# Chapter 9
# A Framework for Designing Smarter Serious Games

**Katherine Smith, John Shull, Yuzhong Shen, Anthony Dean and Patrick Heaney**

**Abstract** As smart technology becomes more ubiquitous, there are increased opportunities to enhance the way that educational content is presented and practiced. There are numerous studies indicating the importance of combining instructional and game design in the process of developing serious games. However, there are no frameworks or processes that have been developed to provide insight on methods for effectively combining these existing design methodologies. To this end, a modular framework has been developed with an accompanying spiral design process that facilitates the design, development, and continued improvement of smarter serious games. The implementation of this framework will be explored through an evolving serious game developed using the framework that is aimed to teach precalculus at the college level.

**Keywords** Smart serious games · Game design · Instructional design · STEM education

K. Smith (✉) · J. Shull · Y. Shen
Department of Modeling, Simulation and Visualization Engineering, Old Dominion University, Norfolk, USA
e-mail: k3smith@odu.edu

J. Shull
e-mail: jshull@odu.edu

Y. Shen
e-mail: yshen@odu.edu

K. Smith
Department of Mathematics and Statistics, Old Dominion University, Norfolk, USA

A. Dean
Engineering Fundamentals Division, Old Dominion University, Norfolk, USA
e-mail: adean@odu.edu

P. Heaney
Department of Mechanical and Aerospace Engineering, Old Dominion University, Norfolk, USA
e-mail: phean001@odu.edu

© Springer International Publishing AG 2018
V.L. Uskov et al. (eds.), *Smart Universities*, Smart Innovation, Systems and Technologies 70, DOI 10.1007/978-3-319-59454-5_9

## 9.1 Introduction

As of 2015, 86% of 18–29 year olds owned smartphones while 78% owned traditional computers [1]. As today's students are constantly engaging with technology, the ability to further engage them in education on that same technology through gaming is an incredible asset. However, many educators and instructional designers have cited lack of content relevance as a major deterrent to the adoption of serious games in education [2, 3]. There are studies showing serious games are highly effective at conveying content from training for Navy recruits [4] to medical instruction and training for both patients and health care providers [5]. While some efficacy studies of serious games have yielded mixed results, this result is not surprising. Just as there are effective and ineffective teachers, there will be effective and ineffective serious games [6]. The challenge then becomes how to develop serious games that are not only effective, but smart. There are many papers focusing on characteristics of serious games that have been developed [7–9], but there has been no overall methodology put forth to allow game developers to design serious games that get smarter as the player model is developed. In particular, due to budget and scheduling constraints it is often necessary for developers to choose a subset of these characteristics to incorporate into an initial version of the game. From there, it will be important that the overall framework is flexible enough to allow developers to remove characteristics that are not effective for their target population and add elements that may increase learning and engagement. In this process, it is vital for developers to focus on three main aspects. The first is the target player and how they may be different from the general population. The second is the content as they carefully determine what content can be effectively delivered through a serious game and the optimum way to present this content to enhance learning and maintain engagement. The final focus is the game itself and using effective game design techniques to build a game that is engaging as well as instructive. The approach highlighted aims to build upon approaches to instructional design [10] and game design [11] to present a clear method for combining them to create a framework for smart serious games. In addition, this framework supports the enhancement of smartness features including adaptation, sensing, and inference. This is demonstrated by showing how serious games developed under this model can be considered smart learning environments and enhance the overall smartness level of a university.

In this chapter, the discussion will focus on the development of a framework which combines game and instructional design and can be applied to produce an evolving smart serious game that becomes smarter as data collection informs the developers. The end result is iMPOS$^2$inG: A Model and Process for creating Smarter Serious Games which is an adaptable framework that contains each of the modules needed for a smart learning environment [12] while ensuring that those modules are able to adapt and change as more information is revealed about the individual users. A spiral process involving (Re)Definition, Development, and Enhancement of player, instructional and game design characteristics is carried out repeatedly to create a final product that is smart, effective, and entertaining. At each stage, the player, content, and game are emphasized to ensure that all three considerations are properly balanced in the final product.

Beginning with assumptions based on research into the target population, game and instructional design principles are combined to create a serious game that targets general deficiencies in the student population. Through data collection from in-game action, each component or module of the game can be updated individually to take into account learned features about individual students. Finally, an intelligent tutoring layer is added to redirect students through the game modules based on the user model learned from initial trials. This customization is facilitated by having all information about the player's path associated with the player model rather than with individual objects within the game.

The remainder of this chapter will be organized as follows. Section 9.2 provides a review of significant literature related to serious games, instructional design, and smart learning environments. Section 9.3 defines the overall goals and objectives that will be addressed in this chapter. Section 9.4 discusses the overall framework and process along with methods used in their development. Additionally, Sect. 9.4 indicates how this framework can be used to develop smarter serious games that enhance particular smartness features at a university. Section 9.5 discusses applications and implementation of the spiral process and framework for designing smarter serious games. This is demonstrated with examples from a serious game following the framework that has been developed by the authors. Section 9.6 provides discussion of results and lessons learned. Section 9.7 addresses conclusions and future work.

## 9.2 Literature Review

**Smart learning environments**. Hwang [12] presents criteria for smart learning environments as well as a set of modules that comprise a smart learning environment. According to Hwang [12], smart learning environments are aware of the learner's context and environment. This can include their physical location and surroundings as well as their behaviors. The smart learning environment is then able to adapt and provide support to the learner based on their specific context. Additionally, smart learning environments are aware and able to respond to unique learner requirements based on information gathered from the user throughout their interaction with the environment. Finally, smart learning environments have user interfaces that are able to change to present content in different ways based on the learner's context and needs. In addition to providing criteria for smart learning environments, Hwang proposes a set of modules that are required to implement a smart learning environment and presents a framework that links content and learner profiles to the learner through a user interface.

**Reviews of serious game effectiveness.** Connolly et al. [6] provide a highly detailed survey of the literature regarding positive outcomes from studies of serious games. They discover that while negative outcomes from serious games are more publicized, there are actually a greater number of studies reporting positive learner effects. They also note that there is great variability in the methodology used to collect these results which they attribute to the multidisciplinary nature of serious game development. In response, they recommend more rigorous methods for evaluating serious games.

**Study of educational game designers.** Ruggiero and Watson [9] focus on how game designers approach praxis in their games by interviewing twenty-two educational game designers. Based on the compiled results, they find that many experienced game designers describe the entire process of game design as an action-reflection cycle that is focused first on the content they are trying to convey to the learners. In addition, they find that it is important to understand and reflect on any design and project constraints as soon as possible. The authors recommend using an action-reflection cycle in the educational game design process.

**Game characteristics in game and instructional design.** Charsky [7] provides an overview of not only how the implementation of game characteristics has evolved over time, but also the importance of collaboration between game and instructional design in implementing these characteristics. The author focuses on competition and goals, rules, choice, challenges, and fantasy and how incorporating these characteristics requires knowledge of topics that span instructional design, game design, and computer programming. He also points out that there is no set procedure for balancing these considerations. More research is required to understand how these various components can be integrated in successful serious games.

**Need for a serious game framework and process.** Bellotti et al. [13] highlight opportunities for research in the design of serious games particularly identifying a need to study different methodologies and architectures for developing serious games that promote the development of serious games that are effective for a variety of learners across a range of content areas. In our work, we aim to provide a methodology for development as well as an overall game architecture that can be adapted to serve the needs of many content areas and user groups.

**Smart features.** Uskov et al. [14] identify key features that indicate the level of development of a smart university beyond a traditional university. These features include adaptation, sensing, inferring, self-learning, anticipation, and self-organization. From our perspective on investigating a model for the development of smarter serious games as smart learning environments, we will highlight how smarter serious games can address adaptation, sensing, and inferring. Uskov et al. [14] define these as follows:

- Adaptation describes a smart university's capacity to alter the way it approaches functions, such as teaching and learning.
- Sensing describes a smart university's ability to collect data regarding changes that may affect its interests.
- Inferring describes a smart university's ability to use collected data to make decisions that change the way the university functions or help students.

**Smart serious gaming.** Uskov and Sekar [15] identify a set of trends which they expect will be followed as serious games progress into smart serious games. A few of these trends are:

- Serious games will evolve by incorporating and enhancing the same smartness features from [14].
- User engagement will tend to increase.
- Collaboration and integration between diverse platforms will be enhanced.

## 9.3 Research Project Goal and Objectives

This framework for smart serious games has been developed as part of a smart learning environment for military veterans that aims to help universities become smarter by anticipating the needs of a target student population and providing adaptive programs to serve those needs. While this particular project focused on student veterans and STEM courses specifically, the framework and lessons learned can be transferred to support the development of smarter serious games for a range of student populations across a broad array of content areas. This chapter will focus mainly on the development of the framework and components of the program that directly impact this aspect of the project. First, a modular smart learning environment with components that continually evolve to become smarter is presented. In addition, an overall design process that accounts for player, instructional, and game design considerations is proposed. The overall goal is to develop and demonstrate a framework that can be used to develop smart serious games.

