An Architecture for Designing Content Agnostic Game Mechanics for Educational Burst Games

by

Tyler Baron

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved March 2017 by the
Graduate Supervisory Committee

Ashish Amresh, Chair
Brian Nelson
Mary Niemczyk

ARIZONA STATE UNIVERSITY
May 2017

ABSTRACT

Currently, educational games are designed with the educational content as the primary factor driving the design of the game. While this may seem to be the optimal approach, this design paradigm causes multiple issues. For one, the games themselves are often not engaging as game design principles were put aside in favor of increasing the educational value of the game. The other issue is that the code base of the game is mostly or completely unusable for any other games as the game mechanics are too strongly connected to the educational content being taught. This means that the mechanics are impossible to reuse in future projects without major revisions, and starting over is often more time and cost efficient.

This thesis presents the Content Agnostic Game Engineering (CAGE) model for designing educational games. CAGE is a way to separate the educational content from the game mechanics without compromising the educational value of the game. This is done by designing mechanics that can have multiple educational contents layered on top of them which can be switched out at any time. CAGE allows games to be designed with a game design first approach which allows them to maintain higher engagement levels. In addition, since the mechanics are not tied to the educational content several different educational topics can reuse the same set of mechanics without requiring major revisions to the existing code.

Results show that CAGE greatly reduces the amount of code needed to make additional versions of educational games, and speeds up the development process. The CAGE model is also shown to not induce high levels of cognitive load, allowing for more

in depth topic work than was attempted in this thesis. However, engagement was low and switching the active content does interrupt the game flow considerably. Altering the difficulty of the game in real time in response to the affective state of the player was shown to increase engagement. Potential causes of the issues with CAGE games and potential fixes are discussed.

DEDICATION

To my parents; for all their help, whether I wanted it or not.

ACKNOWLEDGMENTS

I would like to thank Ashish Amresh for being an amazing and helpful advisor and chair of my committee as I could not have gotten this far without your help.

I would like to thank Mary Niemczyk for both being on my committee and for an excellent class which provided a substantial amount of the theoretical basis for this thesis.

I would like to thank Brian Nelson for being on my committee and helping me make this thesis more refined and much higher quality.

I would like to thank Scotty Craig for his last minute help and his help with the methodology employed in the CAGE study.

I would like to thank the students from the Fall 2016 Game Based Learning class for their feedback on the architecture and using it to develop most of the games used in the study.

I would like to thank Alankrit Shah for not only making a game with the architecture, but also for his help in developing the architecture itself as well as helping with the data collection tools for the study.

TABLE OF CONTENTS

Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

DEFINITIONS

1. Burst Games – A design paradigm for educational games which focuses on short, fast paced levels of action to learn through repetition and keep student frustration low (Amresh, Clarke & Beckwith, 2014).
2. CAGE Framework – The software architecture and framework that was developed to expedite the design and development of educational games that follow the content agnostic game engineering design paradigm.
3. Content Agnostic Game Engineering (CAGE) – The process of developing an educational game that uses the content agnostic mechanics design paradigm.
4. Content Agnostic Mechanics – Game mechanics which are not related to the content being taught and can be reused across multiple content domains without modifications (Baron, Heath & Amresh, 2016).
5. Game Mechanic - A method the player can use to interact with the game world (Sicart, 2008).

CONTRIBUTIONS

Baron, T., & Amresh, A. (2015). Word Towers: Assessing Domain Knowledge With Non-Traditional Genres. In European Conference on Games Based Learning (p. 638). Academic Conferences International Limited.

Baron, T., Heath, C., & Amresh, A. (2016). Towards a Context Agnostic Platform for Design and Assessment of Educational Games. In 10th European Conference on Games Based Learning: ECGBL 2016 (p. 34).

Baron, Amresh, Nelson & Niemczyk (In Progress) The CAGE study.

Baron, Khan, Craig, Amresh & Gonzalez-Sanchez (In progress) Affective Computing in CAGE games.

1 INTRODUCTION

Overview

Video games as a whole suffer from the common perception that they are just a mindless activity with little or no value beyond entertainment (Shaffer, Halverson, Squire & Gee, 2005). In spite of this viewpoint, research into how games can be used as educational tools has been steadily increasing for some time. While this means that interest by both educators and developers has also increased, educational games are not yet a staple of the modern classroom. The slow rate of adoption speed is due to three key problem areas. The first is the difficulties educators face when trying to integrate games into their existing curriculums (Blanco et al., 2012). The second problem area is that while both academia and industry are interested in educational games, industry often does not make use of current research or base much of their work on established learning theory leading to ineffective games (Shaffer et al., 2005). The final problem area is the actual development of these games. It takes a great amount of both time and money in order to produce a single educational game (Moreno-Ger, Burgos, Mertinez-Ortiz, Sierra & Fernandez-Manjon, 2008). Once a game is made, the development team will have to start from scratch again to make another game, as little of the code from the first game will likely be usable in the second project. Content Agnostic Game Engineering (CAGE) tackles this particular problem area by developing a software architecture that will allow developers to efficiently build game mechanics that are effective at making the game educational and entertaining regardless of the knowledge domain being taught, meaning the mechanics can be reused in subsequent games.

The Current Problem

As previously stated, it requires a great deal of time and resources to create one educational game (Moreno-Ger et al., 2008). After development is complete, the developers often have to start completely over with brand new code, or have to modify the existing code from the previous project so much that it amounts to almost the same work load. This extra time and effort results in fewer educational games being produced as developers lose time and money that could be spent on other projects. If developers were able to reuse most of their existing code base with only minor changes, developers would be left with a great deal more time to pursue further projects and would have to spend less money per game. The reason that developers do not already do this is due to a core flaw in the way these games are designed.

This flaw is that the game mechanics and the content being taught are deeply tied together (Van Eck, 2006). At first glance, this approach may appear to be beneficial. A strong connection between the content and the mechanics would be the obvious choice for maximizing the teaching effectiveness of the game and makes it easy to design metrics to assess student progress (Mislevy et. al., 2014). However, designing the mechanics to go hand in hand with the content makes it difficult to reuse those same mechanics in a different content domain. If the mechanics are designed for teaching spelling, it would be difficult to reuse them to teach history without making significant changes. This issue of reusability is where CAGE becomes more beneficial. It is possible to design mechanics that are potentially as effective as existing ones without having a strong connection to the knowledge content (Baron & Amresh, 2015). Adopting a content

agnostic paradigm would solve the reusability problem without sacrificing learning effectiveness.

Research Questions

This work seeks to answer several research questions.

1. Can CAGE be easily used for rapid creation of educational games?

A primary goal of CAGE is to increase the development speed of educational games. By allowing developers to reuse a significant portion of their code base in subsequent games, the CAGE development model should greatly speed up production of subsequent projects. The framework should reduce some of the time needed to build the first content, but the majority of the benefit will be noticeable in all other projects that use the same mechanics. The mechanics will need to be written first, but then the same mechanics can be reused. This means the only portion of the code that needs to be changed for new versions of the game is the content component. The CAGE framework provides a solid base on which the content can be developed, meaning multiple content components can be created with little effort.

2. Does the context or mechanics play a role in the cognitive load measured while playing CAGE games?

Cognitive load can account for differences between learning two topics of seemingly equal rigor (Sweller, 1994). Processing information takes some amount of mental effort. However, it can take additional mental effort to process extraneous information that appears when the content is provided. A student will then spend

cognitive resources processing the extraneous information leaving fewer resources to actually learn. Essentially, the way in which a topic is taught can have an effect on how well a student learns the topic. The two contents available in each CAGE game used in this study may have differences in their cognitive load as one topic may be much more rigorous than the other. Another concern is how the mechanics affect cognitive load. Again, how the topic is taught can have an effect on student learning and some mechanics may require much more of the student's attention and distract from learning. Lower learning gains from some CAGE games may be explained if those games carry a higher cognitive load than the CAGE games that had higher learning gains.

3. Can CAGE games maintain player engagement?

If players are not engaged, they will be less likely to learn. Players are also less likely to continue playing a game they find boring, meaning they will have even less opportunity to learn. Therefore, CAGE games should aim to keep players engaged. Also, differences in engagement levels between individual CAGE games used in this study could help provide an explanation for differences between learning gains.

4. Does changing the context of CAGE games interrupt the game flow?

While CAGE provides a way to easily develop secondary games that could be marketed as separate products, it should be possible to swap in a new content without any interruption to the game flow. The CAGE framework is designed to have the content component switched out without any other changes. While such a switch is certainly possible on a technical level, it is unclear if it can be achieved on a game play level.

Swapping out the content could prove to be a jarring transition which would break player engagement.

5. Does dynamically changing game difficulty in CAGE games based on player affective state improve engagement?

Players will learn best when they are engaged and in a state of cognitive flow (Csikszentmihalyi, 1996). Flow is achieved when the player's skill is well matched with the challenge provided by the game. If the game is too difficult then players will become frustrated, and if the game is too easy players will become bored. Both of these affective states can be visually detected using facial tracking software. By determining the player's affective state, the current difficulty can be adjusted if the player is bored or frustrated. This is expected to lead to an increase in engagement, as the difficulty should be kept close to the player's skill level. This type of adjustment has not been done in real time during the game before, and thus it is not clear if this adjustment actually leads to increases in engagement.

Potential Contributions

The CAGE framework stands to make several important contributions. For one, it provides a software architecture for quick creation and rapid prototyping of CAGE games. This is a boon to both industry and academia. By providing developers with a system for easily creating educational games with mechanics that can be reused easily in future projects, developers in industry can save both time and money on development costs. Full expense will only be needed for the first version of a game, since subsequent

games will only require changes to the content and possibly the art work. However, the game mechanics can remain as is, greatly reducing development time on these subsequent games. While academic researchers in educational games are not bound to profit margins as the industry is, there is still great benefit in their use of CAGE, notably in the reduction of time spent on development. Time spent on developing a game is time that is not being spent on actually conducting, analyzing, and reporting research. CAGE will expedite the development process and allow for easy creation of alternate versions of the games as may be needed for different condition groups within a study.

In addition to the framework, CAGE provides a foundation for disconnecting assessment from the mechanics as well. Within CAGE, student assessments that are shown to be valid for one version of the game would implicitly be valid for the other content as well, given that the player's means of interacting with the problems is identical in both domains. This thesis does not explicitly take time to validate CAGE as an assessment tool, but this does open avenues for future work with and expansions upon CAGE.

2 BACKGROUND LITERATURE

Theoretical Foundation

Intrinsic and Extrinsic Motivation

As psychologists first began to tackle the issue of motivation they envisioned motivation coming from two sources: basic biological needs and extrinsic rewards such as money (Sansone & Harackiewicz, 2000). However, closely examining an individual's daily behavior quickly gave rise to a major problem with this theory. As early as the 1920's, psychologists began to notice that people regularly performed tasks in their free time that did not fit into either of these two sources of motivation. White (1959) stated that motivational theories founded on these instinctual drives cannot explain exploratory or playful behaviors. Engaging in play does not satisfy any biological needs and, unless there is a promised reward for playing or winning the game, there are no external motivators under this theory. This gave rise to the idea of intrinsic motivation, where the rewards are inherent to the task itself (Sansone & Harackiewicz, 2000). These rewards could serve as replacements when biological or extrinsic motivators were missing, but intrinsic rewards could also provide additional reinforcement beyond motivators already present. In short, intrinsic and extrinsic motivations are not mutually exclusive and both types can motivate a given behavior. For the purposes of game based learning, biological motivation is not considered a factor given that games and learning are driven by intrinsic and extrinsic motivation rather than biological needs.

The argument for intrinsic motivation got a notable boost from Deci (1972) when he published the results from an experiment with undergraduate students. The results

showed that students who were paid to solve cube puzzles were less likely to attempt the puzzles on their own during free time, while students who were not paid to solve them were more likely to keep working on the puzzles on their own. If intrinsic motivation were not a factor, then one would expect the paid students to keep working on the puzzles in their free time. The extrinsic motivation would make up the sole basis for desire to work on the puzzles. It is possible that students might not work on the puzzles in their free time without the promise of monetary reward, but then neither group would be expected to continue working on the puzzles. There remains a troubling paradox with these results; if the activity is intrinsically motivating then both groups would be expected to continue working in their free time. These results were a stark example of this issue which Deci (1971) had already encountered. He found that extrinsic reinforcement by paying a person to perform a task could effectively buy out their intrinsic motivation. However, a second experiment provided extrinsic motivation by means of verbal encouragement and participants did not seem to lose intrinsic motivation as a result. Deci (1971) suggested that this may be because social approval is less likely to be viewed as a control mechanism compared to a monetary reward. These results suggest that while an activity can be both intrinsically and extrinsically motivating, the two types can be in competition with each other. Namely, extrinsic rewards can reduce the intrinsic value of a task.

Deci (1971) called these observations the over justification hypothesis which Lepper, Greene & Nisbett (1973) tested with preschool children. They found that children in the group who drew pictures for an expected reward had less intrinsic interest in

drawing after the study, while the control group with no reward showed no impact on intrinsic interest in drawing. However, the study had a third group where the children were surprised with a reward after they finishing drawing. Unlike the group that was expecting a reward, this group showed the same or increased intrinsic interest in drawing after the study. Mehren (1985) supplemented this argument after studying creativity. She found that creative people had a harder time being creative in environments where their work would be judged. These two results further complicate the relationship between the two types of motivation. A given behavior can be both intrinsically and extrinsically motivating (Sansone & Harackiewicz, 2000). Providing extrinsic motivation without care can ruin the intrinsically motivating aspect of an activity (Deci, 1971), but providing extrinsic motivation in an appropriate fashion can increase the intrinsic motivation of a behavior (Lepper, Greene & Nisbett 1973). Partially due to this complexity, researchers have recently begun to push for a more multifaceted viewpoint on motivation rather than try to fit all human motives into only two broad categories (Reiss, 2012). These theories however are currently underrepresented and understudied in the literature. Due to this relative lack of research, the motivation model of intrinsic and extrinsic motivation was applied throughout the CAGE thesis.

The intrinsic and extrinsic motivation model also fits well with the problems surrounding game based learning. Games are often seen as being intrinsically motivating because there are usually no real world rewards for playing them, yet video gaming has become a massive industry (Kong, Kwok & Fang, 2012). Learning may be intrinsically motivating, but our education system tends to add extrinsic rewards such as grades to the

process which can reduce the intrinsic value of learning (Shute & Ventura, 2013). Video game players average twenty six hours of play time a week (Williams, Yee & Caplan, 2008). Most teachers would be astonished to find a student spending that much time on their studies. The desire to capitalize on this highly intrinsically motivating activity has been one of the reasons for the push behind game based learning.

Kong, Kwok and Fang (2012) investigated the role of motivation in games. In particular, they looked at online multiplayer educational games. These online games are of notable interest as a potential education platform because of player interaction. This interaction could lead to collaborative learning. Collaborative learning increases student interest and helps build critical thinking skills (Gokhale, 1995). In addition to critical thinking skills, online games have been shown to be safe learning environments which can teach research methods, game design, cyber culture and has shown potential for teaching law and policy (Delwiche, 2006). However, collaborative learning and its ties to motivation had not been explored in this context of online educational games (Kong, Kwok & Fang, 2012). They proposed two new motivation constructs based on peer interaction in an online game environment. The first construct was peer intrinsic motivation where individuals are driven, as a group, to engage in a behavior depending on the values and standards of the group. The second was peer extrinsic motivation, where a player's desire to advance themselves in the game may be in conflict with their desire for collaborative victory. In an online only game, the results of an action, whether they are rewards or consequences, cannot simply be undone by reloading the game. Such potential consequences for failure can lead players to not attempt a collaborative effort at

all for fear of consequences to themselves individually. There is also the issue of envy caused by competition, as players can have varying skill levels (Kong et al., 2012). This may lead players to focus instead on advancing themselves to catch up to other players. They also do not want to work with others as much, because working with them would push the other player even further ahead. Essentially, extrinsic motivation can lead to a push for dominance (Bénabou & Tirole, 2003). Kong et al. (2012) hypothesized that an increase in peer intrinsic learning would lead to an increase in intention to learn collaboratively, while an increase in peer extrinsic motivation would lead to an increase in intention to lead individually Their results supported both of these manipulations to motivation causing such changes in behavior This demonstrates the role of motivation within multiplayer games and again shows that motivation is complex, even within two broad categories The role of motivation shown here has important implications for both educators and game developers and designers. An educator can change the types of motivation by means of class-wide achievements or a leaderboard to create either a collaborative or competitive environment as they desire (Urdan & Schoenfelder, 2006). Likewise, the developers of a game can build the game to provide a friendly social environment to encourage collaboration and provide educators with tools to create their own learning structure (Kong et al., 2012). Regardless of the intended learning atmosphere, both developers and educators need to be aware of how these motivations are affecting their players and the effectiveness of their learning.

From this previous research, it is clear that motivation is a complex subject that cannot easily be simplified for the purposes of game based learning. While some

researchers are starting to break out of the classic two category model of motivation (Reiss, 2012), the original model from Deci (1971) remains widely used and popular. Kong et al. (2012) took a closer look at motivation within a gaming context and found that simple changes can have a great effect on both student desire to learn and the way in which a student will go about learning. Again motivation was shown to be very complex, but also that a dual model can still be profitably applied to learning through games. The intrinsic and extrinsic model may not provide the most accurate and in-depth view of human motivation, however it is generally suitable to the needs of game based learning. Despite the acknowledged weaknesses of this model, it remains a primary factor in game based learning research.

The dynamic between intrinsic and extrinsic motivation will play a key role in CAGE games. Typically a game that disconnects the content and mechanics will be driven by extrinsic motivation (Van Eck, 2006), and successful students are often driven by both intrinsic and extrinsic motivators (Niemczyk & Savenye, 2001). The gameplay only serves as motivation to continue with the educational topics. There is nothing about the educational content itself that drives students to continue to learn. This is where CAGE games will need to surpass current models by keeping the gameplay and content close enough to keep the experience intrinsically motivating.

Self-Determination Theory

Self-Determination theory is a popular theory of motivation that places motivation in the context of self-improvement (SDT; Gagné & Deci, 2005). According to SDT, a person is continually motivated to improve themselves and achieve mastery over their

environment. When a person encounters something in their environment that they do not yet understand, they will be inherently driven to study and understand this new concept or object. After they have mastered it, the person can incorporate the new skills into their being, thereby improving themselves.

Motivation as a whole and SDT are both important in education (Ryan & Deci, 2000). Students who are motivated to learn show increased engagement in the classroom, give their teachers higher ratings, and are less likely to drop out. Of particular interest to educational gaming is the higher level of engagement. Students who are engaged in their gaming activity are more likely to learn and are less likely to give up on their gaming task (Baron & Amresh, 2015), just as students in a normal classroom setting are (Ryan & Deci, 2000). According to SDT, there are three key factors that motivate individuals in any given setting. These are relatedness, competency, and autonomy (Deci & Ryan, 2000). Relatedness is the need to feel a connection with others. This is considered important, but is considered the least critical of the three. Relatedness is also the hardest to fit into educational games, especially if there is no multiplayer component to the game. Due to this difficulty and its lesser importance, relatedness is not factored into educational games research as much as the other two factors. Instead, relatedness needs to be achieved through either through simple means in the game, such as an in-game chat system, or rely on external means such as having all students collaborate verbally while playing (Denis & Jouvelot, 2005).

Competency is the second key factor from SDT and it is one that is easier, and important, to integrate into educational games (Baron & Amresh, 2015). Competency is

the psychological drive to need to feel that one is doing a sufficient job at the task they are performing (Deci & Ryan, 2000). It is the need for competence, whether it is an everyday task, a classroom activity, or trying to finish a level in a video game. When learning and attempting something new, it is expected that the student will not have full competence when first attempting the task. However, the student should still have some level of perceived competence, otherwise they fall behind and may cease trying (Chen & Jang, 2010).

To prevent students from getting discouraged with a low perceived competency, it is important to build student confidence through positive feedback (Deci & Ryan, 2000). One reason for this is because positive feedback is seen as a sign that the student is doing well in their task, which improves their perceived competency (White, 1959). The second benefit to positive feedback is that immediate feedback on performance helps scaffold the learning and guides the student as they learn (Lester et.al, 2013). Feedback is an element that is already needed in educational games in order to inform the student of their progress, what areas they need to improve on, and help them in achieving the learning objectives. Reflecting on individualized feedback is helpful to students, though students will not always view or focus on feedback (Nelson, 2007). Ensuring the feedback always maintains a positive tone will encourage students to view it and help make sure it maintains a higher level of perceived competency for the students that view it (Lester et.al, 2013).

The final key factor of SDT is autonomy, which is maintaining a feeling of control and free will (Deci & Ryan, 2000). In the context of educational games, it

requires giving the player a level of freedom to do as they choose without being controlling in how the game guides the player (Baron & Amresh, 2015). This can be a difficult state to achieve as there must be some level of guidance in order to ensure that students are working towards the learning objectives of the game. There is also cause for concern regarding providing rewards for student progress. Rewards may seem like an ideal way to motivate students to work towards the goal of the game but offering too many rewards, or not choosing carefully which type of rewards to give to students, can actually decrease their motivation after the task is over (Deci, 1971).

SDT is an important theory for educational games (Ryan, Rigby & Przybylski, 2006). Educational games place students in a virtual environment to teach and assess their knowledge. Since SDT holds that a person is driven to master their environment, this goes hand in hand with the environments of educational games where the goal is for the player to attain mastery over the concepts presented in the game world. Methods for alleviating outstanding issues with both autonomy and competency can be found within the burst games model, which is discussed in detail later (Amresh et. al., 2014). CAGE games are designed as burst games, and so they incorporate these methods into their design.

Cognitive Load Theory

Cognitive Load Theory (CLT) is a learning theory that aims to explain the differences in difficulty in learning equal amounts of content from different topics (Sweller, 1994). Students will often struggle to learn content from one task and domain, while easily picking up a similar amount of knowledge from a different domain. CLT

holds that the reason for these differences is the amount of cognitive load involved in learning the two different topics. Cognitive load is the amount of mental processing needed to perform a certain task. The number of items that a student needs to hold simultaneously in their mind, extraneous information, pressure to perform well or quickly, and many other potential factors increases this load. CLT defines three types of cognitive load (Paas, Tuovinen, Tabbers & Van Gerven, 2003). The first is intrinsic load, which represents the interaction between the nature of the material being taught and any expertise the learner has. This is fixed and cannot be changed by instructional design. It will vary on a student by student basis and will always be present. The best that can be done about intrinsic load is to try and mediate its effects by observing the students and assisting those that fall behind. Due to its inherent and unchangeable nature, it is more important to focus on the two other types of load which instructional design can have an effect on.

The second type of cognitive load is called germane load and is the beneficial type of load in instructional design (Paas et. al., 2003). It is the type of load that contributes to knowledge acquisition. Germane load is caused by the student thinking and processing the problem or information being presented. It is important for educational games to divert unused cognitive power into germane load by giving the students the opportunity to think deeply about the material (Kiili, 2005). In short, it is important to have the students spend as much time as possible thinking about the problem and content itself, rather than focusing on extraneous gameplay elements. Highly immersive games that bring about a strong feeling of virtual presence actually have shown to have

decreased learning gains compared to a low presence group (Schrader & Bastiaens, 2012). This may be because highly immersive environments may distract the player too much from the task at hand, which would lead to reduced learning. On the other hand, prioritizing learning over gaming content often leads to even worse learning outcomes as students become unmotivated in low gaming environments (Peirce, Conlan & Wade, 2008). This means that game design needs to be prioritized, but great care needs to be taken in order to ensure that the gaming aspects do not completely overshadow the educational content. The educational first approach is part of the problem that the CAGE model solves. By disconnecting content and mechanics, developers can take a game design first approach by designing a game and then deciding what content they can teach with it, rather than the other way around.

The final type of cognitive load is extraneous load, which accounts for distractions and other elements of poor instructional design (Paas et. al., 2003). Extraneous load will occupy some of the learner's limited cognitive capacity, limiting the amount of germane load they can sustain. This was the issue that Schrader and Bastiaens (2012) found with their high presence gaming environment. If there is too much detail in the gaming world, or if there are several game mechanics that do not contribute to the learning task, then they become distractors that detract from the learning experience. Striking the balance between good game design and good instructional design remains a great challenge in educational gaming research (Peirce et. al., 2008). In order to be effective teaching tools, educational games must maintain an adequate level of germane load, with as little extraneous load as possible or none at all.

This model of three types of load has been challenged recently due to difficulties in distinguishing germane load from intrinsic load (Kalyuga, 2011). Germane load was added to the model after evidence suggested that cognitive load does not always interfere with learning, as had been proposed with the original model (Sweller, Van Merrienboer & Paas, 1998). Kalyuga (2011) argues that the intrinsic and extrinsic only model explains cognitive load sufficiently and that germane load may be better defined as the working memory resources used for processing intrinsic load.

This need to minimize extraneous load is problematic for CAGE games like all other educational games. Multimedia learning can easily overload a learner by providing too much information at once, or by demanding split attention from the learner (Mayer & Moreno, 2003). CAGE games need to avoid this, but some level of complexity in game play is desirable from a game design perspective. Certain game genres will implicitly require some amount of extraneous load. A real time strategy game will require the player to manage several units and resources at a time while keeping an eye out for incoming threats. It would be difficult to link each of these to learning content, and so some mechanics end up becoming extraneous load. At the same time, a subset of players will find such a game significantly more engaging than a simpler game which would be easier to connect all mechanics to a learning objective. The core goal of CAGE with respect to cognitive load is ensure that the CAGE model itself does not add significant extraneous load to any game created with it. This is what research question two examines as differences in cognitive load may help explain differences in learning gains between two CAGE games (Graesser, 2017).

Cognitive load may come from any number of potential elements of a CAGE game. This means that the overall load measured while playing a CAGE game can come from the user interface, a particular game mechanic, the controls, or any other specific element within the game. Rather than try to pinpoint any specific element within the game that was a source of cognitive load, an overall scale was used (Paas et al., 2003). This was chosen due to the already long length of the survey which could impact game flow, as well as the high variability in the CAGE games. While questions could have been developed to measure the cognitive load from certain elements in some of the CAGE games, these questions would have been wasted on the CAGE games that did not include these elements.

Student Assessment

Evidence Centered Design

Evidence Centered Design is an instructional design theory that assessments should be built with a focus on evidence-based arguments (ECD; Mislevy, Almond & Lukas, 2003). This theory arose in response to criticisms about the ability of traditional assessment methods to meet the needs modern students. Cognitive science and learning theory had advanced, but assessment methods had not. Student understanding can be effectively measured by observing their reaction to changes in observable variables. More important is the student's perceptions about the effects of changes on unobservable variables. However, tasks that allow for this level of complex observation are difficult to set up, maintain and design in a way that will allow for fair assessment of all students. ECD offers a way to do this through multiple models that encompass either the students'

understanding of the content, or how to measure their knowledge. The student model is the representation of the students' knowledge. Evidence models are the explicit set of instructions on how to update the student model based on their task performance, and the measurement model is the component of that model that directly ties the variables in the student model to the observable variables from the task. Incorporating these models into the architecture for the content agnostic mechanics will be important to ensure that the assessment component of the games remains consistent and valid.

Stealth Assessment

Stealth assessment is a method for performing unobtrusive assessment that assesses student knowledge without the student being aware that the assessment is occurring (Shute, 2011). This method of assessment is particularly useful in educational games because obtrusive assessment will usually break player engagement (Baron & Amresh, 2015). When the player is engaged in the game, they will enter cognitive flow (Csikszentmihalyi, 1996). While in flow, the player will perform their best, which makes it an ideal state to maintain so they can be assessed while at their peak. This is why stealth assessment is important to ensure that this level of engagement is not broken during assessment (Shute, 2011). Rather than presenting the student with a question directly, it should be the way in which the player interacts with the problem that provides the opportunity to determine the players' knowledge level. How a student solves a problem is as important as whether or not they are able to solve it. Likewise, if they cannot pass a specific challenge, that itself is a form of feedback to them and self-assessment about their performance. The architecture for content agnostic mechanics will

incorporate stealth assessment in order to ensure that players do not become disengaged from the experience.

Current Status of Educational Games

Despite the great amount of interest and motivation in educational games, both from researchers and students alike, it is difficult to draw distinct conclusions about their effectiveness as teaching tools (Hannifin & Vermillion, 2008, Chapter 2). In fact, there are many bold claims that have been made about the potential for educational games to revolutionize learning in the classroom, but these claims are not backed by meaningful evidence (Mayer, 2014). This is similar to claims made about various new technologies such as motion pictures and television as they have appeared over the last hundred years or so (Cuban, 1986). New entertainment technologies often elicit these types of revolutionizing claims, but as Mayer (2014) points out they often fail to meet expectations.

This is not to dismiss educational games, but rather to temper immediate expectations about their performance. Tobias and Fletcher (2011) conducted a survey of the state of educational games and were able to conclude that people do learn from games. However, the issue lies with who will learn from games and under what circumstances. Many factors such as socio economic status affect whether or not a person will even have access to educational games and whether they will have sufficient time and supportive instruction to learn effectively from the games (Leemkuil & de Jong, 2011, Chapter 13; Dai & Wind, 2011, Chapter 19).

Recent studies back up Tobias and Fletcher's (2011) claim that students do learn from educational games(Freeman & Higgins, 2016). Additionally, recent studies have shown that the learning behavior of the student is an important factor and can increase learning gains (Sung & Hwang, 2017) and have shown ways game play data can be mined for techniques used by successful learners (Rosenheck, Lin, Klopfer & Cheng, 2017). However, these studies often use measures that are very specific to the game which they are testing which follows the approach advocated for by ECD (Mislevy et al., 2014). This makes it hard to generalize their results beyond the games used in the individual studies. The data mining approach used by Rosenheck, Lin, Klopfer and Cheng (2017) is the most content agnostic of these approaches, but these analysis techniques start to become a study design that Mayer (2014) cautioned against, which is collecting a massive amount of data and then determining what can be observed afterwards. Data mining techniques, like those used by Rosenheck et al. (2017) are certainly useful but must be used to detect specific predefined patterns that show student learning.

Learning gains in educational games have been shown (Tobias & Fletcher, 2011; Freeman & Higgins, 2016; Rosenheck et al., 2017) but these measures are not content agnostic, or even game agnostic, and are therefore limited to the specific games used in the studies. Showing learning gains in educational games in general will require measures that can be used across a variety of different games and topics. CAGE games are ideal for this because the mechanics are not tied to specific topics or games and can be applied to other games from the same genre.

Software Architecture

Source Code Refactoring

In software engineering, refactoring is a term for rebuilding a portion of the source code. This rebuilding will take time and resources to complete. The main goal of this thesis is to reduce the amount of refactoring needed across multiple content domains. This is a key reason for adopting the component based approach. Only the content component would need to be refactored across multiple domains. The mechanics component is developed once and then used across the different content domains, and the framework remains unchanged across all versions of the game. To help bridge the gap between component based software architecture design and educational game design, this thesis will make use of the Game Software Model (GSM) from Tang and Hanneghan (2013) shown in *Figure 1*. The GSM divides the individual facets of the game into six categories. Four of these layers are Game Resources, Component Wrappers, Helper Components and Core Components. These four layers will all be part of the framework portion of this architecture as they are always needed in any game, regardless of content or mechanics. The Game Logic and Interactivity layer is the part that will comprise the mechanics component in the architecture. Recall that mechanics are the way in which the player can interact with the game world. The final layer, Game Specific Systems, is the one that matches with the content component in this architecture. Out of the six layers, this is the only one that will need to be refactored for each content domain.

Epistemic Forms and Games

Epistemic forms are specific knowledge structures that are filled out as a student learns a new topic (Collins & Ferguson, 1993). These can be hierarchies or networks of knowledge nodes that fit in a particular pattern to describe how the concepts relate to each other. Different disciplines will have different epistemic frames about knowledge, as each frame sets importance as well as the relationship between concepts (Shaffer, 2006). The importance of particular concepts will naturally vary from one group of professionals to another. Epistemic frames are essentially the context in which knowledge is being used and applied.

