Game Development for Computer Science Education

Chris Johnson
University of Wisconsin, Eau Claire
johnch@uwec.edu

Monica McGill
Bradley University
mmcgill@bradley.edu

Durell Bouchard
Roanoke College
bouchard@roanoke.edu

Michael K. Bradshaw
Centre College
michael.bradshaw@centre.edu

Víctor A. Bucheli
Universidad del Valle
victor.bucheli@correounivalle.edu.co

Laurence D. Merkle
Air Force Institute of Technology
laurence.merkle@afit.edu

Michael James Scott
Falmouth University
michael.scott@falmouth.ac.uk

Z Sweedyk
Harvey Mudd College
z@cs.hmc.edu

J. Ángel Velázquez-Iturbide
Universidad Rey Juan Carlos
angel.velazquez@urjc.es

Zhiping Xiao
University of California at Berkeley
patricia.xiao@berkeley.edu

Ming Zhang
Peking University
mzhang_cs@pku.edu.cn

ABSTRACT

Games can be a valuable tool for enriching computer science education, since they can facilitate a number of conditions that promote learning: student motivation, active learning, adaptivity, collaboration, and simulation. Additionally, they provide the instructor the ability to collect learning metrics with relative ease. As part of 21st Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2016), the Game Development for Computer Science Education working group convened to examine the current role games play in computer science (CS) education, including where and how they fit into CS education. Based on reviews of literature, academic research, professional practice, and a comprehensive list of games for computing education, we present this working group report. This report provides a summary of existing digital games designed to enrich computing education, an index of where these games may fit into a teaching paradigm using the ACM/IEEE Computer Science Curricula 2013 [13], and a guide to developing digital games designed to teach knowledge, skills, and attitudes related to computer science.

1. INTRODUCTION

As part of ITiCSE 2016, the Game Development for Computer Science Education working group convened to examine the current role games play in computer science (CS) education, including where and how they fit into CS education. To guide our discussions and analysis, we began with the following question: in what ways can games be a valuable tool for enriching computer science education?

In our work performed prior to our first face-to-face meeting, we reviewed over 120 games designed to teach computing concepts (which is available for separate download [5]) and reviewed several dozen papers related to game-based learning (GBL) for computing. Hainey [57] found that there is "a dearth of empirical evidence in the fields of computer science, software engineering and information systems to support the use of GBL." This is not unique to CS, however. A review by Papastergiou [95] found limited evidence to support games for learning, and a systematic review by Graafland et al. [54] also found no empirically validated games to support education in the medical field.

Though our reviews were not designed to be comprehensive, our findings support these claims. This lack of evidence prevented us from identifying which of these games are most effective in meeting educational outcomes, because little evidence exists to make such claims, and it further prevented us from stating which game design frameworks for CS education might be most effective across various demographics. This required us to rethink our approach and to consider how we would analyze the games and the relevant research in a way that would provide significant value for the broader computer science educational research community.

The purpose of this working group report, therefore, is:

1. to provide a summary of existing digital games designed to enrich computing education and an index of where these games may fit into a teaching paradigm using the ACM/IEEE Computer Science Curricula 2013 (CS2013), and
2. to provide a guide to developing digital games designed to teach knowledge, skills, and/or attitudes related to computer science.

ITiCSE '16 July 11–13, 2016, Arequipa, Peru

 ISBN 123-4567-24-567/08/06…$15.00
DOI: https://doi.org/10.1145/3024906.3024908

To narrow the broad scope of games, we have chosen to focus on digital games; however, we note that analog games (board games, card games, etc.) can also be an important tool in enriching student learning. Games like those presented in CS Unplugged [22] provide a meaningful way of implementing active learning within a curriculum [23].

Though an initial goal for this report was to also include a guide for evaluating the effectiveness of games for CS education, given the expansiveness of such a task, we will focus on evaluation and assessment in follow-up work.

Game developers, educational theorists, and others may find value within this report. However, the primary target audience is academic researchers interested in developing games for enriching CS education. This perspective is reflected throughout this report.

This working group is unconventional in that its work will span two ITiCSE conferences. This first report is divided into several sections, starting with a background that introduces important vocabulary related to game design theory, provides a summary of previous related computer science education research, and provides a summary of relevant educational psychology. It also provides a brief review of games created for computer science education and how these map to CS2013.

This is followed with case studies of four games designed to teach CS concepts. The games are analyzed in two ways: 1) for their design elements using the Mechanics, Dynamics, and Aesthetics (MDA) framework and 2) for how one might evaluate their effectiveness. Using information from this analysis, our background research, and our previous experience creating games, we provide a set of best practices for creating meaningful games for CS education.

2. BACKGROUND

There is a unique relationship between digital games and computer science, since computer science is the foundation of such games. But more than that, the programming process itself contains many of the same elements found in games. In 1980, Thomas W. Malone [84] stated:

> "In some senses, computer programming itself is one of the best computer games of all. In the 'computer programming game,' there are obvious goals and it is easy to generate more. The 'player' gets frequent performance feedback (that is, in fact, often tantalizingly misleading about the nearness of the goal). The game can be played at many different difficulty levels, and there are many levels of goals available, both in terms of the finished product (whether it works, how fast it works, how much space it requires, etc.) and in terms of the process of reaching it (how long it takes to program, etc.). Self-esteem is crucially involved in the game, and there is probably the occasional emotional or fantasy aspects involved in controlling so completely, yet often so ineffectively, the behaviour of this responsive entity. Finally the process of debugging a program is perhaps unmatched in its ability to raise expectations about how the program will work, only to have the expectations surprisingly disappointed in ways that reveal the true underlying structure of the program."

Though other areas of study may also contain elements of this process, programming offers a unique parallel. Malone's insight into how self-esteem is intertwined into the process is worth noting, as CS educational research supports the notion that learner and instructor self-esteem and self-efficacy are important to the learning process. We mention this here, as it is important to note that although games for CS education may be designed to teach disciplinary concepts, such as programming constructs or computational thinking, they may have positive or negative unintended outcomes that could affect behaviors and beliefs about computing.

This section is designed to provide a contextual background to readers and describe the important elements of research relevant to the remainder of this report. We define the value of using games to teach computer science education. We also provide a summary of vocabulary for game design, as well as CS educational research and the broader educational psychology that is relevant to game design.

2.1 The Case for Educational Games

Games can facilitate learning across a variety of disciplines in multiple ways. Even games designed specifically for entertainment have been shown to have educational value. Though much has been written previously over the last couple of decades about games in education, we provide a brief synopsis for the case of using educational games for teaching computing.

Educational outcomes and competencies of modern education are changing, and the quickly changing nature of computing makes it important for educators to keep pace. Learners are growing up with laptops, tablets and cellphones. Today, people continuously learn and interact daily with information and communications technologies [27]. The modern workforce needs relevant education focused more on solving problems individually and in groups. Jobs are changing and are often characterized by increased technology use, extensive problem solving, networking and complex communication [79].

A recent ESA Essential Facts Report [45] finds that 65% of US households own a device used to play video games. Games are culturally relevant to today's learners, and previous research shows that learners may feel more engaged when culturally relevant tools are be harnessed for education [72]. Previous research demonstrates that digital games sustain engagement and motivation across time [52, 104], and that this engagement is strongly associated with student achievement [119]. In addition, students are more intrinsically motivated [53] and their work can focus on complex thinking and problem solving through games [19].

Traditional instruction can be improved by games given that they can foster collaboration, decision-making, problem-solving, communication, innovation, production, and procedural thinking [66, 115]. The game World of Warcraft, for instance, is an example of a game that drives individual specialization within cross-functional teams working collaboratively to meet goals [53]. Games have broad appeal and the evidence does not support many of the stereotypes perpetuated in popular media that only young males play digital games [129]. Approximately 41% of players are female, suggesting that technology acceptance is not a significant challenge for a wide demographic.

Like other media, they do not automatically do this just by virtue of being games. Freeman et al. [50] make a strong

case for why new active-learning approaches, such as games, are needed in STEM fields. Games can be a valuable tool for enriching computer science education, since they encompass a combination of motivation techniques, student engagement, adaptivity, simulation, collaboration and collection of performance metrics [55]. While traditional instruction is primarily focused on concepts and procedures, game-based learning in computer science can improve the ability for students to apply learning outside of the context in which it is learned, or transversal competence. Additionally, educational games are designed based on learning outcomes and can provide immediate feedback to the learner [40].

2.2 Establishing Vocabulary

What do we mean when we talk about game development for computer science education? As computer scientists, we tend to desire operational definitions capable of making clear distinctions and clear classifications. However, defining the term *game* is philosophically problematic and has a long history of contention and variation. Given the lack of agreement, Ellis [44] concludes:

> "The perplexing problem of how to define play will only be resolved by continually regenerating new definitions that fit current concepts of play behavior."

Arjoranta [15] similarly writes:

> "Games are a sociocultural phenomenon and, therefore, they should be defined and redefined in a hermeneutic circle that enhances our understanding of them."

There is even an online generator that proposes new definitions [8, 47]. Since the 1930s, philosophers and game scholars have proposed more than 60 definitions across multiple contexts [123]. However, the Wittgensteinian approach endorsed by Arjoranta [15] deviates from more essentialist positions to a position that is more pragmatic, framing definitions to deliberately focus on purpose, exclusion, and justification for a particular context and allowing them to be modified as practical needs and contexts change. In this section we further examine how games have been defined.

2.2.1 Games

Strenos [123] observes ten key "points of interest" in the way that game definitions vary. These are: rules, purpose and function, artifact or activity, separate or connected, the role of the player, productivity, competition and conflict, goals and end conditions, construction of category, and coherence. Using these dimensions, we constructed our own conceptual framework and identified several definitions that resembled and delimited, to a small extent, the notion of an artifact that would provide a game-like experience for an educational purpose. Specifically, we adapted and refined those proposed by Juul [67], Deterding [38], and Suites [125] to fit our desired practical context. Thus, we arrive at the following proposed definition for a game for computing education:

> A digital tool expressly designed to directly facilitate the development of computing knowledge, skills, and dispositions within a lusory context by leveraging the principles and elements of gameful design.

We use the term *digital* in order to exclude non-digital games, such as board games like c-Jump and activities like CS Unplugged. Scope is a major consideration of this report, and though we recognize the value that non-digital games have in education, we leave the evaluation of non-digital games for computing education to future research.

The term *tool* is used deliberately to emphasize that the artifacts we are studying are merely educational tools, similar to a book or an in-class exercise. Games are not a panacea that will replace educators, nor are they currently self-contained intelligent tutoring systems. As such, these tools potentially have many modes of deployment which need to be considered in design, including in-class activities, alternatives to homework tasks, and general out-of-class practice.

The clause *expressly designed...leveraging the principles and key elements of gameful design* refers to the use of game elements being explicit and complete. There are many e-learning tools and gamified applications that may include some elements of gameful design, while excluding many others [39]. As such, we focus on those key elements that comprise a full game: "rules, goals, and variable quantifiable [positive and negative] outcomes" [67].

Callois [28] makes the distinction between *paidea*, which is play characterised by no immediate structure or well-defined objective, and *ludus*, which is play characterised by structured rules where players strive to achieve a fixed goal. To this end, we use the term *directly facilitate* to emphasise that our aim is to study *ludus*. In our context, this means that we exclude tools like Scratch—which facilitate free and open learning based on *paidea*. The games we examine embed specific learning objectives and incorporate some form of scaffolding that aids learners to achieve the objectives.

We borrow the term *lusory context* from Suites [125] who uses the term "lusory attitude" to describe players engaged in gameplay. Specifically, we mean the construction of some imagination framework and its adoption during play. This is related to Huizinga's [62] notion of the *magic circle*, which he defines as, "a shield of sorts, protecting the fantasy world from the outside world" [29]. However, we emphasise the word *context* because this membrane is permeable, blurring boundaries between entertainment and learning [132].

The definition is proposed to frame the goal of the current working group and to encapsulate its activities. We anticipate that this definition will become more refined as discourse on game development in computer science education matures. Furthermore, as new evidence emerges to reveal the properties of games that are useful to educators, the very notion of our object of study could shift away from games. Instead, it could focus on artifacts that adopt useful properties from a wide range of immersive and interactive media.