### *9.3.1 Goals and Objectives for Smart Serious Games*

The goal for a smarter serious gaming framework is to target many of the deficiencies that traditionally plague serious games while simultaneously serving the target population. In order to do this, the framework needs to be content focused. In addition, the framework needs to focus particularly on challenging content that can be enhanced by interactivity, compelling visualizations, and dynamic rapid feedback which are all characteristics of any good game. Furthermore, providing an adaptable framework that allows small teams to develop games that become smarter as more information about the player is revealed was of great importance. Designing an adaptive learning environment requires a detailed model of the player population which can usually only be obtained by gathering data as the game is played. This requires the development team to use limited knowledge to develop a smart player model that can adapt to become smarter. Going beyond the traditional idea of an adaptive learning environment which allows players to proceed through a custom path, this framework needed to be adaptable to allow incorporation of additional game play enhancements such as score boards to promote competition and integration of external resources for just-in-time (JIT) assistance.

### 9.3.2 Stern2STEM Program Goals and Objectives

This smart serious game approach has been developed as part of *From Stern to STEM* which is a pilot program designed to develop and investigate techniques to assist military veterans in attaining STEM degrees [16]. This program is designed to recruit driven, capable military veterans with technical STEM experience into engineering and engineering technology degree programs. Once the student veterans are enrolled, the program aims to support the veterans throughout their degree program. First, veterans are aided in preparing for their first college classes and entrance exams by providing STEM leveling assistance and tailored advising. Many of these students have been out of the classroom for an extended period of time and require a refresher on introductory precalculus, calculus, physics, and chemistry courses to allow them to start the program at the correct level. Students are provided with tailored support through tutoring, mentoring, advising, online resources, and interactive gaming to allow them to prepare to begin successfully. The tools provided through this program are designed to combine proven pedagogical practices with smart technology such as interactive gaming to provide the veterans with an experience that is tailored to their individual needs. After graduation, the program aims to provide graduates with career placement resources and provide workplace development throughout the career of Navy STEM professionals.

## 9.4 Development of IMPOS$^2$inG

While many serious games have been designed and analyzed [6], there has been no adaptable framework presented to guide game developers through the complex process of designing smarter serious games. Serious game design requires understanding of both game and instructional design in order to combine game characteristics and content in a way that leads to a motivating game focused on learning [7]. In order to resolve this lapse, a framework that incorporates game design, instructional design, and player consideration has been developed. As recommended by a survey of experienced game developers [9], this framework involves a spiral process that promotes activity and reflection on that activity throughout the development process. After reviewing a model for instructional design and a model for game design, the discussion will focus on combining the two into a unique process as well as the resulting model for a smarter serious game. Finally, the ramifications of this development on smartness features and smart universities will be discussed.

### *9.4.1 Overview of the Successive Approximation Model for Instructional Design*

The Successive Approximation Model (SAM) was introduced by Allen and Sites in 2012 as a replacement for the more traditional Analysis, Design, Development, Implementation, and Evaluation (ADDIE) model that took advantage of iterative design processes [10]. While an extensive discussion of the model is not necessary, the features and components of the model are summarized here for readers not intimately familiar with instructional design models.

The SAM process starts with information gathering to collect background information needed for a successful project start. The process then moves to a "SAVVY Start" phase which is a short brainstorming session meant to collect ideas from key team members and kick off the iterative design phase. In the iterative design phase, the team follows a design, prototype, review cycle in order to solidify a design that will be ready to move into the iterative development phase. During the iterative development phase, developers follow a cycle where they develop, implement, and evaluate to develop and enhance a product that is ready for market. Once a product is released, the iterative design and development phases continue in order to continually improve the product and incorporate feedback.

The entire process is focused on developing a series of effective learning events that are meaningful, motivational, and memorable. These characteristics are often easily incorporated in games as many of the key game characteristics directly support them. In addition to these characteristics, learning events are comprised of four components. The first component is context which provides the background and setting for the learner's task. The next component is challenge which defines the problem the user must solve or the adversity they must overcome. The third component is activity which is the actual set of actions the user can take in their attempt to complete the challenge. The final component is feedback which informs the user about their performance and potentially generates a new learning event based on the outcome of this current event.

The iMPOS$^2$inG model incorporates aspects of the SAM model to support player completion of effective learning activity throughout game play.

### *9.4.2 Overview of a Game Design Model*

The U.S. computer and video game market had a revenue of 22.41 billion dollars in 2014 [17]. As the market has grown steadily over the last decade, the emergence of large studios and distribution companies has steadily followed suit. With so many commercial developers working to release games, it would seem intuitive that there would be many models for game design. However, as market-leading games can take a great deal of time and capital to produce, these studios are hesitant to turn over their models. Thankfully within the past decade, casual game design models

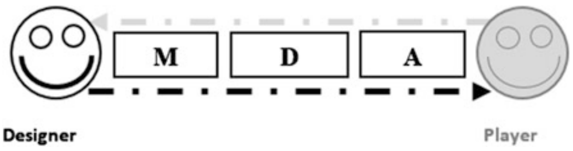

**Fig. 9.1** The MDA model by Hunicke et al. (*source* [18], p. 2)

have emerged. In general, these models share many aspects and some would argue that their differences lay in more semantics than functionality. For our intentions we will identify three models that have been highly referenced and reviewed. These models are the Mechanics, Dynamics, and Aesthetics model (MDA) by Hunicke, et al. [18], a model of lenses by Schell [19], and a playcentric approach by Fullerton [11].

The MDA model was developed using an iterative process through a game design workshop run by Robin Hunicke [18]. A graphical overview of this model from [18] is shown in Fig. 9.1. Within the MDA approach, the three design elements are broken down into specific components based on the interaction between the rules and the system and focusing on the fact that the result of this interaction should be enjoyable gameplay [18]. This framework is linear and viewed by designers and players from different ends of the process. Designers approach the game starting with Mechanics, while players approach it starting with aesthetics. From the developer perspective, mechanics would be addressed first as this is related to the overall structure of the game and how algorithms are developed to implement the game rules. Next, dynamics focuses on how the elements of the game and player interact as the game is played and time passes. Finally, Aesthetics is described by Hunicke as "…desirable emotional response evoked in the player" [18]. These experiences and responses are a result of the player's interaction with the game. Good design would have the goal of evoking these positive responses from the player. Overall, the three design components are considered as a lens through which the developer and player view the game. The idea behind viewing the game through the lens of MDA is to view the game as separate components that are casually linked.

The lens approach is expanded greatly by Jesse Schell who developed the second game design model as a model of lenses [19]. Schell lays out a list of hundreds of lenses that build upon the MDA concept. These lenses present the design element as a series of questions that are intertwined with what Schell calls the four basic elements, *Mechanics, Aesthetics, Technology, and Story* [19]. Schell has these four elements linked as shown in Fig. 9.2. In this model, the definition of mechanics is very similar to the definition from the previous model as it states that the mechanics components consist of the rules and procedures for the game to function. In addition, Schell's model highlights that these rules and procedures should also describe the goal of the game. In this model, a separate story component is identified where the story is the driving engine that helps to define what mechanics will be needed. The Aesthetics are similar to the MDA model in that aesthetics are described as the medium in which the player perceives the game

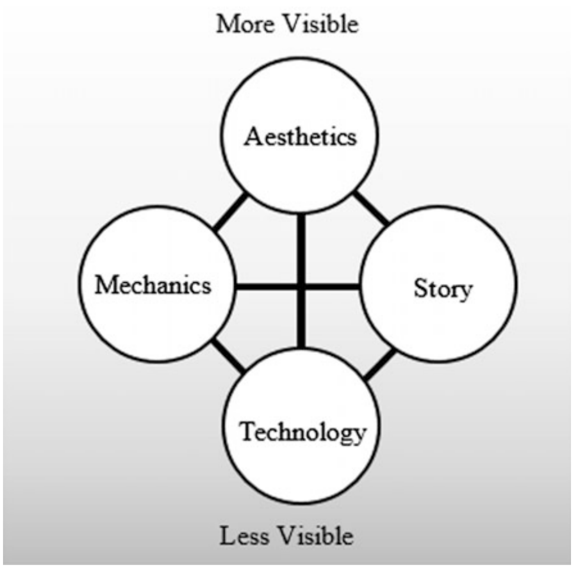

**Fig. 9.2** The elemental tetrad by Jesse Schell (*source* [19], p. 51)

mechanics and the story. This naturally includes any sensory perception as well as emotional response from the player. Schell includes technology as a fourth aspect and particularly emphasizes that the technology used to deliver the game experience deserves consideration in its own right. The definition of technology is loosely stated to encompass anything that can be used to deliver the game experience including pen and paper, cards, computers, and mobile devices. He states that the "… technology you choose for your game enables it to do certain things and prohibits it from doing other things" [19]. These four elements are built upon to show how each plays a role in the overall process of game design. Similar to Hunicke's approach of considering the player perspective, Schell includes an interpretation of which elements are more versus less visible to the player.

