El-Nasr and B. K. Smith, "Learning through game modding," *Computers in Entertainment*, vol. 4, no. 1, pp. 45–64, 2006.

[11] G. Lukas, "Uses of the LOGO programming language in undergraduate instruction," in *Proceedings of the Association for Computing Machinery Annual Conference (ACM '72)*, vol. 2, pp. 1130–1136, Boston, Mass, USA, 1972.

[12] M. Micco, "An undergraduate curriculum in expert systems design or knowledge engineering," in *Proceedings of the 15th Annual Conference on Computer Science (CSC '87)*, pp. 36–39, St. Louis, Mo, USA, 1987.

[13] J. P. Higgins and S. Green, *Front Matter*, John Wiley & Sons, New York, NY, USA, 2008.

[14] K. S. Khan et al., *Undertaking Systematic Reviews of Research on Effectiveness: CRD'S Guidance For Carrying Out or Commissioning Reviews: CRD Report*, Number 4, NHS centre for revies and lissemination, University of York, 2nd edition, 2001.

[15] M. Papastergiou, "Exploring the potential of computer and video games for health and physical education: a literature review," *Computers and Education*, vol. 53, no. 3, pp. 603–622, 2009.

[16] J. Kirriemuir and A. McFarlane, "Literature review in games and learning," Tech. Rep. 8, 2004.

[17] E. Ye, C. Liu, and J. A. Polack-Wahl, "Enhancing software engineering education using teaching aids in 3-D online virtual worlds," in *Proceedings of the 37th Annual Frontiers In Education Conference—Global Engineering: Knowledge Without Borders, Opportunities Without Passports (FIE' 07)*, pp. T1E-8–T1E-13, Milwaukee, Wis, USA, October 2007.

[18] B. Wu, A. I. Wang, and Y. Zhang, "Experiences from implementing an educational MMORPG," in *Proceedings of the 2nd International IEEE Consumer Electronic Society's Games Innovation Conference (GIC '10)*, pp. 1–8, December 2010.

[19] A. Baker, E. O. Navarro, and A. van der Hoek, "Problems and programmers: an educational software engineering card game," in *Proceedings of the 25th International Conference on Software Engineering*, pp. 614–619, May 2003.

[20] F. McCown, "Teaching a game programming class for the first time: tutorial presentation," *Journal of Computing Sciences in Colleges*, vol. 25, no. 5, pp. 131–132, 2010.

[21] C. Leska and J. Rabung, "Learning O-O concepts in CS I using game projects," in *Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, vol. 36, p. 237, June 2004.

[22] E. Ferguson, B. Rockhold, and B. Heck, "Video game development using XNA game studio and C#.Net," *Journal of Computing Sciences in Colleges*, vol. 23, no. 4, pp. 186–188, 2008.

[23] R. H. Seidman, "Alice first: 3D interactive game programming," *ACM SIGCSE Bulletin*, vol. 41, no. 3, p. 345, 2009.

[24] X. Fu, S. Doboli, and J. Impagliazzo, "Work in progress—a sandbox model for teaching entrepreneurship," in *Proceedings of the 40th IEEE Annual Frontiers in Education Conference (FIE '10)*, pp. F2C-1–F2C-2.

[25] M. Kolling, "Greenfoot: introduction to Java with games and simulations," *Journal of Computing Sciences in Colleges*, vol. 25, no. 3, p. 117, 2010.

[26] A. Azemi and L. L. Pauley, "Teaching the introductory computer programming course for engineers using Matlab," in *Proceedings of the 38th Annual Frontiers in Education Conference (FIE '08)*, pp. T3B-1–T3B-23, October 2008.

[27] A. Pardo and C. D. Kloos, "Deploying interactive e-labs for a course on operating systems," in *Proceedings of the 6th Conference on Information Technology Education*, pp. 71–78, Newark, NJ, USA, 2005.

[28] P. Rooney, K. C. O'Rourke, G. Burke, B. Mac Namee, and C. Igbrude, "Cross-disciplinary approaches for developing serious games in higher education: frameworks for food safety and environmental health education," in *Proceedings of the Conference in Games and Virtual Worlds for Serious Applications (VS-GAMES '09)*, pp. 161–165, March 2009.