2.2.2 Formal Elements and the MDA Model

Beyond defining games, a further complication is establishing the vocabulary of game design brought about through the formal study of games. Game studies has yet to mature, which is to be expected, as 2001 marked the year when game scholars declared game studies a self-contained field of study [12]. However, as Costikyan [33] notes in his seminal paper, "I have no words, but I must design," this youth brings complications. The *lingua franca* of games is not firmly established. That is, designers' understanding of

Player	The system requires players which it interacts with. There are usually multiple operators, who can be people or computers.
State	The system has different conditions.
Rules	Rules are operational, constitutive, and implied methods of state transition [106]. They control the setup of the game, the progression of play, and the resolution.
Sequence	The flow of state transition. Some systems are turn-based, while others are real-time, or some combination.
Representation	The system has a means to represent its state using tokens.
Goal	The end-state of the system, which the players strive towards.
Decisions	The system must present players with "a series of interesting choices" [87]. Often this by presenting obstacles or providing some form of challenge.
Interface	The system requires an interface which facilitates feedback (output) and response (input).

Table 1: Adapted from Salen and Zimmerman [106] and Schreiber and Braithwaite [109].

key terms varies and this complicates the communication of design knowledge and how we describe game experiences. For example, terms such as *gameplay* are used in ambiguous ways and, in the absence of a shared understanding, lack practical utility.

This problematizes what we refer to when we discuss the "principles" and "key elements" of gameful design. A game overall can be characterised by a *system of interaction* which possesses abstract qualities like player intention, perceivable consequence, and narrative [31]. Even such a simple notion, however, has weaknesses. The term *narrative* could refer to the story embedded in the game by the designer, but it is also commonly used in a way that includes the emergent story created by players. So, to clarify what these principles and key elements are, we extend Church's [31] notions and summarise them in Table 1.

These elements define the aspects of a game that designers need to consider. However, it is not clear how they work together to form a game. To aid in this endeavour, several models exist (e.g., [107, 65]). One such model, which has seen widespread adoption by both game scholars and games industry professionals, is the MDA Model [65]. This is the notion that a game experience can be understood in terms of three interconnected and interrelated concepts: mechanics; dynamics; and aesthetics.

The *mechanics* are the specific parts of a game design, comprised of elements such as the rules of the game. These could be the game's possible states, transitions from one state to another, ways in which players trigger these transitions, representation of state, and so on. In other words, they are the parts of the game that limit or restrict the player's actions, control the flow of the game, and encode some kind of meaning. The *dynamics* refer to the behaviors of the player-game system as a game is being played. In other words, the actual flow of the game. This flow emerges from the interaction of the mechanics with each other in response to player action. The *aesthetics*, then, refer to the resulting experience of playing the game. This is conceived from the perspective of a player as they play. This is distinct from the theme of the game and the "look and feel" of the user interface. It instead refers to the player's experience itself. It is sometimes framed as the set of emotions that a game may evoke.

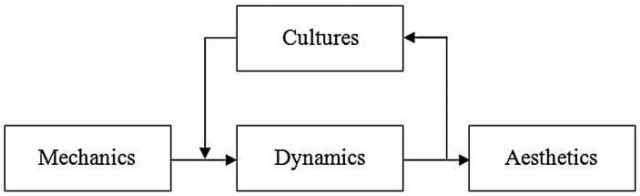

Figure 1: The interplay of mechanics, dynamics, aesthetics, and culture that is used to inform a game's design.

The mechanics drive the dynamics because they constrain the actions the player may take. However, the player also has a highly influential role in the formation of the game dynamics, since games typically provide players with a choice of possible actions. Further, the skills of some players may be insufficient to sustain certain dynamics, while other players discover subversive ways to exploit the available mechanics in ways that designers may not have anticipated. These different dynamics can have a profound effect on the resulting aesthetic of the game. Again, there is variance. For example, solving a puzzle may evoke satisfaction and pride. However, if the puzzles are too repetitive, they may evoke boredom and annoyance. Players can also differ in the way they perceive an experience. Figure 1 illustrates an adaptation of MDA, where Scott [110] proposes culture as a moderating factor alongside dynamics. On the one hand, culture may shape a player's behaviour, while on the other, culture may shape a player's values and thereby influence how they interpret signs and symbols to construct meaning. Hunicke et al. [65] state:

> "From the designer's perspective, the mechanics give rise to dynamic system behavior, which in turn leads to particular aesthetic experiences. From the player's perspective, aesthetics set the tone, which is born out in observable dynamics and eventually, operable mechanics."

It is also important to highlight a game designers' focus on aesthetics. As computer science students and educators, it may be tempting to focus primarily on the mechanics, since art, sound, and user interface design are typically not components of computer science education. However, the aesthetics are often what makes a game fun and more engaging, and considering aesthetics can be pivotal to a game's success.

2.2.3 Players

As with teaching, the target audience of a game is of primary consideration. Players moderate the aesthetics of the games they play, as their value systems will shape their interpretations, and different dynamics may be accessed by different demographics [133, 110]. Several player models have

been considered and used in designing games for different target demographics. Each of the following player models with various levels of complexity have been used to identify player preferences, providing information to the designers who can then integrate those preferences within their game. Though not a comprehensive list, this provides some insight into how to consider player preferences within gameplay. When designing a game, designers can link gameplay features to specific player preferences to achieve specific goals. For example, a game designer's target demographic may be achievers, who are known to enjoy competition and the ability to achieve high scores and earn rewards.

Bartle's Taxonomy. The Bartle's Test was developed by Andreasen and Downey, who based it on Richard Bartle's player type analysis [21]. This test is a questionnaire that was originally designed for players of virtual worlds to take to provide insight into their preferred play style. Four play styles are provided: Killers, Achievers, Socializers, Explorers [20]. Also referred to as Hearts, Diamonds, Clubs, and Spades, Bartle attempted to classify player preferences in an effort to predict players who may enjoy dungeon crawlers. Socializers (hearts) enjoy forming relationships with other players within a game. Achievers (diamonds) enjoy meeting goals in games and being rewarded for their efforts. Killers (clubs) enjoy dominating others in gameplay using in-game means. Explorers (spades) enjoy games that provide them with an opportunity to explore and enjoy free-play. These styles are not mutually-exclusive, and in fact, players often enjoy combinations of these styles.

BrainHex. The BrainHex [1] is a broader model that defines seven player archetypes based in part on neurobiological research: Seeker, Survivor, Daredevil, Mastermind, Conqueror, Socialiser, and Achiever [90]. More closely aligned with genres than Bartle's Taxonomy, it can still be used as a tool within a target demographic (e.g. girls aged 10–13 in a particular middle school setting) to help in determining the types of games they may prefer, thereby creating a game to meet the preferred player styles of that demographic.

Unified Model. The Bartle's Test has been further adapted to include the Keirsey Temperaments and the Bateman DG1 Model in a Unified Model, as well as the alignment of these models with 7 others [124]. This also maps to the MDA framework of game design, which defines a game based on Mechanics, Dynamics, and Aesthetics. Motivation between the different types of players as well as their preferred method of solving problems to achieve goals differs. Their overall goals (Do, Have, Know, or Become) provide a single action word summary into their play styles.

Five Factor Personality Traits. More recently, the Five Factor Personality Traits survey, a validated instrument used in psychology, has been used to predict game play preferences and in designing games. Game designers at Ubisoft employ the Five Factor Personality Inventory (also called OCEAN or the Big Five) to measure five opposing dimensions in an individual's personality, Openness (versus Closedness) to Experience, Conscientiousness versus Lack of Conscientiousness, Extraversion versus Introversion, Agreeableness versus Hostility, and Neuroticism versus Emotional Stability [126]. The Inventory has been used in one study to show to that "conscientiousness was negatively correlated with perceived use of first-person shooter games and extraversion was positively correlated with both liking and perceived ease of dancing games. Agreeableness was positively correlated with liking of dancing games" [37]. An alternative to OCEAN is HEXACO, which assigns personal characteristics through Humility (H), Emotionality (E), Extraversion (X), Agreeableness (A), Conscientiousness (C), and Openness to Experience (O) [75].

Other psychosocial models. Though there are several ways to define player styles, other psychosocial models can be used. For example, Hofstede's cultural model has four dimensions: power distance (weak or strong), uncertainty avoidance (weak or strong), individualism (versus collectivism), and masculinity (versus feminism) [60]. One could form the basis for analyzing player styles and preferences using such models.

These models encourage examining player preferences often based on personality, in a very similar way that culturally relevant pedagogy theory is used to develop curriculum [72]. These models move beyond personal biological identities, with gender, race, and ethnicity, for example, not given consideration. In a classroom with learners, however, additional considerations may need to be made: environment, socioeconomic influences, instructor abilities and motivation, and more [57]. If seen as a tool or only one form of several medium used in the classroom, games should also be contextualized by the instructors. That is, the instructor is still in control of the the learning process and interpreting the in-game learning externally provides a method of instructors integrating this knowledge—which is also referred to as pedagogical content knowledge [118].

2.3 Effect on students

Educational games may produce a number of learning and psychological effects. These effects may be intended or unintended. We list the most relevant effects of educational games, as well as representative measures of these effects:

- Cognitive development. This effect can be summarized by saying that games may present an opportunity for students to learn more deeply. For example, according to Bloom's taxonomy [14], there are six levels of increasing cognitive development, or according to the SOLO taxonomy [24], a student may give responses to a task in five levels. Cognitive development can be measured by means of different variables, such as:
  - Assessment performance.
  - Time necessary to accomplish a task.
  - Accuracy in performing a task.
- Affective and motivational effects. This category clusters a number of subjective phenomena:
  - Acceptance. Technology acceptance is a relevant issue in the area of work and training [36]. Some students (especially adult learners) may refuse the use of games in education.
  - Emotions experienced. There is no universally accepted classification of emotions. Some authors have proposed a set of basic emotions, other being composed of the basic ones. Thus, Zinck and Newen argue that the four basic emotions are joy, anger, fear and sadness [134].
  - Motivation. According to the theory of self-determination, there are four classes of motivation

towards a subject matter or activity: intrinsic, extrinsic via identified regulation, extrinsic via external regulation, and amotivation.

- Transversal skills or competences. In recent years, concern about these general skills has increased. Some examples are self-efficacy, communication or leadership skills.
- Behavior change. Computing students may develop undesirable behaviors, such as cheating or hacking. Educational games can guide students to consider their behaviors in their academic and professional development.

Other effects can also be achieved with games but are not directly relevant to CSE: motor skills (e.g. motor coordination), perceptual and cognitive effects (e.g. attention, or visual or spatial skills), and physiological effects (e.g. heart rate).

3. SURVEY

We examined over 100 unique games that have been or are currently available for use in teaching computing concepts. Each review consisted of inspecting available documentation and commentary and, when possible, playing the game. Using this information, we categorized each game with respect to numerous pedagogical and game characteristics [5]. These games were published over the last several decades, ranging from 1982 to 2016. Of the games reviewed, 34 are available commercially, 51 are freely available, and 15 no longer appear to be available. In this section, we provide a general summary of the games and a classification of the games in context of the ACM/IEEE Computer Science Curricula 2013 [13].

3.1 Summary of Games for CS Education

With respect to the general computing topics addressed by the games, just over half (64) focus explicitly on some aspect of programming. Not surprisingly, within this group, topics typical of introductory programming classes are addressed by numerous games: arrays, assignment, data types, debugging, encapsulation, event handling, expressions, I/O, iteration, modularization, object orientation, parameters, recursion, selection, sequence, testing, and variables. There are also more specialized programming-related games that address artificial intelligence, algorithms, bottlenecks, ciphers, concurrency, critical thinking, fault tolerance, instruction sets, interprocessor communication, messaging, memory access, multiagent systems, problem recognition, registers, sorting, synchronization, tree traversal, and other topics.

We found that 17 of the games focus primarily on computational thinking, in the sense defined by Wing [130]. Of the games that focus on neither programming nor computational thinking, the topics addressed include artificial intelligence, architecture, circuits, data types, security, sensors, and systems. As part of the review process, we attempted to identify specific learning outcomes addressed by each game through either literature or by playing the game. Though these details are provided in the online appendix [5], they could not be summarized in a meaningful way.

We characterized the games in terms of the e-Learning Goals they claimed or that we inferred based on our experimentation. Using Clark and Mayer's cognitive task analysis [32], we identified "Inform" level goals for almost half of the games (55), "Perform Procedure Tasks" for more than half (73), and "Perform Strategic Tasks" for almost half (55).