Tracy Fullerton leads the USC game design lab and has developed an approach similar to Schell's and the MDA framework. Fullerton's approach is composed of three main elements, *Formal, Dramatic, and Dynamic* [11]. These three elements share commonalities with MDA as well as Schell's four elements. Fullerton wraps up mechanics, logic, rules and procedures into the 'formal' element as she agrees with Schell that these elements are what separates games from other media avenues [11]. Fullerton combines the aesthetics and story from Schell's model into her dramatic element. She expands greatly in areas of what makes a game challenging and how to incorporate the nature of play into the process. Fullerton's dynamic element is similar to MDA, but expands into a description of how defining simple rules and logic in the formal element can lead to a changing dynamic environment in which the player interacts. Fullerton presents her model as an iterative game design model which can be seen in Fig. 9.3.

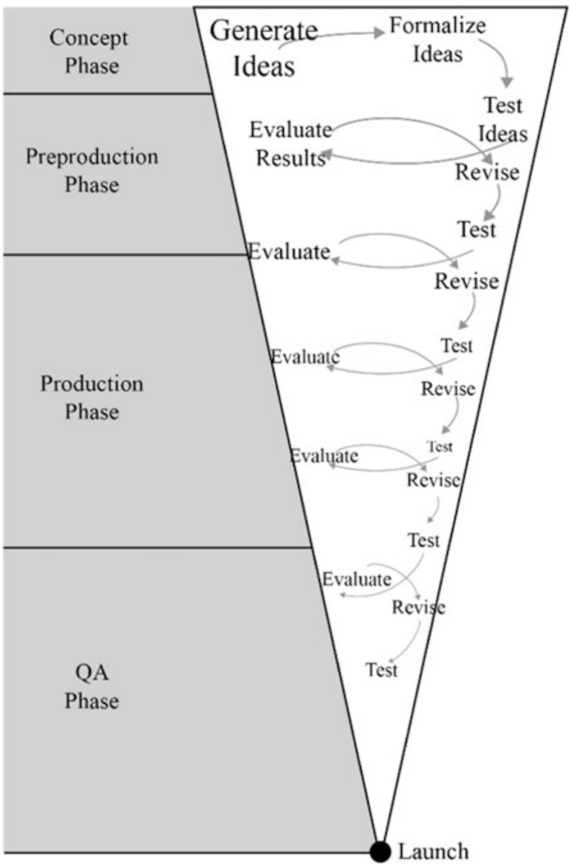

**Fig. 9.3** Model for iterative game design by Tracy Fullerton (*source* [11], p. 272)

Each model uses similar constructs but takes a slightly different approach. The MDA framework focuses on the different ways in which game designers and players approach the game. A key point from this model is that when designing games one should truly consider how the player will experience the game. Schell's model [19] is a detailed breakdown from a developer's point of view, where each of his four elements can be assigned as tasks to various team members. Fullerton's approach defines three components that can be reworked in an iterative process. When considering these three models and factoring in the importance of incorporating instructional design elements, Fullerton's approach was selected as a starting point to build the new framework as it was the most adaptable and allowed the instructional design elements to flow freely.

### 9.4.3 A Model for a Smarter Serious Game

In his discussion of smart learning environments, Hwang [12] discussed seven modules that comprise a smart learning environment including a learning status detecting module, a learning performance evaluation module, an adaptive learning task module, an adaptive learning content module, a personal learning support module, a set of databases, and an inference engine. The iMPOS$^2$inG model for smarter serious games (Fig. 9.4) incorporates each of these modules. Each module will be discussed in detail below. These modules are not only meant to provide conceptual modularity, but also modularity in software design. By designing the code in a modular way, individual modules can be modified much more easily without greatly affecting other modules.

**Capturing in-game actions and events**. Every smart learning environment needs a learning status detecting module. In smarter series games, this is the portion that allows for monitoring of player actions and detection of in-game events. Depending on the type of game, this process can be very simple or very complex. For example, in augmented reality games, the player's physical movement through space, geographic location, surrounding objects, and gesture interactions all need to be tracked. In a physical fitness training game, wearables can be used to monitor the player's heart rate, movement, and location. In contrast, for a simple turn-based game where the player interacts with the mouse, in-game actions would consist of a player taking their turn or interacting with menus.

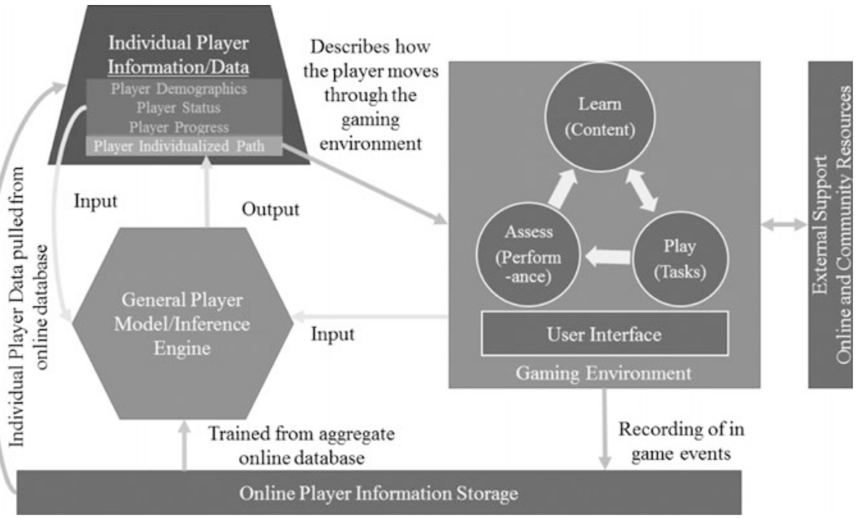
Individual Player Information/Data
Player Demographics
Player Status
Player Progress
Player Individualized Path
Describes how the player moves through the gaming environment
Learn (Content)
Assess (Perform -ance)
Play (Tasks)
User Interface
Gaming Environment
External Support
Online and Community Resources
Input
Output
General Player Model/Inference Engine
Input
Individual Player Data pulled from online database
Trained from aggregate online database
Recording of in game events
Online Player Information Storage

**Fig. 9.4** Model for smarter serious games incorporating modules needed for smart learning environment

**Storing User Data.** In order to record progress and inform a player model, player actions and in-game events need to be stored. The simplest way to do this is using local storage on the user's device. It is relatively straightforward to store data objects in a serialized format that can be accessed throughout game play. Local data storage also has the advantage of allowing users to play without an internet connection. However, storing data locally does not allow players to maintain progress between devices or provide a backup in case the local file is corrupted or lost. In addition, locally stored data is inaccessible to game developers. By using one of the many available online storage systems, developers have access to player data to monitor player progress, improve game play by analyzing player results, aid in identifying user reported problems during troubleshooting, and inform a player model to support an adaptive game.

**Player model and inference engine.** The gold standard for smart educational tools is to provide tools that adapt to the learner to support their learning based on learner characteristics, learning style, as well as past results. This requires a refined player model that serves as an inference engine where the inputs are information about the player and player actions and the outputs are in-game events such as levels being unlocked or additional content and resources being provided.

**Individual player information and data.** Individual player information needs to be accessed and stored throughout the game. By centralizing all of this information as a single entity that is persistent throughout the game, all other elements within the game have a single point of reference for the player's progress, current status, and progression through the game. Other details unique to a certain serious game implementation can easily be added to this player model.

**Learn, play, assess in the gaming environment.** The *Learn, Play, Assess* structure mimics a traditional classroom structure where material is presented, students practice the material, and finally are assessed on what they have learned. In some smart serious games, these three activities could be carried out seamlessly throughout game play so that the player is not aware of a transition between the three. However, the game developer still needs to consciously implement all three. The *Learn, Play, and Assess* modes provide the adaptive learning content module, adaptive learning task module, and learning performance evaluation module, respectively. Even though in reality these three modes of play may have significant overlap in keeping with a system dynamics approach, we will discuss each of these aspects as a mode in which the user is engaged during game play.

**User interface.** An adaptive and responsive interface is critically important in serious games as it supports all components of effective learning activities. The user interface provides context by showing the setting and background and providing the player with any narrative details. The user interface needs to accurately convey the challenge to the player and potentially provide clues on how the player can overcome the challenge. All actions and communication between the player and the game during the activity phase are provided by the user interface. Finally, feedback on performance is provided to the user through the user interface.

**External and internal learning support.** Throughout game play, content resources can be made available to players to facilitate their learning. In the

Stern2STEM program, this involves directing users to other program components including tutoring and advising. However, in general, there is a vast array of resources available for most topics in education. Simply redirecting student to a short video example at an appropriate time can be considered incorporating external resources.

### *9.4.4 The Cycle for Smart Serious Game Development*

Now that a model for a smarter serious game has been presented, a process that will allow for the development of models within that framework is required. A process incorporating game design, instructional design, and player considerations is shown in Fig. 9.5.