The use of frames in this manner leads to issues with transfer of training (Shaffer, 2006). Transfer of training is when existing knowledge or skills affect the performance of new tasks (Cormier & Hagman, 2014). A person trained in one frame will not perform as expected when placed in a new job or context, changing the frame. Their knowledge fits with the old frame and they will attempt to act as the old frame taught them. Despite the knowledge being common to both tasks, the person's training did not transfer from one frame to the other. Over time they can be retrained to fit in the new frame, but this takes time and money. It is preferable for all involved if the person had been able to transfer their training from one frame to the other (Baldwin & Ford, 1988). This is of great concern to employers who invest a large amount of time and money into a new employee, as well as educators who want their students to perform as expected when outside the classroom environment. If the training is strongly integrated with a particular frame, the training will not transfer properly to other frames (Shaffer, 2006).

Transfer of training is also related to cognitive load as balancing germane and intrinsic load has been shown to be an issue (Van Merrienboer, Kester & Paas, 2006). Namely, providing the learner with simpler tasks may reduce cognitive load, but makes the material less motivating and prohibits transfer to more complex situations as often found in the real world. The goal is to reduce intrinsic load for novices to make initial learning more manageable and increase intrinsic load over time as their prior knowledge increases. This increase allows for more complex tasks, which emulate more real world situations and increases transfer of learning.

Epistemic games are the rules and strategies that guide the formation of knowledge within the structure of a particular frame (Morrison & Collins, 1996). This definition of epistemic games covers a broad spectrum and incorporates essentially all forms of guided instruction. The conversations between students and teachers as well as activities done within the class room can all be considered forms of epistemic games. The problem is that these games are a set of rules and strategies that are specific to the particular frame in which the instruction is presented. If those rules cannot be applied in a different frame, then the training will not transfer (Shaffer, 2006). This is because the rules of the game are strongly tied to the knowledge being taught. Like educational games in general, the mechanics of epistemic games are strongly integrated with the frame in which the current content is being taught. As long as this connection remains, the training will not transfer well. By contrast, rules for understanding and applying knowledge that are not frame specific would allow students to apply their training in multiple frames and contexts easily.

Endogenous and Exogenous Games

Endogenous and exogenous games are two classifications for educational games that describes how well connected the educational content is to the game itself (Malone & Lepper, 1987). Exogenous games are games where the content and game play have little or no relation to each other.

Example. Imagine a game where the player pilots a spaceship and shoots aliens, and must complete a math quiz at the end of the level.

The above example game contains both game elements and educational instruction. It is certainly an educational game, but the education and the game elements are distinct. The player's success on the math portion in no way affects the math quiz and their performance does not in any way improve their ability to destroy alien ships in the level. This is what makes this example game an exogenous game (Halverson, 2005). The two parts could each be done separately and would function exactly the same and would stand on their own. There is no interdependence between these two components.

Endogenous games on the other hand are games where the game play and content have a connection and an effect on each other (Malone & Lepper, 1987).

Example. If in the example game above, scoring well on the math quiz empowered the player's spaceship.

This example game would now be closer to being an endogenous game (Halverson, 2005). The player's math skills impact their performance in the alien space ship portion of the game. The player is rewarded for doing well on the math quizzes and

thus is provided motivation within the game to do well on the math questions. There is still no reason to do well in the space ship portion as it has no effect on the math questions, but there is at least a minimal interaction between the two components. The example game would still be seen as an exogenous game, but there are now endogenous elements in it.

Endogenous games are examples of ECD (Mislevy, 2003), and some have been pushing for their use compared to exogenous games which are more common in the classroom (Halverson, 2005). However, there are multiple considerations to take into account when comparing the two. Endogenous games are often much more complex to play, meaning class time is spent on learning how to play the game and less can then be spent on learning the material. At the same time, these games have a strong basis in realism, meaning the game reacts to the player's actions based off of complex models. Proponents of endogenous games claim that this creates a superior learning environment as players can easily investigate alternative history and other subjects difficult to emulate in the classroom. However, exogenous games better fit the current educational model of using widely accepted content standards. These standards define what should be taught in the classroom and teachers have long worked to streamline their delivery of these topics. Since the content delivered in exogenous games does not change with player performance, teachers are assured that students receive the required material and do not accidentally miss some intended content because of choices they have made in the game. This makes endogenous games less appealing to educators who need to stick with a prepared content plan.

Still, endogenous games are seen as a superior choice and as the better way forward (Mislevy et al., 2014). However, having a strong connection between the game play and the content does not explicitly make the game a better teaching tool (Habgood, Ainsworth & Benford, 2005). Instead, it is important to deliver the educational content through the most enjoyable part of the game, allowing the learning to take advantage of the flow state. While this approach still advocates for a connection between the mechanics and the learning, and thus endogenous games, it is important to note that integration alone is not enough. How the material is represented is also a critical factor.

CAGE games are then caught between the classification of exogenous and endogenous. In order to disconnect the content and mechanics, it is necessary to push the games in an exogenous direction. However, it order to maintain principals of ECD (Mislevy, 2003) and link the content with the core game play (Habgood et al, 2013), the games will need to be endogenous. Overall, CAGE games are endogenous, but the mechanics are designed to have any of the potential content components layered on top of them. In this way, CAGE games are a combination of the endogenous and exogenous design paradigms and represent the definition of a new hybrid classification.

3 GAME DESIGN PRINCIPLES

Game Mechanics

A game mechanic is a method the player can use to interact with the game world (Sicart, 2008).

Example Game Mechanic. In Super Mario Brothers[1], the player can jump by pressing the A button on the controller as shown in *Figure 2*.

One specific game design philosophy is a design paradigm that keeps each game level very short, only a minute or two of play time required, and reinforces learning through repetition (Amresh et al., 2014). These games are commonly referred to as burst games because they present the material in short quick "bursts" rather than through long and complex play. By presenting smaller problems that only require a small time investment from the player for each attempt, players who fail the challenge can quickly try the problem again and incorporate the feedback they received. This allows for learning by repetitive game play, and helps prevent students from getting frustrated (Amresh et al., 2014). Students will become much more frustrated when they work on a problem for an hour only to find that they did it wrong and must start over. The smaller problem size also allows for the problems to increase in difficulty in very small increments. A given concept can be tested several times before a new concept is quietly slipped into one of the problems. This approach to learning by repetition and incremental learning has merits because repetition has long been known to be a helpful tool in reinforcing learning (Ebbinghaus, 1913; Horner & Henson, 2008). However, it has also

been shown that excessive repetition can actually be detrimental to memory access time (Smith, 1984). Due to this, it is important to discuss how learning through repetition can be beneficial, its potential drawbacks, and how it fits within the burst games design model.

Following these concerns raised by the findings from Smith (1984), English and Visser (2014) performed a three part study to further investigate the role of repetition in memory recall times. The first study was a replication of a study by Kuhl and Anderson (2011), which found that increased repetition lead to a decrease in recall, just as Smith (1984) did. The study by English and Visser (2014) replicated these results in its first experiment. Participants were asked to repeat a list of words shown to them and then did a cued-recall task to elicit the words they had just been shown. In these first experiments, participants were never told to remember or otherwise make note of repeated items in the list. The results showed that recall for items actually declined with repetition. A second study did explicitly instruct participants to remember repeated items. In that study, participants showed a positive relationship between repetition of an item and their ability to recall it when prompted. The third study confirmed that placing repetition within an explicit learning context did allow for easier recall of repeated items. In short, while repetition in excess can remove value from a word or cause a person to subconsciously ignore it, doing so within a learning environment actually increases a person's ability to recall that word or term later. Since a player is playing a burst game specifically for the purpose of learning something, repetition works in such a game's favor. This repetition also functions as the feedback loop from Gagné's (1977) Nine Events of Instruction.

Decreasing memory access time is an important factor in teaching new material (Horner & Henson, 2008). When a person first encounters a problem, they go through a series of initial mental processes to determine a response. Repetition can lead to a person being able to bypass these steps on subsequent encounters, making the response faster, easier and more reliable. Such improvements are ideal in many learning situations (Horner & Henson, 2008).

Burst games are also ideal for certain types of players. Researchers examining player behavior in online games have attempted to fit players into different categories (Bartle, 1996). Originally, Bartle (1996) proposed four main player types: Achiever, Explorer, Socializer and Killer. Explorers are players who like to see everything the game has to offer and Socializers like interacting with other players above other gameplay elements. These two player types do not fit well into the burst games model, as the time to interact with the world and others is very limited. Achievers are players who seek absolute mastery of a game. They fit well into the burst games model as quick repeated attempts allow for fast skill growth. Killers are players who seek to change the play experience of others, for better or worse. In a competitive multiplayer burst game, Killers would have ample opportunities to alter another player's experience. Marczewski (2013) took these player types further by adding the underlying motivation for each player type. Achievers are motivated by mastery of the game, which is in keeping with self-determination theory. Marczewski (2013) rebrands Killers as Disruptors and ties their motivation to causing change in the game world. Promoting change in the environment

for one's own benefit is also a part of self-determination theory. These two player types also fit in well with the burst games model and exemplify self-determination theory.

Game mechanics are fundamental to defining how the game is played and a given player's enjoyment of the game is tied to their personal enjoyment of the game mechanics. Because of this, game mechanics make up the core of game design and development. Other aspects of development, such as the narrative, are given a lower priority by most development teams (Amresh et. al., 2014). This is because a low quality story may be a small blemish on an otherwise great game, but a fantastic story will not save an otherwise low quality game from harsh review scores. In short, game mechanics are the central piece of the game for both the player and the developer and these mechanics define how the game is played and are the key factor in determining how entertaining the game is.

Design Considerations for Serious Games

A learning game cannot be an effective teaching tool if it is poorly designed and so there are concerns about how to properly design such games (Rilling & Weschselberger, 2011). Virtual learning technologies are especially appealing to education fields that are very hands on, such as automotive servicing, where practice is critical but physical parts for student use may be limited. Virtual reality and the potential to practice on simulated three dimensional objects could greatly alleviate such situations. However, running a full three dimensional virtual reality scenario with user interaction requires extensive, complex and expensive hardware to accomplish. Due to this limitation, educators instead turn to serious games which only require a computer to run.

The obvious issue with using serious games compared to a virtual realty training option is that computer interaction is very different from the way a student would interact with actual machine parts. Despite this issue, games remain a viable alternative (Rilling & Weschselberger, 2011). Another area that game based learning appeals to is safety training (Bloom, 2009). The biggest reason for this is that games would allow a person to be trained in how to handle dangerous or potentially lethal situations without putting themselves at any actual risk. Some situations, such as a gas line explosion, cannot be safely simulated in a real environment. Also, those most likely to suffer an injury in the workplace are persons age eighteen to thirty five and people in that age group are usually already attracted to video games (Bloom, 2009). Finally, games bring a level of interactivity that safety training has previously been lacking. Traditional safety training methods had the learner in a very passive role as they watched videos, presentations and attended lectures. Bloom (2009) states that interactivity is critical to retention of knowledge and that passive training methods alone were not sufficient to keep workers safe.

Rilling and Weschselberger (2011) proposed a framework for designing serious games and applied it to a game to teach students about starting up an automation lab. The first part of their framework is what they refer to as the industrial environment, but would simply be the environment in a more general case. This environment was designed to be as close as possible to an actual automation lab. A high level of fidelity is important in training games so that learning can more easily transfer to the workplace (Rilling & Weschselberger, 2011). The second major component of this framework is curriculum. It

is important to identify what can be easily taught using your environment. By creating a high fidelity realistic environment that the player interacts with, an easy task that can be taught would be to identify the locations of the important buttons. Learning is easier when the environment and curriculum go together, making learning intuitive.

A second serious game design framework was proposed by McNeill (2004). This model had five key instructional elements to it. The first was that the game should be simulation based. This means putting the player into an interactive learning environment that reflects real world scenarios and then providing immediate feedback about their performance. This feedback would be best delivered through a support character who mentors the player as they learn. Intelligent tutoring systems have used avatars to teach students in the past, but users often dislike interacting with them (McNamara, Jackson & Graesser, 2010). This is an issue that would need to be addressed, and is mitigated by the other points in the model (McNeill, 2004). Following on that issue, the second key element of the model is to maintain a conversational style. This is particularly important in highly technical fields with many key terms a player must learn. A conversational tone will make the content easier to learn and help keep them immersed in the content. A very technical or formal writing style could easily break that feeling for a player. The third key element of this design model is that content should be presented as levels, not lessons. Normally, content is provided to a learner through lessons which progress through the topics one at a time, which is not reflective of real world situations. McNeill (2004) argues that all learning content should be provided and present in the game from the beginning. Then, instead of progressing through topics the levels get steadily more

difficult. This is ideal partly because real world application of this knowledge will usually involve multiple aspects of the field, rather than just a single topic at a time. The last two parts of this model are cognitive apprenticeship and driving the learners to the tools. Cognitive apprenticeship is building expertise through guided learning, providing a second reason to deliver feedback to the player through a virtual mentor character (McNeill, 2004). Driving the learner to the tools refers to placing helpful information within the game to encourage them to use such resources in real scenarios. For example, in a game about a technical topic the designer could include a library which contains manuals with the actual technical information. Players can use this to help themselves through challenging levels while also reinforcing that behavior of referring to physical manuals while on the job.

Another design aspect that is important to keep in mind when designing serious games is the narrative (Amresh et al., 2014). The gameplay and interaction has typically driven design of video games with the story of the game being a relative afterthought. While this gameplay focus is important for ensuring that a game is fun and engaging, the narrative structure of a serious game is more important than it is when making entertainment focused games. The goal of a serious game first and foremost is to educate, and one of the best ways to integrate that into a game is through the story. This presents a problem however because a focus on using narrative to encourage learning can interfere with the cycle of action and feedback that serious games need. One solution to this problem is the suggestion by Lester et al. (2013) to use intelligent learning environments. Such an environment is a game built within a game engine which integrates an intelligent

tutoring system into the game. The intelligent tutoring system is already built to handle many of the issues facing learning within a serious game such as immediate feedback, hint progression and dynamic problem difficulty. The narrative then builds on two distinct but related levels. The first level is the global story arc which is built from a series of possible story events. There is a set of key events that keep the story consistent but many of the events that occur are in response to player success or failure. The second narrative level is the behavior of virtual characters. The way in which these characters interact with the player can easily be built to be a form of feedback. Providing feedback in this embedded manner introduces a new set of problems. The characters have to stay believable, stick appropriately to the current version of the story (Riedl & Young, 2004) and retain their role as authoritative figures (Mateas & Stern, 2005). While these can be challenging to address, success allows the feedback to be built into the narrative and keep players highly engaged by not breaking immersion (Lester et al., 2013).

Content Domain

The content domain of a game is the knowledge area or academic subject being taught (Baron & Amresh, 2015). Unlike game mechanics, content domain is not a regular consideration for non-educational games. This is because non-educational games are not trying to teach any specific skills and therefore do not need to define a content domain. Both commercial and educational games need to teach the player the game mechanics. For development of educational games however, it is critical that a content domain be identified and refined in order to ensure that the game is able to help increase player skill in that area. Some teachers have tried using commercially available non-educational

games and adapting them as a teaching tool (Van Eck, 2006). While this may appear to be a practical solution and studies have shown that it can be an effective way of teaching (McFarlane, Sparrowhawk & Heald, 2002), this brings with it a different set of challenges (Van Eck, 2006). Again, these games were not made as a teaching tool originally, which means the content domain is not deeply integrated with the game mechanics. The content may also be inaccurate and teachers may not be prepared to properly modify the game to update the content and bring it more in line with the core game mechanics. This is one of the reasons that development of educational games is on the rise. By developing the game with the content domain in mind from the beginning and keeping it as a central focus, educational games provide stronger content that is more deeply tied to the core game mechanics keeping the game fun and informative at the same time.

Game Mechanics that are Deeply Tied to the Content

As discussed, using commercially available non-educational games poses several problems when adapting them for educational use (Van Eck, 2006). While a few of these problems stem from teachers often not being proficient with making the necessary modifications (Tang, Hannegham & El Rhalibi, 2009), most of the issues come from the content and game mechanics not being fully integrated. Given the frequent inaccuracies of these games, and possible complications from the teacher's modifications, validity of work done with commercially available games can be hard to verify (Charsky & Mims, 2008). These factors all drive the push for educational game development.

Deeply linking game mechanics and content seems the ideal choice, and is the primary design paradigm in educational game development (Van Eck, 2006). However, this is not without its own set of problems. As an example, consider the following game and mechanics.

Example Deeply Tied Mechanic. Consider designing an educational game that will teach spelling. The resulting game would likely resemble Scrabble[2], as it is already a popular spelling and vocabulary game with mechanics that are ideal for promoting these skills.

A development studio makes a successful spelling game, like the one from the example above. As sales wind down, the studio will need to begin a new project. Here is where the encounter a problem.

Problem. How can the example game by used to teach history?

It would be very difficult to effectively teach history using the mechanics of Scrabble. In order to accomplish this, the developers would need to make so many modifications to the mechanics that the resulting game would no longer resemble Scrabble. This means that the studio must either spend a great deal of time and effort on modifications, or start their next project entirely from scratch.

Disconnecting Content and Mechanics

This has brought this discussion back to where it started, mechanics that are deeply tied to the content are not transferrable across multiple content domains. The only way to keep mechanics transferable is to ensure that they are not deeply tied to the

content. This brings up two problems. The first is the same problem found when trying to modify commercial games for educational use, which is that a disconnect between the mechanics and the content can lead to inaccuracies in the content and difficulties using them as effective teaching and assessment tools (Van Eck, 2006). However, this will not be as big of an issue in this circumstance. Commercial games not only have their mechanics separate from the content, but these mechanics are separate from learning as well. Content agnostic mechanics that are designed with learning and assessment as key factors will perform better as educational tools than mechanics that were designed without these in mind (Baron, Heath & Amresh, 2016).

The second issue would be over generalizing the mechanics to the point of mundanity. Indeed, mechanics that are too general are hard to make enjoyable and hard to build a significant case about their effects on learning. There are also a large number of highly specialized skills that require very focused training. For these reasons, truly agnostic mechanics that can teach and assess any given content domain may be beyond reach. However, having mechanics that can work across a broad spectrum of content domains would be an improvement over the current situation. In addition, if a second mechanic could be designed that, while not working in the same content domains as the first, would work in content domains that the first does not work in would also be ideal. This way, a developer could design a mechanic that would work across a set of content domains of interest to them and use it to build several projects. While developers can implement these mechanics and then recycle the code in their next project, what is currently missing is a model to facilitate designing these mechanics.

4 THE CAGE MODEL

Theoretical Model

Examples of Existing Models

While the proposed model will be the first to provide guidelines of this sort, there exist a large number of existing models that the proposed model will draw on. For example, see *Figure 3*, which is the Game Object Model (GOM) from Amory and Seagram (2003). This is a model for developing educational games, and provides a means of matching pedagogical concepts and game elements. The relationships it makes are loosely based on the concept of Object Oriented Programming (Cox, 1986), which is the most common programming paradigm used today, making the GOM easily translate into code. Amory (2007) updated the GOM to the Game Object Model II (GOMII) which is shown in *Figure 4* and provides an updated version of the GOM to include new parameters that the first version was lacking. There is also the LM-GM model shown in *Figure 5*, which shows a one to one line up of learning mechanics to matching game mechanics (Arnab et. al., 2015). Together, these three models provide an example of a model that shows how to match learning and pedagogy to game elements.

The CAGE Model

The current model for game based learning is shown in *Figure 6*. It consists primarily of a one-way loop that begins with the player inputting their commands into the hardware via keystrokes or mouse clicks. These inputs are passed by the hardware to the mechanics component, which translates them into in-game actions.

Example: The player presses the W key, which moves their in-game character forward.

Recall that game mechanics are the way in which a player interacts with the game world (Sicart, 2008), which is why it serves the role of translating player actions to in-game actions. This action is then passed further up the loop to the content component, which evaluates the action. Some action will need to be evaluated by the content, while others may not.

Example that needs evaluation: The player has put in their answer and has clicked submit.

Example that does not need evaluation: The player moves their in-game character across the room to get within range to speak to a tutorial character.

It is up to the content component to evaluate the action the player has taken and determine if it is correct or not. The result of this assessment is then passed along the loop to the screen, where it appears as feedback to the player. The player then sees this feedback and incorporates it into their next action. This completes the loop of action, reaction, and reflection between the game and the player (Kiili, 2007). *Figure 7* shows this player learning loop, for both single loop and double loop game systems. However, the typical game development process being outlined in *Figure 6* is a single loop system.

The CAGE model is shown in *Figure 8* and is similar to the existing development model, but makes two key changes. The first is the addition of a new step in the loop, called the student model. This will maintain a working model of the students' knowledge

and understanding of the content during the game. This is constantly updated since every new action the player takes is reflective of their understanding. However, the functionality of this system is fixed and does not need to be changed between different content domains. The part that will switch out is the content component, which is the other major change from the existing model. Instead of having a single content component, CAGE games have several. Only one is active at a time, but it is possible to switch between them at almost any time without needing to make changes to any other part of the framework. *Figure 8* shows three inactive contents, and a fourth content which is active. It is important to note that this is just an example. There can be as many or as few contents as a developer wants, though there will be some limit on the number of contents that will work with any one set of mechanics. The more content agnostic the mechanics are, the more content components can be made to fit with them. All of the CAGE games developed for this thesis had two content domains, though many had mechanics that could have easily worked for more topics.

Software Architecture and Implementation

Overview

The architecture is the heart of this research and is the main outcome. Providing this architecture to educational game developers will allow them to build content agnostic game mechanics that can be reused across multiple domains. The architecture will be component based, which will maximize compatibility and reusability (Mei, Chen, Feng & Yang, 2003). A component based architecture is one that is composed of multiple individual pieces that work together in the overall architecture. The architecture produced

by this work will have four main pieces. These are the content component, the mechanics component, the student model, and finally the overall framework that those two components are placed in. *Figure 8* shows how these four pieces are laid out and connect with each other. Each of these is described in detail in its own section below.

A central focus behind the design of the framework is the concept of keeping things generic. Keeping the code as general purpose as possible is the best way to ensure that the code will work for multiple components (Mei et. al., 2003). If calls between components get too specific, then they will not work when the component is switched out.

The Framework

This portion of the architecture is the skeleton that holds the components together. In *Figure 8*, the framework is represented as the blue arrows. First, it links player input to game mechanics. The results from that component are then passed into the content component, and then to the student model which returns feedback for the player. This feedback is then presented to the player by the framework portion of the architecture. This in turn should cause the player to change their behavior based on that feedback, and try again providing new input and starting the cycle over again. The framework is designed to be static and be consistent across all games developed with this architecture.

The Mechanics Component

This component plugs into the architecture right after the player input is received. In *Figure 8* this is represented by the green box. This component receives player input and translates it into actions within the game world. This could be considered part of any

computer game's structure (Sicart, 2008), but it is of note in this architecture because it will be designed to be removed and substituted with a different set of mechanics. Recall that the content domain and mechanics are traditionally deeply connected (Van Eck, 2006), making removing them a difficult process. It is here that this architecture first starts to break from the traditional model by allowing the mechanics to easily be switched out for a different set. This component is designed to be easily removed and replaced, but is intended to remain intact across multiple games and content domains, because the mechanics should be content agnostic.

The Content Component

This is the final piece of the architecture. In *Figure 6* it is represented by the orange box. It accepts the results of the player's actions that come from the mechanics component, which are passed to it by the framework, and assesses the player's knowledge and skill in the content domain. After measuring this, the student model in the framework is updated and feedback about the player's performance is given to the framework, which will display it to the player. The content component is the most dynamic and easily swapped portion of the architecture, because it is intended to change with each new game. It is therefore critical that it be easy to remove and replace.

The Student Model

Assessing student knowledge will be a part of the framework portion of the architecture. There are two reasons to place assessment here, rather than in the content component as might be expected. The first is that assessment is always required, no matter what domain is being taught and it is ideal to not require developers to build their

own assessment measures for each content domain. Building off of that concept, the second reason is to maintain consistency and ensure compliance with ECD (Mislevy et. al., 2003) and stealth assessment (Shute, 2011). The key concern with integrating this into the overall framework is maintaining the flexibility required to assess any domain.

The approach that will be used to solve this is educational data mining (EDM; Ocumpaugh, Baker, Gowda, Heffernan N. & Heffernan, 2014). EDM allows educators to handle massive amounts of data to locate specific patterns or behaviors that are common among students that are successful. The important step is to tie these behaviors to learning objectives, which can be achieved with a Bayesian network (Pardos & Heffernan, 2010). A Bayesian network is a model that represents a set of variables and their conditional dependence on one another (Friedman, Geiger & Goldszmidt, 1997). This network can be matched with knowledge tracing, which was first explored by Atkinson and Paulson (1972).

Knowledge tracing has four probabilities as parameters which Pardos and Heffernan (2010) fit to a Bayesian model. Two of the parameters are knowledge parameters, prior knowledge and rate of learning. Prior knowledge is the chance that the student already knows the content in question, while rate of learning is the chance that the student will transfer from an unlearned state to a learned state after an attempt at the task. The other two parameters are performance parameters, guess rate and slip rate. Guess rate is the chance that the student will answer correctly when they do not actually know the material, and slip rate is their chance of answering incorrectly when they actually do know the material. Due to the probabilistic nature of these parameters and conditional

interdependence, they can easily be converted into a Bayesian network as shown in *Figure 9*. This same method can be used to match the parameters to learning outcomes. Each task or potential action within the game can be matched to a specific learning objective and then the student's knowledge can be tested as they play the game. Their success or failure as they attempt each task will update the Bayesian network to develop an accurate model of the student's understanding.

5 CAGE GAMES

Overview

A total of thirteen games were developed for the CAGE thesis. Of these, eleven used the CAGE software framework. The other two were concept games developed prior to the creation of the software framework. Each game was developed by a different developer and each teaches two different content domains. The very first game developed was excluded from the study due to being drastically different from the other twelve. All thirteen of these games are discussed in detail below.

Game Genres

Each of the games used in this thesis fell into one of five different genres. The first genre is shooter. Games in this genre revolved around firing projectiles at enemies while dodging the enemies and their own weapon fire. They are fast paced and require quick thinking and reaction times. The second genre is action games. Games in this genre were similar to shooter games in that they require quick reaction times, but did not involve attacking and dodging enemies. The third genre is puzzle games. These games gave the player ample time to consider and solve the problem presented to them. Quick reaction times are not required when playing the games in this genre. The fourth genre is tower defense. In these games, players cannot directly fight enemies and must instead build towers next to the enemy's path which will attack them. The enemies move in real time, so quick reaction times are helpful but so is strategy for picking locations to place towers. The final genre is quiz games. Games in this genre usually posed a direct multiple

choice question. Many games had a time limit to answer the question, but otherwise the player simply had to click on an answer to proceed.

Non-Framework Games

While the development of CAGE games is new (Baron et. al., 2016), the initial concept has been around for a few years. This led to the development of two proof of concept games that follow the CAGE design paradigm, but were built prior to the creation of the CAGE framework. This means that they do not follow the CAGE model as closely and have some distinct differences from the games built with the framework. See Table 1 for an overview of the CAGE games that did not use the framework.

Word Fighter and Math Fighter

The first pair of games are Word Fighter and Math Fighter. These two games use mechanics similar to Tetris[3] to teach spelling and math respectively. Creating the first one took time and effort, but making the second version took very little time. For example, Word Fighters took a month to develop, but Math Fighters core functionality was created from Word Fighters in a single afternoon and only a few more days of polish. The fighter games were not included in the study for the CAGE thesis. This was because they were built in Adobe Flash[4], not the Unity Engine[5] like all the other games, and the source code was not available for modification. This meant that data would not have been able to be collected the same way as all the other CAGE games and would have been very difficult to work with. However, the games remain important as they were the first games built in accordance with the CAGE paradigm. The games and their mechanics are described below.

Word Fighter

Word Fighter (WF) is a Tetris style game that tasks the player with removing blocks before they reach the top of the screen. The player is playing against an opponent, who is either another student or a computer player if no other players are available. Each player can only see their own board, but they do have a meter for how high the opponent's board is. See *Figure 10* for a screenshot. At the start of the game, four rows with seven blocks each fall from the top to the bottom of the board. Each block contains a single letter. Blocks that contain vowels are a different color to help them stand out easily. Using the blocks, the player must select blocks one at a time to spell out any word they can find among the available letters. If they do spell a valid word, the blocks they used disappear and an equal number of blocks falls onto the opponent's board. When the opponent completes a word, then an equal number of blocks are dropped into the player's board. Every twenty seconds, a few additional blocks fall onto both boards. Like Tetris, the goal is not to allow the blocks to reach the top of the board. Since longer words will both drop more blocks on the opponent's board as well as remove more blocks from their own board, the mechanics encourage the player to find longer words, rather than just use short words.

Math Fighter

Math Fighter (MF) is built from WF and again uses Tetris style game mechanics. However, MF uses these mechanics to teach math instead of spelling. See *Figure 11* for a screenshot. Note that the two games look almost identical. Each block now contains a single digit, and operators are now highlighted instead of vowels. There is a new element

in the user interface, which is a number appearing in the top left of the screen. This number is the target, and the player must pick numbers or an equation that equals that target number.

Example. Let 76 be the target number. The player could select a 7 block and then a 6 block to make 76 directly, but this would only remove two blocks. Instead, the player could use a more complicated expression to reach the target. So, the player could select 7 * 10 – 5 + 1 which evaluates to 76 and uses eight blocks instead of only two.

The core mechanics of MF differ only slightly from those of WF. Changes include the addition of the target number and converting from checking against a dictionary to evaluating expressions. Of those two, only the new target number has an impact on gameplay. There are other changes related mostly to swapping the letters for digits, but these are aesthetic changes and it is expected that art assets will need to be swapped between games.

Word Towers and Math Towers

The second pair of games are Word Towers and Math Towers. These two games are both tower defense games that teach their respective content. While they were built without the CAGE framework, they were included in the CAGE study. Their purpose was to serve as a comparison for testing the effectiveness of the CAGE software framework and verify its usefulness. This will help determine the impact of the framework along with the effectiveness of the CAGE model. It is possible that the CAGE model is effective, but that the software framework does not actually expedite the

development process or possibly hinders it. Including the towers games in the study will help to test for this possibility.

Word Towers

Word Towers (WT) is a tower defense game that challenges players to spell words to build defenses against invading pirates. See *Figure 12* for a screenshot with labeled user interface elements. In tower defense games, the player does not directly attack enemies. Instead, the enemies travel along set paths and the player builds towers on the side of this path which attack the enemies. Strategic placement of towers is key to success in tower defense games. For example, placing a tower on a corner allows it to fire on the enemies for longer. Many of these games have different types of towers with different effects, such as slowing the enemy down. WT only has towers that deal damage, but the damage and range of towers depends on the word used to build it. Players are given a random set of letters to pick from. If the player sees a word they can build from those letters, then they select where they want to build their tower. The player then picks the letters to spell their word and submit their word. If it is a valid word, then the tower is built on that spot and it will begin to fire at nearby enemies. The longer the word, the more powerful the tower. Towers can also be upgraded later if the player has new letters they could mix with the old word to make a longer one.

Example. The player has a tower that was built with the word SAND. The player has the letters S and T in their letter board. They could upgrade SAND to STANDS, making it more powerful.

If an enemy reaches the end of the path, the player will lose some gold, depending on the size of the enemy. There are five types of enemies, each has more health and takes more gold than the preceding one. If the player has any gold left at the end of the level, they pass the level and can attempt the next level. Each level has more enemies than the previous one, meaning the difficulty of the game increases with each level.

Math Towers

Math Towers (MT) is also a tower defense game that challenges players to make eexpressions to reach a target number. See *Figure 13* for a screenshot. Instead of letters, the player is now given digits and operators. A random target number is also provided and, like MF, the player must build an expression that evaluates to the target number. Again, longer equations are rewarded with better towers. An existing tower can also be upgraded to match the new target number, making it longer in the process.

Like the fighter games, there are only minimal differences between the two versions of the towers games. Again, only minor aesthetic changes are needed as well as the additional UI element to show the target number. The fighter games existed as two separate games, but the towers games were combined into a larger single project and players could switch between the two topics by clicking a button on the main menu screen. Portions of the code that needed to be different depending on which content was active needed to verify which content was active and branch to the appropriate portion of the code. Even without using the software framework, because the game followed the CAGE design paradigm, there were only a few places this check was required and it did not lead to massive differences in the order of code execution. However, this was made

even smoother by the CAGE framework such that a single statement would work regardless of the active content.