A majority of the games reviewed specify their targeted demographics in terms of either age group or educational level. All age levels from 3 to adult and grade levels from pre-K to post-graduate are covered. Classifying the games using the five stages of experience levels, novice, advanced beginner, competent, proficient, expert, a fair number of games state that the target specific levels of prior aptitude, education, familiarity, or interest in either computing or gaming [42]. 61 (59.8%) targeted novices, 59 (57.8%) targeted advanced beginners, 23 (22.5%) targeted those competent in computing concepts, and 2 (2.0%) targeted those proficient in computing concepts. We found documentation indicating that two of the games (1.9%) were created to specifically target females.

The games reviewed include representatives of a wide variety of genres, including action, adventure, arcade, board, dance, MMO (massively multiplayer online), puzzle, RPG (role playing game), (turn-based/real-time) strategy. Likewise, the user interface styles employed vary widely, including command line, drag-and-drop, first/third person graphical (key-based, mouse-based, controller-based), and point-and-click. Finally, there is substantial variety in the game mechanics. For example, many games are either single player or competitive multiplayer, but some are cooperative multiplayer. Also, within those games that use drag-and-drop interfaces, in some cases the objects being manipulated represent machine instructions, while in others they are game-specific actions.

Many of the games reviewed are based on established game engines, such as Bioware Aurora, Codea, LibGDX, Moai, OpenFL, Pygame, RPGMaker, Spring engine, XP, Unity, and XNA. Others are built using identified languages and libraries such as .NET, C, C++, C#, CSS, Flash, Git SCM, HTML, iOS, Java, Javascript, Logo, Lua, Ruby, Scratch.

3.2 Games mapped to CS 2013 Categories

We analyzed the games in context of the ACM/IEEE Computer Science Curricula 2013 to identify which of the 18 computing knowledge areas that each game targeted [13]. For the games that we were able to classify, the majority (75.5%) could be used in teaching Software Development Fundamentals (SDF). Nearly one-third (28.4%) taught Algorithms and Complexity (AL) concepts. Table 2 shows the breakdown of the categories. Note that some games could be used to teach concepts in two or even three categories. Additionally, it is worth noting that none of the games reviewed target the following areas: IM (Information Management), NC (Networking and Communications), OS (Operating Systems), and PD (Parallel and Distributed Computing).

We analyzed this further and drilled down to the concept areas. The vast majority of the games target concepts in the SDF category. Within the three concept areas targeted in SDF, the concept areas of Fundamental Programming Concepts (46 or 45.1% of all games reviewed), Algorithms and Design (25 or 24.5% of all games reviewed), and Algorithmic Strategies (19 or 18.6% of all games reviewed) have the most games suitable for teaching these concepts (Table 3).

Based on prior evidence that games are useful for inspiring student interest, it is natural that most of the games are designed for beginners, and thus SDF and AL, which

Knowledge Area	Concepts	#
AR (Architecture and Organization)	Digital Logic and Digital Systems	1
AL (Algorithms and Complexity)	Algorithmic Strategies	19
	Fundamental Data Structures and Algorithms	10
	Advanced Data Structures, Algorithms, and	1
AR (Architecture and Organization)	Machine Level Representation of Data	2
	Assembly Level Machine Organization	1
CN (Computational Science)	Data, Information, and Knowledge	11
DS (Discrete Structures)	Basic Logic	3
GV (Graphics and Visualization)	Fundamental Concepts	10
HCI (Human-Computer Interaction)	Programming Interactive Systems	3
	Foundations	1
IAS (Information Assurance and Security)	Security Policy and Governance	1
	Network Security	2
	Threats and Attacks	1
	Cryptography	2
IS (Intelligent Systems)	Basic Search Strategies	2
	Fundamental Issues	1
PBD (Platform-Based Development)	Game Platforms	1
	Web Platforms	1
PL (Programming Languages)	Object-Oriented Programming	1
SDF (Software Development Fundamentals)	Fundamental Programming Concepts	46
	Algorithms and Design	25
	Fundamental Data Structures	3
SE (Software Engineering)	Tools and Environments	1
	Software Project Management	2
	Software Verification and Validation	1
	Software Reliability	1
SF (Systems Fundamentals)	Computational Paradigms	3
SP (Social Issues and Professional Practice)	Security Policies, Laws and Computer Crimes	1

Table 3: Games by Category and Area.

Knowledge Areas	Count
SDF (Software Development Fundamentals)	77
AL (Algorithms and Complexity)	29
CN (Computational Science)	11
GV (Graphics and Visualization)	10
AR (Architecture and Organization)	9
IAS (Information Assurance and Security)	6
SE (Software Engineering)	5
HCI (Human-Computer Interaction)	4
IS (Intelligent Systems)	4
SF (Systems Fundamentals)	3
DS (Discrete Structures)	3
PBD (Platform-Based Development)	2
SP (Social Issues and Professional Practice)	1
PL (Programming Languages)	1

Table 2: Number of Games classified in CS knowledge areas.

include many basic and introductory topics to computer science, would include the vast majority of the CS educational games. However, we note that there are considerable areas with few or no games that teach these concepts, leaving a wide variety of subjects that researchers could target in future games.

4. CASE STUDIES

Of the games that were evaluated by the group, we selected and performed a more rigorous analysis on four: The Foos, Human Resource Machine, Lightbot, and PicoBot. These games were chosen to represent well-designed games across a range of demographics, their quality, and their usefulness in appearing to meet their stated learning outcomes. We use these games as lenses into the array of games for computer science education, first describing each game using the MDA model and then comparing and contrasting the four in an effort to gauge how design elements affect the player's experience. This review serves as a backdrop for the next section in which we provide suggested best practices for designing games for use in computing education.

4.1 The Foos

The Foos, a commercial game developed by codeSpark [3], teaches basic programming concepts to students 5-10 years old. The game is available for iOS and Android. A simplified version can be played online. This game is endorsed by Code.org as an activity for the Hour of Code [2].

The Foos presents the player with a series of scenes featuring an avatar, some obstacles, some bonus rewards, and a final prize. The player constructs a series of moves her avatar can make in order to capture the prize. For example, in Figure 2, the avatar must jump on and off the wooden boxes in order to get the donut. The green collectibles are bonus rewards the player can collect along the way to the goal. For this level, the available instructions are *move* and *jump*. After constructing an appropriate sequence of steps, the player runs her program, which steps through the pro-

Figure 2: The tile-based programming editor that a player uses to guide the avatar to the donut in early level in the The Foos.

Figure 3: The heads-up display that reports the player's achievements in The Foos.

grammed movements with playful sound effects and music.

If the avatar reaches the prize, the avatar dances a victory jig, while points accumulate in the heads-up display shown in Figure 3. If the player constructs an erroneous sequence, the avatar still follows the instructions but does not reach the final prize. The player is allowed to modify the instruction sequence without penalty until she meets the goal.

Since the game is aimed at young children, it does not require reading skills. It provides visual hints to help the player understand the drag and drop interface and how the available instructions work. This also makes the game easily played by students whose first language may not be English.

As levels progress different types of instructions are introduced that allow the player to solve increasingly difficult puzzles. The game incorporates 10 different scenarios, each with specific learning objectives: sequences, commands, parameters, events, loops, efficiency, endless loops, conditional statements, and debugging.

4.2 Human Resource Machine

Human Resource Machine is a commercial game published in 2015 by Tomorrow Corporation [10], which is available for Windows, Mac, Linux, iOS and Android platforms. It is targeted for players age 9 and above.

The game opens outside the drab-colored office building,

Figure 4: The avatar executing an assembly program that outputs the maximum of each input pair.

whereupon the player chooses an employee avatar with large blinking eyes, dressed in a desaturated suit. Instead of choosing a name, the player chooses an impersonal employee number.

Work begins at once for this employee, who appears in a room with two conveyor belts. One belt is marked "In" and the other "Out." A manager sitting at a desk gives the employee her first task: move all the boxes from In to Out. While the look and feel of Human Resource Machine is very different than The Foos, the game mechanics are quite similar. The player must construct a sequence of moves the employee can follow to accomplish the objective.

In the first level the available commands are `->inbox` and `outbox->`. The former causes the employee to pick up a tile from the input conveyor belt and the latter causes the employee to place the tile she is holding on the output belt. The player drags and drops instructions to create a sequence in the program editor panel on the right. When the command sequence is executed, the employee runs between the two belts, picking up and dropping off tiles as directed. The interface is shown in Figure 4.

If the player fails to compose a sequence that outputs the expected tiles, the manager reprimands the employee. The player modifies the sequence without penalty until the expected and actual results match, at which point, a year passes and the employee is promoted to a more complex task. The player's progress is marked on the "corporate ladder," a game screen shown in Figure 5 that shows a vertically-oriented map of the game's levels.

In subsequent levels, new commands are gradually introduced that allow the employee to accomplish increasingly difficult operations on the input. Additional commands include jump, conditional jump, store or retrieve tiles, add or subtract tile values, increment and decrement tile values, and so on. The final "boss" is a sort routine.

When a player beats a level, a report is generated by management. The player's program is scored using two metrics: the number of instructions used in the program and the number of runtime steps required to execute it. This dual scoring reflects a tension commonly found in software development, in which one often chooses between minimizing code size and minimizing execution time. If the scores are below a certain threshold, the player's achievement is noted. The game informs the player that optimizing one metric may produce a suboptimal score for the other metric. Multiple

Figure 5: The corporate ladder that marks the player's progress in Human Resource Machine.

solutions will sometimes need to be written to earn both achievements, and the player may accordingly save multiple solutions.

By the time the employee has reached the top of the corporate ladder, the player has gained fluency in an 11-instruction assembly language.

The players of Human Resource Machine are not expected to have prior programming experience, as the developers state on the game's website: "Don't worry if you've never programmed before - programming is just puzzle solving. If you strip away all the 1's and 0's and scary squiggly brackets, programming is actually simple, logical, beautiful, and something that anyone can understand and have fun with!" They also identify the intended audience as "expert nerds."

The developers do not discuss particular learning outcomes or assessment. However, the comments from the game's reviews on Steam provide some informal but insightful responses from players:

- "Due to the fact that there is 'optimization challenges,' I do get the feeling I would return to some of the earlier puzzles to beat/match the best possible."
- "Everyone will probably enjoy this game for a little while, if you like solving abstract problems and basic programming you'll enjoy it more."
- "After I finished this, I realised that I'd accidentally learned how to write efficient assembly-code. Hooray!"
- "If you're a CS student that hasn't yet taken an assembly course, this is a great intro. The assembly language is very conventional (unlike TIS-100), so many of the skills and strategies you learn will be directly applicable to, say, MIPS or x86."

4.3 Lightbot

Lightbot is a game developed by Danny Yaroslavski [4], based on a game he first built in high school. It is available as an online flash game or for iOS and Android. The game comes in two versions, one targeted for players aged 4–8 and one for players age 8 and above. Lightbot is endorsed by Code.org as an Hour of Code [2] exercise.

The mechanics of this game are very similar to both The Foos and Human Resource Machine. In this game the player is presented with a grid of tiles and a robot, as shown in Figure 6. The player constructs a sequence of moves to

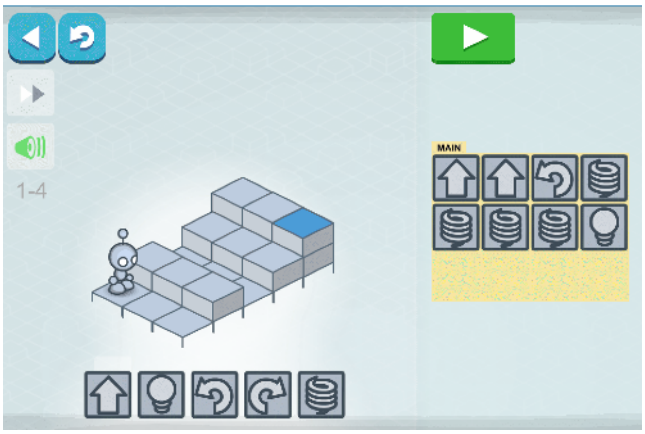

Figure 6: The interface of Lightbot. The player uses a tile-based program editor to navigate a robot across a world of blocks, some of which need to be illuminated.

direct the robot to traverse the grid. Available instructions include *move*, *turn*, *light*, and *jump*. As in the previous games, the player runs the program to step through the programmed sequence. The objective of the game is to have the robot light up every dark blue tile in the grid.