**Environment**. The first step in the process is similar to the collection of background information from SAM. The process begins by identifying and collecting information from subject matter experts (SMEs), target student population, curriculum and content requirements, and any existing resources including existing resources that are smart. All this information will be combined to define the environment within which the smarter serious game will be developed.

**(Re)Define**. The outset of the project will begin in a Define phase. As we will cycle through this phase many times, it will become a Redefine phase that will allow for updates to the existing definitions from feedback information. During this stage, we are defining the target audience needs and characteristics. Initially, this may be a general set of characteristics that will eventually evolve into a player model that can be used in combination with an inference engine to allow the game to become smarter.

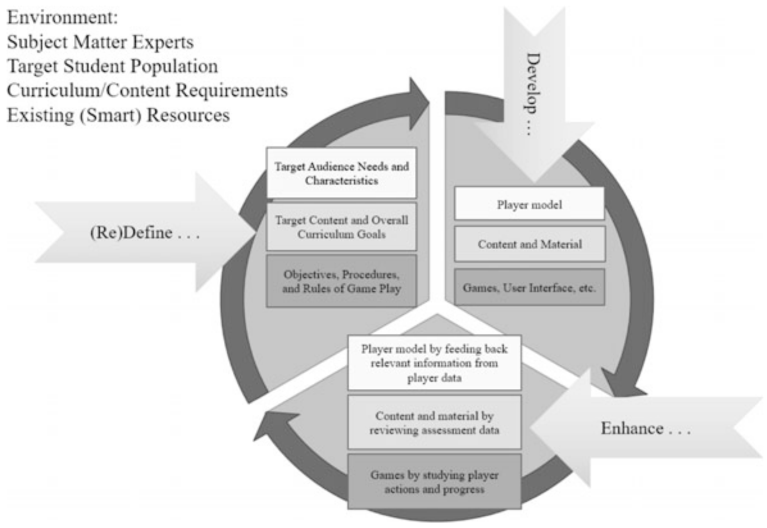
Environment:
Subject Matter Experts
Target Student Population
Curriculum/Content Requirements
Existing (Smart) Resources
Develop . . .
(Re)Define . . .
Target Audience Needs and Characteristics
Target Content and Overall Curriculum Goals
Objectives, Procedures, and Rules of Game Play
Player model
Content and Material
Games, User Interface, etc.
Player model by feeding back relevant information from player data
Content and material by reviewing assessment data
Games by studying player actions and progress
Enhance . . .

**Fig. 9.5** Process for developing smarter serious games that fits within an adaptive framework

The content and overall curriculum goals are also under development during this stage. Initially, this involves selecting a group of topics at a high level. As the process continues, determinations are made about which of those topics contain learning objectives that can be effectively conveyed through game play. Further in the process, learner feedback will be reviewed to determine if there are topics that are not being conveyed effectively or additional topics that can be added to further assist learners. Finally, as part of the game development, the overall game objectives, procedures, and rules are being defined from both a player and developer perspective.

**Develop**. Proceeding into the develop phase, development begins on the software that will eventually become the completed game. The player model will start very simply. As more information is collected, the player model will become more sophisticated with the goal of eventually having a player model that supports an adaptive learning experience. In addition, content and material development begins by creating both in-game and external resources as necessary. At this stage, the structure for the games and user interface is cultivated. As more cycles of the process are completed, these will become more mature. It is best to follow a modular design so that items in the user interface can be swapped out and replaced as needed. In addition, designing games that are reusable for a number of different topics not only shortens development time, but also decreases the burden on the user to learn a large number of different modes of play.

**Enhance**. The key benefit of rapidly developing a playable model is that feedback can be easily gained from demonstrations and players. During demonstrations, developers can collect feedback on how easy the controls are to use, if there is enough direction and help, and what may be frustrating or preventing players from completing certain tasks. Once there are a significant number of players, the data collected during game play can be used to inform and redefine the player model. In addition, the content can be enhanced by reviewing assessment data to see which topics are being effectively addressed and which are not.

This process is designed to be traversed many times throughout the development spiral so that feedback can be incorporated to improve the game. By having a modular framework for the overall game, it becomes easier to modify certain aspects that may not be satisfactory without having to change the entire game. For example, if the current data management system is not working, it can be easily replaced by only changing the parts of the code that are reporting back in-game events. A good code design will have these items isolated to a few scripts so that the modifications are minimal.

### *9.4.5 Focusing on Smartness: How the Framework Supports Smartness Features*

The ability to develop and implement smart serious games as smart learning environments enhances a smart university's ability to adapt, sense, and infer to provide an enhanced teaching and learning experience. These are three of the smartness features based on the Taxonomy of Smart Universities developed by Uskov et al. [14]. Each of these smartness features is discussed as it is highlighted in the iMPOS$^2$inG model as well as an example game that was developed using this system. The game highlighted is called MAVEN and was developed to help students obtain a better understanding of precalculus topics.

**Adaptation**. The very purpose of developing a modular framework for smarter serious game development was to enhance the developer's ability to modify existing elements to allow them to adapt to player needs. In fact, adaptation is highlighted twice in this approach. By using modularity and well-defined interfaces in the software design, the model itself is adaptable by providing the ability to interchange various components to better meet the needs of the learners. In addition, the model provides the opportunity to collect data to inform an adaptive player model and inference engine that will then be implemented to allow the game itself to adapt to the needs of individual learners.

In MAVEN, the adaptive model has been used to allow design changes driven by initial feedback and testing. For example, the original in-game menu user interface was separated and positioned at corners of the screen. Figure 9.6 shows an older implementation of the interface with the menu in the top left, help in the top right, and a back button to return to the main navigation menu in the bottom left. In addition to this physical separation which required users to interact with different parts of the screen to conduct menu interactions, the control of these areas was separated. Changing the menu user interface to one that was more compact and centrally controlled was made easier due to the modularity of the overall design. The only components that had to be modified were the menu items and their interfaces which had been kept simple to support the modular design. The new compact design is shown in Fig. 9.7.

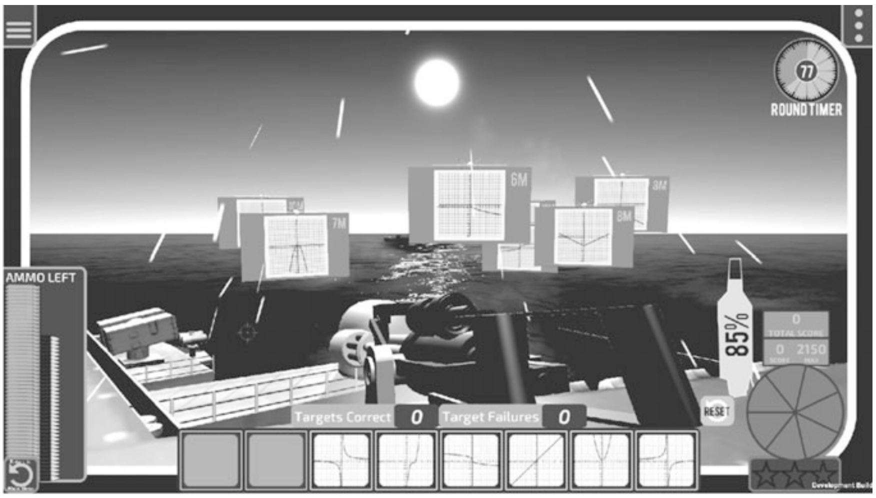

**Fig. 9.6** A scene showing a former implementation of the menu user interface

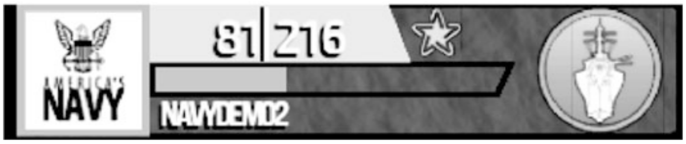

**Fig. 9.7** New menu user interface where menu controls and player status are consolidated and available in one location

Additionally, the player model itself supports adaptation. Currently in MAVEN, the player model adapts to the player's performance by only allowing the player to progress when they have successfully completed previous tasks. During play, the game is constantly collecting feedback and player data that will allow the game designers to adapt the player model to be more intelligent as time goes on. Eventually, it is feasible for the framework to support a player model that adapts the path of the user through the game based on previous performance as well as external factors.

**Sensing**. Since the game is constantly collecting player data, it is continuously monitoring the actions of the players. It collects data about how and when learners are using the game. This can contribute to awareness of unique ways in which students are engaging with the content. For example, a variety of information is collected as individuals play MAVEN. Initially, players are required to create an account and answer a series of demographic and educational background questions. This will allow developers to assess whether or not external factors contribute to the overall effectiveness of the game. Additionally, each gameplay action in MAVEN is mapped to a question that the user gets either right or wrong. For example, one of the games in MAVEN requires players to match the graphs of functions with the correct category of parent function as shown in Fig. 9.8. The player is presented with a set of missiles, each displaying a flag that shows the graph of a function. There is also a set of planes that each display the name of a type of functions.