[29] A. W. B. Furtado, G. D. de Andrade, A. R. G. do Amaral Leitão et al., "Cegadef: a collaborative educational game development framework," in *Proceedings of the Conference on Interaction Design and Children*, Preston, UK, 2003.

[30] H. C. Yang, "A general framework for automatically creating games for learning," in *Proceedings of the 5th IEEE International Conference on Advanced Learning Technologies (ICALT '05)*, pp. 28–29, July 2005.

[31] K. Kardan, "Computer role-playing games as a vehicle for teaching history, culture, and language," in *Proceedings of the Association for Computing Machinery's Special Interest Group on Computer Graphics and Interactive Techniques Symposium on Videogames*, pp. 91–93, Boston, Mass, USA, July 2006.

[32] S. Arakawa and S. Yukita, "An effective agile teaching environment for java programming courses," in *Proceedings of 36th Annual Frontiers in Education Conference*, pp. 13–18, October 2006.

[33] W. W. Y. Lau, G. Ngai, S. C. F. Chan, and J. C. Y. Cheung, "Learning programming through fashion and design: a pilot summer course in wearable computing for middle school students," *SIGCSE Bulletin Inroads*, vol. 41, no. 1, pp. 504–508, 2009.

[34] S. V. Delden, "Industrial robotic game playing: an AI course," *Journal of Computing Sciences in Colleges*, vol. 25, no. 3, pp. 134–142, 2010.

[35] P. H. Tan, C. Y. Ting, and S. W. Ling, "Learning difficulties in programming courses: undergraduates' perspective and perception," in *Proceedings of the International Conference on Computer Technology and Development (ICCTD '09)*, pp. 42–46, November 2009.

[36] T. E. Daniels, "Integrating engagement and first year problem solving using game controller technology," in *Proceedings of the 39th IEEE Annual Frontiers in Education Conference (FIE '09)*, pp. 1–6, October 2009.

[37] A. Striegel and D. van Bruggen, "Work in progress—development of a HCI course on the microsoft surface," in *Proceedings of the 40th Annual Frontiers in Education Conference (FIE '10)*, pp. S3F-1–S3F-6, October 2010.

[38] A. Wang, "Interactive game development with a projector-camera system," in *Proceedings of the 3rd International Conference on Technologies for E-Learning and Digital Entertainment*, pp. 535–543, Springer, 2008.

[39] J. Dempsey, K. Rasmussen, and B. Lucassen, "The instructional gaming literature: implications and 99 sources," Tech. Rep. 96-1, University of South Alabama, College of Education, 1996.

[40] J. Dempsey, B. Lucassen, W. Gilley et al., "Since Malone's theory of intrinsically motivating instruction: what's the score in the gaming literature?" *Journal of Educational Technology Systems*, vol. 22, no. 2, pp. 173–183, 1993-1994.

[41] R. Hays, "The effectiveness of instructional games: a literature review and discussion," Tech. Rep. 2005-004, Naval Air Warfare Center, Training Systems Division, Orlando, Fla, USA, 2005.

[42] M. C. V. Langeveld and R. Kessler, "Two in the middle: digital character production and machinima courses," *SIGCSE Bulletin Inroads*, vol. 41, no. 1, pp. 463–467, 2009.

[43] W. L. Honig and T. Prasad, "A classroom outsourcing experience for software engineering learning," *SIGCSE Bulletin Inroads*, vol. 39, pp. 181–185, 2007.

[44] S. Hrastinski, "What is online learner participation? A literature review," *Computers and Education*, vol. 51, no. 4, pp. 1755–1765, 2008.

[45] S. Papert, *Mindstorms: Children, Computers, and Powerful Ideas*, Basic Books, New York, NY, USA, 1980.

[46] S. Puntambekar and J. L. Kolodner, "Toward implementing distributed scaffolding: helping students learn science from design," *Journal of Research in Science Teaching*, vol. 42, no. 2, pp. 185–217, 2005.

[47] J. Dewey, *Democracy and Education: An Introduction to the Philosophy of Education*, 2005.