The space provided for the tile sequences is limited, forcing the player to consider code size. As the game progresses, the size constraint adds substantial complexity to the puzzles. Similar to The Foos, if the player constructs an erroneous sequence of moves, the robot will follow its directions but will not light up all the dark blue tiles. The player can revise and rerun a sequence without penalty. Later levels incorporate subroutines and recursive calls.

4.4 Picobot

Picobot [9] is a *Karel*-like [96] JavaScript game created by Zach Dodds and Wynn Vonnegut for their introductory computer science course at Harvey Mudd College. In the game, players write rules that instruct the green Picobot to navigate a two-dimensional grid world. The world consists of white, open cells that Picobot can traverse and blue wall cells that are impenetrable, as shown in Figure 7. When Picobot visits a cell, it turns gray. The goal of the game is to have the Picobot visit all of the white cells, turning them all gray.

The game models a finite state machine. The user defines rules that dictate the Picobot's next move and new state based on the current state and which cells are in the immediate neighborhood. For example, the rule "`0: Nxxx -> W 1`" specifies that if Picobot is in state 0 and only its northern neighbor is barred (blue) and its other neighbors are free (white), Picobot should move west and go to state 1. An asterisk can be used as a wildcard. For example, the rule "`1: *S** -> S 1`" specifies that if Picobot is in state 1 and the southern neighbor is open, Picobot should move there and remain in state 1. This rule effectively moves the Picobot south as long as south is open.

Like the previous games, Picobotintroduces the player to basic programming concepts, but using a declarative programming language. It also provides visual analogy for programming problems that provides immediate, clear feedback of both success and failure. The interface also has a textual

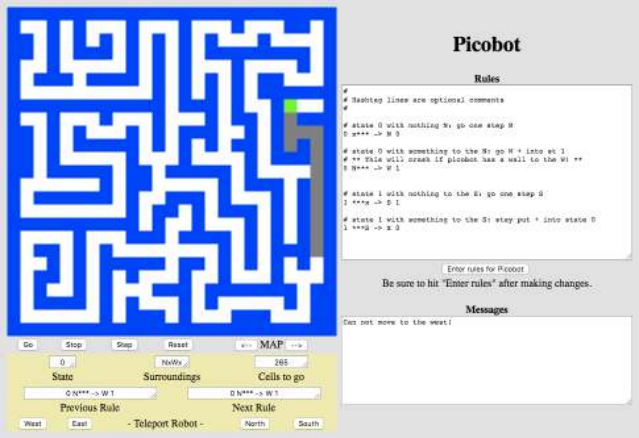

Figure 7: The interface for Picobot, whose 2D game world is navigated much like a Turing Machine with a 2D tape. The goal of each level is to visit all cells using only state machine rules based on neighborhood information.

display of the current state of the program including buttons to step through the execution that is helpful for debugging. *Picobot* has multiple maps with various levels of difficulty that encourage players to seek and solve additional challenges.

Picobot is not polished to the same degree as the previous games. Players enter rules using the keyboard. Its error messages can often be inscrutable to novice programmers. Picobotdoes not provide hints or have a builtin tutorial. It is intended to be introduced in class by the instructor.

4.5 Compare and contrast

The four games we reviewed are similar in many ways, but provide substantially different user experiences. These examples help illustrate the relationship between mechanics, dynamics, and aesthetics and how they work together to make a game unique.

4.5.1 Aesthetics

From the player's perspective, a game is an aesthetic experience. Hunicke, LeBlanc, and Zubek [65] suggest the following taxonomy for categorizing the user experience:

1. Sensation: Game as sense-pleasure
2. Fantasy: Game as make-believe
3. Narrative: Game as drama
4. Challenge: Game as obstacle course
5. Fellowship: Game as social framework
6. Discovery: Game as uncharted territory
7. Expression: Game as self-discovery
8. Submission: Game as pastime

This list is not meant to be exhaustive and for the purposes of educational games we can add a ninth experience:

9. Learning: Game as learning tool

These experiences are not mutually exclusive, in fact, games typically aim to deliver on many of these objectives.

All four of the games we reviewed provide challenge. In each case the difficulty of the challenge is targeted to a specific age group/audience.

The Foos and Human Resource Machine both have a narrative structure. It is strongest in Human Resource Machine where the player is likely to empathize with the hapless employee trapped in the dreary corporate world. In contrast the narrative structure of The Foos is simple and fun and clearly more appropriate for a younger audience.

All games provide some sort of sensation. The Foos and Human Resource Machine use high quality graphics and sound to create a rich sensation in the player. One can hardly resist being drawn into the joyous world of The Foos with its fun characters, bright graphics, fanciful animations, and cheerful sounds. In contrast Human Resource Machine uses desaturated colors, repetitive and mechanistic sounds, and mournful characters to convey the dreariness of the corporate world (and the advantages of automation). While graphics and sound are often considered superfluous in learning games, they can greatly enhance the user's experience [6].

Based on admittedly anecdotal evidence, all of our games we reviewed are effective learning tools for their specific learning objectives and the audiences they target. The Foos and Human Resource Machine provide a richer user experience than Lightbot and Picobot. But that is somewhat essential since they are teaching children programming concepts. The fun of Lightbot and Picobot is derived largely from solving difficult puzzles, which is perfectly fine for an older and possibly captive classroom audience. Who would play if the games weren't fun? But that richness is not always easy or inexpensive to achieve.

4.5.2 Dynamics

Game dynamics work to create the user experience. In these games, challenge is created by the difficulty of the puzzles the player must solve. The sense-pleasure of these games is tied to how effectively the game helps players navigate its challenges.

Each of the games has a progression of difficulty. The Foos, Human Resource Machine, and Lightbot introduce new mechanics across levels. Picobot has a fixed set of tools that are available from the start but the maps become more difficult to navigate over time.

Each of the games is effective in revealing state information to the player the game interface. The Foos, Human Resource Machine, and Lightbot provide a visible action associated with each command the player enters; a player can typically determine what went wrong and debug their code appropriately. Picobot's state machine simulation is more complicated but the game provides excellent debugging tools that let the player step through the state machine they have created.

The Foos has the most sophisticated in-game help systems with visual hints that help the player proceed. Human Resource Machine and Lightbot have non-player characters that provide directions. Picobot does not have in-game instruction. While some players may be able to figure out the gameplay on their own, many will require an instructor's guidance.

All of the games tolerate some variation in the successful solutions. The Foos is particularly forgiving about missteps;

an incorrect sequence played twice (without reset) will often have enough correct moves to achieve success. Because Human Resource Machine has two scoring metrics, it actually encourages players to consider alternative solutions; this contributes to aesthetic goals of Discover and Expression. Similarly, Lightbot grids often have numerous solutions though the constraint on sequence length can limit possibilities. Picobot provides the least constraints on the solution space; for some players this may be liberating but it also allows for complicated solutions that are difficult to debug. But once again, Picobot is intended for classroom use where an instructor can help get students past difficult hurdles.

Each of the games encourage computational thinking by initially focusing on simple problems the player must solve through a series of simple steps. Later levels provide more difficult problems and, in the case of the first three games, provide advanced techniques like recursion to help solve them.

4.5.3 Mechanics

Mechanics are the rules of the game and the actions and controls available to the player. The first three games have very similar mechanics; the player drags and drops instructions to creates a sequence their avatar follows to achieve some goal. This provides a good introduction to imperative programming.

Picobot is different than the other three games in that its rules are defined using a declarative programming language, which may not be appropriate for classrooms that focus on imperative programming. But this difference can also be a strength, particularly an introductory computer science course that aims to expose students to multiple programming language paradigms.

Human Resource Machine has the most sophisticated scoring system providing two separate metrics by which a solution can be evaluated: code size and execution time. Lightbot only focuses on the first of these metrics but does so with a very hard constraint; limiting the space to describe a solution effectively limits the player to efficient solutions. The Foos scoring system is the most forgiving; even wrong solutions can garner points. This is appropriate given its young target audience.

4.5.4 Summary

We consider each of these games to be fun learning tools for their intended audiences, and as such each has the potential to be effective. Games developed by professional designers/developers with access to considerable resources are going to have greater polish. But simple games that are designed for specific use in a classroom can be highly effective in their own right and, therefore, should be considered as another tool for enhancing learning.

5. SUGGESTED PRACTICES FOR DESIGNING GAMES FOR USE IN CS EDUCATION

Based on our review of games, our experience as computer science educators, our understanding of educational research, and our experience designing and developing games, we provide considerations for designing games for use in computer science education. We discuss useful and relevant aspects of educational frameworks to consider when designing, basic game production practices to aid a researcher new to game design, and methods for enhancing the player (student) experience. This is followed by techniques for integrating games into curriculum. Though not an exhaustive list, we propose that these are useful starting points for creating games for use in CS education.

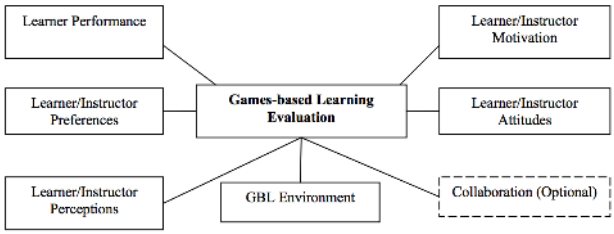

Figure 8: Hainey's evaluation framework for games-based learning evaluation.

5.1 Infusing Educational Frameworks

Hainey's dissertation [57] considers a set of seven factors that influence learning in the context of games. These are shown in Figure 8. Each of these categories of influencing variables are further defined by Hainey to provide a reference for evaluation methods, and each is important for instructors to consider the impact of games as a learning tool. Like books, videos, and other teaching tools, their impact is often heavily influenced by many other factors.

However, although striving to evaluate a newer media is a desirable goal, holding a formal evaluation of games as a litmus test for whether or not they should be used for learning contradicts what instructors do in classroom preparation each day. Shulman [118] refers to this as the "wisdom of practice," or the practice of instructors to choose assignments, activities, discussion questions, lecture topics and more based on their knowledge and wisdom of the content and how their particular set of students learn. In considering wisdom of practice, games can be carefully selected by instructors to incorporate them into their curriculum in meaningful and influential ways.

Many games for learning may be used as stand-alone games for independent learning by the learners; however, providing context for the learner's in-game experiences is certainly part of the game-based learning environment [92]. The manner in which the instructor integrates the learning experience from games into the classroom is very important to its effectiveness.

Based on this, integrating the learning into the classroom is an important aspect of the curriculum design process, and instructors can liken themselves to the Dungeon Master (DM) in Dungeon and Dragons. The DM sets up the context for the dungeon or game. They provide assistance during the game to clarify any questions and can debrief after the game.

5.2 Important Caveats

Though this set of guidelines is not comprehensive, it has the potential of being overwhelming for someone who has not yet created games for learning. We recommend researchers new to this field find one or two areas on which to focus, then expand those areas as more work is considered.

As previously mentioned, though our initial goal was to

provide a guide for assessing games for efficacy, scope and time prevented us from engaging in that activity. This should by no means be taken to imply that this goal is not as worthy. We recognize its importance and intend to pursue that activity in whole or in part during the 2017 ITiCSE Working Group.

We note here that beyond knowledge and skills, attitudes and dispositions are valid reasons for providing games as a learning tool in CS. Motivation, engagement, happiness, satisfaction, and perceptions of the field among students across a wide range of demographics are important aspects of games for CS education. As educators, the "wisdom of practice" may not be enough for this new medium, and we may be interested in various levels of proof that games are effective. A wide range of tools for understanding effectiveness can and should be used. For example, test scores, students surveys, quantitative or qualitative research methods, and/or rigorous evidence (replicated, longitudinal) are all forms of data that can be used to determine whether a game may be effective for a particular group of students.

The next two sections on game development processes and game design for games for use in CS education are presented in the context of games for CS education where possible. These two sections are also enhanced by Section 5.5, which discusses best practices integrating games into the CS curriculum or a CS classroom.

5.3 Basic Game Development Processes

Game design is the process of designing the game. Game production is the process of taking the design and implementing it. There are many books and articles on game production, and in this section we highlight a few important aspects of development that a researcher newer to creating game for CS education might find useful.

Agile processes. Agile processes [7] are often used in managing a game project. A project is broken into tasks or sprints. In a classroom setting, the first sprint, for example, may last two weeks and during that time students may be responsible for creating a game proposal. One of the values of agile development is the reflection at the end of each sprint of what is going well with the project, what needs to be improved, and what might be standing in a team member's way.