The goal is to load the missiles on the planes that correspond to the correct parent function type for the graph. The player controls a cart that they must use to pick up missiles and transport them to the correct planes. In this game, each time the player loads a missile they are actually categorizing a function by type. The player's action is recorded as the response to a question and scored as right or wrong. Information about the problem, including the graph of the function, and the player's answer, and whether it was correct or incorrect, is stored in a time stamped online database of player information. This information can be used by developers to review player performance. It also provides training data for an intelligent player model and will provide inputs to that model for future players. For example, given that a player has missed a certain number of questions of a certain type, the model may recommend they play a different game or provide information on more formal remediation.

**Inferring**. In addition to the player inference model which makes inferences to inform the player's path through the game, the model can be used in a more general

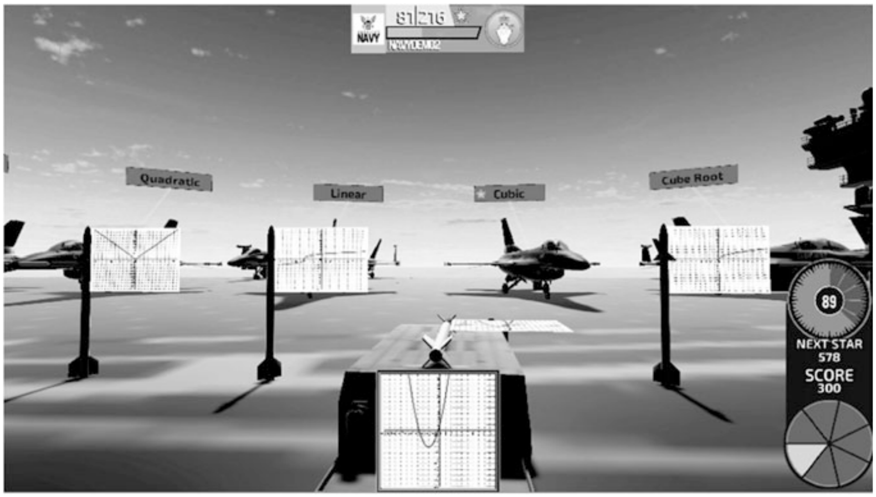

**Fig. 9.8** An example of a game from MAVEN where players practice their knowledge of library functions by matching functions with their parent type

sense. For example, by collecting data and pairing it with student data already maintained by the university, recommendations can be made in order to alert educators of students that are not performing well in particular areas. These early alerts could allow interventions that would help the student to get back on track before it was too late.

Currently, inference is implemented on a smaller scale. The Stern2STEM program provides tutoring to on-campus student veterans as well as providing education games for their use. While students are being tutored, the tutors will often encourage them to download the game and play it either in the tutoring center or at home. The tutors keep a list of the usernames of the students who are being tutored and monitor student progress through the game to help inform them about the student's progress.

## 9.5 The Model and Process in Action: Discussion and Supporting Examples

Now that the iMPOS$^2$inG model has been presented and defined, it is important to demonstrate how it can enhance the development of serious games and help them to become smarter. Following a discussion of how the framework can be applied, this will be demonstrated by specific examples using a series of games that were developed to help military veterans enhance their understanding of precalculus content through the Stern2STEM program [16].

### 9.5.1 *Refining the Player Model*

At the beginning of a project, game developers are unlikely to have a pre-developed, sophisticated player model. However, in the early stages of development, developers can research to find general information about the target audience as well as information about the content that can inform a simple player model. For example, developers may interview potential players to gather information or talk to a content expert or instructional designer to determine the best method or order in which to present content. The simple model developed may be a linear model that describes a common path that all players take through the game, progressing only when they master the previous content. As user data is collected and analyzed, this simple model can be replaced with a higher fidelity model that allows the game to adapt to the user's needs based on performance. In fact, multiple models could be developed and compared to see which best support the learning goals of the game.

As an example, the Stern2STEM project performed a literature review and relied on subject matter experts in both the veteran population and academic community to devise a descriptive set of student characteristics [16]. This set of student characteristics in addition to information from STEM educational content subject matter experts helped game developers design a primitive player model at the onset of development of MAVEN. This primitive player model was used to develop a linear gameplay progression. An illustrative example of this gameplay is described below. An image showing one of the content submenus from MAVEN is shown in Fig. 9.9.

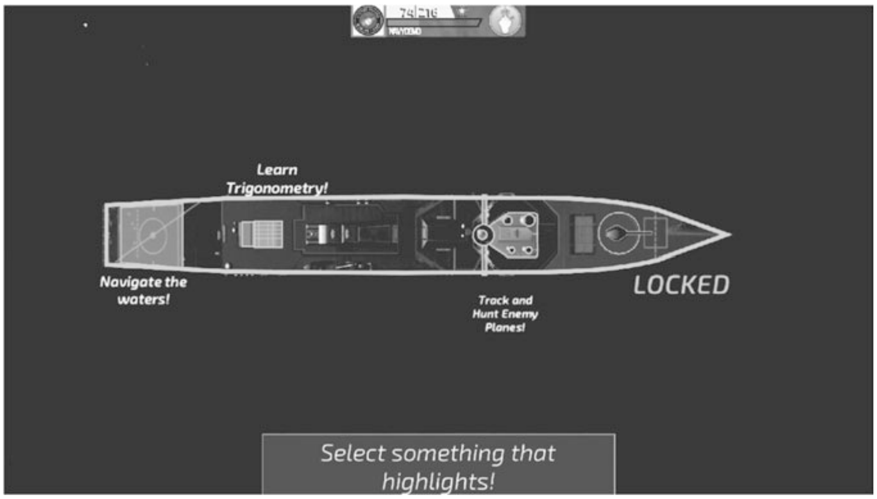
Learn Trigonometry!
Navigate the waters!
Track and Hunt Enemy Planes!
LOCKED
Select something that highlights!

**Fig. 9.9** An menu scene from MAVEN which is a serious game developed to help military veterans obtain a better understanding of precalculus content

This menu shows that the player has limited access to game areas. This access is updated based on completion of specific tasks. For example, when the player first gains access to the Trigonometry Destroyer shown, they can only access the “Learn Trigonometry!” area. Once they have gone through some content introduction in this area, they will gain access to the first play area “Track and Hunt Enemy Planes!” where they will practice the skills they have just learned. They will continue to play until it is determined that they have mastered the previous content and are ready to continue learning. This process of unlocking will continue until the player has unlocked and mastered all available play areas. We can see in Fig. 9.9 that this particular player has one play area left to unlock. Once all play areas are mastered, the player is assessed to determine their level of understanding of the content presented in the learn and play areas on this ship.

While this is certainly a primitive model that makes many assumptions about how the target population learns and employs a simple adaptive approach that is responsive only to player mastery, the modular design of Fig. 9.4 makes it easy to interchange a more advanced player model with this one. This simple model allows the development of the game to move through a full cycle, so that player data can be collected to inform a more sophisticated model. In addition, it is important to note that the player’s individual path through the game is associated with the player profile rather than then overall player model. This is essential as it allows the individual player’s path to be an output of the player model so that it can be easily individualized for more sophisticated models.

In addition, the player inference engine has inputs both from the player’s individual data including demographics, player progress, and player status as well as from the game itself. This makes it possible to develop a model that adapts the game play in a way that takes into account in-game actions, learning styles and behaviors.

### 9.5.2 Learn, Play, Assess: Bringing Together Game and Instructional Design

In the Learn mode, players are presented with content. Technology provides a variety of ways to present different content including interactive visualizations, videos interspersed with checks for understanding, and interactive pop ups that remind players of key principles during game play.

In Play mode, players engage in game play focused on practicing the content knowledge they are learning. As discussed above, in the Successive Approximation Module (SAM) for instructional design, effective learning events involve context, challenge, activity and feedback [10]. Each learning event should be structured to include these four components while also adhering to good game design. In Fullerton’s game design model, there are *Formal, Dynamic, and Dramatic* components that must be implemented throughout the game [11].

#### 9.5.2.1 Structuring Learning Events by Interweaving Instructional and Game Design

Since the goal of a serious game is to help the player learn, the first focus is on the four components of an effective learning event and the second is on ensuring that each of the components needed for game design are incorporated into each component of the learning event. This process is described below.

*Context.* The context is the way that the game is presented to the player. The Formal elements incorporated in context are the rules and the instructions for the player to follow. The Dynamic elements of context relate to how the context changes as the player takes actions within the game as well as the strategies that are promoted to the player through cues from the context. The Dramatic elements of context involve the background story and setting for the game. The context is usually dominated by the Formal and Dramatic elements.

*Challenge.* The challenge is the problem that is presented for the player to solve. The Formal element of the challenge is the actual problem itself. The Dynamic elements of challenge are how the player devises a strategy to overcome the challenge. The Dramatic elements of the challenge are the visual features and feedback that help the student recognize the problem they need to solve. The challenge is usually dominated by the Formal element.