Framework Games

The following eleven games were built by students in a game based learning course at Arizona State University. These students were all in the Masters of Software Engineering program and were provided the CAGE software framework to build their games. While they were in a class about game based learning, only one had used the Unity engine to develop an educational game before. This resulted in a consistent set of core concepts derived from the CAGE framework, but wide variations in quality of the final games. See the CAGE framework study chapter for more information about the developers of the CAGE games. See Table 2 and Table 3 for an overview of the CAGE framework games.

Base Common Elements

Since the CAGE framework was used as a base for all of the framework games, there are some elements that are common to all of them. For example, *Figure 14* shows the default main menu that is included with the CAGE framework. Its use is not required, but it is provided as an easy base to use for a main menu and provides a working example of how to use the framework. All of the CAGE games developed for this thesis used this provided menu setup. Note that in *Figure 14*, the text is incomplete. The title of the game is “Game Title”, the two contents are referred to as “Content1” and “Content2”, and the phrase “No Topic Selected” appears below the content buttons. This is because the game

is not running when that screenshot was taken. These are all placeholder values that are populated by a provided script when the game starts.

Setting up this menu requires only three lines of code to be changed. The string for the name of the game must be changed from “Game Title” to the appropriate title and the two content objects must be changed to be one of each of the developer’s content objects. These start as empty NoContent objects, which serve as placeholders to ensure the framework compiles until the developer is ready to test their own content. Any aesthetic changes to the main menu can be done within the Unity editor itself and does not require any code changes. This means that those three lines of code are the only code level changes required to change the CAGE framework for any new developer. The rest of the framework can be used without any edits to the existing code, though some advanced users may wish to do so.

Bean Man

Bean Man (BM) is an arcade shooter game where the town is being attacked by green slime monsters. The monster names change depending on the active content.

Gameplay. The player is placed in the level shown in *Figure 15*. The player can only move in 2D space and can jump by pressing the space bar. Clicking the mouse fires a rocket and the character aims where the mouse is on the screen. The player gets points for every green enemy they kill. Each enemy takes one shot to kill. At the same time, humans skateboard around the level. Hitting them once causes them to change to an unbalanced stance on their skateboard to warn the player. Shooting them again knocks them out and the player loses points. All enemies and humans drop in on the top side of

the level and travel towards a hole in the bottom center of the level. When they fall in the hole, they are removed from the game. If the player falls in the hole, the game ends. Touching enemies deals damage to the player and the game ends if the player takes too much damage. Each new game, the first time the player touches an enemy with a new name that they have not touched before, a picture of the topic briefly appears above their head as shown in *Figure 16*.

The first content domain is health and disease. The enemies all have names of common diseases. The second content domain is social issues. Here the enemies have names of controversial issues in modern society such as racism. After the level ends, the player is taken to a game over screen where some information about each disease or issue is displayed. The student model recorded the player's score at the end of each level.

Car Racing

Car Racing (CR) is a quiz game that gives multiple choice questions to the player. Correct answers make the player's car go faster, wrong answers make the other cars in the race go faster, and taking too long to answer the question slows the player car.

Gameplay. At the start of the game, there are three cars all of which are going the same speed. The car in the middle is the player's car, while the other two are computer controlled enemy cars. Some other computer controlled cars occasionally pass by, but do not factor into the race at all. *Figure 17* shows what the game looks like at the start of a race, when the game has just been started. The player's rank, or position, in the race is shown on the top left. The rest of the information for the player is shown on the right side of the screen. The current question is shown there and three choices for possible answers

are displayed below it. The answer to the previous question is shown to the right as well as whether the previous answer was correct or not, except at the beginning of the game when it is blank as no questions have been answered yet. The time the player has remaining is shown below the answer choices.

The player controls the car entirely through the multiple choice questions and answers. All the cars always drive straight and turning is not possible and is not required. Selecting the correct answer increases the speed of the player car and a new question is shown. An incorrect answer slows down the player car and moves to the next question. The player has ten seconds to answer each question. If they take longer than the allotted time, then the enemy cars speed up, the question is left unanswered, and the next question is shown instead. Every five seconds, the enemy cars will speed up regardless of how the player is performing. The amount that the cars speed up is random within a range and the two cars speed up at their own independent rates. Both content domains have twenty questions. The race ends when all twenty questions have passed. There can be any combination of correct answers, incorrect answers, and timeouts in order to finish the race. However, the more questions that were answered correctly the more likely the player is to win the race. Due to the varying rate at which the enemy cars speed up, there is not a fixed number of questions that the player must get correctly in order to win the race. Faster players will need to get fewer questions correct, as the enemy cars will not get as many speed boosts. The game was designed to require a sixty percent correct rate in order to place first, but in practice it can vary somewhat.

The first content domain is vocabulary. Each question requires the player to identify a synonym of the provided word.

Example. The word is “controversial” and the options are “incomplete”, “debatable”, and “reduce”.

The second content is United States History. Each question asks the player to provide the name of a person or a place significant to U.S. history.

Example. The question “The first successful permanent English settlement in North America was located at” with the options “Jamestown, Virginia”, “Roanoke Island, North Carolina”, and “New London, Connecticut”.

The student model tracked the answer the player selected for each question, whether that answer was correct or not, and their position at the end of the race. See Appendix II for a full list of all questions in CR.

Fun-O-Sphere

Fun-O-Sphere (FOS) is a puzzle game where the player has to eliminate bouncing spheres until the remaining spheres fulfill a given condition. This can be the chance of picking a certain color sphere at random from the box, or the chemical makeup of a given molecule.

Gameplay. The player clicks on the moving spheres until they believe the remaining spheres satisfy the given condition, at which point they click on a button to have their answer evaluated.

The first content in FOS is probability. At the beginning of the level, there are a certain number of colored spheres bouncing around in a confined area. The player is asked to remove spheres so that there is a given probability of picking a specified color sphere, if one were chosen at random.

Example. The player is asked to have a four out of five chance of picking a black sphere, meaning there must be four lack spheres and one non-black sphere remaining.

Figure 18 shows what this question looks like during the game. Once the player believes that they have met the required probability, they click a button to have it evaluated. If they are correct, the question is answered and the player moves to the next question. If they are wrong, they are told that they are incorrect, but the game still advances to the next question. It is also possible for the player to remove too many of a given sphere, making it impossible to solve the puzzle. If this happens, the player is told that they removed too many and the game advances to the next question.

The second content is chemistry as shown in *Figure 19*. In the probability content, the color of each sphere was what mattered when trying to solve each question. Each sphere was still labeled with the name of its color in text on the sphere in order to help any players who may not be able to see colors correctly. In the chemistry content, the color is irrelevant and instead the labels show what element each sphere represents. The goal in this content is to make the remaining spheres match a certain molecule.

Example: The goal is to make a molecule of water, so the player must have two hydrogens and one oxygen remaining.

The rules for a correct answer, incorrect answer, and eliminating too many spheres remains the same as the probability content. Both contents have five questions each. See Appendix II for a full list of questions. The student model recorded when the student had a correct answer, when they had an incorrect answer and what they had submitted as that answer, and when they eliminated too many of a sphere.

Hat Trick

Hat Trick (HT) is a 2D action game where the player attempts to catch answers to questions in a hat.

Gameplay. In HT, the player can move a hat from left to right. The hat cannot be moved up or down. A question is displayed near the top of the screen as shown in *Figure 20*. Every few seconds, a bowling ball labeled with a potential answer to the question drops from the top of the screen. When the player sees a falling answer they believe is correct, they must try to move the hat under the bowling ball to catch that answer. Players are told whether their answer was correct or not and their score is updated appropriately. Wrong answers take away twenty points and correct answers add fifty points. The score is effectively a time limit as it counts down one per second and the game ends when the player reaches 0 or fewer points. Catching a correct answer changes the current question to a new question. See Appendix II for a list of all possible questions.

The first content in HT is math. Players are presented with a linear equation and must catch the appropriate value. The second content is history and players must catch the year in which the given event occurred. The student model tracks every answer, correct or not, that the player catches for each question.

Modern Scrabble

Modern Scrabble (MS) is a puzzle game where the player forms combinations of elements to clear the board.

Gameplay. In MS the board is given a board of elements. These are either letters or words depending on the active content. The first content is English and provides the player with a board of letters as shown in *Figure 21*. The player can then swap adjacent letters until they form a word. Once a word has been made, the letters that form it are removed from the board. This continues until the board is clear. The second content is biology. In that content the board is filled with names of various organisms, as well as the sun as shown in *Figure 22*. The player must rearrange the words to form food chains of four terms or more.

Example: Having Sun, Plant, Rabbit, and Cheetah in order would form a food chain.

The student model tracks what words or food chains the player forms.

Operation VIP Extraction

Operation VIP Extraction (VIP) is a 2D shooter where the player needs to shoot enemies and allow allies to pass unharmed.

Gameplay. The player can move their ship up and down as well as left and right. Other ships fly from the top of the screen down towards the bottom as shown in *Figure 23*. The ally should be allowed to pass by, but all other ships should be shot. Ally ships vary by content. The first content is social science and the allies are cabinet members of

the Obama admiration. The enemies are fictional characters from movies and comic books. The second content is biology and the enemies are viruses and bacteria as shown in *Figure 24*. The allies are human blood cell types. The student model tracks whether the player was successful in letting the ally escape or not. The model does not track enemies shot down due to the large number of them.

Pattern Recognition Training

Pattern Recognition Training (PRT) is a quiz game where the player needs to pick the correct object from a series of falling objects.

Gameplay. In PRT objects fall from the top of the screen as shown in *Figure 25*. The player is asked to click on objects of a certain type from among those falling. Picking a correct one gives the player a point and changes the target object to another random one. Clicking an incorrect one removes one point from the player, to a minimum of zero. The first content is geometry and the falling objects are shapes. The second one is chemistry and the objects are instead molecules. The objects are molecular structures and the player is asked to pick the correct molecule by name. The student model tracks which object the player clicked and which they were asked to click.

Pedestals

Pedestals (PD) is a quiz game where the player is given a question and must move a ball to the answer they want to select.

Gameplay. In PD a question is show on the screen along with four potential answers as shown in *Figure 26*. The player must move the ball to the answer they think is

correct. If they are correct, the question will change and they pick again. If they are wrong, a message appears on screen to tell them and they get to pick a new answer. The first content is linear equations where the player must solve for x. The second content is English and the player must pick the correct word to fill in the blank in common situations that cause grammatical mistakes. See Appendix II for the full list of questions. The student model tracks which answer the player picked in response to each question.

Quiz Up!

Quiz Up! (QU) is a quiz game where players answer a series of true/false questions about the active content.

Gameplay. Many questions simply display a question, but some show a picture or play an audio clip. The player then clicks on the true button or the false button to make their choice. They are told if they are correct or not, and then the game moves to the next question. The first content is music and the second content is French. The student model tracked the answer to each question and whether it was correct or not.

Think Fast

Think Fast (TF) is a quiz game in which the player answers a given question using one of the provided answers.

Gameplay. In TF a question is displayed along the top of the screen. Three possible answers are shown along the bottom. The space in between has five elements that move back and forth, left to right and back again. These elements are numbered with the top being element one and the bottom being element five. The questions will ask the

player to either provide the correct order of elements or use them to fill in blanks in the problem, depending on the content.

The first content is math. In this content, the elements represent values in a provided equation as shown in *Figure 29*. The elements are chosen at random when the game starts, and as a result the proper answer is also calculated at this time. The equation remains the same every time however "Calculate ((num1 + num2 – num3) * num4 / num5)". The second content is Spanish as shown in *Figure 30*. For this content, the player is asked to find the grammatically correct sentence. Sentences of five words, or four with a preceding question mark or exclamation point, made up the five elements. A list of the sentences can be found in Appendix II. The two incorrect answers were scrambled versions of the proper sentence. The student model tracked the answer the student submitted and whether it was correct or not.

Titanical

Titanical (TT) is a quiz game where the player shoots torpedoes at icebergs to select their answer. The player drives a boat around a 3D environment as shown in *Figure 31*. Torpedoes always fired straight forward, so the player would need to aim their ship at the correct answer. TT was not included in the CAGE study because the final version was lost, and the earlier available build only had one content and was missing multiple mechanics.

Affective State Game

One additional CAGE game was built after the initial set of games. This game integrated affective state detection into the game to adjust the difficulty in real time to better fit the player's skill level. See Chapter 8 for details on affective state and how it affects engagement.

Bingo Bingo

Bingo Bingo (BB) is a puzzle game where the player switches adjacent terms to make three or more identical items in a row or column.

Gameplay. The game begins with the board filled with a seven wide and five high grid of circles. See *Figure 32* for a screenshot example of this starting state. The player can then click on any of these circles and drag them to one of the circles next to it. This is a circle to the left, right, above, or below the first one. Each circle has a term on it, which varies by the active content. The goal is to switch the positions of two circles such that the new placement of one or both of the circles makes three like terms in a row. If the match is valid, the matching circles disappear and new circles fall in from above to fill in the missing spaces. If the move is not valid, nothing happens. On the higher difficulties, some circles appear as a magenta color instead of the regular yellow. These spaces must be matched twice before they will disappear. They also do not fall down into lower places if a space below them becomes empty.

The first content is trigonometry. Each circle has terms on it like cos60 (the cosine of sixty degrees) or sin45 (the sine of forty five degrees). The player can match the exact same values, such as getting three cos60 circles in a row, but they can

also match different circles that have equivalent values. The second content is vocabulary and the goal is for the player to match synonyms.

6 THE CAGE FRAMEWORK DEVELOPMENT STUDY

Overview

Before the CAGE games themselves could be made, the CAGE framework itself had to be developed. This framework was built in the Unity Engine by Unity Technologies using the CAGE model as detailed in that Chapter 4. Once completed, the framework was distributed to a group of software engineering students to develop the framework game detailed in the previous chapter. After development was completed, the students filled out surveys about how the use of the framework impacted their development of CAGE games. This process and the results are detailed in this chapter. This study was approved by the Institutional Review Board (IRB) or Arizona State Unveristy (ASU) as Content Agnostic Mechanics – STUDY00005148.

Software Architecture

The CAGE framework is a set of software methodologies built on top of the Unity 5 game engine developed by Unity Technologies. Unity itself is already a complete game development package for building 2D and 3D games and deploying them to several platforms. This means Unity already handles many of the low level programming tasks such as hardware input and a complete rendering pipeline. This enables CAGE to focus entirely on game building and education, instead of having to worry about these low level processes. The CAGE framework further streamlines the development process by providing a series of classes and pre-built Unity GameObjects that enable rapid

development of CAGE games. Unity uses the C# programming language and so CAGE is built in C# as well.

In order to accommodate as many content domains and game mechanics as possible, a key concept behind the CAGE framework is generic messages. These messages are called Hooks. A game mechanic will send out Hooks as various in-game event occur, such as the player attempting a question or making a mistake. The Hooks are sent to the content component and will be caught if that particular content can make use of that Hook. Otherwise, it is ignored. This enables the developer to send any type of Hook that the mechanics can at any time without consideration for which content is active. Whether the Hook is accepted or not, it is also passed to the student model component for evaluation there as well. Hook is a generic abstract base class so that all Hooks can be accepted in the content component through a single method call via polymorphism. The provided base content class contains a public method for accepting all Hooks which determines which type of Hook it has received and passes it to a stubbed virtual void method. This is how Hooks are able to be ignored, because developers will inherit the content class in their own content classes and override only the stubs they actually use in the current content class. If a Hook of an unknown type is received, it is passed to a protected method. The unknown Hook type would be one defined by the developer for their current game. The developer can override the protected method to provide sorting criteria for the new Hook type and send it to a private method to process it.

While the Hook system is how the mechanics passes information to the content, and the CAGE model is primarily designed to flow in one direction, it is sometimes necessary to pass information from the content back to the mechanics. This may be needed sometimes in order to display information correctly, like displaying a math problem over an enemy's head. It is most often needed when the player attempts to solve a problem, because the mechanics component is often in charge of how the game will react to success or failure. In order to handle information flowing back to the mechanics, the base content class was defined to contain several methods. These include a get for several primitive data types as well as a method to get an Object, which allows the developer to return any wrapper object when more complex data need to be returned. There is also a boolean method to determine whether the last action was valid or not. This is the primary way the mechanics can tell if the player succeeded in a task or not. The mechanics sends a Hook of the player's action to the content which then evaluates the action and the mechanics then checks if the last action was valid.

Study Design

Participants. Participants were recruited from a graduate level course in game based learning. There were eleven (N = 11) students in the class. They were given an assignment to complete their game for this study. Participation in the study was a required part of the class, which was specified by the instructor during the first few days of the class, giving students a chance to drop the course if they did not want to participate. Participant’s age ranged from twenty two to twenty nine (Mean = 24.36). All were students in the Masters of Software Engineering program at Arizona State

University. Of the participants, eight (72.7%) were in their second year of the program, with the remaining students being in the first year. Three of the participants were female.

Three of the participants received their undergraduate degree in software engineering, while the rest all earned their degree in computer science. Nine participants had professional software development experience. Of those, four had one to two years of experience, two had less than one year, and two more had two to three years. The remaining participant had four or more years of professional experience. Out of all the participants, only one had ever used the Unity engine to develop an educational game prior to taking the course.

Method. Participants were given the CAGE (called CAM at the time) framework as a prepared Unity package. This was provided as part of an assignment that gave the participants two weeks to build a content agnostic educational game from scratch. This first part of the assignment asked the students to only build a single content. This was to give them time to become familiar with the framework and account for the time needed to build a new game from scratch. This effectively allowed for a week for the mechanics component and a week for the content component. The students checked in with their progress after one week, and feedback about the games was given by the instructor and the study author. At the end of the second week, students submitted the final version of their games with only one content. Another round of feedback and advice was given and then the students were given another assignment, which was to expand their CAGE game to include a second content component. They were only given a single week to complete

this final version. The version handed in after this week is the version used in the CAGE study and details about them are given in the previous chapter.

Having finished with development of their CAGE game, participants then filled out a questionnaire about the development process. This was divided into two main parts. The first part was questions specifically pertaining to the CAGE framework and development process. These included questions about how many lines of code were written for each part of the assignment and self-reporting on the effect the framework had on development time. The second set of questions was about general software reusability and was taken from software reusability work done by Agresti (2011). These questions revolved around how often code was reused and reasons why it was or was not reused. A full list of questions from both parts can be found in Appendix III.

Code Reusability Results

The first three questions were all statements that were rated on a five point Likert scale, with 1 being Strongly Disagree and 5 being Strongly Agree. On the matter of thoroughly rewriting old code, participants were fairly split with a slight tendency towards keeping old code as is (Mean = 2.27). The next statement dealt with reusing old code for reliability, even if it does not quite fit the new system design. Participants were again slightly against, but remained quite split (Mean = 2.81). Participants indicated they were slightly more likely to try to reuse old code when under time pressure (Mean = 3.45).

The next two questions had participants indicate how often these situations occurred in their software development experience, with percentage frequencies given for each rating. 45.5% of participants indicated that they only occasionally wrote entirely new code when they were asked to reuse existing code. At the same time, 27.3% indicated it happened more often than not. 18.2% indicated often, which is between the two previous ratings, and only one participant indicated it rarely happened. See *Figure 33* for an overview of these results. The second question was how often they reused old code instead of writing new code, despite being asked to write new code. The majority (54.4%) responded that they often reused old code, and over half of the rest (27.3%) claimed it only happened occasionally. Of the two remaining respondents one indicated that it happened rarely, but the other indicated that it happened most of the time. See *Figure 34* for an overview of these results.

The last six questions were each a potential reason for not reusing existing code, and participants indicated how often each reason was the cause for reusing code. Most participants (63.6%) indicated that the code not being on the computer was the reason for not reusing code. Only one participant claimed it happened often, with the rest claiming it had happened only a few times. The old code being in an incorrect language was more likely to be the cause with only 18.2% reporting that it was never a problem. It had been a problem occasionally for 45.5% of respondents and a frequent issue for the remaining 36.4% of participants. The old code not meeting the requirements was also a critical issue with only one participant claiming it had never been an issue. It was a frequently reoccurring issue for most (54.5%) respondents.

The issue of defects being present in the old code was evenly split across the possible responses. Often and occasionally both were claimed by 36.4% of respondents with the remaining 27.3% claiming it had never been an issue. Almost half (45.5%) claimed that not being able to understand the existing code had occasionally been an issue that could them to write new code. Most (36.4%) of the remaining participants claimed it had never been an issue and only two indicated that it was a frequent problem. For the final issue of the old code being too complex, almost half (45.5%) claimed it was an occasional issue. The remaining two possible responses, that it never happened or that it happened frequently, were evenly split with 27.3% of responses each.

Code Reusability Discussion

The results indicated that participants had a preference for reusing old code when possible. This is seen in their desire to leave old code unmodified and well as often reusing old code despite being asked to write new code. This indicates that reusability across multiple projects is desired, and yet most of them still rewrite old code. All but one respondent indicated that they rewrote old code at least ten percent of the time, and half of those indicated that it occurred more than twenty five percent of the time. In the interest of time invested by developers, old code should not have to be rewritten so frequently, and should only be needed in the event of a major change in implementation. This is particularly true in situations where the old code was specifically supposed to be part of the new project. It is a serious problem for projects that are then more likely to take longer than expected and run over budget (Agresti, 2011).

Investigating why old code was rewritten so often revealed that the most commonly indicated cause was that the old code did not meet the requirements of the new system. This is a particularly troubling cause given that this code was supposed to be reusable for this new project. If a module is determined to be reusable for a new project, it should only require minimal edits to make it fit within the new project. There are many possible reasons why an inappropriate module would be selected for use in a system it does not meet the requirements of, but there are three likely causes. The first of these is that the module was selected for use by someone who was not actually familiar with the module and possibly not familiar with the requirements of the new project either. The second reason is that the requirements of the new project changed since the decision was made to reuse the old module. The third reason, and one of greatest interest to the CAGE thesis, is that the module was written in a way that was too specific to the original requirements. This meant that when the developers try to reuse the module in a new context, it does not fit well with the new system. This is similar to the main problem CAGE is attempting to solve, that of mechanics being too tightly integrated with the learning content and thus not being usable when trying to teach different topics. A module that is not deeply connected to the specific requirements of the project for which it is written will be easier to reuse in a new project later on.

The second most common reason for not reusing old code, even when asked to, is that the old code was not in the proper programming language. This most likely ties back into the second reason for the module not meeting the requirements, which is that the requirements changed since the module was selected. Changing which language a project

is written is possible at the early stages or after a major issue with the current language is found. However, there are many methods by which it is possible to use a module from another language in a new project in a different language. The issue with this is that it is often difficult or impossible to then make changes to the module in the other language. This means that even minor changes to requirements can render the module inadequate for the new project.

The code being too complex was a fairly evenly distributed cause among the participants. This may be because there is a fairly even distribution of experience across the participants. Some of the less experienced ones may have considered the code too complex because they could not understand it. All of the respondents who indicated that they often rewrote old code because they could not understand it had industry experience, but less than two years of such experience. The two participants who did not have any industry experience indicated that it was rare that they did not understand the code. Given that they had not worked in industry, they likely have not had a great amount of exposure to more the complex code that is often found in industry compared to the classroom environment.

The frequency of rewriting because of defects in the old code was evenly distributed across the participants. There is no apparent relationship between participant experience and how often they reported this as a frequent cause. The final reason for rewriting an old module for was that the old code was not on the computer they were using to write the new code. This was the least common reason by a notable margin. This is likely an artifact of Agresti's (2011) work being several years old by the time this study

was conducted. While they existed at the time, means of moving and maintaining large amounts of data and source code have drastically improved since then. With the current power and availability of cloud computing and source code version control tools, it is very unlikely that there is no way to transfer code to a new computer.

The results on participant's perceptions regarding reusing old code were not as conclusive as the other results. In general, participants indicated that they did not feel it was best to always completely rewrite old code, even though they frequently did. This is not necessarily a contradiction as developers may not want to rewrite old code, but still do so out of necessity. Participants also indicated that they were more likely to try to reuse old code if they were under time pressure, which is an understandable result. If the code already contains an implementation of the current problem a developer is tackling and they are short on time, it is reasonable that they would try and use the existing solution. A possible reason that this was not given a higher rating is the risk of failure. If a developer tries to reuse old code, it would take some time to set up but less than building all new code from scratch. However, if the old code does not actually work more time gets spent trying to properly integrate it with the new project. This still may not work, and now the developer is even further behind as they have now spent this time on the old code so they are no closer to finishing and have even less time remaining. Many of the participants may not be willing to take the risk of this happening, and will continue to write new code instead.

CAGE Framework Results

The number of lines of code participants reported writing for the first portion of the assignment varied widely from as low as 65 to as high as 750 (Mean = 366.7). Note that one participant reported sixty to seventy lines, which was averaged to 65 as a single number. One participant did not report numbers for this group of questions, and so is excluded from this portion of the results. The number of lines for the second portion of the assignment also had a great deal of variation, but were lower overall going from as little as 5 to 300 (Mean = 107.7).

The next group of questions asked participants about the number of hours spent on the two portions of the assignment. Two participants provided answers that could not be used, as so are excluded from these results. The number of hours spent on the first part ranged from 4 to 72 (Mean = 19.7). For the second portion of the assignment, participants reported generally lower numbers from only 1 hour to 48 (Mean = 8.8).

When reporting how useful the CAGE framework was in speeding up their development process, no participants gave below a 3 rating, and the average score was a favorable 4. Results were similar for reusing the framework for future projects, which also had no negative responses and had an average of 4.

The participants were asked to include any other comments they had about the framework, or suggestions for improvements that could be made. Two participants did not provide any comments, but all of the ones that did asked that the framework add support for additional data types. This was most often asked for in terms of getting data

from the content component in order to display content appropriate information to the player. A few participants mentioned the same concept, but were asking for more generic hook types for passing data from the mechanics to the content.

CAGE Framework Discussion

The results indicate that the CAGE framework was working as intended because both the number of lines of code written and the hours spent working were drastically reduced from the first part of the assignment to the second part. The average lines of code written decreased by over 250 lines between the first and second portions of the assignment. That would indicate that the mechanics component was those 250 lines of code, since that component should not have been changed during the second part of the assignment. Likewise, the mechanics component took about 8 hours of work to create as the number of hours spent on the second part of the assignment was about 8 hours less.

These findings have significant implications for developers of educational games. Developing the second content component took a third of the code and less than half the time on average than building the mechanics and the first content component. Part of this can be attributed to the participants being unfamiliar with the framework for the first part of development, but the difference in time and lines needed is still significant and may well be improved with developer familiarity.

Participants also viewed the CAGE framework favorably, indicating that they felt it helped speed up their development process and that they would reuse it in future

endeavors. This means that CAGE not only accomplished the goal of decreasing development time needed, but also did so in a way that was ideal for developers.

Limitations

There are several limitations that limit the generalizability of findings from this study. The first problem is the small number of participants. There were only eleven students enrolled in the course that this study took place in. It would be better to redo this study with a significantly higher number of participants. This would allow for a greater range of differences in industry and programming experience, which in turn would allow for more meaningful connections between experience, perceptions, and practices. The current sample was also rather homogenous in many aspects. Most were male, all were students enrolled in a Master's program, and most were not U.S. citizens. These are all dimensions that may affect perceptions and more diversity is needed in any future work.

Another issue is the classroom environment that this study was conducted in. The classroom environment and the industry environment differ in some significant ways. Some key differences include the complexity of code the developer will have to work with, strictness of requirements, and amount of control the developer has over their code. In industry, newer developers will often work with code written by older developers with considerably more experience than them. This leads to potential issues with complexity and readability. In the classroom, any code provided to students comes from the instructor who has written it at an appropriate level with specific learning goals in mind. Any missing parts or defects are specifically placed to train students and test their knowledge.

Another issue, which is similar to the classroom environment issue, is that all of the participants worked entirely alone on their CAGE games. They may have consulted each other, but each game was the student's own individual work. In most development environments, development is done as part of a team. Given that any CAGE games developed in industry would likely be built by a team, this study should be redone with teams working collaboratively on games. This difference could have an impact on the framework's perceived usefulness. In addition, all of the CAGE games used in this study were relatively simple as a result of limited time and being built by developers working alone. Team built games that are developed with a longer time line will likely be more complex and this is likely to have an effect on the time spent and lines of code written when creating additional content components.

Finally, the framework was provided to the participants shortly after it was developed. Their feedback has helped improve and reshape the framework into the next version. Many of the usability and usefulness results should be considered pilot results and the framework would likely benefit from another round or two of testing and improvements. Additionally, no formal documentation was provided as it was not available at the time. This meant that some participants were confused on how to use the framework and either misused it or spent more time than should have been needed on the first content. Providing full documentation should resolve this issue.

Conclusion

The first research question of the CAGE thesis was to determine if CAGE could be used to allow for easy and rapid creation of educational games. The results showed

that CAGE decreased the time needed for creating secondary content domains by more than half and reduced the amount of code needed by two thirds. Participants indicated issues they had with the CAGE framework, but still noted that they would be likely to reuse the framework for future educational games they may develop. This means that research question one is supported by the results.

These results have notable implications for developers of educational games. The CAGE model stands to significantly reduce the amount of time and resources required to develop secondary projects. By reducing the portion of the game that needs to be rewritten to only a single interchangeable module, the majority of the work is handled up front with the development of the first version of the game. This initial work load is itself reduced by the framework, which provides the skeleton for designing a CAGE game. This means that only the mechanics component and the first content component need to be built for the first version. Subsequent versions can simply reuse all but the content component, which would then be the only part that needs to be rewritten. This allows for either multiple different versions of a game, or a single game that contains multiple different content domains.

There were several limitations to the generalizability of these results that leave room for further work. This study should be redone with a higher number of participants to ensure validity. There is also room to experiment with the differences between individual and team development efforts, as well as examining CAGE when utilized for more complex projects. As with many software tools, CAGE needs consistent updates and refinement, particularly at this early stage. It would therefore be best to do several

repeated studies with it, incorporating feedback into the next iteration each time in order to truly refine CAGE for mass development efforts. However, these initial results show that the CAGE framework has merit and can greatly reduce the amount of time needed for development of subsequent projects.

7 THE CAGE STUDY

Overview

This is the primary study of this thesis. The goal of this study was to determine the effectiveness of the CAGE game model and answer research questions two, three, and four. A total of eleven CAGE games were created for this study. They are detailed in the CAGE games chapter, and the means by which they were created were detailed in the CAGE framework study chapter. Of these eleven games, ten used the CAGE framework while the last game was created prior to the creation of the framework, though it still follows the CAGE model.

Study Design

This study was designed as a five by two by two repeated measures design. The first of the three factors is game genre. See Table 5 for a list of the CAGE games by genre. The second factor was the order in which the two contents were played. The third factor was the cognitive load measure for the cognitive load results and engagement for the engagement results. The genre and order were independent variables and the cognitive load and engagement were dependent variables. Demographics information about the participants is grouped with the results for each game. This study was run in conjunction with a study approved by the IRB of the National Science Foundation (NSF) as Project App Maker Pro (AMP): Motivating STEM Study and Teacher Updating through App Development. NSF Award number 1509105.

Procedure

The ten framework games were exported as WebGL builds and were posted on a website. The main web page for this site would choose one of the CAGE games at random for each individual visitor to the website and provide them with a link to that particular game. The link to this main home page was then sent out in a mass email to the parents of hundreds of middle and high school students who had previously participated in programming camps at ASU. The link was also sent to several high school teachers who were participating in an NSF project at ASU, who had agreed to distribute the link to their students and possibly play the games as part of their class. The one non-framework game was too large to run in WebGL and so an exported executable of the game was brought in during the NSF project sessions for the student participants to play in person.

Participants filled out a background information survey for demographic information, but were allowed to leave any of these blank if they did not wish to provide an answer. They were then given instructions on how to play their game. Once they finished reading the instructions, the participant could press a play button to begin their game. Each game had two content topics, A and B. Each game had two versions, one that played A first and then B, while the other version played B first and then A. Participants were invited to play for as long as they liked and once they were finished, they progressed to the cognitive load measurement where they reported their mental effort on a scale of one to ten (Paas et. al., 2003). Next, participants filled out a User Engagement Scale (UES; Wiebe, Lamb, Hardy & Sharek, 2014). The UES consists of twenty six statements and participants report on a scale of one to four how much they agree with

each statement. Once all of UES was filled out, participants played the second content for their version of their game and then filled out the cognitive load measurement and UES again for the second content. Finally, the participants had a place to leave any additional thoughts or comments, after which their game ended. All of the questions were integrated into the game itself and answers were uploaded automatically to an online spreadsheet.