Moodboards, Storyboards, Ripomatics. Moodboards provide a visual representation of the art style, typography, and early visual design and themes of the game to convey its look and feel. An example is shown in Figure 9. Storyboards provide a visual representation of gameplay and narrative. An example is shown in Figure 10. Ripomatics are scenes that are "ripped from" existing media such as film, TV, or games to provide a concept of the look and feel of the proposed game. All of these can be used to create a consistent look and feel across the game and are particularly useful when multiple people are working on the game.

Paper prototyping and testing. Paper prototyping is the process of presenting a game idea through storyboards or mockups of a game's scenes (or a combination) to present your game in a paper format to potential players. This form of test has been proven to be very effective to gauge how fun is one's game, to determine whether the flow of the game is effective, and to determine if players are learning. This process enables the developers to get feedback very early in the design process, before actual production of the game

Figure 9: Sample moodboard for the game Wake Up, Koala!, a game developed by undergraduate students to raise awareness about Sjögren's Syndrome.

begins. It is much easier to correct major issues with flow and game mechanics before starting to program rather than after time and effort has been spent on programming the early prototype of the game.

Quality Assurance and User Testing. Testing is a necessary and important process and is critical for enhancing the user experience. No one likes to have flow broken when learning due to problems with their learning aides! We have used two types of testing, one with users to gain critical feedback on concept builds, early prototypes, alpha, and beta builds. In this type of test, users are solicited for feedback on how to improve any part of the system. The other, quality assurance, is the development team's responsibility. This type of test includes the painstaking process of testing each element within the game (buttons, scene transitions, scoring, general functionality, and more) as well as art, animation, and sound.

5.4 Designing for the User Experience

Within this section we provide a set of design principles for game development. Though not meant to be comprehensive, we carefully selected elements that are important to consider when designing a game for use in CS education. The intent is to provide a best practice cheat sheet for creating games for CS education to those newer to game design and development. This toolset considers best practices in each area and is designed to suggest methods for improving the impactfulness of the games.

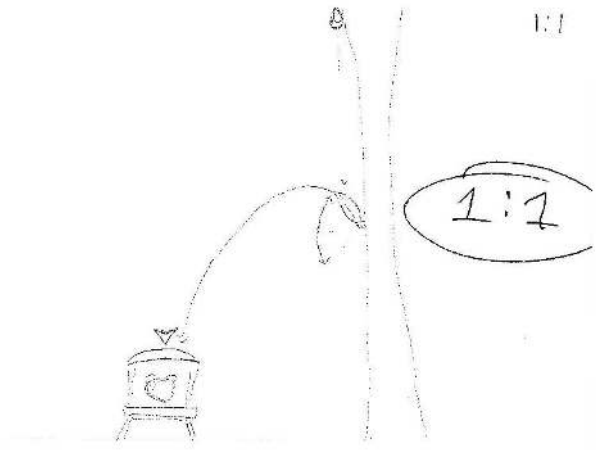

Figure 10: Sample storyboard for the game Wake Up, Koala!

5.4.1 Define the Learning Outcomes

Games have been shown to improve learning in some computer science education domains, such as computer memory [95]. Embedding sound instructional design into games for learning is needed if we want the games to be effective. O'Neill, Wainess, and Baker [92] note that many games for learning have historically missed the mark on this. They make the case that CRESST and Kirkpatrick's models for learning can be evaluated and applied within a game's design.

The Understanding by Design model [128] is also relevant to game design for education. The three major stages of this backward design model is to 1) identify the desired results, 2) determine the acceptable evidence, and 3) plan the learning experiences and instruction. These stages can be tailored for the development of games for computing education as follows:

- Stage 1:
  - Establishing learning goals for the player
  - Defining the set of essential questions for achieving these goals
  - Identify the understandings that are desired
  - Specifying what key knowledge and skills students will acquire as a result of playing the game
- Stage 2:
  - Identify the tasks that will be performed in the game to provide the evidence needed to ascertain the player's understanding of the learning material
  - Identify any other evidence that should be collected to determine if the desired results in Stage 1 have been met. Include time playing, time spent in each level, incorrect and correct moves, etc.
  - Provide a manner for the players to reflect on their learning experiences
- Stage 3:
  - Once Stages 1 and 2 have been drafted, design the sequence of experiences that the players will need to engage with in the game, develop by playing the game, and demonstrate within the game those desired understandings identified in Stage 1.

Though there are many aspects in instructional design to consider, for both development and evaluation purposes, learning outcomes are at the top of the list. Defining the goals and learning outcomes for a game will not only provide clear direction when designing, but also provide a set of metrics for which to evaluate the game for effectiveness.

5.4.2 Research target users

Referring back to Section 2.2.3, knowing the player preferences of your target learners is important when designing games. As noted by O'Neil et al. [92], games for learning can be effective for developing learning strategies, but "players apply those strategies differentially, with some players more effective than others. These differences come not only from our knowledge, skills and abilities, they are socially motivated as well." Learner (and instructor) preferences are important variables to consider [57].

There has also been much discussion within the game industry that game designers have historically chosen to create games that appeal to them. Anthropy [117], a game designer, noted that game designers are primarily white and male, and this has influenced the types of games that they have created and has, in turn, influenced the types of players that are attracted to such games. It is important to consider the preferences of the learners one aims to target, especially in light of the continued lack of diversity within computing [48]. The game industry has begun to mature beyond stereotypes like "girls like pink and boys like blue," recognizing that the player, regardless of their sex, may prefer an avatar that is blue or pink or another color altogether.

The tools mentioned in Section 2, such as BrainHex or a Big 5 Personality Survey, can provide a starting point for the type of game that might be most engaging for a set of learners—particularly for high school and adult learners. Considering age and ability in your design choices can also make your game more effective. From a pedagogical point of view, by so doing, researchers can engage in developing a form of culturally relevant pedagogy that will be more inclined to motivate and engage learners [72].

There is also value in bringing a representative group of targeted users into your design process. Whether they become part of the design process by making choices about the theme or genre, or whether they are asked for their input at various stages of the game's development, the process will help ensure that the game is engaging for the learners that they represent.

5.4.3 Engagement

In both video games and assignments the notion of overcoming a challenge is one of the primary motivators for engagement [105]. The presentation of challenge must be carefully balanced to be effective. This concept is called flow in video games and scaffolding in educational delivery [34]. If the material is too hard, student-players will give up, and if it is too easy, the student-player becomes bored and will lose engagement. In both cases, as the skills of the student-player increase the difficulty of the class/game must also increase.

While there are always exceptions, assignments and video games have separate avenues for generating additional engagement for the student-player. In crafting assignments the inclusion of personal meaningfulness can provide dramatic increases in engagement. Hulleman et al. [63] asked students to write essays about how their learning of math or psychology was relevant to a student's life. Interestingly, being told by an instructor that something was good for them only helped the high ability students. When students wrote their own reasons, they were more engaged with the material. A similar technique calls on instructors to utilize socially relevant causes in order to increase student engagement and learning [26]. Seeing direct applications of their discipline of study generates excitement in the students.

In video games, the designer can use a multitude of tools in order to increase engagement. However, not all are equally powerful. In a survey of gamers the four most reported game aesthetics (after Challenge) that increased engagement are, in order, Narrative, Fellowship, Sensory, and Discovery [108]:

1. Narrative was reported as the most common reason that players would continue playing a game. The story becomes part of the reward structure. In order to uncover the entire story, a players must typically finish a significant part of the game content. Narrative can be delivered in the form of traditional dialog elements, such as characters talking or a background story. However, the best stories are formed through the combined use of dialog, art, sound, level design, and mechanics.

2. Fellowship with other players fulfills a psychological need in people to connect with others. The desire to meet up with friends or adversaries within the game can keep players returning long after they have completed the game content.

3. Sensory aesthetic has traditionally been the use of visual and audial components to create an immersive experience. However, new technologies such as Kinect and Wii controllers have introduced the use motion as a sense pleasure. New advances in augmented realities, such as Pokemon Go, will further expand the means by which we can offer sensory aesthetics.

4. Discovery appeals to curious players who wish to discover all that their world has to offer. This can be simple exploration of the environment, but also experimentation with mechanics or unlocking all of the secrets within the game. Games which provide player agency in accomplishing goals are most likely to fuel the Discovery aesthetic.

5.4.4 Positive/Negative Reinforcement

Games are designed to shape their players. This shaping occurs at many levels, ranging from the simple cues that teach a player how to navigate a game world to the promotion of beliefs and skills that the player will carry outside the game. How exactly the player is shaped is an important ethical concern of game design.

Skinner introduced the idea of *operant conditioning* [121], which he defined as the encouragement of behaviors through positive and negative reinforcement and their discouragement through punishment. These consequences increase or decrease the probability of the associated behavior in the subject to whom they are applied. Within a game, consequences are often delivered through game mechanics that hold significant value to the player, like scores, wealth, visual and auditory feedback, character health, and leveling systems. The feedback delivered through these tools selects out certain player behaviors while diminishing others.

Games do much of their shaping through positive reinforcement, by rewarding a player for performing a desired task. The schedule and structure of these rewards is a key component of player engagement. In early gameplay, rewards may need to be delivered more or less continuously to draw in a player [80]. However, rewards dispensed continuously and without merit quickly lose their value.

To avoid this, designers soon alter the reward schedules to deliver only *partial reinforcement*, in which rewards are dispensed intermittently. The reward schedule may be a *ratio schedule*, in which the player is rewarded only after completing a certain number of actions. Alternatively, the reward schedule may be an *interval schedule*, in which the player is rewarded only after a certain amount of time has passed. Further, the schedule may be *fixed* or *variable,* indicating that the gap between rewards is either known or effectively random, respectively. Each combination of these two qualities leads to different profiles of gameplay. In general, a variable ratio schedule tends to keep the player most consistently engaged [61], but fixed schedules can produce bursts of activity just before the anticipated reward.

Reinforcement may be spoiled in a number of ways. Rewards not delivered in a timely manner can lead to player frustration and the *extinction* of the desired behavior. Rewards not valuable to the player do not have the desired effect on behavior. For this reason, games offer a variety of different reward types. A player may be content receiving a reward of small magnitude. But if one of larger magnitude is introduced, any subsequent smaller rewards are likely to be treated with disdain.

Nearly all games use some form of operant conditioning to keep the player playing. Frattesi et al. [49] identify several means of promoting replay value within a game: its difficulty should balance challenge and frustration, a player's completion status should be clearly communicated, it should draw on social aspects to facilitate community, its gameplay should have elements of randomness to produce feelings of unexpectedness and novelty, and it should provide a unique experience that cannot be found elsewhere. Educators designing games have different goals than commercial game designers, and replayability should be considered only in the context of sustaining learning. If a student has mastered a game's learning outcomes, then replayability is undesirable and the student should move on to new challenges. Game designer Raph Koster says, "In the end, that is both the glory of learning and its fundamental problem: once you learn something, it's over. You don't get to learn it again" [71].

5.4.5 Reward Systems

Reward systems can shape players towards desirable behaviors like longer game play, more effective use of time, or breadth of coverage. Points, level, badges, or virtual goods feed the player's need for status and accomplishment. Even though these artifacts do not exist in real life, digital artifacts symbolize the player's status among other players and increase their feelings of self worth. Completion bars and

checklists allow users to see goals and plan out how to complete them. Leaderboards allow a person to compete with others in a competitive setting. Analytics like heat maps and histograms allow a player to monitor their performance and provide course correction for personal goals.

Many reward systems are orthogonal to core gameplay mechanics. Therefore they can be added and altered without significantly changing the original game. Because of their disconnectedness from the actual game but their profound influence on player behavior, reward systems are the core of gamification, which is the addition of game-like elements to non-games to increase engagement [89].

There are three caveats in utilizing reward systems to change player behavior. First, reward systems are a force multiplier and not a substitute for the interesting underlying activity. Rewards will not cause players to enjoy the underlying activity if they did not do so in the absence of rewards. However, reward systems will increase how players self-report their overall experience playing the game.

Second, not all reward systems are meaningful for all players. Hakulinen [58] found that students responded to different rewards based on underlying goals. For instance, students with a fear of being seen as incompetent favored heatmaps, which warned how much danger they were in, compared to badges, which indicated that they had succeeded in a particular topic.