*Activity.* The activity is the set of actions that the player needs to complete in their attempt to solve the challenge. The Formal aspect of the activity is related to the boundaries on the gameplay and the resources the player has available to them. The Dynamic element of the activity is the set of actions that the player actually takes as they attempt to solve the challenge. This includes player strategy and behavior within the game. The Dramatic element includes the visuals that change as the game state changes. This may be due to a player action or simply the passage of time within the game. The activity is dominated by the Dynamic elements of game play.

*Feedback.* Feedback is provided to the player to inform them of changes in the game due to player action or the passage of time. For example, if they solve a challenge successfully, they would be rewarded with positive feedback. If there is a time limitation, they would be made aware when the time to complete the challenge was running out. The Formal element of feedback involves the rules for awarding player score and resources based on how they behave in the game. The Dynamic aspect of feedback focuses on the relationship between game and the player. For example, if a player fails to solve a challenge, they receive an indication that encourages them to modify their strategy. The Dramatic element of feedback includes visual effects, sound effects, and text or narrative feedback that result from a player's action during activity. The feedback component is dominated by the Dramatic elements of gameplay.

#### 9.5.2.2 An Example Learning Event

Following this general overview of how instructional design and game design are interwoven throughout learning events, it is helpful to go through an example of learning events in a game that follows this model. For this example, the focus will be on the game shown in Fig. 9.10 which shows a serious game designed to help players practice their manipulation of trigonometric identities.

*Context.* The player is presented with a set of missile expressions. Each missile type has a different trigonometric expression and limited ammunition. In addition, there are ships positioned on a grid. Each ship has between two and four targets, where each target is associated with a trigonometric expression. The Formal elements are the defined rules of gameplay that stipulate that the player is able to cycle through their missile expressions and search through the target expressions. In addition, the restricted number of missiles is also a Formal element in that it places a resource restriction on the user. The Dynamic elements are involved with the ability of the player to cycle through the missile expressions and search through the ships to find target expressions. The player has a choice between keeping a set missile expression and selecting different ships, keeping a set ship and cycling through missile expressions, or some combination of the two. Each of these strategies may result in varied degrees of success. The Dramatic elements are the background story and setting. In this game, the player is attacking enemy ships in an attempt to protect their own ship. They can see their battleship in the foreground while hunting through the open ocean in the background.

*Challenge.* The player needs to fire missiles at targets that match the missile type. A match is determined by whether or not the trigonometric expression

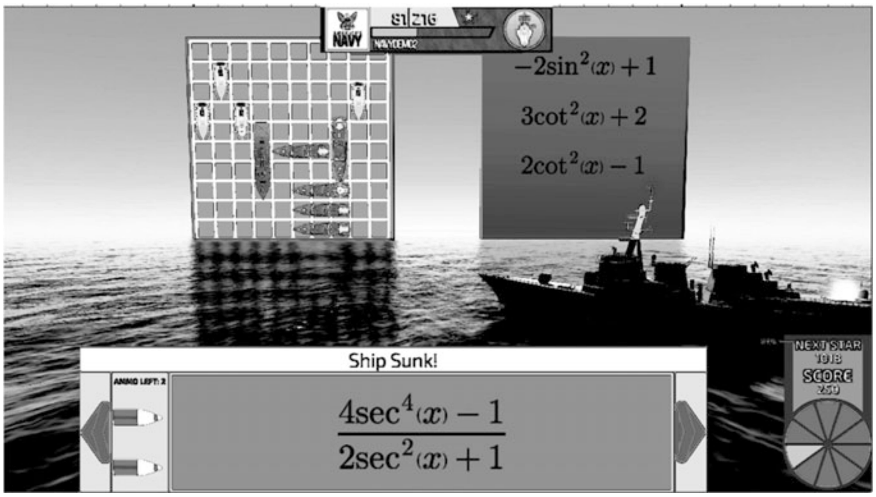

**Fig. 9.10** An example of Play mode in a serious game designed to help players practice their knowledge of trigonometric identities

associated with the selected missile and target are mathematically equivalent. This defines the Formal element of the challenge by formally identifying the problem. The Dynamic element focuses on how the player changes their strategy to address this challenge. For example, they may at first try to only cycle through the missile expressions while focusing on a single ship, but find that this takes a long time because there are a small number of target equations on each ship and most of the missile expressions will not match. They could adapt their strategy to better address the challenge. The Dramatic element of the challenge is how the challenge is interwoven into the narrative. In this game, the player needs to sink ships.

*Activity.* The player can cycle through missiles and search through various ships to find a match. They must use algebraic manipulation and trigonometric identities to decide if they have found a match. Once they have found a match, they indicate their answer by clicking on the target expression. The Formal aspect is the set of affordances and limitations that are placed on the user's activities. They can use the arrows to cycle through expressions and click to select different ships. However, their actions are limited to the number of ships on the game board and the number of missiles remaining for each missile expression. The Dramatic element is revealed in how the player interacts with the game. Their activities during game play and behaviors are both Dramatic elements. The Dramatic element is how the setting and user interface change as game play moves forward. In this case, the expressions are different on each ship and the player can see which targets they have already hit.

*Feedback.* Once the player fires a missile, feedback is immediately provided in a number of ways. The Formal aspect defines the rules by which a hit or a miss is awarded to the player. In addition, the rules for the number of points awarded for each hit and how ammunition is removed are Formal elements. The Dynamic element involves how the player may modify their strategy based on the feedback. For example, if a player got a hit, they may look to see if there are any other target expressions that are the same as the one they just hit. The Dramatic element is revealed in the way the sound and visual effects are used to indicate the result of the player's activity. In the event of a hit, an explosive sound effect is triggered, the target expression is replaced with "Hit!", and the player is rewarded with an increase in score. In the event of a miss, the player hears the missile splash into the water near the target and "Miss!" is displayed in the text feedback area directly above the missile expressions. In addition, feedback on overall progress is displayed at the bottom right of the screen showing a green segment for ships that have been sunk and a red segment for ships that are no longer able to be sunk due to a lack of ammunition.

Assess mode evaluates the player's content retention over a range of previous topics. The assessment could be built into game play, take the form of a more traditional test, or be presented sporadically throughout game play for players to earn bonus points.

### *9.5.3 Topic Selection: Focusing on Topics Appropriate for Gaming*

While topic selection may at first seem like a purely instructional or top-level project consideration, game developers quickly realize that their input in topic selection can have a large effect on the overall success of a serious game. Because of the time and expense required to develop a serious game, it is first important to narrow down the list of topics to those that students struggle with that are not appropriately addressed by existing methods. If there is no demand for a new way to teach a certain topic, developing a game to do so will have little chance of being successful. Once the list of topics has been narrowed to a subset, the game developers need to consider how effectively the content can be conveyed within the confines of a game. Sometimes this takes more than a little imagination; but making sure that the game play is well designed and based solidly in the content is important [20]. In a traditional classroom, getting students to do their homework is a constant struggle; however, one of the main assets favoring gamifying education is that when students are engaged in game play, they are effectively practicing skills. Whether they are practicing skills through homework or game play, the outcome of practicing and engaging with the material is the same [21].

The list below defines a set of criteria that are important in selecting content areas which can be successfully incorporated into games. While it is not necessary for each topic to meet all the criteria, identifying topics that meet most of the criteria helps to narrow the range of topics to those that are more easily adaptable to being presented through game play.

- The topic is not satisfactorily addressed by an existing resource.
- There is a known deficiency in the understanding of the topic in the target population.
- The topic would be enhanced by interactive visualizations.
- The topic involves some process of cause and effect that is not easily visualized by traditional methods.
- The topic involves a learning process that can be broken down into manageable steps that can be turned into effective learning events.

It is important to note that some topics may not be covered completely by serious games. However, taking the most common stumbling block in a process and creating a game around that portion can help students gain the skills and confidence they need to learn the rest of the topic by more traditional methods. Game designers should not be afraid to mix serious games with external resources to present the student with the best possible learning environment. In fact, one of the key features of a smart learning environment is that it provides timely access to external resources to promote student understanding.

**Example topic selection.** Students in precalculus traditionally struggle with graphing using transformations. This topic is particularly important as they move on in mathematics as more advanced courses are often dependent on student

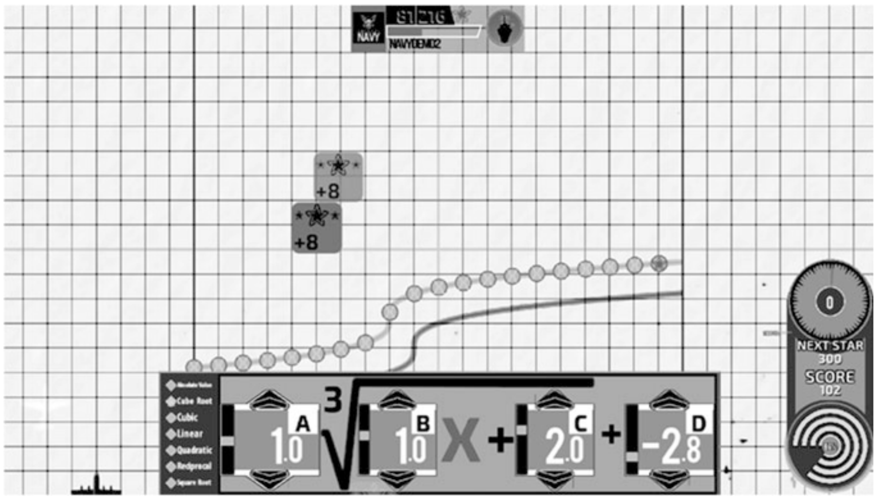

**Fig. 9.11** A serious game that helps students learn about graphing by transformations

understanding of how functions change. The existing method to teach this in the classroom is to process the function transformations one at a time and move the individual function points. This leads to difficulty for the students as they attempt to visualize the overall function transformations. This topic meets all five of the conditions above.