Results

See Table 6 for an overview of all the results for all games and groups.

Non-Framework Game Towers

Participant Demographics

Nine participants (N = 9) played the spelling content first, and six (N = 6) played the math content first for a total of fifteen (N = 15) players of the Towers games. All participants were males in high school and averaged being in tenth grade (Mean = 10.07). Eleven of the participants considered themselves to be gamers, and all participants played about fourteen hours of video games a week on average (Mean = 14.10). Participants who identified themselves as gamers played about four and a half more hours per week on average (Mean = 15.40) than the other participants (Mean = 11).

Results

The group that played the spelling content first had a slightly below average score for the cognitive load of the content (Mean = 4.44), where an average score would be five out of a possible ten. This is slightly higher than the group that played the math content first (Mean = 4.00). The group that played the math content first gave it a just below

average score (Mean = 4.83) while the group that played it second rated it a little above average cognitive load (Mean = 5.77). Overall, the average cognitive load for both groups for the first content was a little below average (Mean = 4.59) while the second content was marginally above average (Mean = 5.11).

For the UES, each item was rated from one to four where four is the best response and two and a half would be an average rating. The group that played the spelling content first gave it a well below average score for engagement (Mean = 1.52), while the other group gave it a better but still below average score (Mean = 2.09). Conversely, the math content was given higher ratings by both groups with the group that played it first giving it a higher score than the spelling (Mean = 2.13) and the group that played it second giving it a higher score than the spelling, but still lower than either of the math first group's scores (Mean = 1.91). Overall, both groups preferred their second content (Mean = 2.00) to their first content (Mean = 1.82). The total engagement for Towers was more than a half point below average (Mean = 1.91).

Bean Man

Participant Demographics

Twelve participants (N = 12) played the social issues content first, while ten (N = 10) played the health and disease content first, for a total of twenty two (N = 22) players of the BM game. Of them, only one was in middle school while the rest reported being in high school, except for four participants (two from each group) who did not report their school level. Most participants did not report their grade level, but the three in the group that played social issues first were in tenth and eleventh grades (Mean = 10.33). The

other group that played the health content first had five participants respond, including the one middle schooler (Mean = 9.60) making tenth grade the average grade overall (Mean = 9.96). Of the participants in the group that played social issues first that reported their gender, six were male and four were female. The other group had five males and three females. The social issues group had four gamers and six non-gamers among those who reported their status and averaged about nineteen hours of game playing per week (Mean = 19.10). The other group had six reported gamers and one non-gamer among those that reported, but reported overall about half as many hours per week of game playing (Mean = 10.80), for an overall average of about fifteen hours per week overall (Mean = 14.97). However, it should be noted that both of these averages for the groups are skewed by excessively high outliers. The social issue group had one participant report one hundred hours of game play per week and another claimed sixty three, while the other group had one claim fifty four hours. These values are significantly higher than any values reported by other participants. Excluding these values gives adjusted averages of three and a half (Mean = 3.50) for the social issues group, slightly higher for the health group (Mean = 3.66) and an overall of just over three and a half hours per week (Mean = 3.58).

Results

The group that played the social issues content first rated slightly above average on the cognitive load scale (Mean = 5.16). The other group which played it second rated it as significantly less cognitively demanding (Mean = 2.20). Likewise, the health group rated the health content only slightly higher than the social issues content (Mean = 2.50)

while the social issues group rated the health content higher, but still below average and below the social issues rating (Mean = 4.25). Both groups found their first content to be more cognitively demanding (Mean = 3.83) than the second (Mean = 3.22).

All of the engagement scores for both contents from both groups were well below an average rating. The social issues group gave that content a rating of a little more than half a point below average (Mean = 1.90) and the health content an additional half point below that (Mean = 1.41). The other group that played the health content first gave even lower ratings for engagement, giving the health content more than a full point below average score (Mean = 1.39) and the social issues content an even lower rating than that (Mean = 1.27). Overall, both groups preferred their first content (Mean = 1.65) to the second content (Mean = 1.34), but rated the game as about a full below average for engagement overall (Mean = 1.49).

Car Racing

Participant Demographics

The two groups for CR both had six participants (N = 6) for a total of twelve people (N = 12) playing the CR game. There were originally seven participants in the history group, but one participants was excluded for reporting that she was 99 years old and not a student. The group that played the vocabulary content first averaged between thirteen and fourteen (Mean = 13.66) years old, while the group that played the history content first was over a year older on average (Mean = 14.83). Half of the participants in the vocabulary group were in middle school, while only two of the six were in middle school in the history group. The vocabulary group had three females, two males, and one

who did not report their gender. The history group had only one female while the rest were male. The vocabulary group had only one participant report themselves as a gamer, while four reported as a non-gamer and the last did not report at all. The history group had four gamers and two non-gamers. As expected, the history group reported a much higher average play time per week (Mean = 15.16) than the vocabulary group (Mean = 4.10).

Results

The vocabulary content was given a below average cognitive load rating by the group that played it first (Mean = 2.66) and the other group rated it even lower than that (Mean = 2.33). However, both groups rated the history content notably higher, with the group that played it first giving it an above average rating (Mean = 6.33) and the vocabulary group rating it below average, but still higher than their rating for vocabulary (Mean = 4.66). Overall, both groups found their first content more cognitively demanding (Mean = 4.50) than their second (Mean = 3.50) but both found the history significantly more demanding (Mean = 5.50) than the vocabulary (Mean = 2.50).

The vocabulary group showed a lower engagement with their first content (Mean = 1.58) than their second content (Mean = 1.63). The history group showed the opposite trend giving their first content a higher overall engagement (Mean = 1.44) than their second (Mean = 1.32). This means both groups found the history content more engaging (Mean = 1.53) than the vocabulary content (Mean = 1.45). Overall engagement decreased from the first content (Mean = 1.51) to the second (Mean = 1.47) and gave CR a full point below average overall engagement (Mean = 1.49).

Fun-O-Sphere

Participant Demographics

The group that played the probability content first had seven (N = 6) participants, and the group that played the chemistry content first had five members (N = 5) for a total of twelve participants (N = 11) for the FOS game. Of the three participants in the probability group that reported their age, they averaged a little more than fifteen and half a years old (Mean = 15.66), and the four from the chemistry group that reported their age averaged a little less than fifteen years old (Mean = 14.75). Only three participants from the probability group reported their student status and all three reported being in high school, though no participants recorded their grade level. The chemistry group reported two high schoolers, two middle schoolers, and one did not report at all. Only the two middle schoolers reported their grade level, with both being in eighth grade. The probability group had one gamer, two non-gamers, and three did not report. The gamer reported thirty hours of game time per week. Only one of the non-gamers reported but reported a play time of zero hours. The chemistry group had only one participant not report their gamer status, or their hours played. The other four all reported being gamers who played between five and thirty hours per week (Mean = 13.75) with about fourteen hours per week between the two groups (Mean = 14.37).

Results

The probability group rated both contents equally for cognitive load (Mean = 2.66). The chemistry group gave chemistry a significantly higher cognitive load rating (Mean = 4.0) over the probability content (Mean = 1.40). Due to this, there was a

decrease in reported cognitive load of more than a point from the first content (Mean = 3.33) to the second content (Mean = 2.03).

The probability group had only a marginal increase in engagement from the first content (Mean = 1.34) to the second (Mean = 1.35). The probability group showed a slightly larger decrease in engagement from chemistry (Mean = 1.32) to probability (Mean = 1.15). Overall, the probability group found the game slightly more engaging (Mean = 1.34) than the chemistry group (Mean = 1.23), though overall the engagement score was very low (Mean = 1.29).

Hat Trick

Participant Demographics

There were ten (N = 10) participants in the group that played the history content first and nine (N = 9) in the group that played math first for a total of nineteen (N = 19) participants that played the HT game. Four participants from the history group did not report their age, but those that did averaged just under fourteen years old (Mean = 13.83). Only one participant from the math group did not report their age, and those that did averaged a little over fourteen years old (Mean = 14.37). The history group had three males, three females, and the other four did not report their gender. The math group had one person not report their gender, but all the other participants were male. The history group had four middle schoolers, two high schoolers, and four who did not report. The math group had four middle schoolers, four high schoolers, and one that did not report. Only five participants from the history group reported their grade level, and their average reflected their split between middle and high school (Mean = 8.60). Six participants

reported their grade level in the math group and they were slightly higher on average (Mean = 9.50). The history group had four gamers, two non-gamers, and four that did not report. The math group had one that did not report and one non-gamer, while the rest all identified as gamers. Only four participants from the history group reported their game hours per week and had almost two hours less play time on average (Mean = 8.62) than the math group (Mean = 10.42) which had all but two participants report their game time.

Results

The history group rated the cognitive load of the history content over a full point (Mean = 3.10) higher than the math content (Mean = 2.00). The math group saw an even greater decrease from their first content (Mean = 4.66) to their second (Mean = 2.22). Overall, the cognitive load of the second content was almost two points lower on average (Mean = 2.11) than the first (Mean = 3.88).

The history group gave their first content a slightly higher engagement rating (Mean = 1.46) than their second content (Mean = 1.26). The math group saw a nearly identical drop in engagement from the math content (Mean = 1.44) to the second (Mean = 1.23). This led to an overall decrease of about 0.21 engagement points from the first content (Mean = 1.45) to the second (Mean = 1.25). HT was given a low engagement score overall (Mean = 1.36).

Modern Scrabble

Participant Demographics

Six participants (N = 6) played the vocabulary content first and twelve (N = 12) played the biology content first for a total of eighteen (N = 18) participants played the MS game. Only one person from the vocabulary group reported their age and was sixteen years old. Five participants in the biology group did not report their age, but the rest were fifteen years old (Mean = 15.00) on average. Only one participant from the vocabulary group reported their gender, and they were female. The same participant was the only to report their student status and was in high school, and eleventh grade. The biology group had one middle schooler, six high schoolers, and five that did not report. Only four participants in the biology group reported their grade level, and they averaged ninth grade (Mean = 9.00). Only one participant from the vocabulary group reported their gamer status and reported as a non-gamer and reported playing zero hours per week. The biology group had five gamers, two non-gamers, and five participants that did not report their status. The biology grouped averaged twenty five hours (Mean = 25.00) of game play per week.

Results

The vocabulary group had a slight increase in cognitive load from the vocabulary content (Mean = 1.33) to the biology content (Mean = 1.66). The biology group gave a

much higher cognitive load rating to the biology content (Mean = 3.58) than the vocabulary group gave to either content. This rating decreased sharply for the vocabulary content (Mean = 1.91). Overall, the high biology rating from the biology group causes an average drop in cognitive load from the first content (Mean = 2.45) to the second content (Mean = 1.79).

The vocabulary group had a slight decrease in engagement from the vocabulary content (Mean = 1.16) to the biology content (Mean = 1.12), though it should be noted that both of these ratings are extremely low. The biology group had the same average for their engagement for the biology content (Mean = 1.16) as the vocabulary group had for their vocabulary rating. The biology group also had a similar decrease with their second content, giving the vocabulary content almost the lowest engagement rating possible (Mean = 1.05). Overall, engagement decreased from the first content (Mean = 1.16) to the second content (Mean = 1.08) and engagement was very low overall (Mean = 1.12).

Operation VIP Extraction

Participant Demographics

Six participants (N = 6) played the health content first, while forty two (N = 42) played the social content first for a total of forty eight (N = 48) participants that played the VIP game. Each group had one participants excluded from that count for reporting excessively high ages (70 and 10,000) and one of them also reported excessively high weekly play times (10,000,000,000). Two participants from the health group reported their age and averaged at fifteen years old (Mean = 15.00). Seventeen participants from the social group did not report their age, and those that did averaged just below fourteen

years old (Mean = 13.88). The health group had two males, with none of the other participants reporting their gender. The social group had ten females, sixteen males, and the rest did not report their gender. The health group had two high schoolers with none of the other participants reporting their school status. The social group had eleven middle schoolers, fifteen high schoolers, one participant who claimed to be in college (but also in twelfth grade), and the rest did not report their school status. Only one participant from the health group reported their grade and there were in ninth grade. Ten participants in the social group reported their grade and they averaged about ninth grade (Mean = 9.30). The health group had two gamers, with the rest not reporting their status. The social group had thirteen gamers, twelve non-gamers, and the rest did not report their status. Two participants in the health group reported their hours per week which averaged to twenty two hours (Mean = 22.00), due largely to the great disparity between the two values (2 and 42). The social group had twenty five participants report their hours per week, which averaged to be significantly less than the health group because a few high outliers were counter acted by several zeros (Mean = 7.32).

Results

The health group gave the health content a very low cognitive load rating (Mean = 1.16) and all participants gave their second content, social issues, a one out of ten for cognitive load (Mean = 1.00). The social group gave both contents significantly higher ratings, but both were still well below average, and cognitive load decreased from the first content (Mean = 2.48) to the second (Mean = 1.53). Both groups therefore gave their

first content a higher cognitive load rating (Mean = 1.82) than their second content (Mean = 1.26).

Similar to their ratings for cognitive load, the health group gave very low engagement ratings to the first content (Mean = 1.17) and ones across the board for the second content (Mean = 1.00). The social group gave the social content a low engagement rating (Mean = 1.26) and nearly gave the health content the lowest rating possible (Mean = 1.04). Overall, both groups saw a decrease in engagement from their first content (Mean = 1.21) to their second (Mean = 1.02) and rated the engagement low for the game as a whole (Mean = 1.12).

Pattern Recognition

Participant Demographics

Twenty seven (N = 27) participants played the chemistry content first and seven played the geometry content first (N = 7) for a total of thirty four (N = 34) participants who played the PRT game. Five participants in the chemistry group did not report their age, and those that did averaged between fifteen and sixteen years old (Mean = 15.77). One participant in the geometry group did not report their age, and those that did averaged over a year younger than the chemistry group (Mean = 14.50). The chemistry group had eleven females, eleven males, and the rest did not report their gender. The geometry group had one female, four males, and two that did not report their gender. The chemistry group had two middle schoolers, nineteen high schoolers, and the remaining six did not report their school status. The geometry group had one middle schooler, two that did not report, and the remaining four were in high school. The chemistry group

averaged just below tenth grade (Mean = 9.93) and the geometry group averaged slightly below that (Mean = 9.66). The chemistry group had thirteen participants that identified as gamers, eight that were not gamers, and the rest did not report. The geometry group had four gamers, two non-gamers, and one that did not report. The chemistry group average just under nine hours of game play per week (Mean = 8.95) and the geometry group averaged closer to twelve (Mean = 11.6).

Results

The chemistry group saw a decrease in cognitive load from the chemistry content (Mean = 3.51) to the geometry content (Mean = 2.66). The geometry group gave the geometry content an above average cognitive load rating (Mean = 6.00) and saw a similar decrease in cognitive load for the second content (Mean = 4.71). Overall, the first content had a slightly below average cognitive load (Mean = 4.76) and the second content was over a point lower (Mean = 3.69).

The chemistry group gave the chemistry content an almost one full point below average engagement rating (Mean = 1.55), and gave the geometry a lower engagement score (Mean = 1.15). The geometry group saw an opposite trend, giving geometry a lower engagement score (Mean = 1.38) than the chemistry content (Mean = 1.47). Overall, the first content earned a higher engagement rating (Mean = 1.47) than the second (Mean = 1.31).

Pedestals

Participant Demographics

Seven participants (N = 7) played the math content first and nine (N = 9) played the English content first for a total of sixteen participants (N = 16) that played the PD game. Only two participants in the math group reported their age and they averaged to be fifteen years old (Mean = 15.00). The English group had two participants that did not report their age, and the rest averaged just over fifteen years old (Mean = 15.14). Of the two participants in the math group that reported their gender, one was male and one was female. The English group had two females, four males, and three that did not report their gender. The math group had one middle schooler, one high schooler, and the rest did not report their schooling. The English group had two that did not report their schooling, but the rest all reported being in high school. No one in the math group reported their grade level, and the four in the English group that did report their grade averaged between ninth and tenth grade (Mean = 9.25). The math group had one reported gamer, one reported non-gamer, and the rest did not report. They averaged twenty one and a half hours (Mean = 21.50) between them, though this is largely due to the large difference between them (1 and 42). The English group had five gamers, two non-gamers, and two that did not report their status. That group averaged between ten and eleven hours of game play per week (Mean = 10.71).

Results

The math group gave a very low cognitive score rating to the math content (Mean = 1.14) and the lowest possible score to the English content (Mean = 1.00). The English group gave the English content a just below average rating (Mean = 4.44) and decreased in cognitive load when playing the math content (Mean = 3.00). Overall, there was almost a full point drop in cognitive load from the first content (Mean = 2.79) to the second content (Mean = 2.00).

The math group had almost no engagement rating for the math content (Mean = 1.04) and gave the lowest score possible to the English content (Mean = 1.00). The English gave higher ratings overall, but still saw a decrease in engagement from the first content (Mean = 1.50) to the second content (Mean = 1.23). This meant that both groups saw a decrease in engagement from the first content they played (Mean = 1.27) to the second one (Mean = 1.11) and overall engagement was low (Mean = 1.19).

Quiz Up!

Participant Demographics

Five participants (N = 5) played the music content first and seven played the French content first (N = 7) for a total of twelve (N = 12) participants that played the QU game. The French content group total does not include one participant who was excluded for reporting his age as 340 years old. One participant in the music group did not report their age, but the other four were split evenly between thirteen and fourteen (Mean = 13.50). The French group had two that did not report their age, and the rest averaged just

over fifteen years old (Mean = 15.20). The music group had two males, two females, and one that did not report their gender. The French group had two males, three females, and two that did not report their gender. The music group had one that did not report what their schooling was, but the other four were split evenly between middle and high school. The French group had two that did not report their schooling, with all the rest being in high school. The three participants that reported their grade in the music group averaged between seventh and eighth grade (Mean = 7.66). The two participants in the French group that reported their grade level were both in tenth grade. The music group had two gamers, two non-gamers, and one that did not report their gaming status. The French group had two gamers, three non-gamers, and two that did not report. The music group averaged ten hours (Mean = 10.0) of game play per week and the French group averaged almost two hours less (Mean = 8.40).

Results

The music group gave the music content a respectable mild cognitive load rating (Mean = 3.60) and showed a decrease of two full points for the French content (Mean = 1.60). The French group had higher values for both contents, but had a similar decrease going from the French content (Mean = 4.28) to the music (Mean = 2.14). The overall cognitive load decrease by just over two point from the first content (Mean = 3.94) to the second content (Mean = 1.87).

The music group gave a low engagement rating to the music content (Mean = 1.22) and showed a decrease in engagement when moving to the French content, almost giving it the lowest score possible (Mean = 1.09). The French group had even lower

scores for both contents, giving the French content a lower rating than the music group gave their first content (Mean = 1.14) and gave the lowest rating possible to the music content (Mean = 1.00). This means both groups showed a decrease in engagement from their first content (Mean = 1.18) to their second content (Mean = 1.04) and a low engagement overall (Mean = 1.11).

Think Fast

Participant Demographics

Five participants (N = 5) played the math content first, while eight (N = 8) played the Spanish content first for a total of thirteen participants (N = 13) that played the TF game. One participant in the math group did not report their age and the rest averaged between fourteen and fifteen years old (Mean = 14.50). Three participants in the Spanish group did not report their age and the rest averaged only slightly younger than the math group (Mean = 14.40). The math group had two males, one female, and the other three did not report their gender. The Spanish group had three males, two females, and three that did not report their gender. The math group had one middle schooler, one that did not report their schooling, and the rest were in high schooler. The Spanish group had one middle schooler, three that did not report their schooling, and the rest were in high school. None of the participants in the math group reported their grade level, and only two from the Spanish group did, but both were in ninth grade. The math group had two gamers, one non-gamer, and two that did not report their gamer status. The Spanish group had four gamers, one non-gamer, and the rest did not report their gaming status. Three participants in the math group reported their hours of gaming per week and averaged

between twelve and thirteen hours per week (Mean = 12.33). Four in the Spanish group reported their play time per week and they average about four hours less than the math group (Mean = 8.50).

Results

The math group gave the math content a mild cognitive load rating (Mean = 3.40) and gave the Spanish content the lowest possible rating (Mean = 1.00). The Spanish group actually saw an increase in cognitive load rating from their first content (Mean = 3.37) to their second content (Mean = 4.00). Overall, there was still a decrease in cognitive load from the first content (Mean = 3.38) to the second (Mean = 2.50).

The math group gave the music content a low engagement score (Mean = 1.17) and saw a decrease in engagement in the second content (Mean = 1.12). The Spanish group had a greater change in engagement starting at a higher engagement rating for the Spanish content (Mean = 1.36) but decreasing to a very low engagement score for the math content (Mean = 1.02). Finally, the engagement ratings decreased from the first content (Mean = 1.26) to the second content (Mean = 1.07) for both groups.

Discussion

Cognitive Load

The cognitive load reported by participants almost universally decreased from the first content to the second content, regardless of which content was played first. A factorial one way analysis of variance (ANOVA) shows that genre does have a significant effect on cognitive load at the $P < 0.05$ level [$F(4,4) = 3.401$, $P = 0.010$]. The

ANOVA analysis indicated that order was not significant at the $P > 0.05$ level [$F(1,1) = 0.875$, $P = 0.351$] and order still did not have an effect when considered along with genre at the $P > 0.05$ level [$F(4,4) = 1.657$, $P = 0.161$]. There were only four groups that showed an increase in cognitive load from their first content to their second. The first of these was the CR group that played the vocabulary content first but gave the history content a higher cognitive load rating. The second was the MS group that played the vocabulary content first, instead giving the higher score to the biology content. The third group was TF group that played the Spanish content first, giving the math content a higher cognitive load instead of the Spanish content. The final group was the Towers group that played the spelling content first, giving math a higher cognitive load rating instead. In all four cases, the other group for that game followed the normal pattern of rating their second content lower than their first content. The only other exception to the trend was the FOS group that played the probability content first. That group rated both contents the exact same (2.67) for cognitive load. The other FOS group showed the regular decrease from first to second content.

The results mean whichever content is played second will get a lower cognitive load rating. The order in which the contents are played does not have an effect on the cognitive load reported, because this decrease occurred regardless of the content played first. This general decrease in cognitive load from the first to second content may come from the player needing more cognitive resources to learn the game while playing the first content. When they play the second content, because the mechanics are identical, they already know how to play the game and so those cognitive resources are free while

playing the second content. The four contents that were exceptions to this were history, biology, and two math contents. The higher load for these contents may be because math and biology require more in depth problem solving that tends to result in higher cognitive loads (Sweller, 1988). They are more technical topics than the other content from those games. The history content may be explained by looking at the content topics from those games that received lower ratings. They were vocabulary for three of them, including the CR game with the history content, and the fourth was Spanish. Participant's first language was not recorded, but since they were all middle and high school students in the United States, English was likely the first language of the majority of participants, or the participant would have at least had a high proficiency in English. Vocabulary tasks require a low amount of cognitive effort to process for first speakers (Yeung, Jin & Sweller, 1998). So, while history may not be a cognitively demanding content, the partner content of vocabulary may have just required a significantly lower amount of cognitive load, causing history to get a higher rating. The FOS group that gave both contents an equal rating may be due to both contents, probability and chemistry, being technical skills that participants found equally challenging (Sweller, 1988).

Another possible explanation for these exceptions is problems with the games themselves. Namely, the four games with a more cognitively demanding content may have not been properly content agnostic, and the mechanics fit better with one of the contents over the other. Whether this is the content with the higher rating, or the one with the lower rating is unclear. The same groups for both CR and Towers showed a similar

increase in engagement for these contents, but the other two groups showed a decrease in engagement instead.

Overall cognitive load ratings were low, with only two groups (PRT geometry and Towers vocabulary) giving an above average rating for both contents combined. This indicates that CAGE games are not overly cognitively demanding by design. The Towers game was one of the most complex games used in this study, so it was expected for it to have one of the higher cognitive loads due to its mechanics. PRT on the other hand was a very simple game, but both contents were technical in nature (geometry and chemistry), meaning its higher cognitive load rating may have come from its content rather than its mechanics (Sweller, 1988).

Research question two examines whether or not the mechanics and context play a role in cognitive load measured while playing CAGE games. Results indicate that the answer is no, since analysis indicated that only genre has an effect on cognitive load. This means that the order in which the contents are played does not have an effect on the cognitive load and so a CAGE game is as cognitively demanding on both contents.

Engagement

Participant engagement decreased from the first content to the second content. ANOVA showed that genre had an effect on engagement at the $P > 0.01$ level [$F(4,4) = 7.525$, $P = 0.001$] but that order in which contents were played did not have an effect engagement at the $P < 0.05$ level [$F(1,1) = 2.516$, $P = 0.114$] and that order still did not have an effect at the $P < 0.05$ level when combined with genre [$F(4,4) = 1.766$, $P = 0.137$]. There were four groups that showed an increase in engagement from the first to

second content. Two of these groups, the CR vocabulary group and the Towers vocabulary group, were ones that showed an increase in cognitive load from the first to second content. The two other groups that showed an increase in engagement from the first content to the second were the HT math group and the PRT geometry group. One group, the FOS probability group, showed equal engagement from the first to second content. However, this is the same group that showed an equal cognitive load rating for both contents.

The decrease in engagement from the first to second content is not unexpected. Since CAGE games employ the same mechanics for both contents, the two versions of the game are the same from a gameplay perspective. Players are likely then to be less engaged during the second content, because they've already seen what the game has to offer. They are largely replaying the same experience. This expectation was a driving force behind splitting participants into two groups for each game and playing the contents in a different order. The ANOVA analysis confirms that the order the content is played does not actually have an effect on the engagement and it is only player apathy that causes the second content to receive a lower rating.

Two of the groups that had an increase in engagement from the first to second content also had an increase in cognitive load from their first content to their second. These two groups may have found the more cognitively stimulating content to be more engaging than the other content they played. The other two groups had the regular decrease in cognitive load from the first to second content. As explained above, this decrease in cognitive load may be due to the player fully understanding how to play the

game when playing the second content. In terms of engagement, the participants may have enjoyed the game more when playing the second content because they understood how to play better. They might have spent most of their time during the first content being confused on how to play, instead of being engaged.

Overall engagement is rather low, with no groups giving an average rating of 2.5 or better overall and three groups giving their second content the lowest score possible. The highest engagement rating, and the only one over a rating of 2, was the Towers math group with an overall engagement of 2.11. The other Towers group had the second highest score, giving Towers a 1.72 for overall engagement. This means Towers, the one game made without the CAGE framework, was the most engaging CAGE game used in this study. This may be due to the CAGE framework itself, which all the other games used in this study were built off of. However all the games, including Towers, used the CAGE model. The underlying software framework is unlikely to have an effect on engagement. Instead, this difference is more likely to be caused by Towers itself. Towers is the most complex of the games used in this study and it had over two years of development put into it before being deployed in this project. The games that were based off the CAGE were all relatively simple games, having been built in only three weeks total. It is likely that the games themselves were too simple to hold player interest. The low engagement scores could also be caused by the games tending to be more exogenous in nature, due to the CAGE model (Malone & Lepper, 1987).

Another consideration is engagement by game genre. See Table 7 for an overview of the CAGE study results by genre. When the CAGE games were being created, no

specifications were given in regards to genre. Even so, the eleven CAGE games can be separated into six different genres. Three of these had only one game in them. Tower Defense only had Towers, Action only had HT, and the Racing genre had only CR in it. BM and VIP were the two games that made up the shooter genre, and FOS and MS were the only games to make up the Puzzle genre. The last, and largest, genre was the Quiz game genre which had four games in it: PRT, PD, QU, and TF. Comparing the differences in mean engagement for each group in each genre with an ANOVA reveals that the differences in engagement between genres is statistically significant ($P < 0.01$, $P = 0.008$). Looking at the mean engagement for each genre, the Tower Defense genre was the most engaging, but this is not surprising given that the Towers game already received the highest engagement score. The Quiz genre was the least engaging, which is also not surprising. The Quiz games asked the players direct questions, which does not utilize stealth assessment and is therefore less likely to be engaging (Shute, 2011). See *Figure 35* for an overview of the mean engagement ratings for each genre.

Research question three examines whether or not CAGE games can maintain player engagement. The results indicate that the answer is no, and that engagement decreases with further play. This is because the second content was shown to be less engaging than the first, regardless of the order in which the contents were played.

Game Flow

The fourth research question examines if switching the context interrupts the game flow. Given the significant decreases in both cognitive load and engagement, the results indicate that switching the context has a notable, negative, impact on game flow.

This may be because participants filled out several survey questions between the first and second content, which may be the basis of the interrupt in game flow and may have created a fatigue effect which is common in repeated measures studies.

Limitations

There are a few concerns that limit the generalizability and validity of the CAGE study. One of these is the mismatch in the number of participants per group. At least five participants were needed per group, but some groups were randomly sorted to have significantly more than the other group for the same game. Most notably the social group for the VIP game which had forty two participants, in a row, sorted into it. This may have been an unlikely error by the random number generator used to sort the participants into their groups, or the incredibly small chance that the generator returned the same number forty two times in a row. It is more likely to have been caused by the next potential issue, which is that a great deal of participants played the game as part of a class, led by a teacher who was not under the supervision of the researcher. The teachers had instructions, but their adherence to the directions cannot be verified. The extra-large grouping may have been caused by the teacher not sharing the proper link with their students, sharing the direct link for the game instead of the general link which would sort the visitors. There may also be a coercion issue, if students were made to play the game as part of a class activity rather than volunteering for participation as was intended. Finally, the test itself may have had an effect on game flow and engagement measurement for the second content. The two contents of the games were broken up by a large series of questions, which required participants to stop playing and consider their

answers. This may explain the consistently lower engagement score for the second content.

The biggest threat to validity for the CAGE study is mono-method bias (Shadish, Cook & Campbell, 2002), which is brought on by participant apathy. All measures used were self-report, so participants may have added false information. The reason to suspect this possibility is the chance that participants may have become bored with the survey and simply skipped ahead, rather than providing honest answers. The UES was twenty six questions long and participants may have been dismayed when they realized they needed to fill it out not once, but twice for a total of fifty two questions. This possibility is backed up by the lower engagement scores for the second content and several participants who gave a rating of one to all the questions for engagement in the second content. Two groups had all their members provide one to all of the questions, bottoming out their engagement rating for the second content. A rating of one was the default value, so it is unclear whether participants who rated everything a one simply clicked through without actually considering the questions, or if they considered all the questions carefully but could not justify a higher rating on any of the questions.

Conclusion

The CAGE study successfully answered research questions two, three and four. CAGE games do not carry a high cognitive load, and it does not matter which content is presented first when considering cognitive load. This is encouraging as CAGE games themselves should not cause a high level of cognitive load. Given that the exceptions to

the trend of decrease in cognitive load mostly occurred with a technical content, the CAGE model itself does not lead to extraneous load.

Unfortunately, the results indicate that CAGE games are not very engaging and that switching the context has a notable impact on game flow. The issues with game flow indicate that it may be better to use the CAGE model to develop multiple separate games with different contents, rather than try to package multiple contents into one game. It may also be that players need a significant break between two different versions of the game to prevent boredom. The low engagement may result from the simplicity of the games, rather than the CAGE model itself. CAGE is designed to allow for rapid creation of educational games, but the decrease in time spent for the second version is relative and more time might have been needed by the individual developers to create more compelling games.

8 AFFECTIVE STATE INTEGRATION IN CAGE GAMES

Overview

Affective state is an individual's current cognitive state which is highly influenced by their emotions (Harding, Paul & Mendl, 2004). A person's affective state has a great effect on their perception of external stimuli. In particular, neutral stimuli are more likely to be interpreted as negative by a person in a negative affective state, and are more likely to be seen as a positive stimulus by a person in a positive affective state.

Affective states also have an effect on engagement, both in video games (Sykes & Brown, 2003) and in learning (Kort, Peilly & Picard, 2001). This means that the importance of the player's affective state is compounded in game based learning, as players in a negative affective state will neither learn well nor play well. A player's affective state has an effect on not just their learning, but also their persistence and incoming knowledge (Shute et al., 2015).

This part of the CAGE thesis uses video capture and facial analysis software to detect the player's affective state in real time, without interrupting the game play. When negative states are detected, the difficulty of the game is adjusted in response. If the player is bored, the game is currently too easy and is less of a challenge than the player is equipped to handle. If the player is frustrated, the game is too difficult and needs to lower its challenge for the player to succeed. By adjusting the difficulty up or down as needed, without interrupting the game flow, the game can keep the player in cognitive flow (Csikszentmihalyi, 1996) and maintain an ideal level of challenge for any given student.