Third, be mindful that reward systems are subtle ways to affect and shape player behavior. Sometimes the behavior that you shape is undesirable. O'Rourke et al. [94] utilized reward systems in the game Refraction to reward players that used mindful approaches to problem solving. When students in the treatment group displayed mindful approaches in the game, they were awarded "Brain Points." O'Rourke found that students in the treatment group did play longer. However, the control group displayed more growth mindset behaviors during game play. Reward structures can be difficult to fine tune for alternative behaviors.

5.4.6 Formative and Summative Assessment

According to the Theory of Multimedia Learning, "people learn more deeply from words and pictures than from words alone" [86]. This is a concept accepted by many in the computer science education community because visualizations can be used to not only engage learners but more powerfully they can make feedback of an abstract problem more concrete. Consider a middle school introduction to computer science where students write programs to create animations using Alice [35]. When a program does not produce the desired animation, the learner is given immediate and obvious feedback. Or consider a college-level computer science course for non-majors using Guzdial's media computation [56], where students write programs to manipulate images. Specifying the behavior of an assigned program in such an environment may be as simple as giving before and after images.

Work by a previous ITiCSE working group [91] concluded that visualizations alone are not a sufficient learning tool. Visualizations must be used to engage students in active learning. This conclusion is corroborated by later work showing that visualizations alone do not demonstrably show learning benefits [102], but a visual metaphor in conjunction with a gaming context does.

We as educators know that formative assessment, with frequent, quick, clear feedback of student work, can have a significant effect on student learning [25]. Games are an opportunity to provide students with faster, more frequent, and clearer feedback. Formative assessment can also be used by instructors to customize the classroom experience to better utilize class time and to maximize learning opportunities. This is seen in pedagogical techniques like Just-in-Time Teaching (JiTT) [17], where out of class readings and quizzes are used to help make modifications to class immediately prior to class. Instructors use of formative assessment is also seen in peer instruction [120], where student responses to in-class clicker questions are used by the instructor to guide classroom discussion to topics that need more attention.

Games for computer science education can also be used to adapt content to address student needs. But games are able to do this on an individual basis by tracking what an individual player has mastered and what a player is still learning. This information can be used to modify the gameplay, to repeat lessons, and to give hints [70, 59]. Learning analytics research has sought to give instructors even more information about their students by monitoring and analyzing students interactions with online learning management systems to help instructors optimize learning [103]. The video game industry has its own game analytics, often called game telemetry, that allows game developers to continually fine tune and improve a game after it has been released [81]. Unity Analytics for the Unity game engine [11] provides an opportunity for computer science education games to analyze player behavior to not only improve the educational efficacy of the game but to also help instructors better understand students and how best to help them learn.

One potential problem with in-game computer generated feedback is that it may not be as clear as instructor feedback. However, difficult to understand feedback is an issue students are already encountering whenever they are presented with errors from a compiler or interpreter. Games for computer science education provide an opportunity to mitigate this. Games have a world or a context in which the feedback can be contextualized. For example, research by Lee and Koh [77] show that a more human-like feedback system can increase player engagement with an educational game.

5.4.7 Learning Communities

The benefits of learning communities in higher education are well-established, and include "higher academic achievement, better retention rates, diminished faculty isolation, and increased curricular integration" [78]. At the same time, the emergence of communities around games, and especially around digital games, is both commonly observed and a carefully studied phenomenon (e.g., by Pearce and Artemesia [97]). Furthermore, numerous researchers have observed that the communities that emerge around games often exhibit many of the characteristics of learning communities. In particular, Gee focuses on the similarity of the experiences afforded to the members of learning communities, even though they may have little else in common, and as such considers them a particular type of "affinity group" [51]. Viewed from the perspective of the educator considering the inclusion of games in the curriculum, this means that game design determines students' potential shared experiences, which in turn determine the characteristics exhibited by that partic-

ular learning community. In his doctoral thesis describing his qualitative study on the use of *Civilization III* in world history education, Squire [122] writes:

> "Cooperative and competitive social arrangements frame game play activity. In some cases, the social context of game play—the kinds of reflection activities, discussion, collaboration, and competition that emerge in game play are as important as the game itself in determining what activity emerges and what learning occurs."

The question at hand, then, is how to design and develop games and how to integrate them into computing curricula so that the emerging communities exhibit the characteristics desired for the learning communities. Guidelines for such design, development, and integration are still an area of open research. As such, the following recommendations based on Shaffer's [115], Squire's and other user studies are necessarily tentative and incomplete.

Given that one goal is to stimulate conversation amongst the students regarding the subject matter, it is worthwhile to consider the types of desired discussion as a part of the requirements development phase. For example, Squire [122] observes that "having students responsible for joint presentations that glean information from multiple games might be [an effective way] to encourage collaboration and knowledge building." If such presentations seem desirable, then the game's design must provide opportunities for meaningful cooperation between players. This implies, of course, that the game should support a multiplayer mode, and that the game rules allow multiple players to be successful.

Similarly, Squire reports that "students took great pride in their games and saw value in using them as a point of exploration." Thus, if the goal is to encourage student participation in open discussion, then the game's design should provide players with interesting and varied experiences that they can report and about which they can ask questions.

Squire noted that students essentially ignored the preparatory lectures he offered, but that they were attentive to his responses to their questions about game play. As such, games should be designed such that very little is required in the way of introduction. Regarding the integration of the game into the curriculum, instructors should allow time to support and encourage impromptu informal discussions emerging from game play. Such discussions could be viewed as instances of "just-in-time teaching."

Finally, Prensky [100] observes that along with developing the skills of information absorption, assimilation, decision-making, and multi-tasking, gamers "increasingly, gamers get good at collaborating with others, over a range of networks." In order to leverage this quality of gamer development, games used for educational purposes must, of course, be multiplayer and support some form of in-game communication between players.

There is wide variation in students' previous gaming background, including the amount and frequency of their play, the types of games they have played, and their reasons for playing. As such, it is unlikely that any one game would engage all students in the same way. At first glance, this observation seems to present a significant challenge for the design of games for education. However, it is also an opportunity. Games that are intentionally designed to engage different players in different ways build in the potential for classroom conversations around the contrasts between the various players' experiences. For example, Squire observes that "discussions between different player types drove them to articulate and defend different strategies, even rethinking their orientation to the game."

For concreteness, as noted in Section 2.2.3, Bartle [20] offers a classification of players into four types, achievers, explorers, socializers, and killers. The "socializer" and "killer" player types require a multiplayer mode for realization, and the competitive and collaborative aspects of the game hold significant implications for these player types.

Debriefing is critical to using games in education [74]. Teachers can facilitate the transfer of skills by leading pre- and post-game discussions which connect the game with other things students are learning in class [16]. Ke [68] concluded that instructional support features are necessary in order for the lessons learned in computer games to transfer to other contexts.

5.4.8 *Differentiated Instruction*

Differentiated instruction serves two purposes. First, it seeks to ensure that each student fulfills the learning outcomes to the greatest extent possible. Second, it seeks to adapt curricula for students whose background and learning styles require it.

The guidelines offered by Lawrence-Brown [73] may be of use:

- Provide "additional supports" for struggling students. Enable them to access content and demonstrate learning. All students benefit. Two categories: access general curriculum, lend structure to curriculum.
- For access to curriculum, use assistive technologies. Provide resource materials to support "finding" and reduce "guessing" (restrict access to those who need it). Provide personal assistance when absolutely necessary, but avoid creating dependency by offering too much help.
- To add structure, emphasize key topics and skills. Provide explicit and precise expectations and examples of successful work. Provide systematic decompositions of specific strategies, skills, and concepts. Make specific connections with prior knowledge and experiences. Work toward increased independence by fading assistance systematically.
- Adapt goals for differently-abled students, which is at least as important for advanced students as for those who struggle. Form cooperative groups with individualized roles.
- To evaluate effectiveness, gather data about students' learning.

Kickmeier-Rust, et al. [69] describe the ELEKTRA system, in which players' actions are used to continuously update a probabilistic model of the players' competencies within an ontological model of the subject domain. The competency model is then processed by a pedagogical rule-based system to adjust game play to provide appropriate interventions to support skill acquisition, skill activation, and motivation.

5.4.9 Social Cognitive and Psychological Influences

When Malmi et al. [83, 82] reviewed the psychological theories that appear in the computer science education research literature, one of the most common was the theory of self-efficacy proposed by Bandura [18]. This is notable, because it is considered a predictor of programming achievement (e.g. [101]). Lee and Ko [76] used this theory as the foundation for their research into play, showing that the personification of a compiler could be used to improve motivation when learning computer programming. Further to this, other games such as RAPUNSEL have been shown to improve learner's programming self-efficacy [99].

Other related theories also appear in the literature. For example: achievement emotions [98], mindset [41, 43], and self-concept [116, 85]. Together, these have been shown to predict of programming practice [111, 113, 114]. Furthermore, games seem to be able to influence these psychological constructs. Notably, helping learners to develop a growth mindset in the mathematics domain [93] and their self-concept in the programming domain [112].

While, in some cases, these effects are small and further research is needed to isolate the particular properties that cause the improvements, the use of games to enrich learners' psychological constructs is worth exploring. Many theories align with what we already know about pedagogy. For instance, some games follow a growth mindset incentive structure which serves to maintain a learner's effort within their zone of proximal development [127]. Another example is Gee's [52] notion of a psychosocial moratorium where low-stakes play buffers learners from negative emotions associated with failure, helping learners feel safe to experiment.

5.4.10 Deliberate Practice

One of our goals as educators is to help our students develop expertise. Implicit in this goal is the belief that expertise can be attained through deliberate practice, which Ericsson et al. [46] define as activity explicitly designed to improve performance. Four key conditions are required for practice to be considered *deliberate* and lead to expertise:

> "The most cited condition concerns the subjects' motivation to attend to the task and exert effort to improve their performance. In addition, the design of the task should take into account the preexisting knowledge of the learners so that the task can be correctly understood after a brief period of instruction. The subjects should receive immediate informative feedback and knowledge of results of their performance. The subjects should repeatedly perform the same or similar tasks."

These conditions exclude certain activities, including "playful interaction, paid work, and observation of others."

A well-designed game provides a medium for deliberate practice by adhering to these conditions. Lightbot, for example, demonstrates all four:

- Many game elements appeal to various aesthetics as defined by the MDA framework: sensation, fantasy, narrative, and challenge. These aesthetics are designed to motivate the player, who in turn willingly engages in a series of increasingly difficult levels that lead to mastery.
- Compared to Lightbot, Lightbot Jr. takes into account that its players have less prior knowledge and therefore provides more opportunities to use a new command before introducing any further commands.
- When the player runs her program, the robot traces out the program in an animated way, giving clear feedback about the program's intermediate behavior and correctness. Each program command is highlighted as it executes.
- The game is organized into worlds, within which the various levels require repeated but slightly varied use of an operation.

A key aspect of deliberate practice is *scaffolding*, which according to Wood, Bruner, et al. [131], "refers to the steps taken to reduce the degrees of freedom in carrying out some task so that the child can concentrate on the difficult skill she is in the process of acquiring." Well-designed games provide scaffolding for players in many ways: by situating the player within a simplified model of a complex subject; by delivering instruction within the game in a just-in-time fashion; and by providing tutorial levels where concepts are introduced slowly and in a non-threatening environment.

5.5 Integrating Games into the curriculum

A common finding about the use of different educational technologies is that educational success depends less on the technology itself and more on its integration into instruction [92]. Examples of these evidences can be found in the fields of program and algorithm visualization [64, 91] as well as microworlds [88].

Identifying successful educational practices for integrating games still is an open challenge. However, we identify several best practices that should be taken into account:

- Model progression. The game model is introduced incrementally, so that the learner does not become overwhelmed.
- Prompting. Prompts may take different forms, such as a question or passing a challenge. Prompts may be aimed at non-knowledge factor. For instance, a prompt may be aimed at increasing the student's self-esteem. Within games, prompts can take the form of trivia questions or answering prompts to further a character's progress in a game.
- Assignments. An assignment is a more complex form of prompt relevant to the subject matter and aligned with the game goals. Assignments can be incorporated within a quest, for example, in the same way that word problems help contextual learning for the field of mathematics. This integration focuses on desired outcomes and enabling students to succeed in solving the given problem within the game.
- Feedback. Students need to obtain feedback to have an external judge on their actions and readjust their activity accordingly. Feedback in games typically describes the degree of achievement or performance in the game. In its simplest form, it may be a score. More elaborate forms include comparing the student's score with others' or reporting about conditions still not met

to achieve a goal. Alternatively, the game provide may feedback on the student's process.