[48] J. Dewey, *Experience and Education*, Simon and Schuster, New York, NY, USA, 1997.

[49] E. Smeets, "Does ICT contribute to powerful learning environments in primary education?" *Computers and Education*, vol. 44, no. 3, pp. 343–355, 2005.

[50] L. S. Vygotskij, *Mind in Society: The Development of Higher Psychological Processes*, Harvard University Press, Cambridge, Mass, USA, 1978.

[51] H. S. Barrows, "A taxonomy of problem-based learning methods," *Medical Education*, vol. 20, no. 6, pp. 481–486, 1986.

[52] B. Lennartsson and E. Sundin, "Fronesis—the third dimension of knowledge, learning, and evaluation," in *Proceedings of the 31st Annual Frontiers in Education Conference*, vol. 1, pp. T2B/14–T2B/19, October 2001.

[53] W. M. Thomas, "What makes things fun to learn? Heuristics for designing instructional computer games," in *Proceedings of the 3rd ACM SIGSMALL Symposium and the 1st SIGPC Symposium on Small Systems*, Palo Alto, Calif, USA, 1980.

[54] A. I. Wang, "Extensive evaluation of using a game project in a software architecture course," *ACM Transactions on Computing Education*, vol. 11, no. 1, article 5, 2011.

[55] A. D. Ritzhaupt, "creating a game development course with limited resources: an evaluation study," *ACM Transactions on Computing Education*, vol. 9, no. 1, pp. 1–16, 2009.

[56] A. McGovern and J. Fager, "Creating significant learning experiences in introductory artificial intelligence," *SIGCSE Bulletin Inroads*, vol. 39, pp. 39–43, 2007.

[57] R. Angotti, C. Hillyard, M. Panitz, K. Sung, and K. Marino, "Game-themed instructional modules: a video case study," in *Proceedings of the 5th International Conference on the Foundations of Digital Games (FDG '10)*, pp. 9–16, Monterey, Calif, USA, June 2010.

[58] G. Fesakis and K. Serafeim, "Influence of the familiarization with "scratch" on future teachers' opinions and attitudes about programming and ICT in education," in *Proceedings of the 14th Annual ACM SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE '09)*, pp. 258–262, Paris, France, July 2009.

[59] Y. Rankin, A. Gooch, and B. Gooch, "The impact of game design on students' interest in CS," in *Proceedings of the 3rd International Conference on Game Development in Computer Science Education (GDCSE '08)*, pp. 31–35, Miami, Fla, USA, March 2008.

[60] M. Al-Bow, D. Austin, J. Edgington et al., "Using game creation for teaching computer programming to high school students and teachers," *ACM SIGCSE Bulletin*, vol. 41, pp. 104–108, 2009.

[61] M. Eagle and T. Barnes, "Experimental evaluation of an educational game for improved learning in introductory computing," *SIGCSE Bulletin Inroads*, vol. 41, no. 1, pp. 321–325, 2009.

[62] W. K. Chen and Y. C. Cheng, "Teaching object-oriented programming laboratory with computer game programming," *IEEE Transactions on Education*, vol. 50, no. 3, pp. 197–203, 2007.

[63] Yulia and R. Adipranata, "Teaching object oriented programming course using cooperative learning method based on game design and visual object oriented environment," in *Proceedings of the 2nd International Conference on Education Technology and Computer (ICETC '10)*, pp. V2-355–V2-359, June 2010.

[64] R. Lawrence, "Teaching data structures using competitive games," *IEEE Transactions on Education*, vol. 47, no. 4, pp. 459–466, 2004.

[65] J. Huang, "Improving undergraduates' teamwork skills by adapting project-based learning methodology," in *Proceedings of the 5th International Conference on Computer Science and Education (ICCSE '10)*, pp. 652–655, August 2010.

[66] B. Wu, A. I. Wang, J. E. Strøm, and T. B. Kvamme, "An evaluation of using a game development framework in higher education," in *Proceedings of the 22nd Conference on Software Engineering Education and Training (CSEET '09)*, pp. 41–44, February 2009.

[67] J.-F. Weng, S.-S. Tseng, and T.-J. Lee, "Teaching boolean logic through game Rule tuning," *IEEE Transactions on Learning Technologies*, vol. 3, no. 4, pp. 319–328, 2010.