Doing this should help increase player engagement and help counter the low engagement recorded during the CAGE study.

The Effect of Affective State

A student's emotions and affective state have a great impact on their motivation and their ability to learn (Kort et al., 2001). Students who are in a negative affective state are less motivated and will not be as persistent when trying to learn new topics. They are also more likely to interpret neutral stimuli as negative stimuli, which only reinforces the negative affective state (Harding et al., 2004). It is best then to try and prevent the student from falling into a negative affective state in the first place. This is not always achievable and so it becomes important to be able to detect when a player has fallen into a negative state and reverse it as early as possible.

There are many possible positive affective states, but the one of particular interest for the purposes of game based learning is cognitive flow (Csikszentmihalyi, 1996). Since players will learn best while in a state of flow, it is important to get them into that state and maintain it throughout the game. Detecting the player's affective state and adjusting the difficulty in real time to match the player's skill level would help reach the match of skill and difficulty that is needed to induce cognitive flow. Unfortunately, flow is one of the harder affective states to detect (Craig, Graesser, Sullins & Gholson, 2004). However, it can be detected briefly and so it is possible to ensure that the player is in a flow state at a given time.

CAGE makes use of three affective states. The first is flow, which is always the target state. The second is boredom, which occurs when the player's skill level is higher than the challenge the game is currently providing. This means that the game is too easy and the player will become bored and disengaged. They are also likely to not learn very much given that their skill level is higher than the material being presented. The player may be reviewing or practicing, but these activities can still be challenging and engaging. If the player is bored, it is a sign that the current difficulty needs to be increased.

The third state that CAGE makes use of is frustration. Frustration is a sign of the opposite problem that leads to boredom. That is, the player's skill is lower than the challenge being presented by the game. The game is too hard, and the player is not able to keep up. The player may have missed material they were supposed to have covered already, or may have been advanced to the current content when they did not actually have mastery of previous topics. However this may have occurred, the player is not ready to learn and perform at the current level. They become frustrated by their failure and are likely to give up. When the player becomes frustrated, it is a sign that the current difficulty needs to be lowered to better fit the player's skill.

Real Time Affective State Analysis

In order to ensure that affective state detection was not distracting and did not impede the player's ability to play the game, the detection was done entirely through the webcam. CAGE uses facial tracking software called Visage|SDK (Visage) from Visage Technologies. Visage offers a full suite of facial tracking and analysis features, but CAGE makes use of the emotion estimation feature. This feature analyses an image,

determines if a face is present, and if there is a face it will estimate what emotion the person has in the image. This works on either a still image or a frame from a video, which is why it could be used on the webcam video.

The Visage emotion analysis is done using the Facial Action Coding System (FACS; Ekman & Friesen, 1977). FACS defines a set of feature points on the face and how their alignment indicates particular emotions. FACS defines six basic emotions: Anger, Disgust, Fear, Happiness, Sadness, and Surprise. These are the same six emotions that are returned by Visage. The emotion estimation in Visage returns six floating point numbers which represent the six basic emotions. Each of these numbers is between zero and one and represents the confidence that the person in the image is feeling that particular emotion in that image. So, a value of zero means Visage detects nothing that indicates that emotion and so there is zero chance that the person is feeling that emotion in that image. Likewise, a value of one is complete confidence that the person is feeling that emotion. Values of zero and one are unlikely, and most of the time the values will fall somewhere in between. The sum of all six values does not have to be equal to one either, as a person can show multiple emotions at a time. Affective states are particular combinations of emotions.

Given that affective states are combinations of emotions, it is important to define these combinations so as to use these six basic emotions to detect a particular affective state (Craig, D'Mello, Witherspoon & Graesser, 2008). The first affective state that is one of the easier ones to detect is boredom. Boredom is defined as when none of the confidence values are high. That is, when the player is displaying a neutral expression.

Frustration is defined as showing a high confidence in anger and a low confidence in happiness. Flow is defined as showing surprise and having a low sadness value. A fourth possible state was defined as none. This was the state reported when none of the above conditions were satisfied. It represented the player being in a state that is not of interest for altering the game's difficulty. This was also the state reported for situations where the player's face could not be found or some error had occurred with Visage or the web cam.

An affective state is only detectable on a person's face for a brief period of time which is usually less than two seconds (Craig et al., 2008). Another problem is that the data from the emotion estimator contains a large amount of noise. It is not reliable to make difficulty adjustments off of a single frame, but the affective state will not be shown for very long. To address these issues, CAGE maintained the last full second's worth of states detected. Each frame the detected state was stored and the total number of frames stored was one second's worth of frames as calculated from the frames displayed per second. The most common state from that data was used to make difficulty adjustments.

Study Design

This study follows a two by two design since only a single game was used so genre was not a factor. The dependent measures are cognitive load and engagement, similar to the CAGE study and the independent variable was order. The goal of this study is to examine the effect the dynamic difficulty has on player cognitive load and engagement. No data about the player's affective state was recorded or analyzed for this

study. This study was approved by the IRB of ASU as Content Agnostic Mechanics – STUDY00005148.

Participant Demographics

Seventeen students in a game engine architecture class at Arizona State University took part in this study for five points of extra credit on a previous assignment. This study was delivered in person, rather than online as the CAGE study was. However, all other aspects of the study remain identical to the CAGE study. Participants were sorted by alphabetical last name into content group A or B. Due to the odd number of students in the class group A had an additional person. Participants were instructed to ensure their computer's webcam was open, but were not told what the webcam was used for yet to avoid intentional tampering with the difficulty during the study.

Procedure

Like the CAGE study, participants filled out the background and demographics form. They then played their first assigned content. After finishing, they filled out a cognitive load form (Paas et. al., 2003) and a UES (Wiebe et al., 2014). Participants then played the second content for their group, after which they filled out a second copy of both forms, had an opportunity to make any final comments, and then were finished. All of these forms are identical to the ones used in the CAGE study. After all the participants were finished, they were debriefed about the nature and purpose of the study.

Results

There were nine (N = 9) participants in the math group and eight (N = 8) in the vocabulary group for a total of seventeen (N = 17) participants that played the BB game. All participants in both groups were in university. Five of the students were Masters in Software Engineering students, one was a PhD student in Simulation, Modeling, and Applied Cognitive Science, and the rest were undergraduate students in the Software Engineering program. The background form did not ask for this information, so it cannot be determined how these differing levels of students were distributed between the groups. The math group averaged to be almost twenty four years old (Mean = 23.70) and the vocabulary group came out to be almost twenty seven years old on average (Mean = 26.80).

Of the nine people in the math group, only three identified as non-gamers, while the vocabulary group had only two non-gamers among them. The math group reported about nine and a half hours of game time per week on average (Mean = 9.44) and the vocabulary group averaged six hours per week (Mean = 6.00).

The math group showed a moderate cognitive load on average for the first content (Mean = 3.20) which decreased by over a point for the second content (Mean = 1.80). The vocabulary group had a slightly higher cognitive load for the first content (Mean = 3.50) but likewise saw an equivalent decrease in cognitive load for the second content (Mean = 2.12) like the math group did.

For the engagement ratings, the math group had a decent level of engagement (Mean = 1.87) and saw only a slight decrease for the second content (Mean = 1.71). The vocabulary group had a higher engagement for their first content (Mean = 2.03) and also saw a decrease in engagement for their second content (Mean = 1.86).

Discussion

Both groups saw a decrease in cognitive load from the first content to the second, which follows the pattern that emerged during the CAGE study. ANOVA shows that this effect is significant at the $P < 0.01$ level [$F(1,1) = 10.621$, $P = 0.005$], and order does not have an effect at the $P < 0.05$ level [$F(1,1) = 0.267$, $P = 0.613$] which follows the pattern from the CAGE study. This shows that the BB game is comparable to the other CAGE games in relation to cognitive load. It also shows that the real time affective state detection does not appear to have an effect on cognitive load.

Similar to the CAGE study, both groups saw a decrease in engagement from the first content to the second content. Once again, this decrease is significant at the $P < 0.05$ level [$F(1,1) = 6.350$] and the order is not significant at the $P < 0.05$ level [$F(1,1) = 0.340$, $P = 0.569$]. This shows that the dynamic difficulty from the affective state detection does not appear to have an effect on engagement. Given that the decrease is significant, this means that CAGE games that integrate this real time difficulty adjustment system still experience a meaningful drop in engagement when the content is switched. Overall engagement is relatively high, with both group giving average ratings close to the highest rated game from the CAGE study.

Research question five examined whether or not adding affective state detection and dynamic difficulty increased engagement or not. The results of this study indicate that no, changing the difficulty in real time does not increase engagement.

Limitations

The main limitation with this study is the low number of participants, only seventeen, compared to over two hundred for the CAGE study. The participants were also older, being in college as compared to middle and high school. These two issues limit the generalizability of these results and mean that this study should be taken as a pilot.

Another key issue is the facial tracking software itself. There is no calibration process to acquaint the system with the user's face prior to analyzing their facial features. This means that various individual characteristics of any given participant may have thrown off the analysis. Potential issues include wearing glasses, hair that obscures part of the face, facial hair, differing skin tones, scars, moles, and other differences in a person's face. This means that the analysis results returned by the system may have been incorrect for some participants. In addition, the process used to convert the emotions to the affective states had only basic testing before being used in this study. A separate study should be done to verify and refine the affective states being determined from the emotion data.

Conclusion

This study shows that making adjustments to the game difficulty based off of the affective state of the player do not appear to increase engagement. Engagement for both groups was higher than all but one of the groups from the CAGE study, showing that the dynamic difficulty may have an effect on overall engagement, but it does not remove the significance of the general decrease from the first content to the second.

9 CONCLUSION

Future Work

There are multiple outlets for potential future work in the area of CAGE games. The first area for further work is the framework itself. The CAGE framework was in an initial prototype state when it was used in this study. Further improvements and iterations could be made to help speed up development and increase code reusability even further. The framework could be expanded to include a much wider variety of default hooks to account for many more common occurrences in educational games. One such specific set of hooks would be to design a series of genre specific hooks, hooks that are commonly needed in a specific game genre, but rarely used in others. Allowing developers to easily plug in a series of hooks for the genre they are going to develop would speed up development for them, and leaving these out of the default set can help reduce complexity and clutter for projects that will not utilize them.

Another potential area for further work with the CAGE framework, and possibly the CAGE model itself, is accessibility. Accessibility is enabling those with disabilities a means with which to play the game, with educational games giving particular concern for those with visual impairments (Morelli, Foley & Folmer, 2010). The CAGE framework currently does not have anything built into it that enables or facilitates accessibility in CAGE games. This is something that should be considered to prevent the risk of CAGE games becoming useless for a portion of the student population.

The CAGE model outlines the student model component, which would track the player's understanding of the material in real time as the player performed actions within the game. This component was not implemented within this thesis. This is due to the amount of work that will be needed to implement what would be content agnostic assessment. In order to work properly, the student model will need a way to measure learning outcomes of any potential content domain. This is itself a great under taking which is left for further work. One will have to define a series of alignments which link a learning objective to a game mechanic. Consideration will be needed to ensure that these alignments are both flexible enough to be matched between any learning objective and any mechanic, while still be rigid enough to ensure their validity as a measurement tool.

More work is needed in order to increase engagement of CAGE games. A good first step is to build a series of more complex games. Several CAGE games were created for this thesis in order to have a sufficient number of games to provide evidence for claims about CAGE games. However, this seems to have created an issue of quantity over quality and the most engaging game was a game that had been in development for years, as opposed to three weeks. Creating a smaller number of CAGE games, with more development time being spent on each game, is needed to validate the engagement results found in the CAGE study. If low engagement is confirmed, more work is needed to determine how CAGE games can be made more engaging. If higher engagement results from the more complex games, then it is important to investigate the baseline needed to establish engagement.

Using dynamic difficulty based off of player affective state was not shown to be effective in countering the low engagement problem with CAGE games. However, more work can still be done with this system. As discussed previously, one of the limitations with the current system is that the affective state detection is rough. It would be good to try to refine this, by recording a video of the participants as they play the game and have the participants replay the video afterwards, self-report their affective state throughout the video, and then compare their statement against what the game reported their affective state was at the same time. The state detection can then be adjusted to better fit the participants report.

The final area that needs additional work is the issue of interruption in game flow that occurs when switching the active content. In this case, the flow could have been broken by the survey itself, but it is worth investigating if the flow is always broken when switching the content. An ideal way to do this is to switch the content during the game, rather than having any sort of interruption in order to switch the content. Giving the player some sort of in-game warning that the content will switch should help minimize the effect the switch has on game flow.

Conclusions

All five research questions have been successfully answered. The CAGE framework was shown to make a noticeable difference in the amount of time and effort needed to build educational games. The amount of development time decreased drastically from the development of the first to the second content. In addition, a significant amount of the code could simply be reused without edits. This means that the

CAGE framework and model greatly help to reduce the amount of time needed to develop secondary games, after an initial one has been built. Those who used the framework for this study showed interest in using it again, providing proof that they found it useful. Reducing development time and increasing code reusability was a key goal of the CAGE framework and model, and this goal was met.

CAGE games were shown to have a fairly low level of cognitive load. This is encouraging because it means that the CAGE model itself does not induce high levels of extraneous load by design. There was a significant decrease from the first content to the second. This means that the load was not dependent on the order in which the contents were played, and helped verify that CAGE games are not cognitively burdensome. Future CAGE games can get deeper into material and teach more complex topics, because the player will have cognitive power to spare. Keeping cognitive load low was a goal of the CAGE model and that goal was met.

CAGE games were also shown to have a fairly low level of engagement. Engagement also decreased significantly from the first content to the second, regardless of the order in which they were played. The upside of this is that it, like the cognitive load results, show that CAGE games are successful at being content agnostic. This is because the order in which contents were played would have an effect if one content was better fit to the mechanics than the other. The downside is that if CAGE games are not engaging, they will be less effective as teaching tools. The most engaging game was the one that had the most development effort put into it, and so more work is needed to build higher quality more complex games to verify whether or not CAGE games have

consistently low engagement, or if there is some baseline complexity needed to make CAGE games engaging. This was also the part of the results that were most likely to suffer from participant apathy from the long survey, which may have skewed the results to show low engagement. Making engaging games was a key goal of CAGE, and this goal appears to have not been met.

Additionally, switching the active content does have a significant effect on the game flow. This is evidenced by the low engagement scores that participants reported after the content was switched. This may be more from the long engagement survey interrupting the game flow and future work should try switching the active content during the game with no other form of interruption. One of the goals of CAGE is to be able to switch the active content without interrupting the game flow, and it appears that goal has not been met.

Finally, integrating dynamic difficulty which changes based on the affective state of the player was not shown to increase engagement. Additionally, there was still a decrease in engagement from the first to second content. There were several limitations of this pilot study and more refinement is needed before final conclusions should be made about the use of dynamic difficulty.

Overall, the CAGE model and framework were a success. All the research questions were conclusively answered. Two of the main goals of CAGE were successfully met. The other two were not, but there is a clear path for further work to address those outstanding issues. The CAGE framework greatly lowers the amount of time needed to develop multiple educational games and greatly increases the amount of

code that can be reused. Game mechanics can be developed in a way that allows for multiple content domains to be layered on top of them without any changes to the mechanics themselves. Finally, the CAGE model does not lead to the development of games that need a high cognitive load in order to be played, allowing the games to add more depth to the topics being presented.

FOOTNOTES

1. Super Mario Brothers by Nintendo, 1985.
2. Scrabble by Hasbro, 1938.
3. Tetris by Alexey Pajitno, 1984.
4. Adobe Flash by Adobe Systems, 2005.
5. Unity Engine by Unity Technologies, 2005.

TABLES

Table 1

Overview of the Non-Framework CAGE Games

Game Title	Genre	Mechanics	Content A	Content B	Screenshot
Fighters	Arcade Puzzle	Elminate Blocks to Send Blocks to Opponent	Spelling	Math	*Figure 10*
Towers	Tower Defense	Build Towers from Letters or Numbers	Spelling	Math	*Figure 12*

Table 2

Overview of the First Set of CAGE Framework Games

Game Title	Genre	Mechanics	Content A	Content B	Screenshot
Bean Man	2D Shooter	Shoot Enemy Afflictions	Health and Disease	Social Issues	*Figure 15*
Car Racing	Quiz	Answer Questions Correctly to Speed Up	Vocabulary	History	*Figure 17*
Fun-O-Sphere	Puzzle	Eliminate Spheres to Achieve the Correct Composition	Probability	Chemistry	*Figure 18*
Hat Trick	2D Action	Catch the Correct Answer	Math	History	*Figure 20*
Modern Scrabble	Puzzle	Line up Answers	English	Biology	*Figure 21*

Table 3

Overview of the Second Set of CAGE Framework Games

Game Title	Genre	Mechanics	Content A	Content B	Screenshot
Operation VIP Extraction	Horizontal 2D Shooter	Only Let the Ally Pass	Social Science	Biology	*Figure 23*
Pattern Recognition Training	Quiz	Pick the Correct Object	Geometry	Chemistry	*Figure 25*
Pedestals	Quiz	Select the Correct Answer	Linear Equations	English	*Figure 26*
Quiz Up!	Quiz	Pick the Correct Answer	Music	French	*Figure 27*
Think Fast	Quiz	Form the Correct Answer	Math	Spanish	*Figure 29*
Titanical	Quiz	Shoot the Correct Answer	History	Uncompleted	*Figure 31*

Table 4

Overview of the Affective Computing CAGE game.

Game Title	Genre	Mechanics	Content A	Content B	Screenshot
Bingo Bingo	Puzzle	Swap two adjacent items to match three in a row.	Math/Trigonometry	Vocabulary	*Figure 32*

Table 5

Overview of the CAGE games grouped by genre.

Genre	Games
Action	Hat Trick
Puzzle	Fun-O-Sphere, Modern Scrabble
Quiz	Car Racing, Pattern Recognition Training, Pedestals, Quiz Up!, Think Fast
Shooter	Bean Man, Operation VIP Extraction
Tower Defense	Towers

Table 6

Overview of the mean values for cognitive load rating and engagement ratings for all groups and games in the CAGE study.

Game	Group	Cognitive Load 1	Cognitive Load 2	Overall Load	Engagement 1	Engagement 2	Overall Engagement
Bean Man	A	2.50	2.20	2.35	1.40	1.27	1.33
Bean Man	B	5.17	4.25	4.71	1.90	1.42	1.66
Car Racing	A	2.67	4.67	3.67	1.58	1.63	1.61
Car Racing	B	6.33	2.33	4.33	1.45	1.32	1.38
Fun-O-Sphere	A	2.67	2.67	2.67	1.35	1.35	1.35
Fun-O-Sphere	B	3.33	2.03	2.68	1.33	1.15	1.24
Hat Trick	A	4.67	2.22	3.44	1.52	1.56	1.54
Hat Trick	B	3.10	2.00	2.55	1.47	1.26	1.36
Modern Scrabble	A	1.33	1.67	1.50	1.17	1.13	1.15
Modern Scrabble	B	3.58	1.92	2.75	1.17	1.05	1.11
Operation VIP	A	2.49	1.53	2.01	1.27	1.05	1.16
Operation VIP	B	1.17	1.00	1.08	1.17	1.00	1.09
Pattern Recognition	A	6.00	4.71	5.36	1.39	1.47	1.43
Pattern Recognition	B	3.52	2.67	3.09	1.56	1.15	1.35
Pedestals	A	1.14	1.00	1.07	1.05	1.00	1.02
Pedestals	B	4.44	3.00	3.72	1.51	1.23	1.37
Quiz Up	A	3.60	1.60	2.60	1.22	1.09	1.16
Quiz Up	B	4.29	2.14	3.21	1.15	1.00	1.07
Think Fast	A	3.40	1.00	2.20	1.17	1.13	1.15
Think Fast	B	3.38	4.00	3.69	1.37	1.02	1.19
Towers	A	4.44	5.78	5.11	1.52	1.91	1.72
Towers	B	4.83	4.00	4.42	2.13	2.10	2.11

Table 7

An overview of the CAGE study results with the games organized by genre.

Game	Group	Engagement 1	Engagement 2	Overall Engagement	Overall Load	Overall Engagement	
Bean Man	A	1.40	1.27	1.33	2.35	1.33	Shooters
Bean Man	B	1.90	1.42	1.66	4.71	1.66	
Operation VIP	A	1.27	1.05	1.16	2.01	1.16	
Operation VIP	B	1.17	1.00	1.09	1.08	1.09	
Hat Trick	A	1.35	1.35	1.35	2.67	1.35	Action
Hat Trick	B	1.33	1.15	1.24	2.68	1.24	
Fun-O-Sphere	A	1.52	1.56	1.54	3.44	1.54	Puzzle
Fun-O-Sphere	B	1.47	1.26	1.36	2.55	1.36	
Modern Scrabble	A	1.17	1.13	1.15	1.50	1.15	
Modern Scrabble	B	1.17	1.05	1.11	2.75	1.11	
Car Racing	A	1.58	1.63	1.61	3.67	1.61	Quiz
Car Racing	B	1.45	1.32	1.38	4.33	1.38	
Pattern Recognition	A	1.39	1.47	1.43	5.36	1.43	
Pattern Recognition	B	1.56	1.15	1.35	3.09	1.35	
Pedestals	A	1.05	1.00	1.02	1.07	1.02	
Pedestals	B	1.51	1.23	1.37	3.72	1.37	
Quiz Up	A	1.22	1.09	1.16	2.60	1.16	
Quiz Up	B	1.15	1.00	1.07	3.21	1.07	
Think Fast	A	1.17	1.13	1.15	2.20	1.15	
Think Fast	B	1.37	1.02	1.19	3.69	1.19	
Towers	A	1.52	1.91	1.72	5.11	1.72	Tower Defense
Towers	B	2.13	2.10	2.11	4.42	2.11	

FIGURES

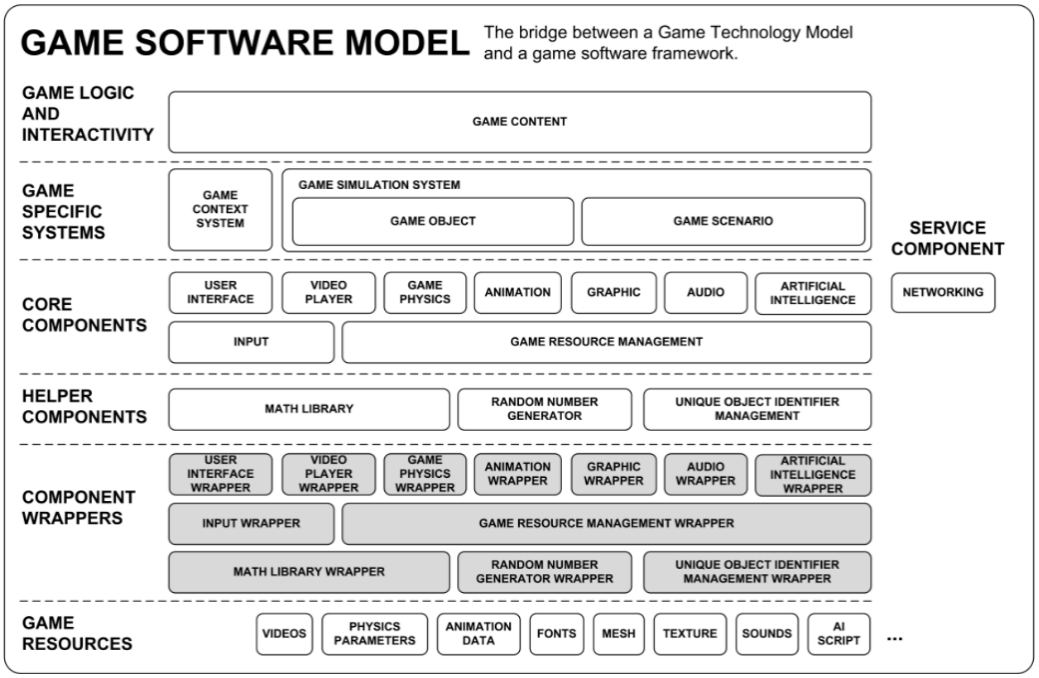

Figure 1. The Game Software Model (Tang & Hanneghan, 2013).

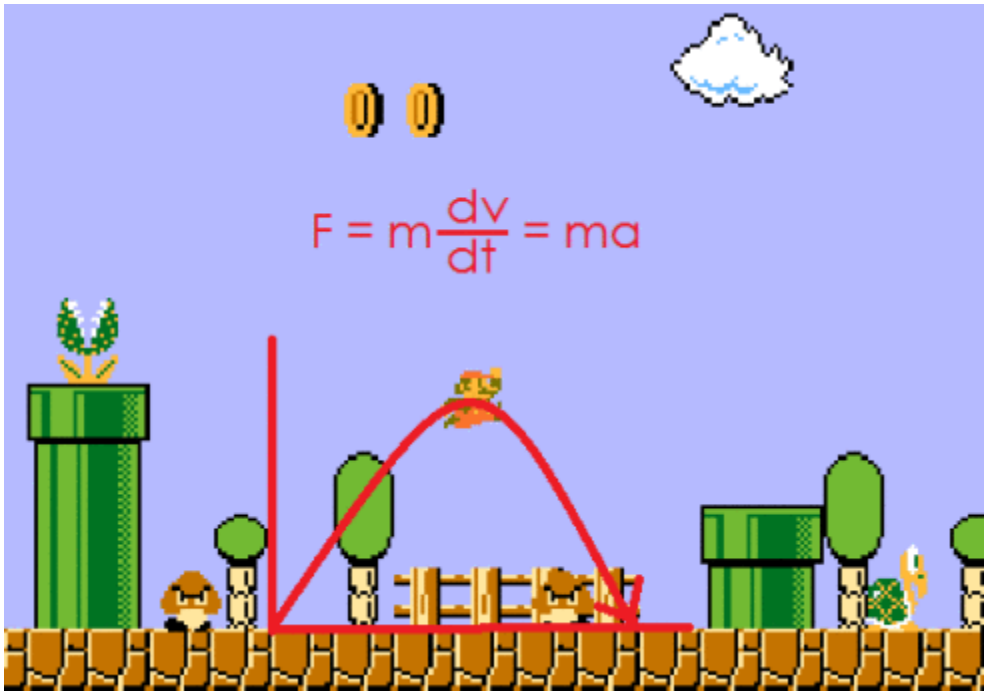

Figure 2. An example showing how high and how far the player character will jump with a running start (Tucker, 2014).

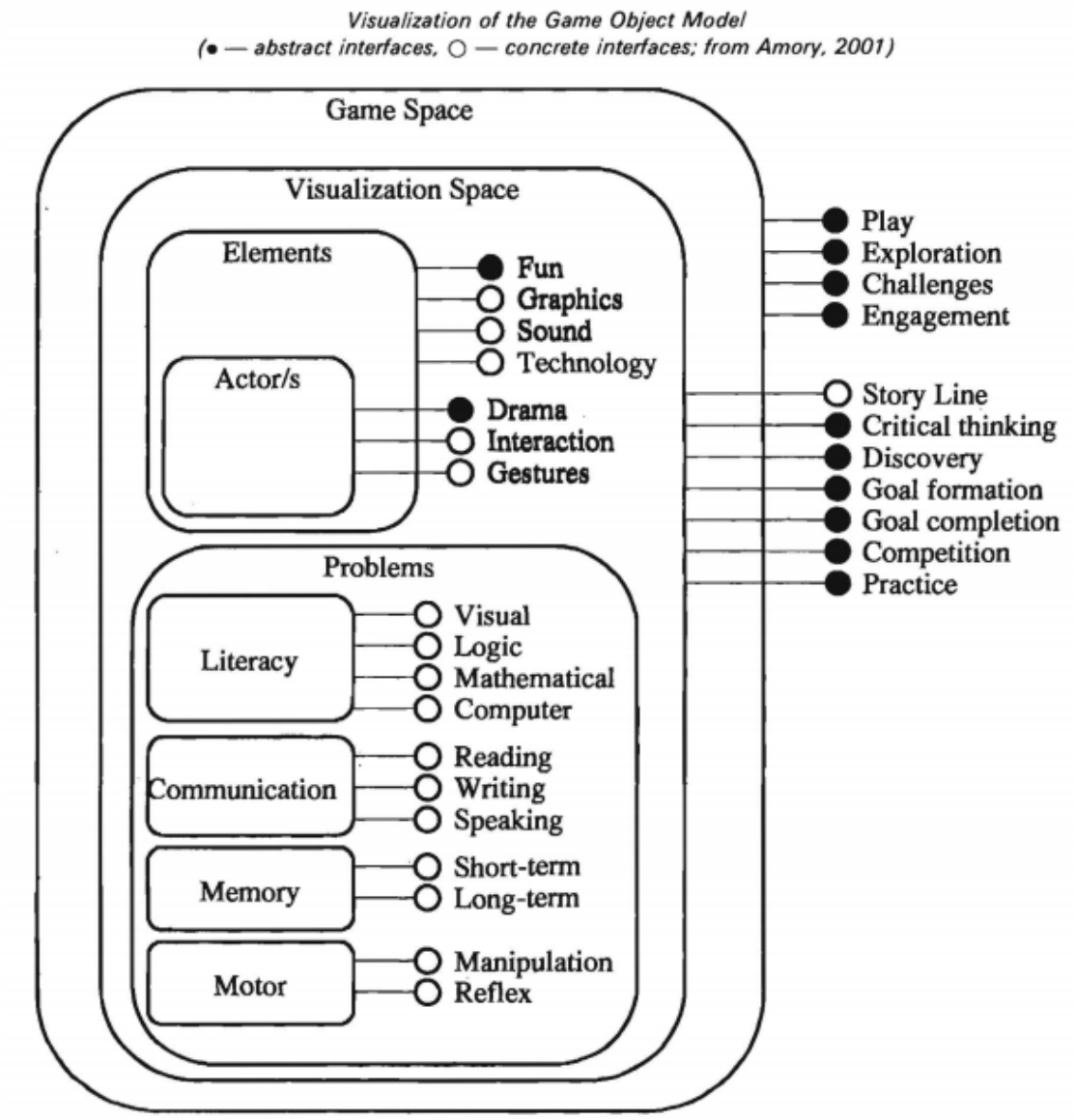

Figure 3. The Game Object Model from Amory and Seagram (2003).

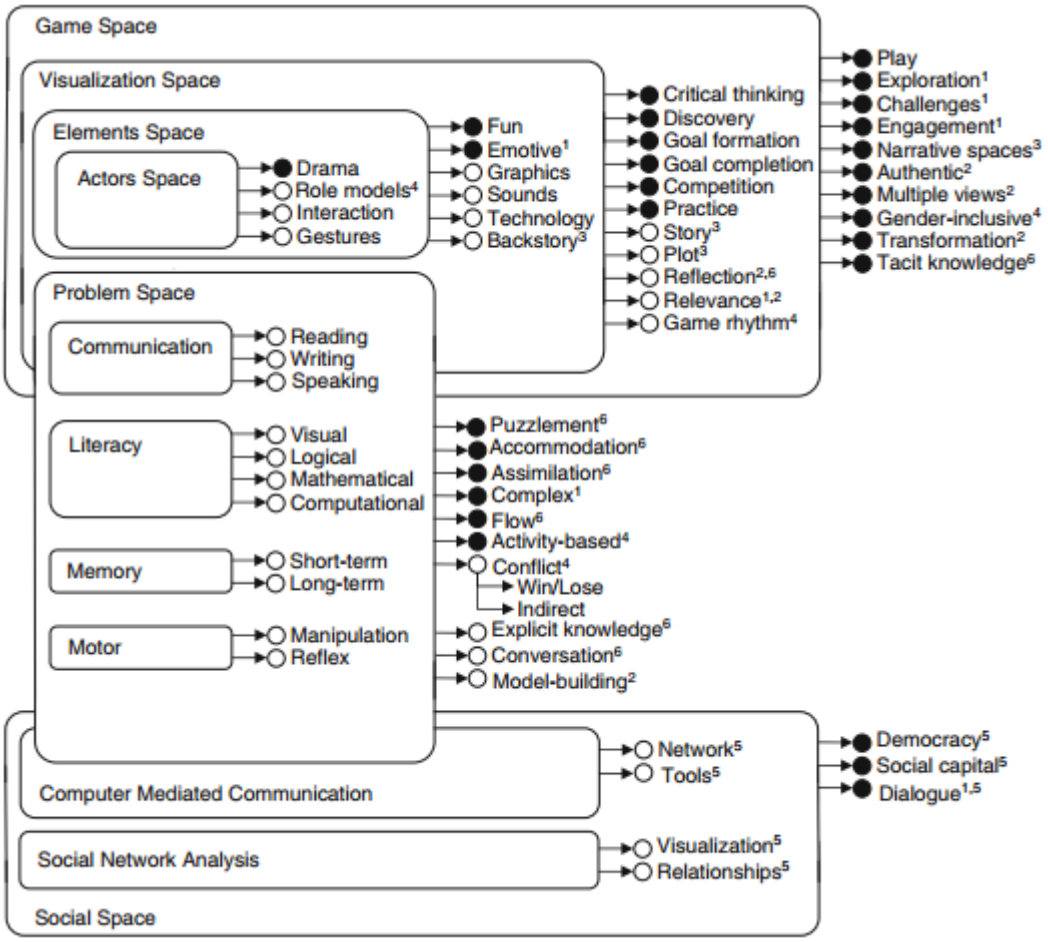

Figure 4. The Game Object Model II from Amory (2007). Numbered items correspond to core concepts: 1. Game Definition; 2. Authentic Learning; 3. Narrative; 4. Gender; 5. Social Collaboration; 6. Challenges-Puzzles-Quests.