- Additional information. Frequently, more information is necessary for the learner to make a right decision. Just-in-time information is the most successful way of assisting the learner.
- Monitoring facilities. In games, especially in complex situations, it seems to be of great value to have the opportunity to inspect the history of the interaction.
- Reflection and debriefing. During a game, the learner often acquires knowledge or adopts a strategy informally. By introducing a final phase of reflection and debriefing learners may acquire more explicit knowledge. This can be done orally but also the learner also may write it (however, it takes more time).
- Explicitation. Here, knowledge must also be made explicit, but in the context of the game. This feature is especially useful in cooperative or collaborative situations, where students have to explain to each other what they intend to do or they have done.

In case the students are the constructors of their games, other approaches need to be adopted. Thus, Basawapatna et al. (2013) propose a project-based approach, where the students must develop games of increasing difficulty. Students who need assistance are given external scaffolding so that they do not get into a state of anxiety.

6. CONCLUSION

The Operating System 4 Computer Science (OS4CS) [30] initiative lists five key challenges for computing education. One of these key challenges identifies the need for more comprehensive, quality instructional resources with a call to the community to create these resources. Games can serve as an important, engaging instructional resource for many of the reasons defined throughout this study. However, the community fails to provide comprehensive and high-quality games. It has also failed in measuring the effectiveness of games in achieving their stated educational goals.

This workgroup study provides a resource for the reader interested in designing and developing games for computing education. The high-level summary provides a starting point for examining important underlying theories about digital game-based learning, best practices to consider when designing and developing a game, and relevant educational psychology, all in the context of computing education. In addition, it defines a set of games that can be used in computing education, and it identifies gaps where future games can be developed to enable the creation of a comprehensive set of resources for the education community.

The second part of this working group study will focus on the evaluation of games to identify their efficacy. As part of that process, several new games are currently being created using the design guidelines provided in this report. These games will add to the list of games for teaching computing education and provide analysis of players' learning.

We encourage the community to consider games that could be created to make the available games more comprehensive. We also encourage the community to keep in mind the need for contextualizing learning and providing the necessary interpretation of games in the context of computing. Just as we interpret assignments, required readings, and other subsidiary materials, our active incorporation of games into our educational environments is key to achieving the learning that external media can provide.

7. REFERENCES

[1] BrainHex. http://survey.ihobo.com/BrainHex/. [Online; accessed 29-August-2016].

[2] Code.org. https://code.org/learn. [Online; accessed 29-August-2016].

[3] codeSpark. http://codespark.org. [Online; accessed 29-August-2016].

[4] Developer spotlight: Danny Yaroslavski. http://www.openfl.org/blog/2014/11/07/developer-spotlight-danny-yaroslavski. [Online; accessed 29-August-2016].

[5] Game survey. http://www.twodee.org/forothers/game_survey_iticse2016.csv. [Online; accessed 15-September-2016].

[6] Juice it or lose it. https://youtu.be/Fy0aCDmgnxg. [Online; accessed 29-August-2016].

[7] Manifesto for agile software development. http://agilemanifesto.org. [Online; accessed 29-August-2016].

[8] Molleindustria. http://www.gamedefinitions.com/. Accessed: 2016-07-13.

[9] Picobot. https://www.cs.hmc.edu/picobot/. [Online; accessed 29-August-2016].

[10] Tomorrow corporation. http://tomorrowcorporation.com/. [Online; accessed 29-August-2016].

[11] Unity3d. https://unity3d.com/. [Online; accessed 29-August-2016].

[12] E. Aarseth. Computer game studies, year one. *Game studies*, 1(1):1–15, 2001.

[13] ACM/IEEE. *CS Joint Task Force on Computing Curricula. 2013*. Computer Science Curricula ACM Press and IEEE Computer Society Press, 2013.

[14] L. W. Anderson, D. R. Krathwohl, P. W. Airasian, K. A. Cruikshank, R. E. Mayer, P. R. Pintrich, R. Raths, and M. C. Wittrock, M. C Wittrock. *A Taxonomy for Learning, Teaching and Assessing*. Addison Wesley Longman, 2001.

[15] J. Arjoranta. Game definitions: a wittgensteinian approach. *Game Studies*, 14(1), 2014.

[16] K. Ash. Digital gaming goes academic. *Education Week*, 30(25):24–28, 2011.

[17] T. Bailey and J. Forbes. Just-in-time teaching for cs0. *ACM SIGCSE Bulletin*, 37(1):366–370, 2005.

[18] A. Bandura. Self-efficacy: toward a unifying theory of behavioral change. *Psychological review*, 84(2):191, 1977.

[19] S. Barab and C. Dede. Games and immersive participatory simulations for science education: an emerging type of curricula. *Journal of Science Education and Technology*, 16(1):1–3, 2007.

[20] R. Bartle. Hearts, clubs, diamonds, spades: Players who suit muds. *Journal of MUD research*, 1(1):19,

1996.

[21] R. Bartle. *Designing Virutal Worlds*. New Riders, 2003.

[22] T. Bell, J. Alexander, I. Freeman, and M. Grimley. Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology*, 13(1):20–29, 2009.

[23] T. Bell, F. Rosamond, and N. Casey. Computer science unplugged and related projects in math and computer science popularization. In H. L. Bodlaender, R. Downey, F. V. Fomin, and D. Marx, editors, *Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday (Vol. LNCS 7370,)*., pages 398–456. The Multivariate Algorithmic Revolution and Beyond, 2012.

[24] J. B. Biggs and K. F. Collis. *Evaluating the quality of learning: The SOLO taxonomy (Structure of the Observed Learning Outcome)*. Academic Press, New York, USA, 1982.

[25] P. Black and D. Wiliam. Assessment and classroom learning. *Assessment in Education: principles, policy & practice*, 5(1):7–74, 1998.

[26] M. Buckley, J. Nordlinger, and D. Subramanian. Socially relevant computing. In *Proceedings of the 39th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '08, pages 347–351, New York, NY, USA, 2008. ACM.

[27] G. Burkhardt, M. Monsour, G. Valdez, C. Gunn, M. Dawson, C. Lemke, E. Coughlin, V. Thadani, and C. Martin. engauge 21st century skills: Literacy in the digital age. *Retrieved June*, 2:2008, 2003.

[28] R. Caillois and M. Barash. *Man, play, and games*. University of Illinois Press, 1961.

[29] E. Castronova. *Synthetic worlds: The business and culture of online games*. University of Chicago press, 2008.

[30] J. Century, M. Lach, H. King, S. Rand, C. Heppner, B. Franke, and J. Westrick. Building an operating system for computer science. Technical report, CEMSE, University of Chicago with UEI, University of Chicago, 2013.

[31] D. Church. Formal abstract design tools. gamastutra, july 16, 1999. http://www.gamasutra.com/view/feature/3357/formal_abstract_design_tools.php. Accessed: 2016-07-13.

[32] R. C. Clark and R. E. Mayer. *E-learning and the science of instruction: Proven guidelines for consumers and designers of multimedia learning*. Pfeiffer, 2011.

[33] G. Costikyan, I. H. N. Words, and I. M. Design. Toward a critical vocabulary for games. In *Computer Games and Digital Cultures Conference Proceedings*, pages 9–33, 2002.

[34] M. CsÃ■kszentmihÃąlyi. *Flow: The Psychology of Optimal Experience*. Harper and Row, New York: USA, 1990.

[35] W. Dann, S. Cooper, and R. Pausch. Making the connection: programming with animated small world. In *ACM SIGCSE Bulletin*, volume 32, pages 41–44. ACM, 2000.

[36] F. D. Davis. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13:319–339, 1989.

[37] C. Degraft-johnson, Y.-c. Wang, M. B. Sutherland, and K. L. Norman. Relating five factor personality traits to video game preference. 2013.

[38] S. Deterding, D. Dixon, R. Khaled, and L. Nacke. From game design elements to gamefulness: defining gamification. In *Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments*, pages 9–15. ACM, 2011.

[39] S. Deterding, M. Sicart, L. Nacke, K. O'Hara, and D. Dixon. Gamification. using game-design elements in non-gaming contexts. In *CHI'11 Extended Abstracts on Human Factors in Computing Systems*, pages 2425–2428. ACM, 2011.

[40] M. D. Dickey. Engaging by design: How engagement strategies in popular computer and video games can inform instructional design. *Educational Technology Research and Development*, 53(2):67–83, 2005.

[41] C. Diener and C. Dweck. An analysis of learned helplessness: Continuous changes in performance, strategy, and achievement cognitions following failure. *Journal of Personality and Social Psychology*, 36(5):451–462, 1978.

[42] H. L. Dreyfus and S. E. Dreyfus. *Mind over machine: The power of human intuition and expertise in the era of the computer*. The Free Press, New York, 1986.

[43] C. Dweck. *Mindset: The new psychology of success*. Random House, 2006.

[44] M. Ellis. *Why People Play*. Prentice-Hall, 1973.

[45] Entertainment Software Association. Essential facts about the computer and video game industry. 2016.

[46] K. A. Ericsson, R. T. Krampe, and C. Tesch-Römer. The role of deliberate practice in the acquisition of expert performance. *Psychological review*, 100(3):363, 1993.

[47] G. Ferri. Rhetorics, simulations and games: The ludic and satirical discourse of molleindustria. *International Journal of Gaming and Computer-Mediated Simulations (IJGCMS)*, 5(1):32–49, 2013.

[48] N. S. Foundation. Women, minorities, and persons with disabilities in science and engineering: 2013. Technical report, National Center for Science and Engineering Statistics, Directorate for Social, Behavioral and Economic Sciences, 2013.

[49] T. Frattesi, D. Griesbach, J. Leith, T. Shaffer, and J. DeWinter. Replayability of video games. *IQP, Worcester Polytechnic Institute, Worcester*, 2011.

[50] S. Freeman, S. L. Eddy, M. McDonough, M. K. Smith, N. Okoroafor, H. Jordt, and M. P. Wenderoth. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences*, 111(23):8410–8415, 2014.

[51] J. P. Gee. Identity as an analytic lens for research in education. *Review of research in education*, 25:99–125, 2000.

[52] J. P. Gee. What video games have to teach us about learning and literacy. *Computers in Entertainment*

(CIE), 1(1):20–20, 2003.

[53] J. P. Gee. Good video games and good learning. In *Phi Kappa Phi Forum*, volume 85, page 33. THE HONOR SOCIETY OF PHI KAPPA PHI, 2005.

[54] M. Graafland, J. M. Schraagen, and M. P. Schijven. Systematic review of serious games for medical education and surgical skills training. *British journal of surgery*, 99(10):1322–1330, 2012.

[55] B. Gros. Integration of digital games in learning and e-learning environments: Connecting experiences and context. In T. Lowrie and R. Jorgensen, editors, *Digital Games and Mathematics Learning: Potential, Promises and Pitfalls*, pages 35–51. Springer, 2015.

[56] M. Guzdial. A media computation course for non-majors. In *ACM SIGCSE Bulletin*, volume 35, pages 104–108. ACM, 2003.

[57] T. Hainey. *Using games-based learning to teach requirements collection and analysis at tertiary education level*. PhD thesis, University of the West of Scotland, 2010.

[58] L. Hakulinen et al. Gameful approaches for computer science education: From gamification to alternate reality games. 2015.

[59] A. Hicks, B. Peddycord III, and T. Barnes. Building games to learn from their players: Generating hints in a serious game. In *International Conference on Intelligent Tutoring Systems*, pages 312–317. Springer, 2014.

[60] G. Hofstede. Dimensionalizing cultures: The hofstede model in context. *Online Readings in Psychology and Culture*, 2(1), July 2011.

[61] J. Hopson. Behavioral game design. http://www.gamasutra.com/view/feature/131494/behavioral_game_design.php. [Online; accessed 29-August-2016].

[62] J. Huizinga. *Homo Ludens: A Study of the Play-element in Culture*. Beacon Press, 1955.

[63] O. H. B. L. H. J. M. Hulleman, Chris S.; Godes. Enhancing interest and performance with a utility value intervention. *Journal of Educational Psychology*, 102(4):880–895, Nov 2010.