In order to address this topic, an interactive game has been developed. This game is shown in Fig. 9.11. This game will be presented in terms of the components of an effective learning event to give another example of how to apply the iMPOS$^2$inG framework. After giving this overview, emphasis will be placed on the components that make this game a good match for the selected topic.

*Context.* The player is presented with two graphs that are identified as paths for their ship to take through the ocean. The darker grey path is the desired path while the lighter gray path is the route that the player is charting. The equation for the lighter gray graph is shown at the bottom of the screen.

*Challenge.* The player needs to choose the correct parent function from the grey area and the correct equation coefficients to match the lighter gray graph to the darker grey graph. There is limited time as the ship is already traveling along the path.

*Activity.* The player can use the toggle menu to the left to select the correct parent function. Then, the player can use the up and down arrows to increase or decrease the values of the four coefficients in the equation to modify the graph to match the target graph.

*Feedback.* Every second, the player is awarded points based on how closely their graph matches the target graph. These points are shown using score pop-ups. This rewards players for working more quickly. In addition, players can see how close their graph is by viewing the bull's-eye pattern at the bottom right that indicates a percent match between the two curves.

This game overcomes deficiencies in the traditional method of instruction by allowing the player to manipulate the equations themselves and immediately see how their changes affect the graph. This is a quicker way for them to understand the function behavior than graphing by hand. Once they have a firm understanding of the effects of changing the different coefficients, they will be better equipped to move to the next learning objective which would be sketching the graphs by hand.

## 9.6 The Model and Process in Reflection: Discussion and Lessons Learned

### *9.6.1 Broader Applications*

Though the game used as an example in this work was developed to help learners master precalculus skills, the overall framework and model developed can be used to develop games for a variety of disciplines. Particularly, it would be easy to extend the model to cover courses that are problem solving focused since the model depends on mapping in-game actions to questions that can be scored as either correct or incorrect. Examples of subjects that meet this description include calculus, physics, chemistry, statics and dynamics. Students in all of these subjects traditionally have difficulty visualizing problems and situations that arise, and a virtual interactive environment would therefore support learning.

In fact, if a series of interconnected games were to be developed, instructors could use a similar environment across a range of classes while receiving feedback and early warnings when students were struggling with particular sections of content. Stretching this idea further, virtual and augmented laboratories could be incorporated as well to bring students a common virtual experience that would allow them to engage with a number of smart systems as they completed their coursework.

One thing that is interesting about the overall process is that it is general enough to be applied to a variety of courses at a smart university. Though some of the games developed for MAVEN are specifically tailored towards precalculus or mathematics content, other games could easily be modified to include material from a variety of subjects. For example, the destroyer game shown in Fig. 9.10 is used to help students learn how to solve trigonometric identity problems. However, the root game play action is matching. Therefore, any class which poses questions that can be expressed in a matching context can be employed in this game. This includes a variety of subjects including science, history, and business.

## *9.6.2 Comparison to Existing Products*

At this time, there are no other serious game in mathematics for higher education that truly incorporate mathematics into gaming. There are other excellent interactive mathematics resources, but they are mainly sources of content with interactive questions interspersed. Khan Academy [22] is one excellent example of an interactive educational tool that is available on a wide variety of devices and presents a broad range of topics including precalculus and calculus. Khan Academy presents content mainly as videos with some typed notes interspersed. Students are invited to practice what they have learned by completing questions between videos. There are a variety of question types, including multiple choice, text entry, and a graphing utility. Even though Khan Academy provides some aspects of gamification including badges and achievements, it is not technically a game. Serious games, such as the one developed here, complement the existing content resources by encouraging students to engage in practice in the form of gameplay.

## *9.6.3 Lessons Learned*

Throughout the development of this process and framework, there have been quite a few bumps in the road. By being fastidious about employing modularity to both the conceptual and software design, the flexibility required to adapt to changing requirements and incorporate feedback along the way has been maintained. In this section, a discussion of several issues that were encountered and how these issues were surmounted will be provided.

### 9.6.3.1 Developing and Integrating External Tools for Equation Display

The first issue was the display and manipulation of symbolic expressions and graphs. The solution was to develop a custom toolkit to support the development of the games [23]. Once this toolkit was implemented, it underwent many revisions as more content was continually added to the game. By having a modular design where the toolkit was only accessed through a single problem generating script, the ability to make changes easily was preserved. In fact, even switching to another tool, if required, would not have been difficult. In addition, there were instances where real-time player interaction with functions was required. Because the toolkit was calling Python scripts from the game, the implementation was not fast enough to provide the desired real-time performance. In these instances, custom solutions were developed while only modifying the interface between the problem generating code and the specific game under development.

#### 9.6.3.2 Using Modularity and Communication to Support Collaboration

Another issue encountered was with version control. A common problem in game development is the handling of asset files by version control software. The solution to this turned out to be a combination of modularity and communication. Before the framework presented in this chapter was being utilized in the project, an unsuccessful attempt was made to implement version control. This led to a great deal of copying files, sending updates back and forth through email, and a significant amount of repeated work. Since the framework has been implemented and project components have been carefully set up to be independent, different developers are able to work on different aspects of the project simultaneously. As long as there is good communication in place, there are very few issues merging after modifying asset files in binary format. One of the keys to this success has been the use of Bitbucket [24] as a cloud hosting service for the entire project. Using Bitbucket along with the git bash terminal rather than a git tool that had a graphical user interface, provided a high level of control when merging files which prevented many issues.

#### 9.6.3.3 User Interface Redesign and Implementation

Another issue faced involved the user interface. Initially, the assumption was made that the user interface would be essentially the same in every scene, so that it would be smart to have it be one consolidated piece where individual components could be turned on and off. This design seemed modular at first because the components of the user interface were all together. However, over time the user interface became unwieldy to work with and modify and was a bottleneck in the work flow. Looking back at the overall framework, there was not enough separation in this original model. While the user interface can be viewed as one component, different pieces of the user interface are intimately involved with different modules in the framework. After this realization, it made much more sense to break the user interface into functional pieces that were associated with different framework modules. This current implementation has a player menu, feedback menu, and user input menu. Each of these menus is highly customizable and closely tied to the module it is associated with rather than being tied to other user interface elements.

#### 9.6.3.4 Integrating an External Player Data Management System

Data management also became a problem early on. Initially, all of the player data was kept locally on the user's machine, but this limited the model use case and jeopardized the player's experience by prohibiting developer access to player data and not allowing players to maintain progress between devices. As the development of MAVEN continued, this concept was partially reworked as the decision was made to host player data on a server so that the players could access their own data from different devices and the developers had access to player data to allow for analysis that would lead to game improvements.

This solution was implemented quickly by taking advantage of the existing project web hosting services and using a Wordpress plugin within the development engine. It turned out that this setup is limited and should only be used for small amounts of player profile data including email, player avatar name, and other non-duplicate data structures. The approach does not work well for storing data that is generated from in-game actions and events because of the sheer quantity of data generated during play. Since a great deal of data needed to be collected in-game to improve the game and eventually inform an adaptive player model, alternative solutions were considered. A better approach is to use a system that stores data in a relational database system. This allows not only for the storage of data, but the eventual retrieving and processing of data to inform developers based on user progress and actions. While there are many existing services that would meet the project needs, one example of a suite of packages that addresses data management problems is Amazon Web Services (AWS) [25]. A major benefit of these services is their ability to be integrated with existing social media sites where users may already have accounts. This can be vitally important to retaining users and being able to compare various user player data models. The combination of offloading data management to a system like AWS and using a complementary social media platform allows more interaction among players and allows us to take advantage of leading cloud storage systems. The cloud model is an industry standard in software development as it provides the developers with an enhanced package of tools, takes advantage of economies of scale, and automatically adjusts to handle various user capacities. Players will be able to log in from a wide range of platforms and experience the same result.