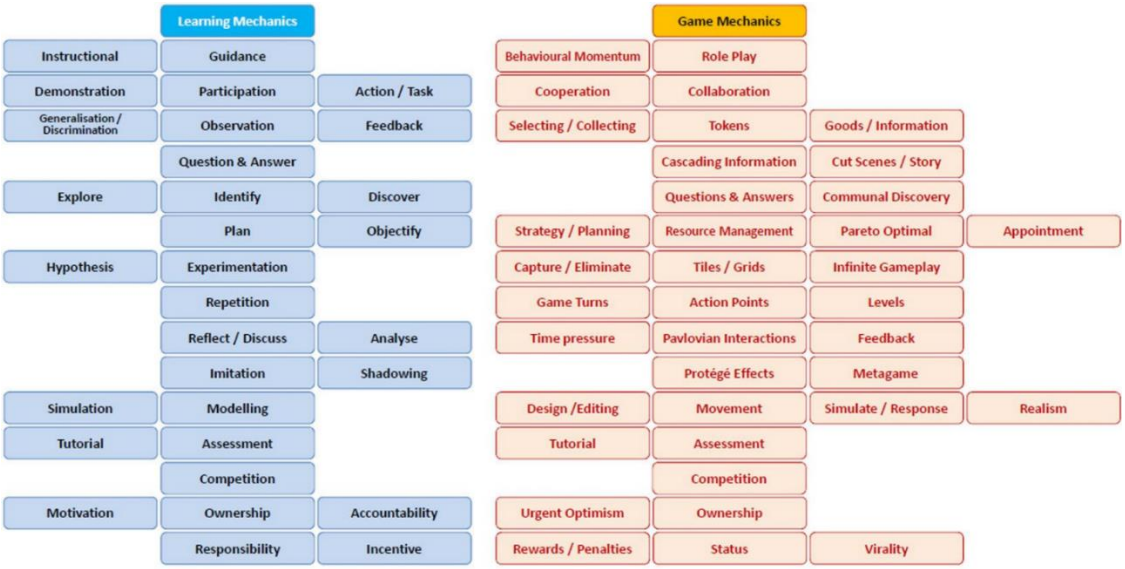

Figure 5. The LM-GM model (Arnab et. al., 2015).

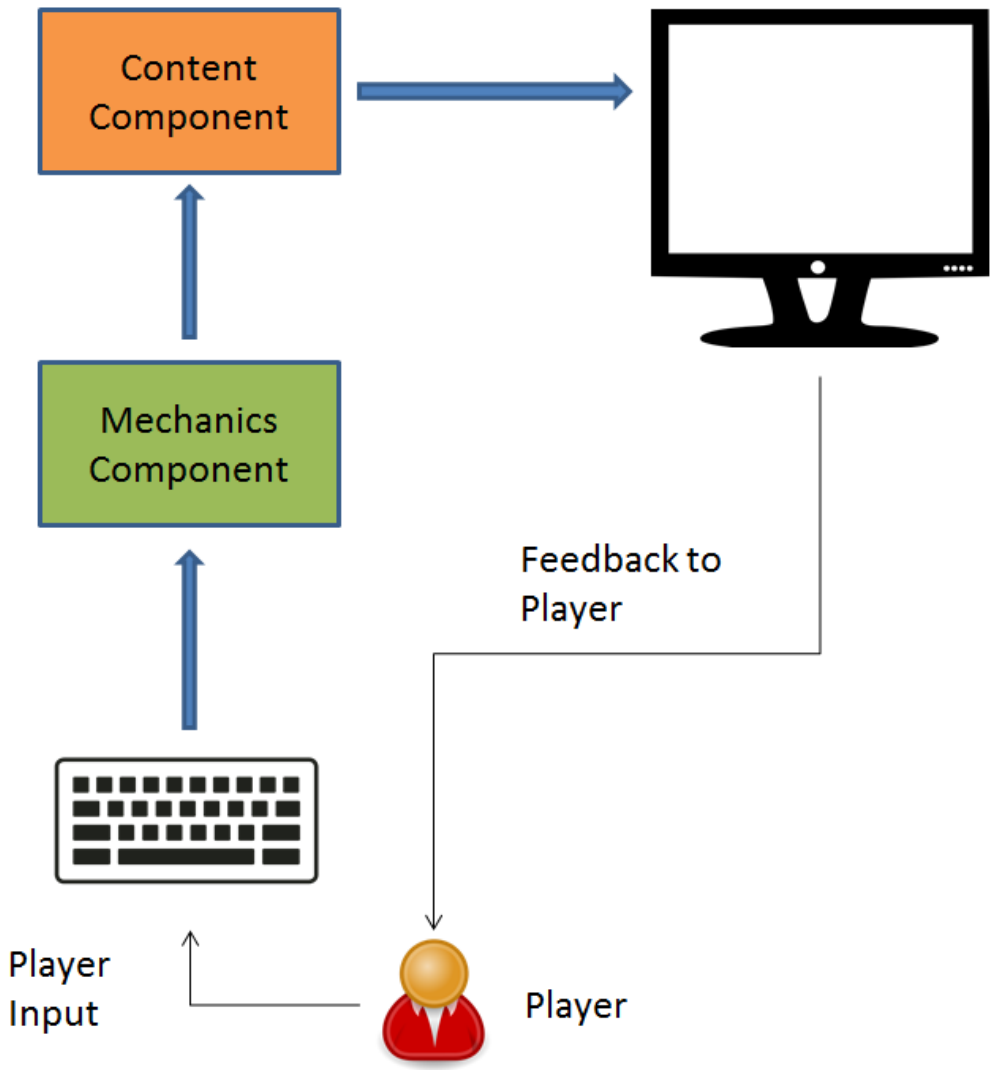

Figure 6. A diagram that outlines the current development model for game based learning.

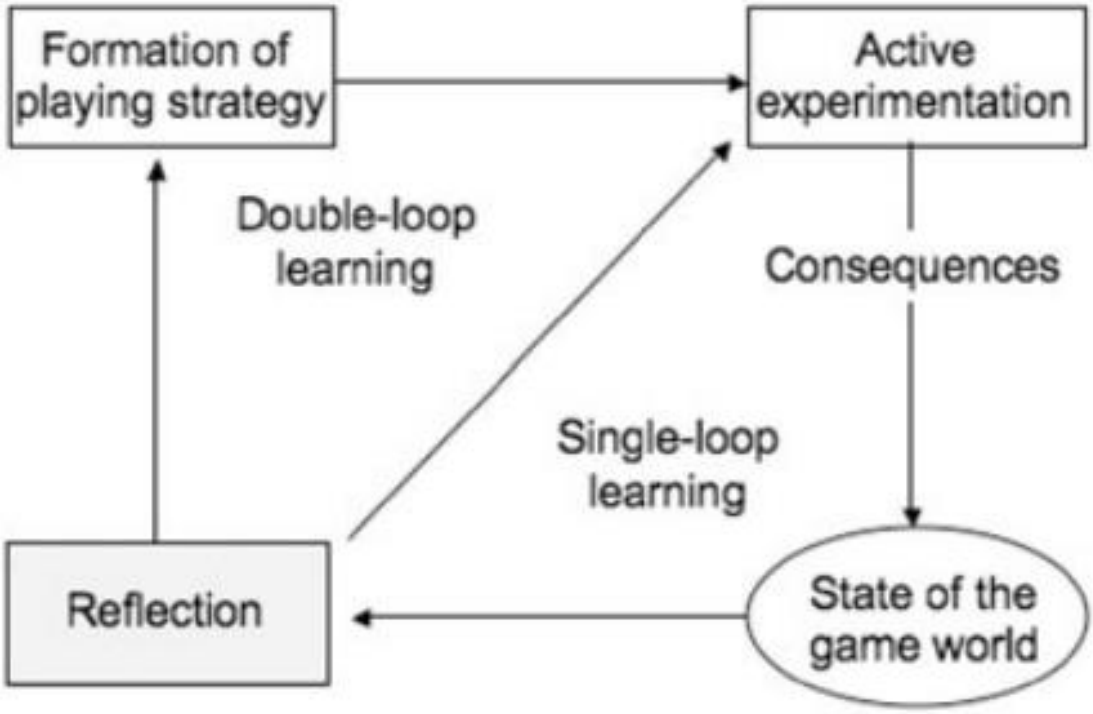

Figure 7. The Problem Based Gaming Model which describes the learning process in educational games (Kiili, 2007).

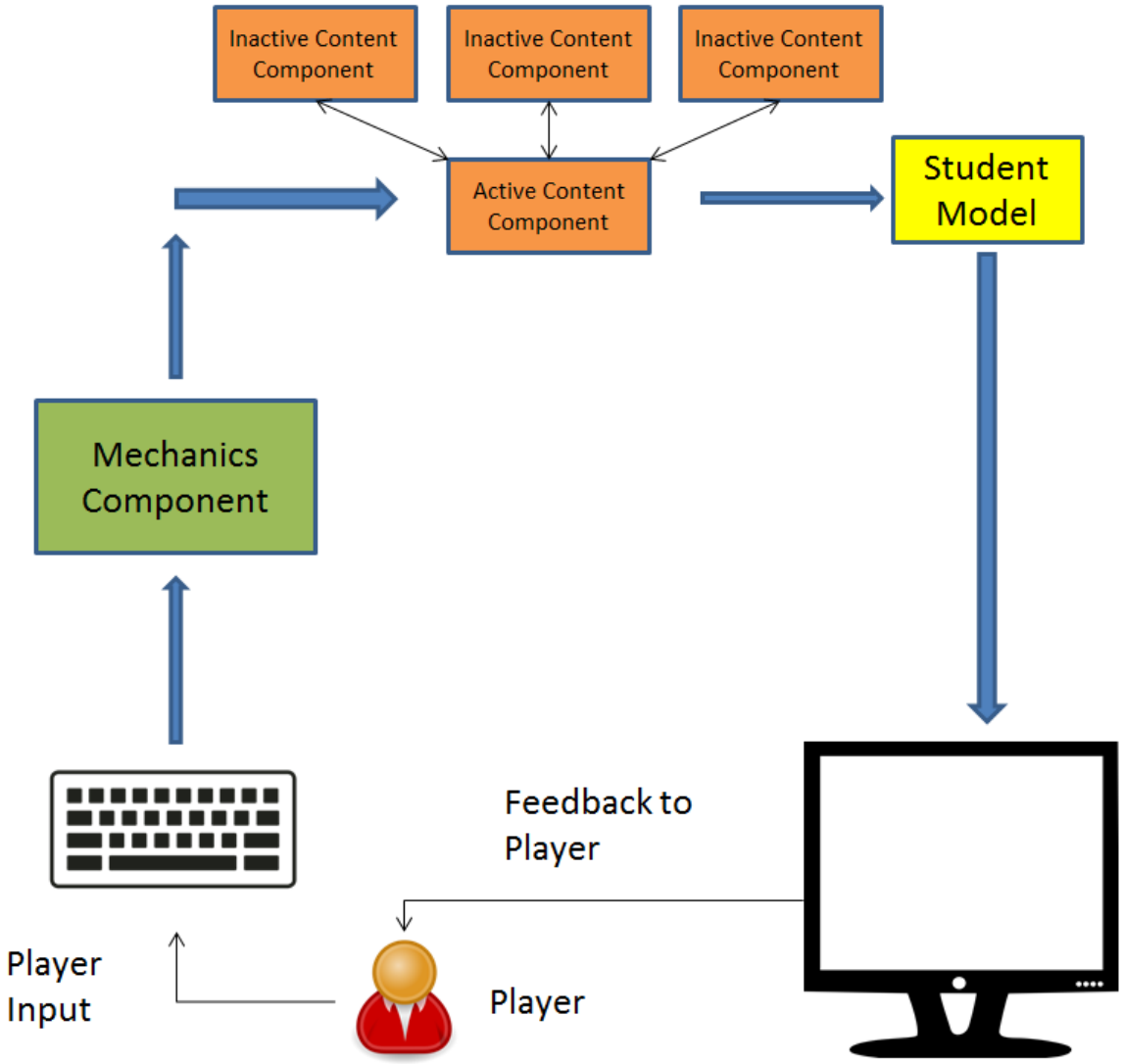

Figure 8. The CAGE model for educational game development.

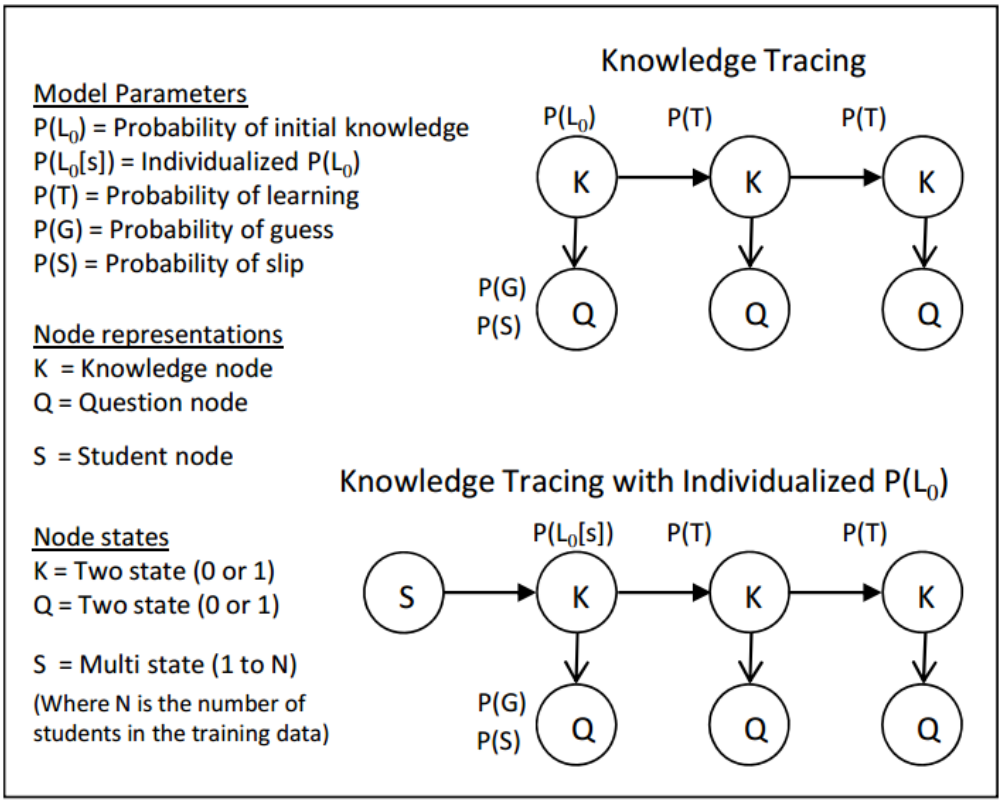

Figure 9. An example showing knowledge tracing as a Bayesian network (Pardos & Heffernan, 2010).

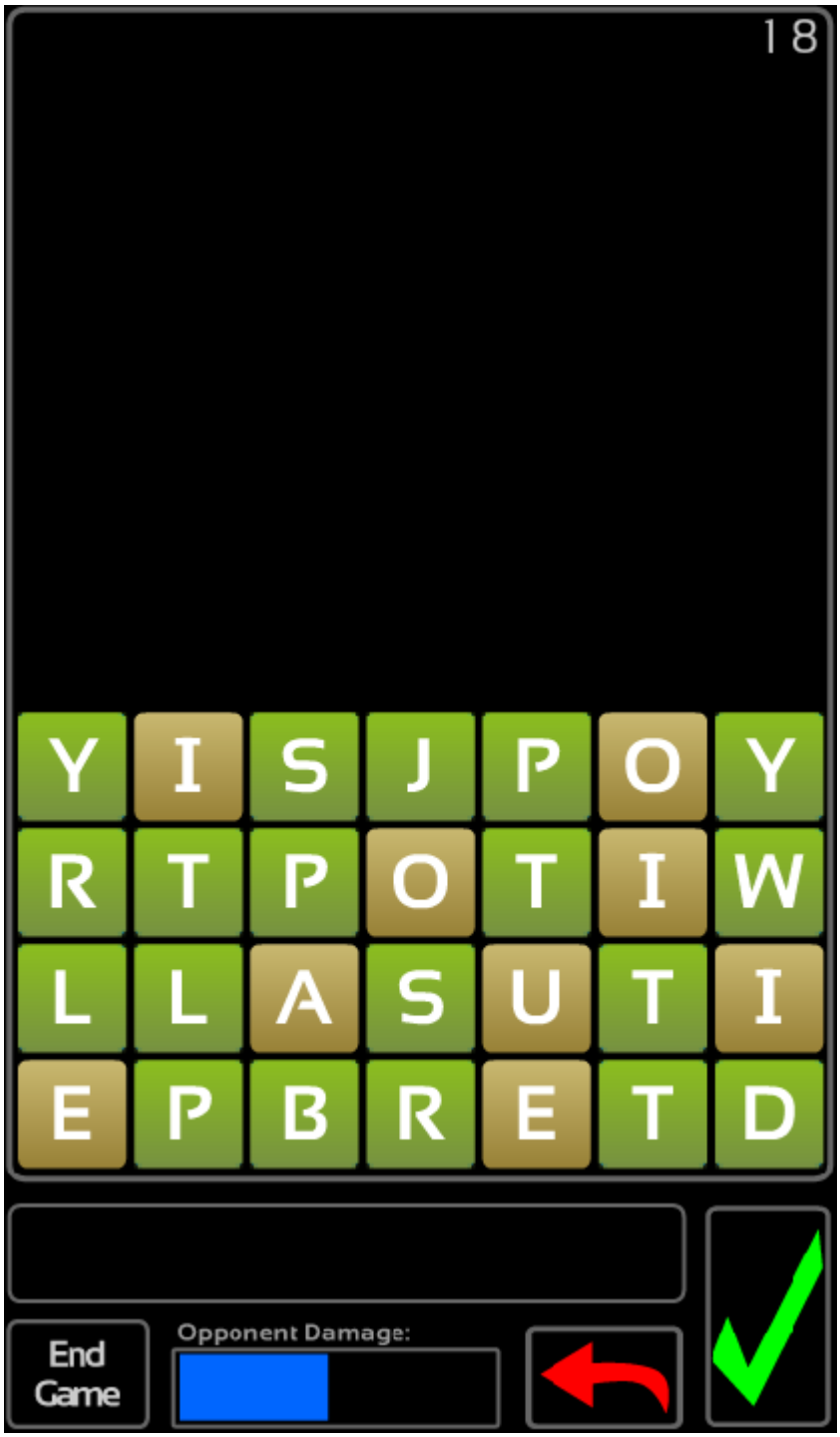

Figure 10. A screenshot of Word Fighters.

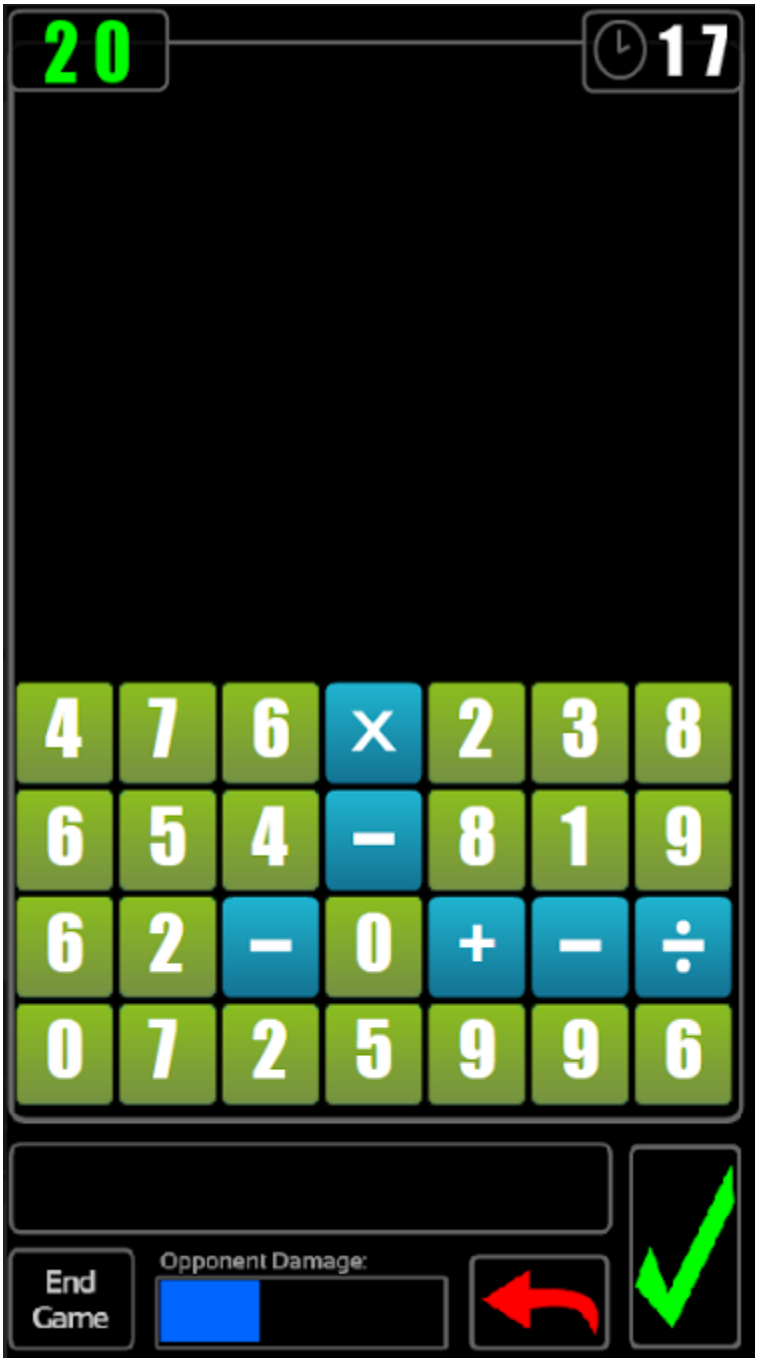

Figure 11. A screenshot of Math Fighters.

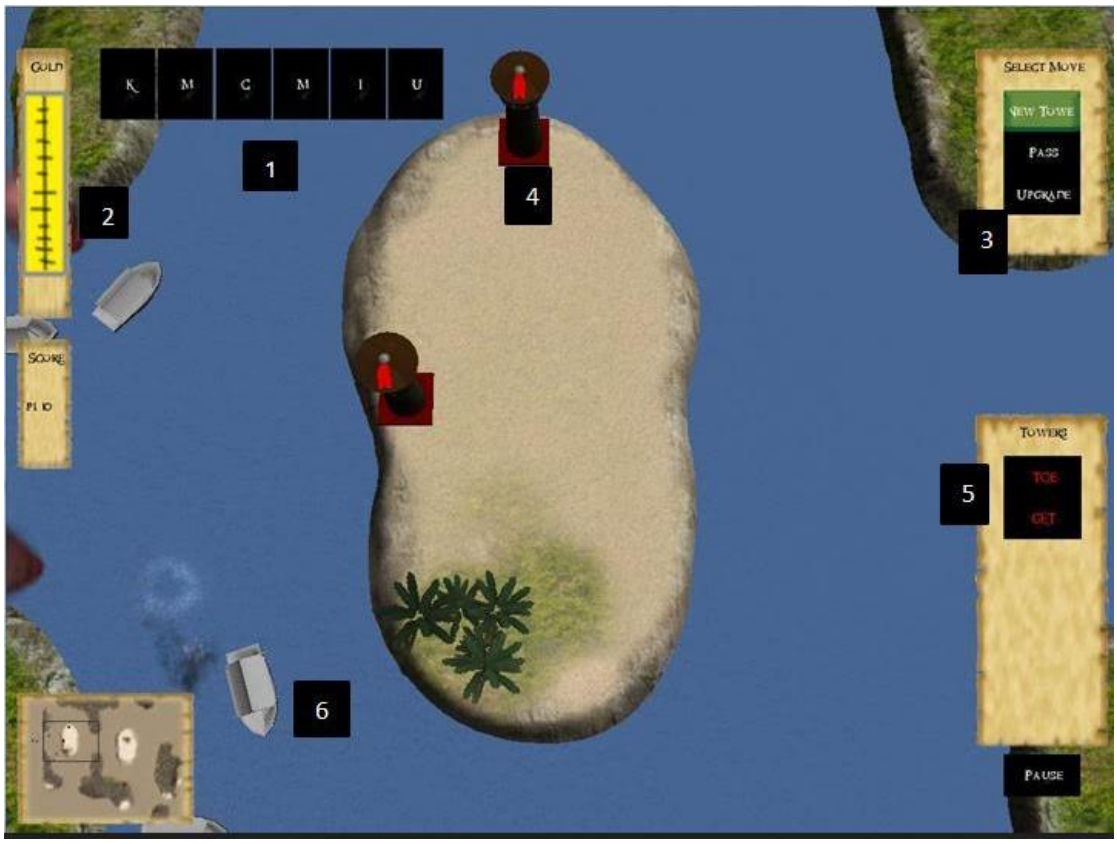

Figure 12. A screenshot of Word Towers. The numbers correspond to different key elements of the game and user interface: 1. The letter board the player can currently choose from; 2. The player's remaining health, referred to as Gold; 3. The buttons to make a new tower, discard the current letter board for new letters and upgraded the selected tower; 4. A tower the player has built; 5. The list of current towers; 6. An enemy ship.

Figure 13. A screenshot of Math Towers. Note the one new UI element compared to Word Towers near the top right of the screen. That is the target number that the tower equation must be equal to.

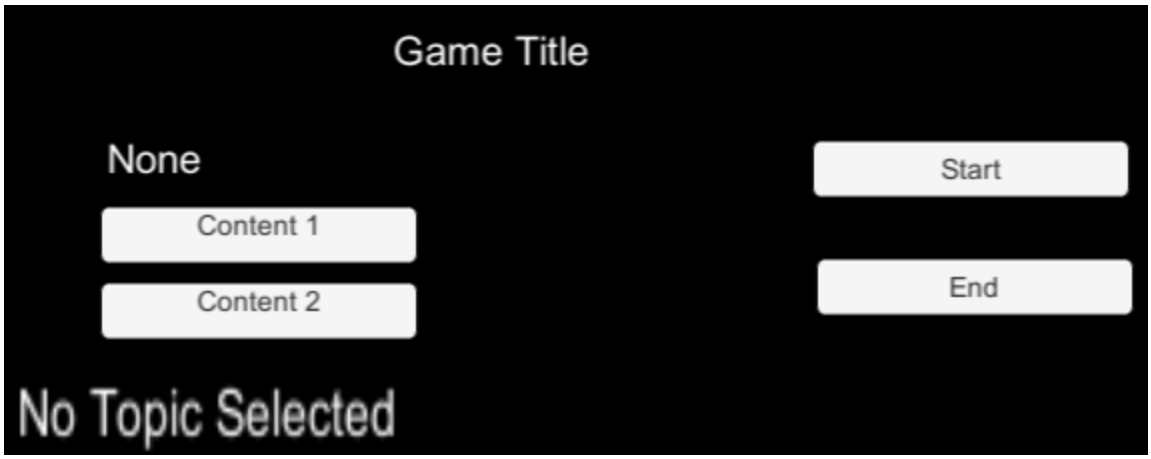

Figure 14. A screenshot of the default main menu provided with the CAGE framework.

Figure 15. A screenshot of Bean Man with health as the active content.

Figure 16. A screenshot of Bean Man where the player has run into a Flu enemy.

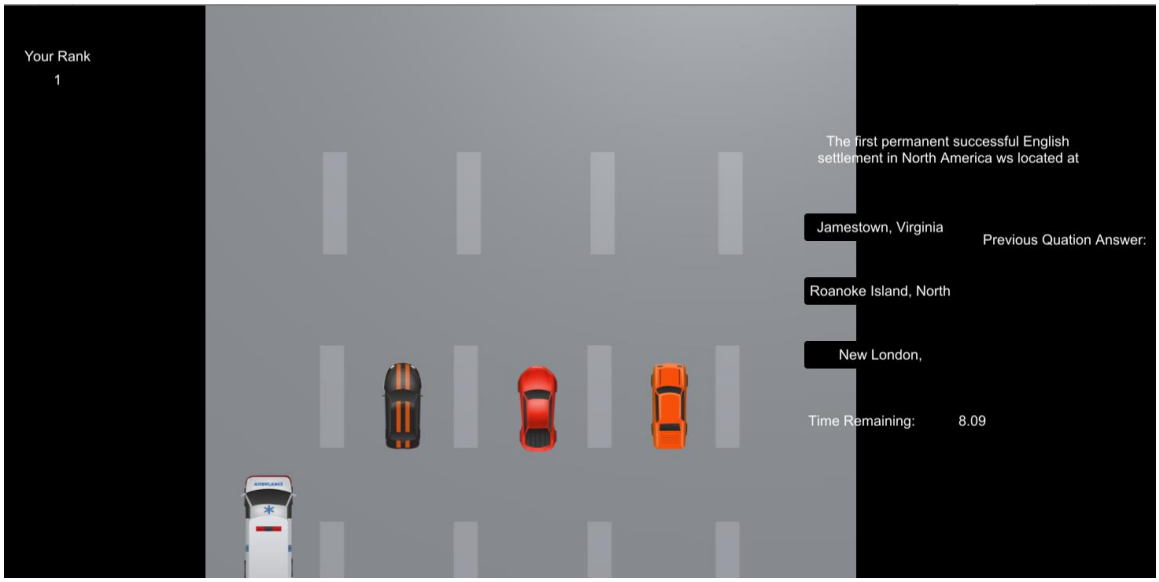

Figure 17. A screenshot of Car Racing at the start of the race. The current question is shown on the right along with the possible answers. History is the active content in this screenshot.

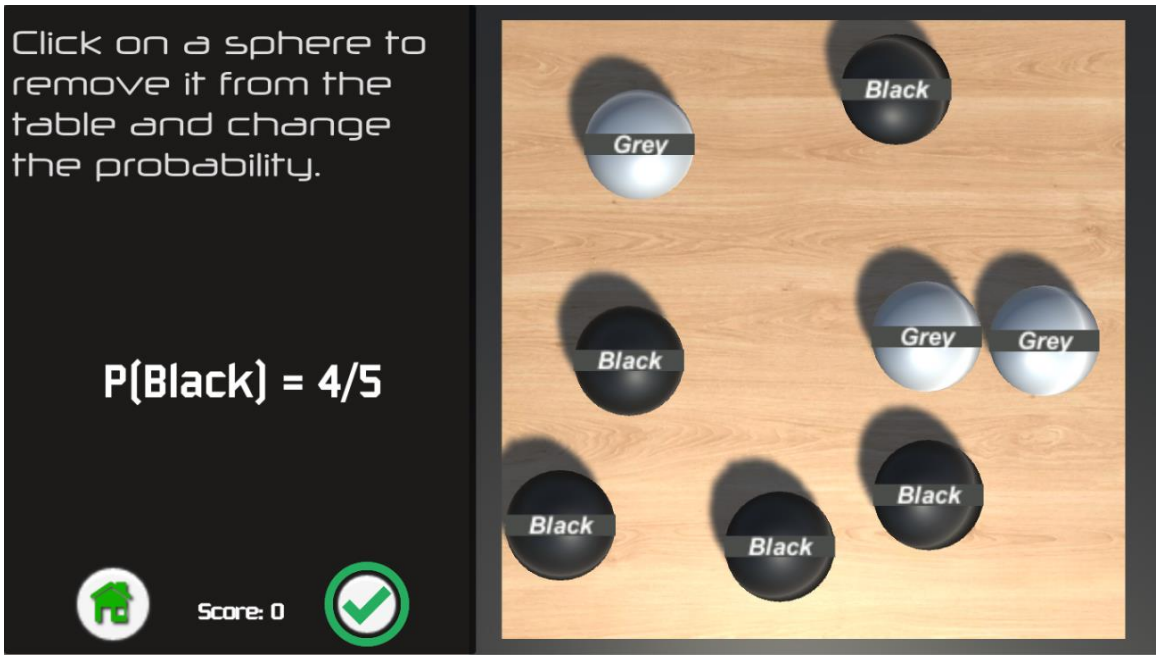

Figure 18. A screenshot of Fun-O-Sphere. The balls on the right side of the screen constantly move and bounce around. The probability content is active.