[64] C. Hundhausen, S. Douglas, and J. Stasko. A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages and Computing*, 13(3):259–290, 2002.

[65] R. Hunicke, M. LeBlanc, and R. Zubek. Mda: A formal approach to game design and game research. In *Proceedings of the AAAI Workshop on Challenges in Game AI*, volume 4, page 1, 2004.

[66] L. Johnson, R. Smith, H. Willis, A. Levine, and K. Haywood. The 2011 horizon report. Technical report, The New Media Consortium, Austin, Texas, USA, 2011.

[67] J. Juul. *Half-real: Video games between real rules and fictional worlds*. MIT press, 2011.

[68] F. Ke. A qualitative meta-analysis of computer games as learning tools. *Handbook of research on effective electronic gaming in education*, 1:1–32, 2009.

[69] M. D. Kickmeier-Rust, C. Hockemeyer, D. Albert, and T. Augustin. Micro adaptive, non-invasive knowledge assessment in educational games. In *Digital Games and Intelligent Toys Based Education, 2008 Second IEEE International Conference on*, pages 135–137. IEEE, 2008.

[70] K. Kiili. Digital game-based learning: Towards an experiential gaming model. *The Internet and higher education*, 8(1):13–24, 2005.

[71] R. Koster. *Theory of fun for game design*. "O'Reilly Media, Inc.", 2013.

[72] G. Ladson-Billings. Toward a theory of culturally relevant pedagogy. *American educational research journal*, 32(3):465–491, 1995.

[73] D. Lawrence-Brown. Differentiated instruction: Inclusive strategies for standards-based learning that benefit the whole class. *American secondary education*, pages 34–62, 2004.

[74] L. C. Lederman and K. Fumitoshi. Debriefing the debriefing process: A new look. *Simulation and gaming across disciplines and cultures. London: Sage Publications*, 1995.

[75] K. Lee and M. C. Ashton. Psychometric properties of the hexaco personality inventory. *Multivariate Behavioral Research*, 39:329–358, 2004.

[76] M. J. Lee and A. J. Ko. Personifying programming tool feedback improves novice programmers' learning. In *Proceedings of the seventh international workshop on Computing education research*, pages 109–116. ACM, 2011.

[77] M. J. Lee and A. J. Ko. Investigating the role of purposeful goals on novices' engagement in a programming game. In *2012 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 163–166. IEEE, 2012.

[78] O. T. Lenning and L. H. Ebbers. *The Powerful Potential of Learning Communities: Improving Education for the Future. ASHE-ERIC Higher Education Report, Vol. 26, No. 6*. ERIC, 1999.

[79] F. Levy and R. J. Murnane. Education and the changing job market. *Educational leadership*, 62(2):80, 2004.

[80] G. R. Loftus and E. F. Loftus. *Mind at play; The psychology of video games*. Basic Books, Inc., 1983.

[81] C. S. Loh. Information trails: In-process assessment of game-based learning. In *Assessment in game-based learning*, pages 123–144. Springer, 2012.

[82] L. Malmi, J. Sheard, Simon, R. Bednarik, J. Helminen, P. Kinnunen, A. Korhonen, N. Myller, J. Sorva, and A. Taherkhani. Theoretical underpinnings of computing education research: What is the evidence? In *Proceedings of the Tenth Annual Conference on International Computing Education Research*, ICER '14, pages 27–34, New York, NY, USA, 2014. ACM.

[83] L. Malmi, J. Sheard, Simon, R. Bednarik, J. Helminen, A. Korhonen, N. Myller, J. Sorva, and A. Taherkhani. Characterizing research in computing education: A preliminary analysis of the literature. In *Proceedings of the Sixth International Workshop on Computing Education Research*, ICER '10, pages 3–12, New York, NY, USA, 2010. ACM.

[84] T. Malone. What makes things fun to learn? a study of intrinsically motivating computer games. Technical report, Xerox, Palo Alto, CA, USA, 1980.

[85] H. Marsh and A. Martin. Academic Self-Concept and Academic Achievement: Relations and Causal Ordering. *British Journal of Educational Psychology*, 81(1):59 – 77, 2011.

[86] R. E. Mayer. Cognitive theory of multimedia learning. *The Cambridge handbook of multimedia learning*, 43, 2014.

[87] S. Meier. Sid meier on how to see games as sets of interesting decisions. gamastutra, march 7, 2012. http://www.gamasutra.com/view/news/164869/GDC_2012_Sid_Meier_on_how_to_see_games_as_sets_of_interesting_decisions.php. Accessed: 2016-07-13.

[88] C. S. Miller, J. F. Lehman, and K. R. Koedinger. Goals and learning in microworlds. *Cognitive Science*, 23(3):305–336, 1999.

[89] C. I. Muntean. Raising engagement in e-learning through gamification. Proc. 6th International Conference on Virtual Learning, 2011.

[90] L. E. Nacke, C. Bateman, and R. L. Mandryk. Brainhex: A neurobiological gamer typology survey. *Entertainment Computing*, 5:55–62, 2014.

[91] T. Naps, G. Roessling, V. Almstrum, W. Dann, R. Fleischer, C. Hundhausen, A. Korhonen, L. Malmi, M. McNally, S. Rodger, and J. A. VelÃązquez-Iturbide. Exploring the role of visualization and engagement in computer science education. *SIGCSE Bulletin*, 35(2):131–152, June 2003.

[92] H. F. O'Neil, R. Wainess, and E. L. Baker. Classification of learning outcomes: evidence from the computer games literature. *The Curriculum Journal*, 16(4):455–474, December 2005.

[93] E. O'Rourke, K. Haimovitz, C. Ballweber, C. Dweck, and Z. Popović. Brain points: a growth mindset incentive structure boosts persistence in an educational game. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 3339–3348. ACM, 2014.

[94] E. O'Rourke, E. Peach, C. S. Dweck, and Z. Popovic. Brain points: A deeper look at a growth mindset incentive structure for an educational game. In *Proceedings of the Third (2016) ACM Conference on Learning@ Scale*, pages 41–50. ACM, 2016.

[95] M. Papastergiou. Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation. *Computers & Education*, 52(1):1–12, 2009.

[96] R. E. Pattis. *Karel the robot: a gentle introduction to the art of programming*. John Wiley & Sons, Inc., 1981.

[97] C. Pearce, T. Boellstorff, and B. A. Nardi. *Communities of play: Emergent cultures in multiplayer games and virtual worlds*. MIT Press, 2011.

[98] R. Pekrun. The control-value theory of achievement emotions: Assumptions, corollaries, and implications for educational research and practice. *Educational Psychology Review*, 18(4):315 – 341, 2006.

[99] J. L. Plass, R. Goldman, M. Flanagan, and K. Perlin. Rapunsel: Improving self-efficacy and self-esteem with an educational computer game. In *Proceedings of the 17th International Conference on Computers in Education*, pages 682–689, 2009.

[100] M. Prensky. *Don't bother me, Mom, I'm learning!: How computer and video games are preparing your kids for 21st century success and how you can help!* Paragon house St. Paul, MN, 2006.

[101] V. Ramalingam, D. LaBelle, and S. Wiedenbeck. Self-efficacy and mental models in learning to program. In *ACM SIGCSE Bulletin*, volume 36, pages 171–175. ACM, 2004.

[102] L. P. Rieber and D. Noah. Games, simulations, and visual metaphors in education: antagonism between enjoyment and learning. *Educational Media International*, 45(2):77–92, 2008.

[103] C. Romero and S. Ventura. Educational data mining: A survey from 1995 to 2005. *Expert systems with applications*, 33(1):135–146, 2007.

[104] A. A. Rupp, M. Gushta, R. J. Mislevy, and D. W. Shaffer. Evidence-centered design of epistemic games: Measurement principles for complex learning environments. *The Journal of Technology, Learning and Assessment*, 8(4), 2010.

[105] R. M. Ryan, C. S. Rigby, and A. Przybylski. The motivational pull of video games: A self-determination theory approach. *Motivation and Emotion*, 30:347–363, 2006.

[106] K. Salen and E. Zimmerman. *Rules of play: Game design fundamentals*. MIT press, 2004.

[107] J. Schell. *The Art of Game Design: A book of lenses*. CRC Press, 2014.

[108] H. Schoenau-Fog. The player engagement process - an exploration of continuation desire in digital games. *Think Design Play: Digital Games Research Conference*, 2011:14–17, Sept 2011a.

[109] I. Schreiber and B. Brathwaite. Challenges for game designers. 2008.

[110] M. Scott. A monument to the player: Preserving a landscape of socio-cultural capital in the transitional mmorpg. *New Review of Hypermedia and Multimedia*, 18(4):295–320, December 2012.

[111] M. J. Scott and G. Ghinea. Educating programmers: A reflection on barriers to deliberate practice. In *Proceedings of the 2nd HEA STEM Conference*, pages 28–33, Birmingham, UK, April 2013.

[112] M. J. Scott and G. Ghinea. Integrating fantasy role-play into the programming lab: Exploring the 'projective identity' hypothesis. In *Proceedings of 43rd ACM Technical Symposium on Computer Science Education*, pages 119–122, Denver, CO, USA, March 2013.

[113] M. J. Scott and G. Ghinea. Measuring enrichment: the assembly and validation of an instrument to assess student self-beliefs in cs1. In *Proceedings of the tenth annual conference on International computing education research*, pages 123–130. ACM, 2014.

[114] M. J. Scott and G. Ghinea. On the domain-specificity of mindsets: The relationship between aptitude beliefs and programming practice. *IEEE Transactions on Education*, 57(3):169–174, Aug 2014.

[115] J. R. Shaffer. *Online and offline gaming social preferences of students*. PhD thesis, George Mason University, 2012.

[116] R. J. Shavelson, J. J. Hubner, and G. C. Stanton. Self-concept: Validation of construct interpretations. *Review of educational research*, 46(3):407–441, 1976.

[117] B. Sheffield. Token video game characters distract from real stories - anna anthropy. *Gamustra*, 2012.

[118] L. Shulman. Those who understand: Knowledge growth in teaching. *Educational Researcher*, 15(2):4–14, Feb. 1986.

[119] V. J. Shute, M. Ventura, M. Bauer, and D. Zapata-Rivera. Melding the power of serious games and embedded assessment to monitor and foster learning. *Serious games: Mechanisms and effects*, 2:295–321, 2009.

[120] B. Simon, M. Kohanfars, J. Lee, K. Tamayo, and Q. Cutts. Experience report: peer instruction in introductory computing. In *Proceedings of the 41st ACM technical symposium on Computer science education*, pages 341–345. ACM, 2010.

[121] B. F. Skinner. Operant behavior. *American Psychologist*, 18(8):503, 1963.

[122] K. Squire and S. Barab. Replaying history: engaging urban underserved students in learning world history through computer simulation games. In *Proceedings of the 6th international conference on Learning sciences*, pages 505–512. International Society of the Learning Sciences, 2004.

[123] J. Stenros. The game definition game: A review. *Games and Culture*, 2016.

[124] B. Stewart. Personality and play styles: A unified model. Gamasutra, 2011.

[125] B. Suits. *The Grasshopper-: Games, Life and Utopia*. Broadview Press, 2014.

[126] J. VandenBerghe. The 5 domains of play: Applying psychology's big 5 motivation domains to games. Game Developers Conference, GDC Vault, 2012.

[127] L. S. Vygotsky. *Mind in society: The development of higher psychological processes*. Harvard university press, 1980.

[128] G. Wiggins and J. McTighe. *Understanding by Design*. Association for Supervision and Curriculum Development, Alexandria, Virgina, USA, 2005.

[129] D. Williams, N. Yee, and S. E. Caplan. Who plays, how much, and why? debunking the stereotypical gamer profile. *Journal of Computer-Mediated Communication*, 13(4):993–1018, 2008.

[130] J. M. Wing. Computational thinking. *Commun. ACM*, 49(3):33–35, March 2006.

[131] D. Wood, J. S. Bruner, and G. Ross. The role of tutoring in problem solving. *Journal of child psychology and psychiatry*, 17(2):89–100, 1976.

[132] N. Yee. The labor of fun how video games blur the boundaries of work and play. *Games and Culture*, 1(1):68–71, 2006.

[133] N. Yee. Motivations for play in online games. *CyberPsychology & behavior*, 9(6):772–775, 2006.

[134] A. Zinck and A. Newen. Classifying emotion: a developmental account. *Synthese*, 161(1):1–25, 2008.