[68] R. Owston, H. Wideman, N. S. Ronda, and C. Brown, "Computer game development as a literacy activity," *Computers and Education*, vol. 53, no. 3, pp. 977–989, 2009.

[69] I. J. Timm, T. Bogon, A. D. Lattner, and R. Schumann, "Teaching distributed artificial intelligence with RoboRally," in *Multiagent System Technologies*, vol. 5244, pp. 171–182, Springer, Berlin, Germany, 2008.

[70] L. Werner, J. Denner, M. Bliesner, and P. Rex, "Can middle-schoolers use Storytelling Alice to make games? Results of a pilot study," in *Proceedings of the 4th International Conference on the Foundations of Digital Games (ICFDG '09)*, pp. 207–214, Orlando, Fla, USA, April 2009.

[71] K. Wang, C. McCaffrey, D. Wendel, and E. Klopfer, "3D game design with programming blocks in StarLogo TNG," in *Proceedings of the 7th International Conference on Learning Sciences*, pp. 1008–1009, Bloomington, Ind, USA, 2006.

[72] C. H. Huang, P. C. Ho, and S. M. Chung, "Computer game programming course for art design students by using flash software," in *Proceedings of the International Conference on Cyberworlds (CW '08)*, pp. 710–713, September 2008.

[73] B. Lennartsson and E. Sundin, "Experience from a course aiming at understanding system development with focus on system design and integration," in *Proceedings of the 32nd Annual Frontiers in Education*, vol. 1, pp. T3G-1–T3G-6, November 2002.

[74] J. Ryoo, F. Fonseca, and D. S. Janzen, "Teaching object-oriented software engineering through problem-based learning in the context of game design," in *Proceedings of the 21st*

Conference on Software Engineering Education and Training (CSEET '08), pp. 137–144, April 2008.

[75] J. Robertson and C. Howells, "Computer game design: opportunities for successful learning," *Computers and Education*, vol. 50, no. 2, pp. 559–578, 2008.

[76] W.-C. Chang and Y.-M. Chou, "Introductory C programming language learning with game-based digital learning," *Advances in Web Based Learning*, vol. 5145, pp. 221–231, 2008.

[77] S. Kurkovsky, "Can mobile game development foster student interest in computer science?" in *Proceedings of the 1st International IEEE Consumer Electronic Society's Games Innovation Conference (ICE-GIC '09)*, pp. 92–100, August 2009.

[78] K. J. Bierre and A. M. Phelps, "The use of MUPPETS in an introductory java programming course," in *Proceedings of the 5th Conference on Information Technology Education (CITC5 '04)*, pp. 122–127, Salt Lake City, Utah, USA, October 2004.

[79] B. Wu, A. I. Wang, J. E. Strøm, and T. B. Kvamme, "XQUEST used in software architecture education," in *Proceedings of the 1st International IEEE Consumer Electronic Society's Games Innovation Conference (ICE-GIC '09)*, pp. 70–77, August 2009.

[80] B. Wu, A. I. Wang, A. H. Ruud, and W. Z. Zhang, "Extending google android's application as an educational tool," in *Proceedings of the 3rd IEEE International Conference on Digital Game and Intelligent Toy Enhanced Learning (DIGITEL '10)*, pp. 23–30, April 2010.

[81] H. C. Jiau, J. C. Chen, and K. F. Ssu, "Enhancing self-motivation in learning programming using game-based simulation and metrics," *IEEE Transactions on Education*, vol. 52, no. 4, pp. 555–562, 2009.

[82] A. Garrido, J. Martinez-Baena, R. Rodriguez-Sanchez, J. Fdez-Valdivia, and J. A. Garcia, "Using graphics: motivating students in a C++ programming introductory course," in *Proceedings of the 20th European Association for Education in Electrical and Information Engineering Annual Conference (EAEEIE '09)*, pp. 1–6, June 2009.

[83] A. Barella, S. Valero, and C. Carrascosa, "JGOMAS: new approach to AI teaching," *IEEE Transactions on Education*, vol. 52, no. 2, pp. 228–235, 2009.