Figure 19. A screenshot of Fun-O-Sphere when chemistry is the active content.

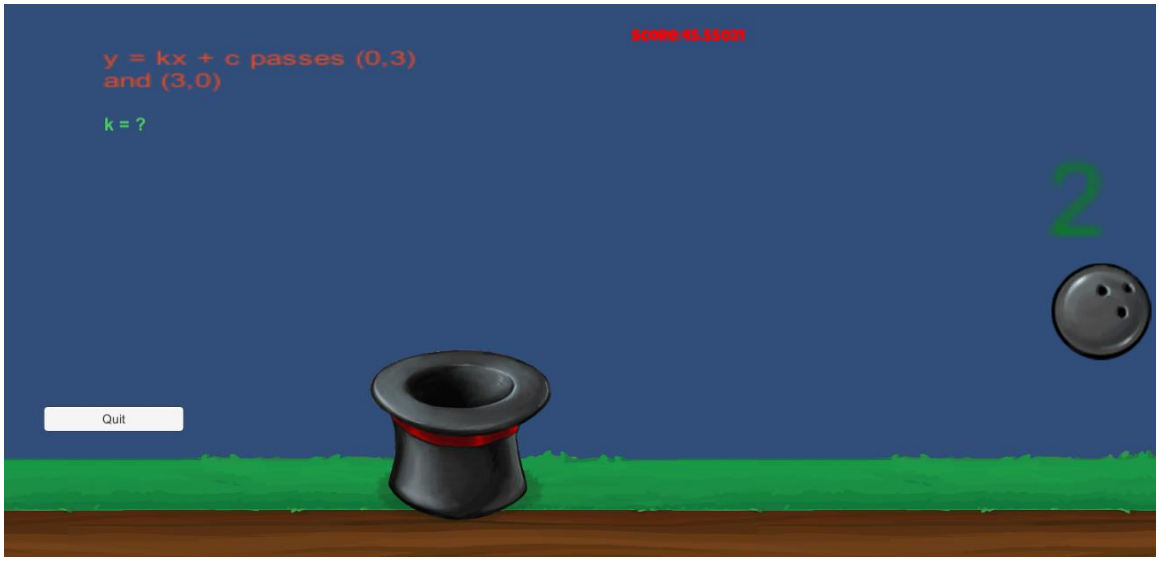

Figure 20. A screenshot of Hat Trick with Math as the active content.

Figure 21. A screenshot of Modern Scrabble. The missing letters formed the words shown on the top left, and were removed.

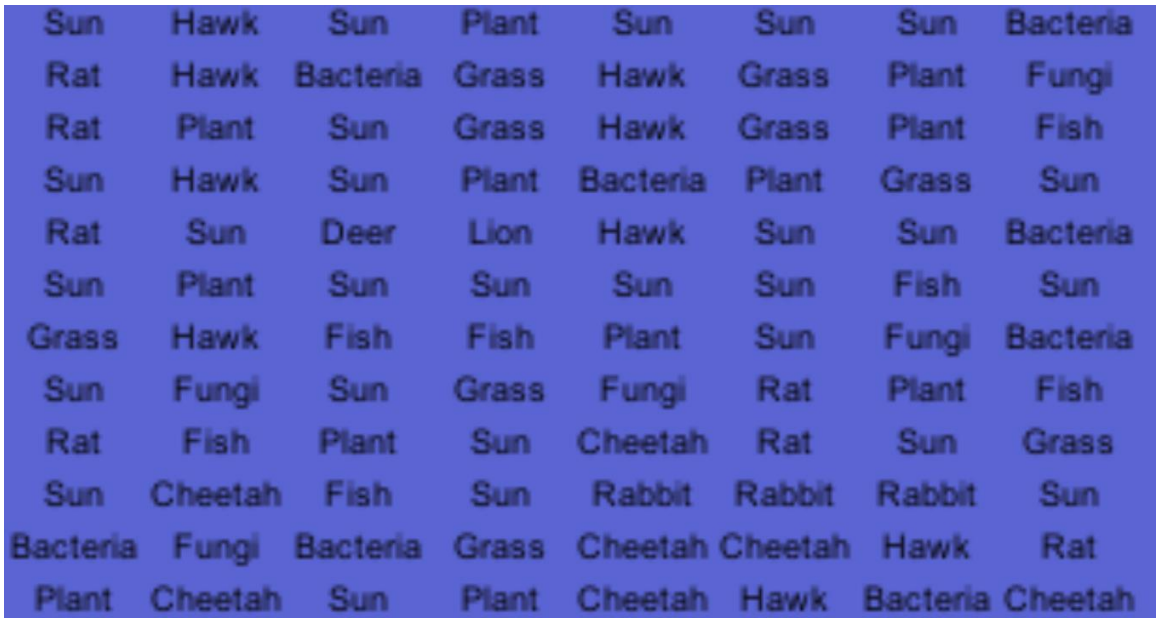

Figure 22. A screenshot of Modern Scrabble with biology as the active content.

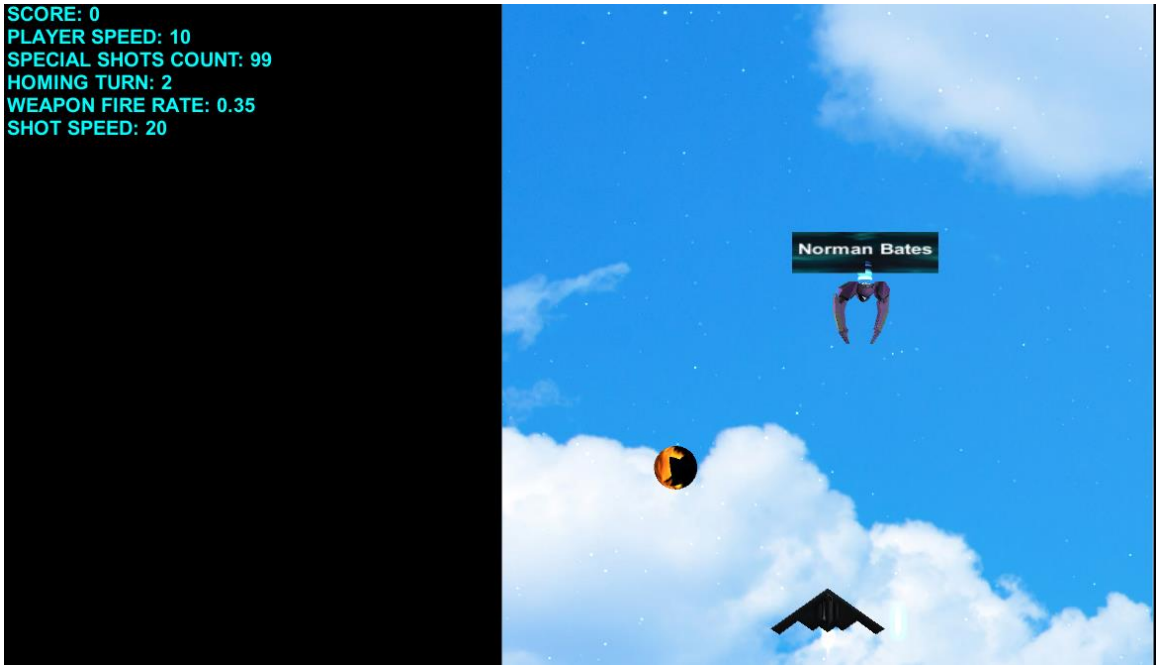

Figure 23. A screenshot of Operation VIP Extraction with social science as the active content.

Figure 24. A screenshot of Operation VIP Extraction with biology as the active content.

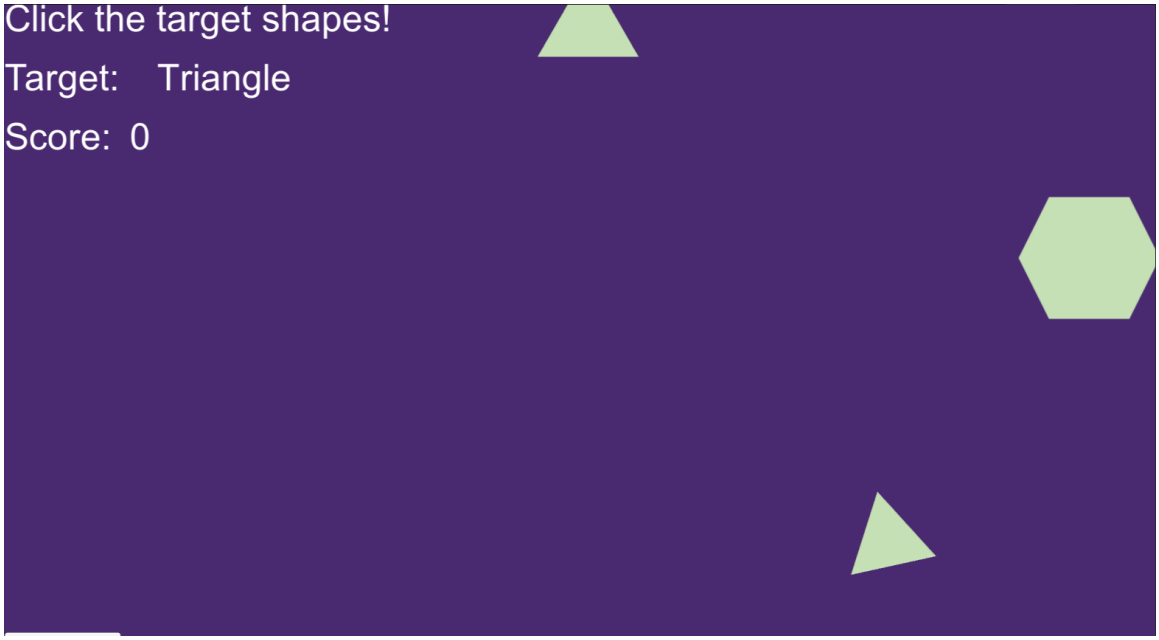

Figure 25. A screenshot of Pattern Recognition Training with geometry as the active content.

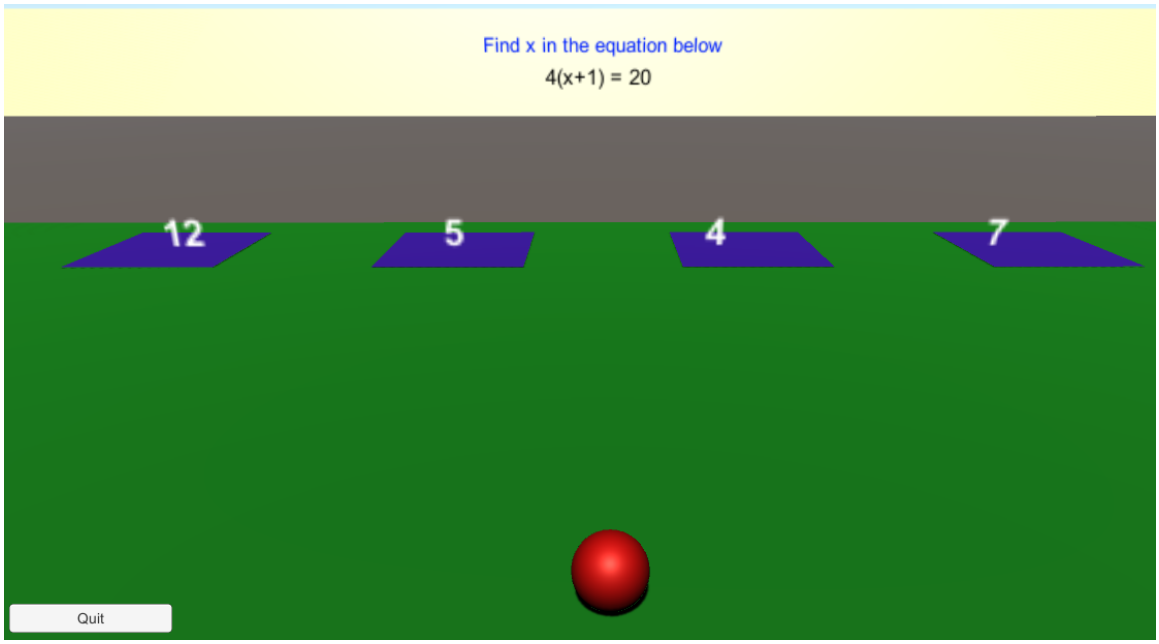

Figure 26. A screenshot of Pedestals with Linear Equations as the active content.

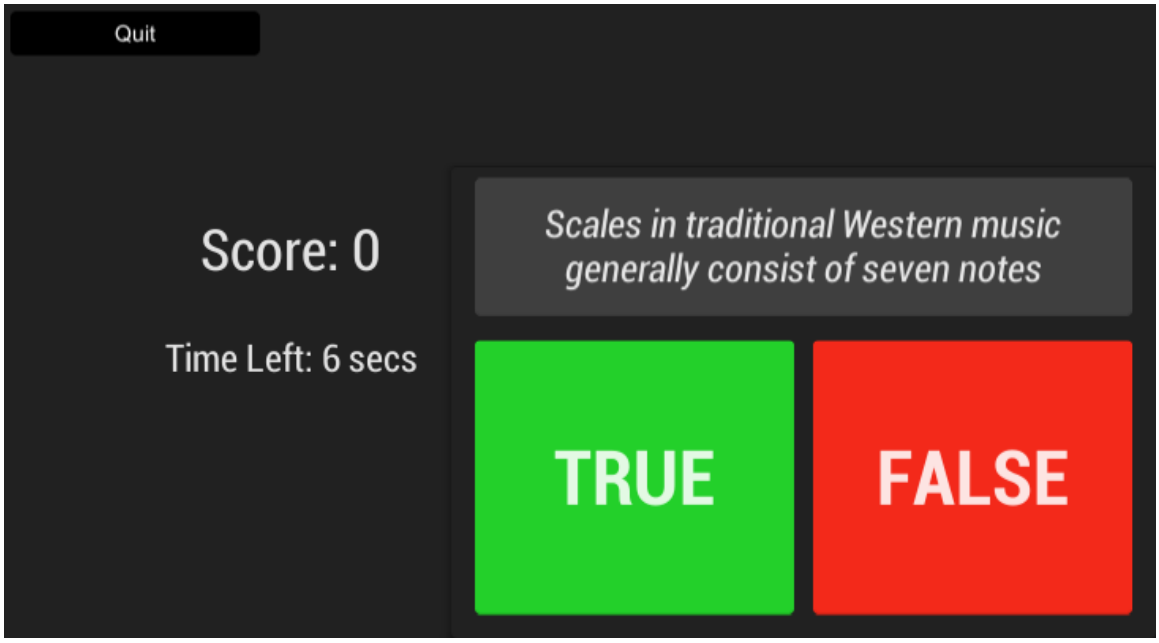

Figure 27. A screenshot of Quiz Up!. All questions are true/false questions. Western Classical music is the active content in this screenshot.

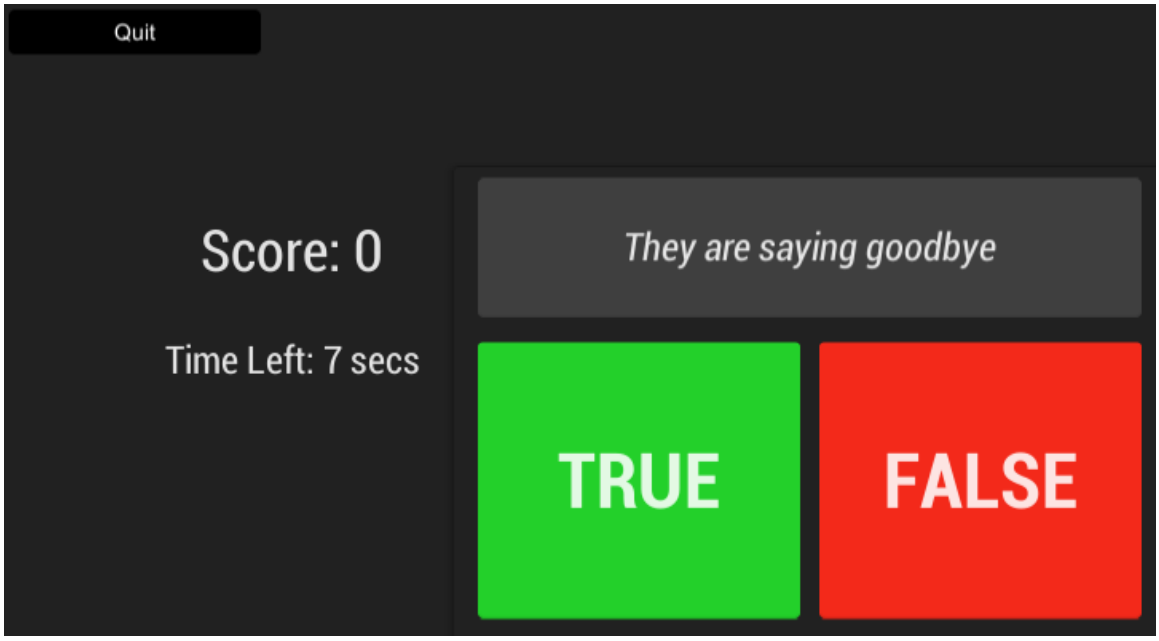

Figure 28. A screenshot of Quiz Up! with the French content active. On this screen an audio clip of a male voice and a female voice saying "Au Revoir" is repeatedly playing.

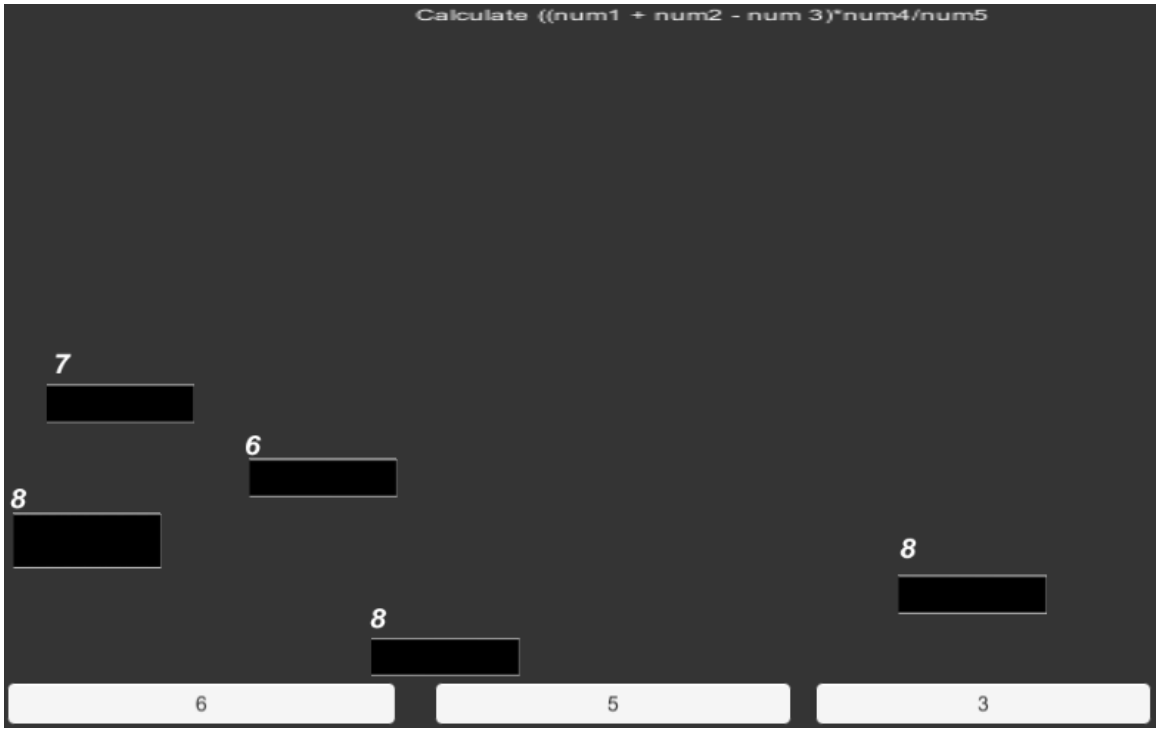

Figure 29. A screenshot of Think Fast. The black boxes quickly move left and right. For the math problem given at the top of the screen, the number on the top black box is num1, the next one down is num2, etc.

Figure 30. A screenshot of Think Fast with Spanish as the active content.

Figure 31. A screenshot of Titanical with history as the active content.

Figure 32. A screenshot of Bingo Bingo with vocabulary as the active content.

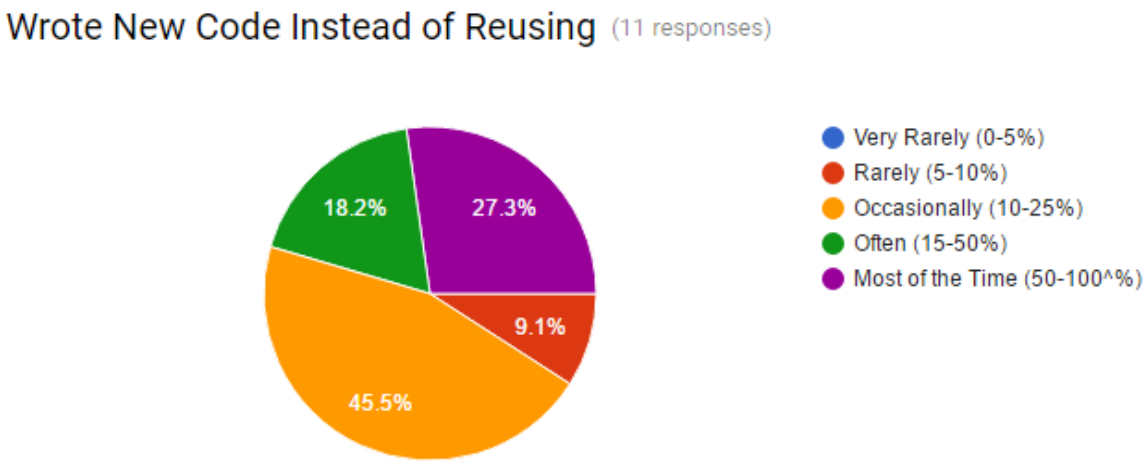

Figure 33. A pie chart showing the distribution of times that participants wrote new code instead of reusing old code when they were asked to reuse the old code.

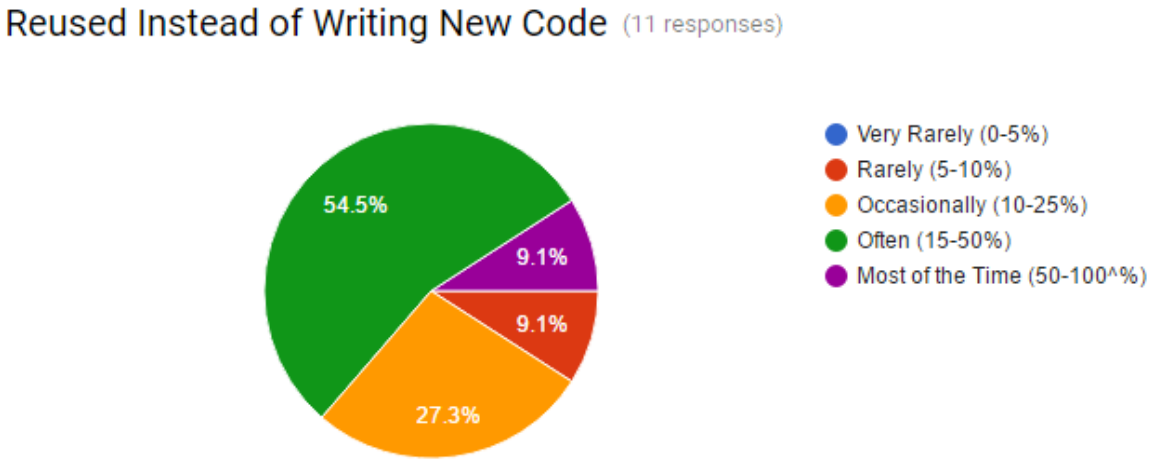

Figure 34. A pie chart showing the distribution of times that participants reused old code when they were supposed to write new code.

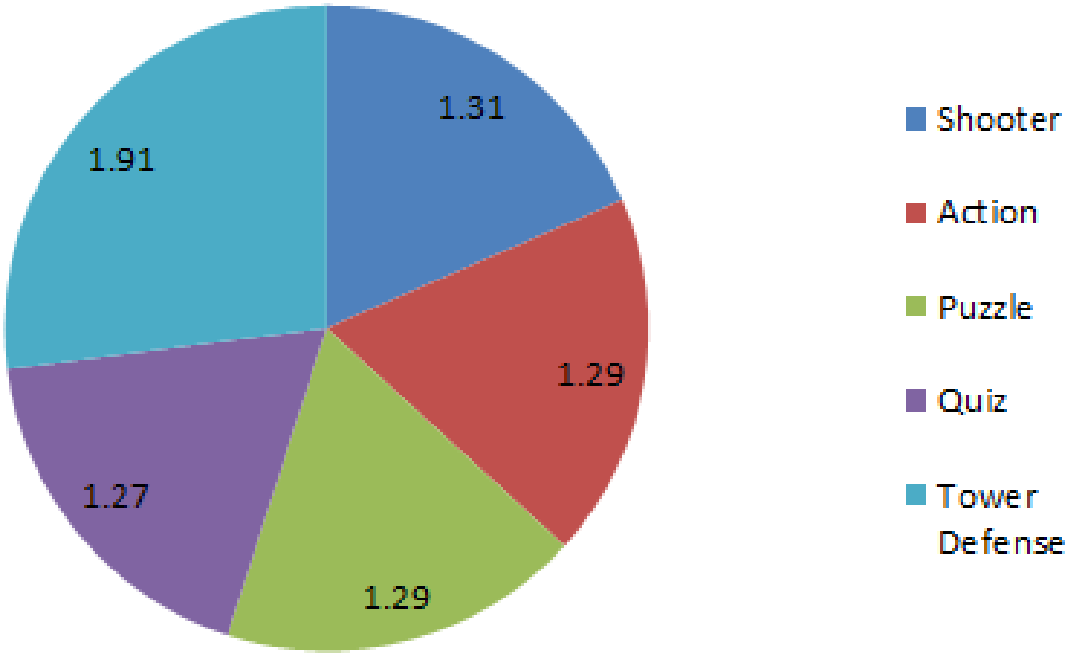

Figure 35. A figure showing engagement of CAGE games by genre.

REFERENCES

Agresti, W. W. (2011). Software reuse: developers' experiences and perceptions. Journal of Software Engineering and Applications, 4(01), 48.

Amory, A., & Seagram, R. (2003). Educational game models: conceptualization and evaluation. South African Journal of Higher Education,17(2), 206-217.

Amory, A. (2007). Game object model version II: a theoretical framework for educational game development. Educational Technology Research and Development, 55(1), 51-77.

Amresh, A., Clarke, D., Beckwith, D. (2014). GameScapes and SimApps: New techniques for integrating rich narratives with game mechanics. 8th European Conference on Games Based Learning: ECGBL2014, 18.

Arnab, S., Lim, T., Carvalho, M. B., Bellotti, F., Freitas, S., Louchart, S., ... & De Gloria, A. (2015). Mapping learning and game mechanics for serious games analysis. British Journal of Educational Technology, 46(2), 391-411.

Atkinson, R. C., & Paulson, J. A. (1972). An approach to the psychology of instruction. *Psychological Bulletin*, *78*(1), 49.

Baldwin, T. T., & Ford, J. K. (1988). Transfer of training: A review and directions for future research. Personnel psychology, 41(1), 63-105.

Baron, T., & Amresh, A. (2015). Word Towers: Assessing Domain Knowledge With Non-Traditional Genres. In European Conference on Games Based Learning (p. 638). Academic Conferences International Limited.

Baron, T., Heath, C., & Amresh, A. Towards a Context Agnostic Platform for Design and Assessment of Educational Games. In 10th European Conference on Games Based Learning: ECGBL 2016 (p. 34).

Bartle, R. (1996). Hearts, clubs, diamonds, spades: Players who suit MUDs. *Journal of MUD Research, 1*(1), 19.

Bénabou, R., & Tirole, J. (2003). Intrinsic and extrinsic motivation. *The Review of Economic Studies, 70*(3), 489-520. doi:10.1111/1467-937X.00253

Bloom, S. (2009). Game-based learning. *Professional Safety, 54*(7), 18-21.

Charsky, D., & Mims, C. (2008). Integrating commercial off-the-shelf video games into school curriculums. *TechTrends*, *52*(5), 38-44.

Chen, K. C., & Jang, S. J. (2010). Motivation in online learning: Testing a model of self-determination theory. Computers in Human Behavior, 26(4), 741-752.

Collins, A., & Ferguson, W. (1993). Epistemic forms and epistemic games: Structures and strategies to guide inquiry. Educational psychologist, 28(1), 25-42.

Cormier, S. M., & Hagman, J. D. (Eds.). (2014). Transfer of learning: Contemporary research and applications. Academic Press.

Cox, Brad J. "Object-oriented programming: an evolutionary approach." (1986).

Craig, S., Graesser, A., Sullins, J., & Gholson, B. (2004). Affect and learning: an exploratory look into the role of affect in learning with AutoTutor. Journal of educational media, 29(3), 241-250.

Craig, S. D., D'Mello, S., Witherspoon, A., & Graesser, A. (2008). Emote aloud during learning with AutoTutor: Applying the Facial Action Coding System to cognitive–affective states during learning. Cognition and Emotion, 22(5), 777-788.

Csikszentmihalyi, M. (1996). Flow and the psychology of discovery and invention. *New York: Harper Collins*.

Cuban, L. (1986). Teachers and machines: The classroom use of technology since 1920. Teachers College Press.

Dai, D. Y., & Wind, A. P. (2011). Computer games and opportunity to learn. In Implications for teaching students from low socioeconomic backgrounds (pp. 477-500). Information Age Publishing Charlotte, NC.

Deci, E. L. (1971). Effects of externally mediated rewards on intrinsic motivation. Journal of personality and Social Psychology, 18(1), 105.

Deci, E. L. (1972). Intrinsic motivation, extrinsic reinforcement, and inequity. *Journal of Personality and Social Psychology, 22*(1), 113-120. doi:10.1037/h0032355

Deci, E. L., & Ryan, R. M. (2000). The" what" and" why" of goal pursuits: Human needs and the self-determination of behavior. Psychological inquiry, 11(4), 227-268.

Del Blanco, Á., Torrente, J., Marchiori, E. J., Martínez-Ortiz, I., Moreno-Ger, P., & Fernández-Manjón, B. (2012). A Framework for Simplifying Educator Tasks Related to the Integration of Games in the Learning Flow. *Educational Technology & Society*, *15*(4), 305-318.

Delwiche, A. (2006). Massively multiplayer online games (MMOs) in the new media classroom. *Journal of Educational Technology & Society*, *9*(3), 160-172.

Denis, G., & Jouvelot, P. (2005, June). Motivation-driven educational game design: applying best practices to music education. In Proceedings of the 2005 ACM SIGCHI International Conference on Advances in computer entertainment technology (pp. 462-465). ACM.

Ebbinghaus, H. (1913). Memory: A contribution to experimental psychology University Microfilms.

Ekman, P., & Friesen, W. V. (1977). Facial action coding system.