In particular, cloud based storage systems offer the opportunity to use their services to handle data management, employ industry standard encryption for user data, integrate social media platforms, broaden the user base, and importantly build a complex system of player data models. These player models can take advantage of leading technology trends in machine learning, adaptive learning, and smart learning models. Using a cloud based relational storage system in combination with an accurate player model can provide developers and educators with valuable information and allow that information to inform complex player models that will provide players with the smartest player model. This player model can dynamically adapt to the player's current level of knowledge. Figure 9.12 provides a visual representation of how these updated services would connect into the existing

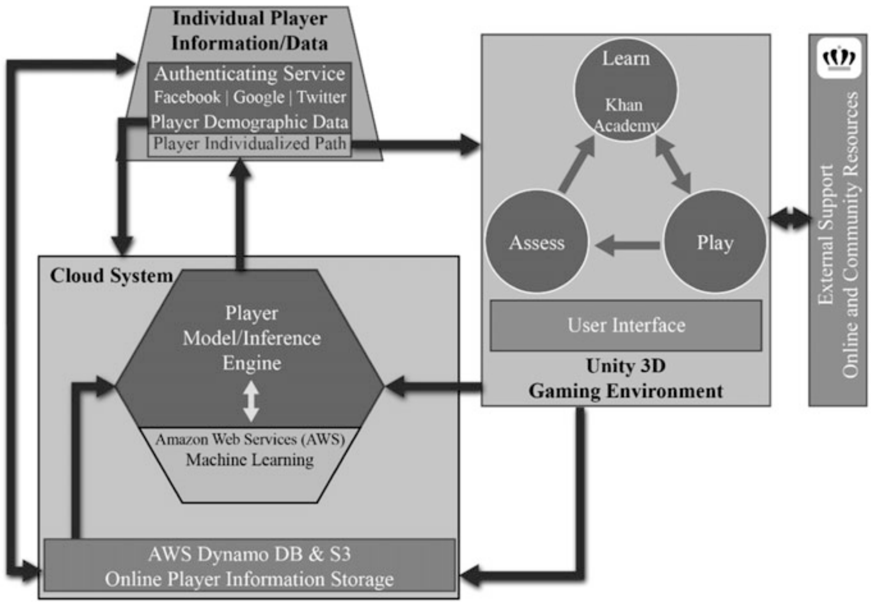
Individual Player Information/Data
Authenticating Service
Facebook | Google | Twitter
Player Demographic Data
Player Individualized Path
Learn
Khan Academy
Assess
Play
User Interface
Unity 3D Gaming Environment
External Support Online and Community Resources
Cloud System
Player Model/Inference Engine
Amazon Web Services (AWS) Machine Learning
AWS Dynamo DB & S3 Online Player Information Storage

**Fig. 9.12** Using the model for smarter serious games incorporating AWS service modules needed for an adaptive, smart learning environment

model. The overall conceptual model does not change, but the modular components are replaced to provide a smarter learning environment.

### 9.6.3.5 Game Manager Implementation

The final lesson learned was about overall game control. Since the design is modular, it makes sense to have a state machine that can transition the game between a set of known states. This allows for events that are common for all modules to be centralized while individualized events can be handled by their respective modules. These states are easy to implement from a single script and it is relatively simple to add another state should the need arise. Currently, MAVEN has the following six states:

- Initialize: Allows for initial variables to be set at the beginning of a menu or game. The online data functions that need to be called at the beginning of every game are centralized.
- Warm-up: Allows for a variable length pause between player entry into a scene and the start of a game.

- Demo: Incorporates the use of guided prompts to instruct a user in how to play a certain game.
- Play: Regular mode of play where the user is completing learning events.
- Menu/Pause: Allows the game to pause and open the menu at any point. Allows for central handling of pause and menu functions from any scene.
- End: Allows for a common end to each game where a round over menu with feedback on the level just completed and navigation buttons is presented to the player.

Centralizing the transition of these events to a single game manager which interfaces with individual game controllers in every game has proved to be a successful way to handle transitions between states as well as assisted in getting new games up and running quickly by handling the common events that occur in every game.

## 9.7 Conclusion and Future Work

In this chapter, instructional design and software design models have been combined into a novel modular framework and accompanying development process aimed to facilitate the design and development of smarter serious games. Under this framework, each smart serious game can be viewed as a smart learning environment that enhances the smartness of an overall university by contributing to the development of individual smartness features. In addition, this framework and process have been demonstrated through their application to the development of an evolving serious game designed to assist military veterans in enhancing their understanding of precalculus topics as they return to pursue engineering degrees. This process demonstrates how to maintain a focus on the target player population, instructional design principles, and game design principles in order to develop a serious game that has interchangeable components to facilitate continued enhancement and development over time.

In the future, additional data will be collected from this game and others developed using this framework. It will be particularly interesting to perform an efficacy study and then incorporate the results back into the games that have been developed. In addition, future work will include incorporating additional elements that are known to engage users in game play, such as high score boards to facilitate competition, additional rewards including unlocking special items for successful game play, and interactive in-game assistance to provide just-in-time learning during game play. Additionally, it is of interest to further modularize the user controls in order to modify controls by player preference for a variety of devices and to address accessibility concerns. Finally, the work will be continued by developing additional games within STEM fields including calculus, physics, and chemistry.

**Acknowledgements** Many of the games that have been provided as examples to elucidate game and instructional design principles were developed as part of MAVEN: Mathematics for Veteran Education, which is part of the Stern2STEM program made possible through the Office of Naval Research STEM under ONR GRANT11899718.

## References

1. Anderson, M.: Technology Device Ownership (2015)
2. Rubin, A.: Technology Meets Math Education: Envisioning a Practical Future Forum on the Future of Technology in Education (1999)
3. Williams, D.L., Boone, R., Kingsley, K.V.: Teacher beliefs about educational software: A delphi study. J. Res. Technol. Educ. **36**(3), 213–229 (2004)
4. Hussain, TS., Roberts, B., Menaker, ES., Coleman, SL., Centreville, V., Pounds, K., Bowers, C., Cannon-Bowers, JA., Koenig, A., Wainess, R.: Designing and developing effective training games for the US Navy. M&S J., 27 (2012)
5. Kato, PM.: Video games in health care: Closing the gap. Rev. Gen. Psychol. **14** (2): 113 (2010)
6. Connolly, T.M., Boyle, E.A., MacArthur, E., Hainey, T., Boyle, J.M.: A systematic literature review of empirical evidence on computer games and serious games. Comput. Educ. **59**(2), 661–686 (2012)
7. Charsky, D.: From edutainment to serious games: A change in the use of game characteristics. Games Cult. (2010)
8. Murphy, C.: Why games work and the science of learning. In: Interservice, Interagency Training, Simulations, and Education Conference, Citeseer, pp. 260–272 (2011)
9. Ruggiero, D., Watson, W.R.: Engagement through praxis in educational game design common threads. Simul. Gaming **45**(4–5), 471–490 (2014)
10. Allen, M., Sites, R.: Leaving ADDIE for SAM: An agile model for developing the best learning experiences. Am. Soc. Training Develop. (2012)
11. Fullerton, T.: Game Design Workshop: A Playcentric Approach to Creating Innovative Games. CRC press, Boca Raton (2014)
12. Hwang, G-J.: Definition, framework and research issues of smart learning environments-a context-aware ubiquitous learning perspective. Smart Learn. Environ. **1**(1):1(2014)
13. Bellotti, F., Berta, R., De Gloria, A.: Designing effective serious games: Opportunities and challenges for research. iJET **5**(SI3): 22–35(2010)
14. Uskov, VL., Bakken, JP., Pandey, A., Singh, U., Yalamanchili, M., Penumatsa, A.: Smart university taxonomy: features, components, systems. In: 2016 Smart Education and e-Learning, pp 3–14. Springer, Cham (2016)
15. Uskov, A., Sekar, B.: Smart gamification and smart serious games. In: Fusion of Smart, Multimedia and Computer Gaming Technologies, pp 7–36. Springer, Cham (2015)
16. Dean, AW.: A Pilot Program for the Recruitment and Education of Navy Veterans Based on System-level Technical Expertise and Leadership Maturation Developed during Service
17. Entertainment Software Association: Essential facts about the computer and video game industry: 2015 sales, demographic and usaged data. Entertainment Softw. Assoc., Washington, DC (2015)
18. Hunicke, R., LeBlanc, M., Zubek, R.: MDA: a formal approach to game design and game research. In: Proceedings of the AAAI Workshop on Challenges in Game AI, p. 1 (2004)
19. Schell, J.: The Art of Game Design: A Book of Lenses. CRC Press, Boca Raton (2014)
20. Squire, K.D., Jan, M.: Mad city mystery: developing scientific argumentation skills with a place-based augmented reality game on handheld computers. J. Sci. Educ. Technol. **16**(1), 5–29 (2007)

21. Bavelier, D., Green, C.S.: The brain-boosting power of video games. Sci. Am. **315**(1), 26–31 (2016)
22. Khan Academy (2016). https://www.khanacademy.org/. Accessed 18 Nov 2016
23. Smith, K., Shull, J., Dean, A., Shen, Y., Michaeli, J.: SiGMA: A software framework for integrating advanced mathematical capabilities in serious game development. Adv. Eng. Softw. **100**, 319–325 (2016)
24. Atlassian Bitbucket (2016). https://bitbucket.org/. Accessed 27 Sept 2016
25. Amazon Web Services Inc. (2016). https://aws.amazon.com/. Accessed 22 Sept 2016