English, M. C. W., & Visser, T. A. W. (2014). Exploring the repetition paradox: The effects of learning context and massed repetition on memory. *Psychonomic Bulletin & Review, 21*(4), 1026-1032. doi:10.3758/s13423-013-0566-1

Freeman, B., & Higgins, K. (2016). A randomised controlled trial of a digital learning game in the context of a design-based research project. International Journal of Technology Enhanced Learning, 8(3-4), 297-317.

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine learning*, *29*(2-3), 131-163.

Gagne, R. M. (1977). The Conditions of Learning, 3rd. New York: Holt, Rinehart and Winston.

Gagné, M., & Deci, E. L. (2005). Self-determination theory and work motivation. *Journal of Organizational behavior*, *26*(4), 331-362.

Graesser, A. C. (2017). Reflections on Serious Games. In Instructional Techniques to Facilitate Learning and Motivation of Serious Games (pp. 199-212). Springer International Publishing.

Gokhale, A. A. (1995). Collaborative learning enhances critical thinking.

Habgood, M. J., Ainsworth, S. E., & Benford, S. (2005). Endogenous fantasy and learning in digital games. Simulation & Gaming, 36(4), 483-498.

Hannifin, R. D., & Vermillion, J. R. (2008). Technology in the classroom. In 21st century education: A reference handbook, 2, 209-218.

Harding, E. J., Paul, E. S., & Mendl, M. (2004). Animal behaviour: cognitive bias and affective state. Nature, 427(6972), 312-312.

Horner, A. J., & Henson, R. N. (2008). Priming, response learning and repetition suppression. Neuropsychologia, 46(7), 1979-1991. doi:http://dx.doi.org.ezproxy1.lib.asu.edu/10.1016/j.neuropsychologia.2008.01.018

Kalyuga, S. (2011). Cognitive load theory: How many types of load does it really need?. Educational Psychology Review, 23(1), 1-19.

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

Kiili, K. (2007). Foundation for problem-based gaming. British journal of educational technology, 38(3), 394-404.

Kort, B., Reilly, R., & Picard, R. W. (2001). An affective model of interplay between emotions and learning: Reengineering educational pedagogy-building a learning companion. In Advanced Learning Technologies, 2001. Proceedings. IEEE International Conference on (pp. 43-46). IEEE.

Kuhl, B. A., & Anderson, M. C. (2011). More is not always better: Paradoxical effects of repetition on semantic accessibility. *Psychonomic Bulletin & Review, 18*(5), 964-972.

Kong, J. S. L., Kwok, R. C. W., & Fang, Y. (2012). The effects of peer intrinsic and extrinsic motivation on MMOG game-based collaborative learning. Information & Management, 49(1), 1-9.

Leemkuil, H., & de Jong, T. (2011). Instructional support in games. Computer games and instruction, 353, 370.

Lepper, M. R., Greene, D., & Nisbett, R. E. (1973). Undermining children's intrinsic interest with extrinsic reward: A test of the "overjustification" hypothesis. Journal of Personality and Social Psychology, 28(1), 129-137. doi:10.1037/h0035519

Lester, J. C., Ha, E. Y., Lee, S. Y., Mott, B. W., Rowe, J. P., & Sabourin, J. L. (2013). Serious games get smart: intelligent game-based learning environments. AI Magazine, 34(4), 31-45.

Malone, T. W., & Lepper, M. R. (1987). Making learning fun: A taxonomy of intrinsic motivations for learning. Aptitude, learning, and instruction, 3(1987), 223-253.

Marczewski, A. (2013). A Player Type Framework for Gamification Design. Retrieved from http://www.gamified.uk/user-types/

Mateas, M., & Stern, A. (2005). Structuring content in the façade interactive drama architecture. *Aiide,* 93-98.

Mayer, R. E., & Moreno, R. (2003). Nine ways to reduce cognitive load in multimedia learning. Educational psychologist, 38(1), 43-52.

Mayer, R. E. (2014). Computer games for learning: An evidence-based approach. MIT Press.

McFarlane, A., Sparrowhawk, A., & Heald, Y. (2002). *Report on the educational use of games*. TEEM (Teachers evaluating educational multimedia), Cambridge.

McNamara, D. S., Jackson, G. T., & Graesser, A. C. (2010). Intelligent tutoring and games (ITaG). *Gaming for classroom-based learning: Digital role-playing as a motivator of study*, 44-65.

McNeill, D. (2004). Using Digital Experiential Learning to Deliver Corporate Policy Training. *Learning Solutions Magazine.*

Mehren, E. (1985, Jul 31, 1985). Intrinsic motivation lost creativity turns into psychologist's life study. *Los Angeles Times (Pre-1997 Fulltext)*, pp. 1.

Mei, H., Chen, F., Feng, Y. D., & Yang, J. (2003). ABC: An architecture based, component oriented approach to software development. Journal of Software, 14(4), 721-732.

Mislevy, R. J., Almond, R. G., & Lukas, J. F. (2003). A brief introduction to evidence-centered design. *ETS Research Report Series*, *2003*(1), i-29.

Mislevy, R. J., Oranje, A., Bauer, M. I., von Davier, A., Hao, J., Corrigan, S., Hoffman, E., DiCerbo, K. & John, M. (2014). Psychometric considerations in game-based assessment. *GlassLab Report*.

Morelli, T., Foley, J., & Folmer, E. (2010, October). Vi-bowling: a tactile spatial exergame for individuals with visual impairments. In Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility (pp. 179-186). ACM.

Moreno-Ger, P., Burgos, D., Martínez-Ortiz, I., Sierra, J. L., & Fernández-Manjón, B. (2008). Educational game design for online education. *Computers in Human Behavior*, *24*(6), 2530-2540.

Morrison, D., & Collins, A. (1996). Epistemic fluency and constructivist learning environments. Constructivist learning environments, 107-119.

Nelson, B. C. (2007). Exploring the use of individualized, reflective guidance in an educational multi-user virtual environment. Journal of Science Education and Technology, 16(1), 83-97.

Niemczyk, M. C., & Savenye, W. C. (2001). The Relationship of Student Motivation and Self-Regulated Learning Strategies to Performance in an Undergraduate Computer Literacy Course.

Ocumpaugh, J., Baker, R., Gowda, S., Heffernan, N., & Heffernan, C. (2014). Population validity for Educational Data Mining models: A case study in affect detection. *British Journal of Educational Technology*, *45*(3), 487-501.

Paas, F., Tuovinen, J. E., Tabbers, H., & Van Gerven, P. W. (2003). Cognitive load measurement as a means to advance cognitive load theory. Educational psychologist, 38(1), 63-71.

Pardos, Z. A., & Heffernan, N. T. (2010, June). Modeling individualization in a bayesian networks implementation of knowledge tracing. In *International Conference on User Modeling, Adaptation, and Personalization* (pp. 255-266). Springer Berlin Heidelberg.

Peirce, N., Conlan, O., & Wade, V. (2008, November). Adaptive educational games: Providing non-invasive personalised learning experiences. In Digital Games and Intelligent Toys Based Education, 2008 Second IEEE International Conference on (pp. 28-35). IEEE.

Reiss, S. (2012). Intrinsic and extrinsic motivation. *Teaching of Psychology, 39*(2), 152-156. Retrieved from http://top.sagepub.com/content/39/2/152.full.pdf+html

Riedl, M. O., & Young, R. M. (2004). An intent-driven planner for multi-agent story generation. Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1, 186-193.

Rilling, S., & Wechselberger, U. (2011). A framework to meet didactical requirements for serious game design. *The Visual Computer, 27*(4), 287-297. doi:10.1007/s00371-011-0550-6

Rosenheck, L., Lin, C. Y., Klopfer, E., & Cheng, M. T. (2017). Analyzing gameplay data to inform feedback loops in The Radix Endeavor. Computers & Education.

Ryan, R. M., & Deci, E. L. (2000). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. American psychologist, 55(1), 68.

Ryan, R. M., Rigby, C. S., & Przybylski, A. (2006). The motivational pull of video games: A self-determination theory approach. Motivation and emotion, 30(4), 344-360.

Sansone, C. & Harackiewicz, J. M., I. (2000). Intrinsic and extrinsic motivation: The search for optimal motivation and performance. San Diego: Academic Press.

Shadish, W. R., Cook, T. D., & Campbell, D. T. (2002). Experimental and quasi-experimental designs for generalized causal inference. Wadsworth Cengage learning.

Shaffer, D. W., Halverson, R., Squire, K. R., & Gee, J. P. (2005). Video Games and the Future of Learning. WCER Working Paper No. 2005-4. *Wisconsin Center for Education Research (NJ1)*.

Shaffer, D. W. (2006). Epistemic frames for epistemic games. Computers & education, 46(3), 223-234.

Schrader, C., & Bastiaens, T. J. (2012). The influence of virtual presence: Effects on experienced cognitive load and learning outcomes in educational computer games. Computers in Human Behavior, 28(2), 648-658.

Shute, V. J. (2011). Stealth assessment in computer-based games to support learning. *Computer games and instruction*, *55*(2), 503-524.

Shute, V., & Ventura, M. (2013). Stealth assessment : Measuring and supporting learning in video games. Cambridge, MA, USA: MIT.

Shute, V. J., D'Mello, S., Baker, R., Cho, K., Bosch, N., Ocumpaugh, J., ... & Almeda, V. (2015). Modeling how incoming knowledge, persistence, affective states, and in-game progress influence student learning from an educational game. Computers & Education, 86, 224-235.

Sicart, M. (2008). Defining game mechanics. Game Studies, 8(2), 1-14.

Smith, L. C. (1984). Semantic satiation affects category membership decision time but not lexical priming. *Memory & Cognition, 12*(5), 483-488.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. Cognitive science, 12(2), 257-285.

Sweller, J. (1994). Cognitive load theory, learning difficulty, and instructional design. *Learning and instruction*, *4*(4), 295-312.

Sweller, J., Van Merrienboer, J. J., & Paas, F. G. (1998). Cognitive architecture and instructional design. Educational psychology review, 10(3), 251-296.

Sung, H. Y., & Hwang, G. J. (2017). Facilitating effective digital game-based learning behaviors and learning performances of students based on a collaborative knowledge construction strategy. Interactive Learning Environments, 1-17.

Sykes, J., & Brown, S. (2003, April). Affective gaming: measuring emotion through the gamepad. In CHI'03 extended abstracts on Human factors in computing systems (pp. 732-733). ACM.

Tang, S., Hanneghan, M., & El Rhalibi, A. (2009). Introduction to games-based learning. Games Based Learning Advancements for Multi-Sensory Human Computer Interfaces. New York: IGI Global.

Tang, S., & Hanneghan, M. (2013, January). A model driven serious games development approach for game-based learning. In *Proceedings of the International Conference on Software Engineering Research and Practice (SERP)* (p. 1). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

Tobias, S., & Fletcher, J. D. (2011). Computer games and instruction. IAP.

Tucker, B. (2014, May 7). Five Reasons Why Super Mario Bros. Can Be A Pain in the Butt(z). [Web log Post]. Retrieved from: https://thesurrealityproject.com/2014/05/07/five-reasons-why-super-mario-bros-can-be-a-pain-in-the-buttz/

Urdan, T., & Schoenfelder, E. (2006). Classroom effects on student motivation: Goal structures, social relationships, and competence beliefs. *Journal of School Psychology*, *44*(5), 331-349.

Van Eck, R. (2006). Digital game-based learning: It's not just the digital natives who are restless. *EDUCAUSE review*, *41*(2), 16.

Van Merriënboer, J. J., Kester, L., & Paas, F. (2006). Teaching complex rather than simple tasks: Balancing intrinsic and germane load to enhance transfer of learning. Applied cognitive psychology, 20(3), 343-352.

White, R. W. (1959). Motivation reconsidered: the concept of competence. Psychological review, 66(5), 297.

Wiebe, E. N., Lamb, A., Hardy, M., & Sharek, D. (2014). Measuring engagement in video game-based environments: Investigation of the User Engagement Scale. Computers in Human Behavior, 32, 123-132.

Williams, D., Yee, N., & Caplan, S. E. (2008). Who plays, how much, and why? debunking the stereotypical gamer profile. *Journal of Computer-Mediated Communication, 13*(4), 993-1018. doi:10.1111/j.1083-6101.2008.00428.x

Yeung, A. S., Jin, P., & Sweller, J. (1998). Cognitive load and learner expertise: Split-attention and redundancy effects in reading with explanatory notes. Contemporary educational psychology, 23(1), 1-21.

APPENDIX I

INSTITUTIONAL REVIEW BOARD APPROVAL

APPROVAL: EXPEDITED REVIEW

Ashish Amresh
CIDSE - Software Engineering at Polytechnic
480/727-1253
amresh@asu.edu

Dear Ashish Amresh:

On 11/14/2016 the ASU IRB reviewed the following protocol:

Type of Review:	**Error! Hyperlink reference not valid.** Study
Title:	Content Agnostic Mechanics
Investigator:	Ashish Amresh
IRB ID:	STUDY00005148
Category of review:	(7)(b) Social science methods, (7)(a) Behavioral research
Funding:	None
Grant Title:	None
Grant ID:	None
Documents Reviewed:	• User Engagement Scale.pdf, Category: Measures (Survey questions/Interview questions /interview guides/focus group questions); • Tyler Baron CITI IRB Training, Category: Other (to reflect anything not captured above); • Child-Assent-Form.pdf, Category: Consent Form; • ADULT PARTICIPANT CONSENT SOCIAL BEHAVIORAL.pdf, Category: Consent Form; • TowersScreenshot.pdf, Category: Other (to reflect anything not captured above); • HRP-503a-TEMPLATE_PROTOCOL_SocialBehavioralV02-10-15.docx, Category: IRB Protocol; • Cognitive Load Form.pdf, Category: Measures (Survey questions/Interview questions /interview guides/focus group questions); • Tyler Baron doctoral study letter.pdf, Category: Recruitment Materials; • Background Info.pdf, Category: Measures (Survey questions/Interview questions /interview guides/focus group questions);

	• PARENTAL CONSENT SOCIAL BEHAVIORAL.pdf, Category: Consent Form;

The IRB approved the protocol from 11/14/2016 to 11/13/2017 inclusive. Three weeks before 11/13/2017 you are to submit a completed Continuing Review application and required attachments to request continuing approval or closure.

If continuing review approval is not granted before the expiration date of 11/13/2017 approval of this protocol expires on that date. When consent is appropriate, you must use final, watermarked versions available under the "Documents" tab in ERA-IRB.

In conducting this protocol you are required to follow the requirements listed in the INVESTIGATOR MANUAL (HRP-103).

Sincerely,

IRB Administrator

cc: Tyler Baron
Tyler Baron

APPENDIX II

LIST OF QUESTIONS AND POSSIBLE ANSWERS FROM CAGE GAMES THAT POSED DIRECT QUESTIONS

Car Racing

Content 1: Vocabulary

All questions asked the player to select a synonym for the provided word.

1. Controversial
    a. Incomplete
    b. Debatable
    c. Reduce
2. Inflate
    a. Observer
    b. Enlarge
    c. Terminal
3. Rural
    a. Rustic
    b. Bogus
    c. Consider
4. Meager
    a. Overturn
    b. Endorse
    c. Poor
5. Prudent
    a. Wary
    b. Competent

    c. Careless

6. Quest
    a. Entrap
    b. Reject
    c. Venture
7. Potential
    a. Possibility
    b. Sad
    c. Crucial
8. Mar
    a. Disfigure
    b. Able
    c. Decrease
9. Haggle
    a. Ferocious
    b. Attend
    c. Bargain With
10. Fugitive
    a. Clear
    b. Runaway
    c. Compensation
11. Incentive

a. Terminate

b. Leading

c. Stimulus

12. Culminate

a. Provide

b. Terminate

c. Incomplete

13. Trivial

a. Wear Away

b. Readable

c. Insignificant

14. Remnant

a. Capricious

b. Remainder

c. Prevent

15. Mutual

a. Two-Sided

b. Waver

c. Chaos

16. Lubricate

a. Oil

b. Ailment

c. Adept

17. Insinuate

    a. Imply

    b. Imperfect

    c. Endorse

18. Vicious

    a. Unconcern

    b. Wicked

    c. Reduce

19. Foremost

    a. Paramount

    b. Hesitant

    c. Surface

20. Luster

    a. Shine

    b. Illness

    c. Shrewd

Content 2: United States History

1. The first permanent successful settlement in North America was located at

    a. Jamestown, Virginia

    b. Roanoke Island, North Carolina

    c. New London, Connecticut

2. Puritan society was organized around
    a. The individual
    b. The family
    c. Elaborate church ritual
3. During the first two years of the war, the Americans were victorious at all of the following battles except the battle of
    a. Long Island
    b. Princeton
    c. Ft. Ticonderoga
4. Which of the following was an ally of the British during the American War for Independence?
    a. France
    b. Spain
    c. None of the above nations.
5. The political figure who took the lead in establishing the Washington administration's domestic priorities was
    a. Alexander Hamilton
    b. George Washington
    c. Thomas Jefferson
6. The Shawnee Indians, led by The Prophet, were defeated by Will Henry Harrison's forces at
    a. New Orleans

b. York

c. Tippecanoe

7. The theory that the Union is a compact among the states and that a state has the right to override a federal law is known as

    a. Nullification

    b. Statism

    c. Sectionalism

8. Which of the following groups in the 1840s believed that the end of the world was imminent?

    a. Mormons

    b. Unitarians

    c. Millerite

9. Which of the following was a reform movement of the 1840s?

    a. Abolitionism

    b. Public education

    c. All of the above

10. The battle that allowed Lincoln to issue the Emancipation Proclamation was

    a. Bull Run

    b. Antietam

    c. Vicksburg

11. White settlers migrating to the West gave the Plains Native Americans

a. Quinine

b. Opium

c. Smallpox

12. Which of the following candidates in the 1912 presidential election advocated the most far-reaching changes for American society?

a. Woodrow Wilson

b. Eugene Debs

c. Theodore Roosevelt

13. Which sector of the economy did not prosper in the 1920s

a. Manufacturing

b. Financial services

c. Agriculture

14. Which of the following did not offer radical criticism of Roosevelt and FDR programs?

a. Huey Long

b. Harry Hopkins

c. Dr. Francis Townsend

15. Japan demonstrated its expansionist policies in 1931-1932 by invading

a. Manchuria

b. Pearl Harbor

c. The Philippines

16. Which of the following groups experiences the least amount of discrimination and prejudice in the U.S. during World War II?
    a. German-Americans
    b. Mexican-Americans
    c. Japanese-Americans
17. Which of the following statements most accurately describes conditions in the American army in Vietnam by 1969?
    a. Morale had plummeted
    b. Drug use had soared
    c. All of the above
18. Martin Luther King's philosophy of civil disobedience incorporated
    a. Nonviolent resistance
    b. Direct action
    c. All of the above
19. The most serious conflict between the U.S. and the Soviet Union during the Kennedy administration occurred over
    a. Berlin
    b. Cuba
    c. Macao
20. Which of the following issues did evangelical Christians target in the 1970s?
    a. Abortion

b. Homosexuality

c. All of the above

Fun-O-Sphere

In both content domains, the player was asked to remove all spheres such that a specific condition would be met with the remaining spheres.

Content 1: Probability

1. P(Black) 4/5
    a. Starts with 5 black and 3 grey.
2. P(Red) 2/5 & P(Black) 3/5
    a. Starts with 3 red, 1 blue, 4 black, and 1 grey.
3. P(Grey) 1/5 & P(Black) 2/5
    a. Starts with 5 red, 2 black, and 3 grey.
4. P(Red) 1/7 & P(Blue) 4/7
    a. Starts with 3 red, 4 blue, and 4 black.
5. P(Red) 1/8, P(Black) 4/8 & P(Grey) 3/8
    a. Starts with 3 red, 3 blue, 4 black, and 4 grey.

Content 2: Chemistry

1. 1 molecule of common salt contains:
    a. Starts with 1 bromine, 2 boron, 1 chlorine, and 2 sodium.
2. 1 molecule of water consists of:
    a. Starts with 2 copper, 3 oxygen, 2 carbon, and 3 hydrogen.

3. 1 molecule of laughing gas consists of:
    a. Starts with 2 bromine, 3 oxygen, 1 boron, and 2 nitrogen.
4. 1 molecule of limestone consists of:
    a. Starts with 3 copper, 4 oxygen, 4 carbon, and 2 calcium.
5. 1 molecule of sulphuric acid consists of:
    a. Starts with 5 oxygen, 2 boron, 3 hydrogen, and 3 sulphur.

Hat Trick

Content 1: Math

Players were asked to solve either for k or for c in the given equations.

1. $y = kx + c$ passes (0,3) and (3,0)
2. $y = kx + c$ passes (1,8) and (3, 12)
3. $y = kx + c$ passes (2,2) and (6,14)

Possible answers were shared between all questions.

1. -1
2. 3
3. 2
4. 6
5. 3*
6. -4

* Answers that fell were picked at random from the list. By having 3 in the list twice it was more likely to appear.

Content 2: History

Players were asked to pick the year in which each of these events occurred.

1. The American Revolutionary War brings freedom and rights for the United States.
2. World War I destroys the world's peace and economy.
3. The first financial crisis has seriously affected the economic development of the US.
4. The establishment of the Soviet Union brings communism to much of the world.
5. The discovery made by Columbus had an enormous impact in the historical development of the modern western world.
6. The English revolution spreads capitalism across Europe.

The potential answers were shared across all questions.

1. 1775
2. 1914
3. 1857
4. 1922
5. 1492
6. 1640

Pedestals

Content 1: Linear Equations

For all questions, the player is asked to find the value of X.

1. $3x = 21$
2. $7x = 56$
3. $2x + 3 = 9$
4. $11x + 24 = 68$
5. If $y = 2$, and $3x + 4y = 23$
6. $y = 2x + 1$ & $y = 5$
7. $5x = 6 + 3y$ & $y = 8$
8. $f(x) = 2x – 3$ & value of function is 15
9. $f(x) = 4x + 13$ & value of function is 57
10. $5x – 12 = 3x + 6$
11. $4(x + 1) = 20$
12. $6(x + y) = 48 + 6y$
13. $f(x) = f(y) + 2$ & $f(y) = 2y + 3$. Find for $y = 2$.
14. $f(x) = 2x + 7$ & value of function is 21
15. $f(x) = 3x + 11$ & value of function is 47

Four answers were shown at a time. One was the correct answer and the other three were picked at random.

1. 7
2. 8
3. 3

4. 4
5. 5
6. 2
7. 6
8. 9
9. 11
10. 9
11. 7
12. 12

Content 2: English

The player was asked to fill in the blank with the correct word. The blanks are represented as _ here.

1. The ball is kept over _.
2. We had a few neighbors move in next door. We love _ car.
3. Mike and Chris think Santa is real. _ are going to be disappointed when they find out the truth.
4. _ rhymes with tough.
5. _ rhymes with though.
6. _ rhymes with through.
7. Cough rhymes with _.
8. The closest synonym of _ is extraordinary.
9. The closest synonym of _ is furious.

10. Dangerous means the same as _.
11. Alfred nodded saying he was positive about the instructions. He meant he was _.
12. Matt wanted to go to the park. He was _ to meet his friends.
13. Peevish is the synonym for _.

As with the first content, four possible answers were shown at a time.

1. There
2. Their
3. They’re
4. Rough
5. Bow
6. Brew
7. Scoff
8. Astounding
9. Anger
10. Perilous
11. Sure
12. Eager
13. Temperamental

Quiz Up!

All questions are true/false. Note that some of the questions make use of an audio clip that plays while the question is on the screen. Other questions show a small picture next to the question.

Content 1: Western Classical

1. Chalumeau is the lower register of the clarinet's playing range.
2. Moonlight Sonata is composed by Mozart.
3. Allegro tempo ranges between 120-168 bpm.
4. Verdi wrote his Requiem when he was 50 years old.
5. Flute is a wind instrument.
6. Scales in traditional Western music generally consist of seven notes.
7. This is a tuba.
8. This is note 'A'.
9. This clip is in A minor.
10. Is this in G sharp?
11. This is a D major chord.
12. This is an Oboe.
13. Images shows note 'D'.
14. This is note 'E'.

Content 2: French

1. 'the woman' is translated as 'la femme'.
2. 'a boy' is translated as 'le garcon'.
3. The number 3 is trois.

4. The number 10 is cinq.
5. They are saying goodbye.
6. It is 3 o clock.
7. Clip is saying ‘Hello’.
8. Text and clip match: ‘Ten Eleven Twelve’
9. Clip asks, ‘Do you exercise?’
10. French is the most spoken language in the world.
11. French is written with the 26 letters of the basic Latin script.
12. ‘The grandfather’
13. ‘We play football’
14. Is the clip the correct translation of: ‘We learn French’?

Think Fast!

The questions for the first content, math, were randomly generated.

Content 2: Spanish

The following sentences were given as the correct answer, while incorrect answers were scrambled versions of the correct one. The commas separate out each element shown in game.

1. ¿, Querría, bailar, conmigo, ?
2. ¿, Vienes, aquí, a, menudo?
3. ¡, Llame, a, la, policía!
4. Un, idioma, nunca, es, suficiente

5. Por, favor, hable, más, despacio

APPENDIX III

QUESTIONARIES FOR THE CAGE FRAMEWORK DEVELOPMENT STUDY

Content Agnostic Mechanics Development Process

This is a survey of your experience using the CAM framework to develop your your game. Please fill everything out as completely and accurately as possible. You do need to fill this out, but your answers do not impact your grade. Be honest!

Demographics Info

Please enter your name

Please enter your age

Please select your gender

- Male
- Female
- Prefer not to say

Please select which year of the Master's program you are currently in

- Year 1
- Year 2
- Year 3+

What was your Bachelor's in

- Computer Science
- Software Engineering
- Computer Engineering

- Other

Do you have software development industry experience?

- Yes
- No

If YES, how many years?

- Less than 1
- 1-2
- 2-3
- 3-4
- 4+

Have you used Unity to develop an Educational Game in the past (before this course)?

- Yes
- No

Software Framework

Please give approximately the number of lines of code you wrote for the FIRST part of the CAM assignment

Please give approximately the number of lines of code you wrote for the SECOND part of the CAM assignment (this does not include the lines from the previous question!)

Please provide the approximate number of hours spent on the FIRST half of the CAM assignment

Please provide the approximate number of hours spent in the SECOND half of the CAM assignment

Note: Until noted again, the below questions each had a five point scale where 1 was the lowest, or least favorable, rating possible.

How useful do you feel the framework was in speeding up your development process?

If you were making another educational game, how likely would you be to reuse this framework (or an updated version of it) for this new project?

Note: The next question was a free response, after which the answer format returned to the scales.

What changes would you suggest for improving the framework?

Software Development Part 1

To what extent do you agree or disagree with the following statements?

I think it's generally better to thoroughly rewrite old code.

For good reliability old code should be reused even if it slightly perturbs the system design.

If I am under time pressure I am more likely to try and reuse old code.

Software Development Part 2

In your experience in the coding phase of projects, how often did you end up extensively modifying or completely rewriting a module when the design called for a module to be reused or reusing a module (in whole or in part) when the design called for a new module to be written?

Note: Both of the following statements had these response options.

- Very Rarely (0-5%)
- Rarely (5-10%)
- Occasionally (10-25%)
- Often (25%-50%)
- Most of the Time (50%-100%)

Wrote New Code Instead of Reusing

Reused Instead of Writing New Code

Software Development Part 3

If there were times when you did not reuse code because the old code was not accessible or acceptable, what were the reasons?

Please indicate how often each of the provided reasons prevented you from reusing existing code.

Note: Each of the following statements had the following response options.

- Never/Rarely

- A Few Times
- Several Times/Often

Not on Computer

Not in Language

Did not Meet Requirements

Defects in Code

Could not Understand

Code too Complex

APPENDIX IV

QUESTIONARIES FOR THE CAGE STUDY

Background Info

Don’t worry, we’ll fill this part out for you.

ID#: _____

Game Played: _____

Please answer the following questions. If there are any questions you would prefer not to answer, just leave it blank.

Age: _____

Gender: M F

I am a student in: Middle School High School College I am not a student

If applicable, I am in Grade: _____

I consider myself: Not a Gamer A Gamer

Either way, I play about _____ hours of video games a week (put 0 if you don’t play games at all).

Cognitive Load Form

ID#: _____

Content: A B

Please circle a number:

How much mental effort do you feel it took you to play the game?

1 means “None at all”

10 means “All my mental effort”

1 2 3 4 5 6 7 8 9 10

User Engagement Scale

Please put one check mark for each question.

1 = Strongly Disagree

4 = Strongly Agree

Question	1	2	3	4
When I was playing the game, I lost track of the world around me.				
I blocked out things around me while I was playing the game.				
The time I spent playing the game just slipped away.				
I was absorbed in my gaming task.				
I was so involved in my gaming task that I lost track of time.				
During this gaming experience I let myself go.				
I lost myself in this gaming experience.				
I was really drawn into my gaming task.				
I felt discouraged while playing the game.				
I felt annoyed while playing the game.				
Playing the game was mentally taxing.				
I found the game confusing to play.				
I felt frustrated while playing the game.				
I could not do some of the things I needed to do in the game.				
The gaming experience was demanding.				
This gaming experience did not work out the way I had planned.				
I liked the graphics and images used in the game.				
The game appealed to my visual senses.				
The game was aesthetically appealing.				
The screen layout of the game was visually pleasing.				
The game was attractive.				
The content of the game incited my curiosity.				
I would continue to play this game out of curiosity.				
I would recommend playing the game to my friends and family.				
Playing the game was worthwhile.				
I felt interested in my gaming task.				
My gaming experience was rewarding.				
This gaming experience was fun.				

Content Agnostic Mechanics Development Survey

This is a survey of your experience using the CAM framework to develop your game. Please fill everything out as completely and accurately as possible. You do need to fill this out, but your answers do not impact your grade. Be honest!

Demographics Info

Please enter your name:

Please enter your age:

Please select your gender:

- Male
- Female
- Prefer not to say

Please select which year of the Master's program you are currently in:

- Year 1
- Year 2
- Year 3+

What was your Bachleor's in:

- Computer Science
- Software Engineering
- Computer Engineering
- Other

Do you have software development industry experience?

- Yes
- No

If YES, how many years?

- Less than 1
- 1-2
- 2-3
- 3-4
- 4+

Have you used Unity to develop an Educational Game in the past (before this course)?

- Yes
- No

Software Framework

Please give approximately the number of lines of code you wrote for the FIRST part of the CAM assignment.

Please give approximately the number of lines of code you wrote for the SECOND part of the CAM assignment (this does not include the lines from the previous question!)

Please provide the approximate number of hours spent on the FIRST half of the CAM assignment.

Please provide the approximate number of hours spent on the SECOND half of the CAM assignment.

How useful do you feel the framework was in speeding up your development process?

- Participants had a 5 point rating scale, with 1 being “Not Helpful at All” and 5 being “Very Helpful”.

If you were making another educational games, how likely would you be to reuse this framework (or an updated version of it) for this new project?

- Participants had a 5 point rating scale, with 1 being “Would not use it” and 5 being “Would definitely use it”.

What changes would you suggest for improving the framework?

Software Development Part 1

To what extent do you agree or disagree with the following statements.

- Each of the following statements had a 5 point rating scale with 1 being “Stongly Disagree” and 5 being “Strongly Agree”.

I think it’s generally better to thoroughly rewrite old code.

For good reliability old code should be reused even if it slightly perturbs the system design.

If I am under time pressure I am more likely to try and reuse old code.

Software Development Part 2

In your experience in the coding phase of projects, how often did you end up extensively modifying or completely rewriting a module when the design called for a module to be reused or reusing a module (in whole or in part) when the design called for a new module to be written?

- For both of the statements below, which respond to the situation posed above, participants were given five choices.
    1. Very Rarely (0-5%)
    2. Rarely (5-10%)
    3. Occasionally (10-25%)
    4. Often (15-50%)
    5. Most of the Time (50-100%)

Wrote New Code Instead of Reusing

Reused Instead of Writing New Code

Software Development Part 3

If there were times when you did not reuse code because the old code was not accessible or acceptable, what were the reasons?

Please indicate how often each of the provided reasons prevented you from reusing existing code.

- All reasons below had the following three options.
    1. Never/Rarely
    2. A Few Times
    3. Several Times/Often

Not on Computer

Not in Language

Did not Meet Requirements

Defects in Code

Could not Understand

Code too Complex