The Game Engine Space for Virtual Cultural Training: Requirements, Devices, Engines, Porting Strategies and a Future Outlook

Technology White Paper

June 9, 2010

The University of Texas at Dallas
Arts and Technology
Marjorie A. Zielke, Ph.D.
margez@utdallas.edu
The Game Engine Space for Virtual Cultural Training: Requirements, Devices, Engines, Porting Strategies and a Future Outlook

Executive Summary

This paper discusses current trends in gaming platform capabilities, and their relevance to cultural modeling and simulation. We explore future developments in these industries and identify how our development strategy will accommodate changes in devices, technology or target platforms. We outline our creation of a middleware and central repository strategy to allow for quick and robust development across multiple platforms, and we propose the best engines for current development needs.

To begin our analysis, we identify ten distinct development criteria for virtual cultural training in a game-based environment discussing factors such as behavioral, psychological and cultural representation and ability to link to OneSAF, JNEM and other similar models and the respective data repositories that feed them. Through this lens we analyze the engines, platforms, and suggested optimal development processes.

Next, the impact of social media and the social-networked communications space is weighed in context of the affect on game development. We suggest that a social-networked communications space is quick to market, affordable, secure, recursive, and leverages organic, community-based development.

In order to strengthen our analysis and reinforce our guiding concept of non-platform-specific development, this paper investigates other platforms besides the traditional PC/Mac model. This investigation illustrates the many ways that FPCT development might have to change to accommodate a new platform, and the new functionality that these platforms might bring to current FPCT development. Platforms discussed include iPhone, Android, and iPad among others.

Relevant game engines are then presented and compared to a list of key engine attributes for cultural training development. Snapshots of five pertinent engines to include Unreal Engine 3, CryENGINE 3, GameBryo, Unity and VBS2 are presented. These snapshots include strengths and weaknesses, pricing strategy, and developer culture and community. The engines are scored and displayed in a chart comparing “meets requirements” with “crossplatformness.” Per this analysis, and taking into account current development experience, Unreal Engine 3 is identified as the best choice for current development, and Unity 3D is recommended as the best choice if a mobile platform strategy becomes a requirement.

Next, our middleware development strategy is discussed. Assets are rated according to how inherently portable they are, and our development strategy of creating a central repository of assets and using a custom middleware solution is outlined.

Finally, our view of the future of game engines describes key factors in this future development space. We see development moving away from increasing hardware demands, towards portability of development, increased connectedness, and crowd-sourced development and testing.
# Table of Contents

I. Introduction ............................................................................................................. 3

II. What is the Purpose of a Game-Based Virtual Cultural Trainer? .......................... 3

III. Development Criteria of Virtual Cultural Training in a Game-Based Environment ........................................ 4

IV. I. The Social-Networked Communications Space and its Affect on Game Development ......................... 8

V. Other Types of Platforms: The Need for Cross-Platform Development .................. 9

VI. Game Engine Analysis and Ranking ..................................................................... 13
   a. Unreal Engine 3 ................................................................................................. 14
   b. CryEngine 3 ..................................................................................................... 15
   c. Virtual Battle Space 2 (VBS2)/Real Virtuality 3 ............................................... 16
   d. Gamebryo Lightspeed ....................................................................................... 18
   e. Unity 3D ........................................................................................................... 19

VII. The Way Ahead: FPCT Core — A Central Repository and Middleware ................. 27

VIII. The Future of Game Engines ............................................................................. 32

IX. Summary ............................................................................................................. 33

X. Appendix A: Game Engine Overview on Specific Technical Variables ................ 34

XI. Bibliography ...................................................................................................... 42
The Game Engine Space for Virtual Cultural Training: Requirements, Devices, Engines, Porting Strategies and a Future Outlook

Introduction

This white paper outlines the game engine space particularly as it applies to virtual cultural training. To this end, this report includes specific functionality and trends in virtual cultural training in a game-based environment and factors in social-networked driven communications which also affect the space in which game engines operate. These factors are then used as evaluation criteria to analyze gaming engines currently on the market with their specific features. The paper discusses other sector devices potentially important to the project such as iPhone and Android mobile platforms as well as the iPad. The paper also discusses a recommendation to pursue a non-platform specific development strategy to more easily take advantage of various technologies and platforms that become available over time. Finally, a forecast of game engine technology evolution likely to occur over the next few years is presented. Most of the ideas and analysis presented here is a result of two years worth of development on the First Person Cultural Trainer (FPCT) as well as hands on technology reviews and secondary research sources.

FPCT is a high-fidelity, game-based simulation, which trains cross-cultural decision-making within a 3D representation of a Middle-Eastern society. The current version of FPCT is a computer-based video game built in the Unreal Engine 3. The project is currently in its second year or spiral of development. FPCT is part of the Hybrid Irregular Warfare Improvised Explosive Device Network Defeat Toolkit (HI2NT) sponsored by TRADOC G2 Intelligence Support Activity. HI2NT is a federation of virtual, constructive and gaming models, providing a combat training center (CTC)-like experience for individual through brigade-staff training audiences. The overall HI2NT program is also in its second year of development. FPCT functions as a stand-alone cultural trainer, or as part of the Federation.

Due to the fast-changing nature of technology, it is anticipated that this white paper will be a living document which will need to be updated often perhaps in a whiteboard space – as both needs and technology evolve.

What is the Purpose of a Game-Based Virtual Cultural Trainer?

A game-based virtual cultural trainer will help to educate, train and internalize understanding of different cultures, verbal and nonverbal communication, rapport and mission essential tasks that will assist in solving problem. Game-based virtual cultures help to prepare war-fighters for nonkinetic tasks. Inherently, game-based cultural trainers must be nonlinear and reflect unpredictable outcomes. This type of development needs to capture the essence of group behavior such as that displayed in social networks and different types of culturally appropriate ways of approaching and working with different members of the populace, to include elders, women and community leaders. Game-based virtual cultural trainers emphasize verbal and nonverbal communications and ability to affect mood and emotion in the community through appropriate gameplay decisions.
Development Criteria of Game-Based Virtual Cultural Trainers

Ten distinct development criteria have been identified for virtual cultural training in a game-based environment. These criteria are important to consider because they may be different from inherent capabilities of more traditional first person shooter game platforms. Further, some of these capabilities are not in any engine platform, per se and require custom development. These are:

Table 1. Ten Distinct Development Criteria for Virtual Training Game-Based Simulations

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Data-base driven culturally appropriate character, group and community development with nonverbal and verbal communication</td>
</tr>
<tr>
<td>2.</td>
<td>Behavioral, psychological and cultural representation</td>
</tr>
<tr>
<td>3.</td>
<td>Ability to link to OneSAF, JNEM and other similar models and the respective data repositories that feed them.</td>
</tr>
<tr>
<td>4.</td>
<td>Well-developed conversation system for story telling that calls other interactive story development elements such as animations</td>
</tr>
<tr>
<td>5.</td>
<td>Emphasis on ongoing development of layered and nuanced realism to include multi-layered sound. This includes character modeling, environment modeling, sound representation and cultural representation.</td>
</tr>
<tr>
<td>6.</td>
<td>Speech-to-text/text-to-speech capability</td>
</tr>
<tr>
<td>7.</td>
<td>Nonlinear gameplay</td>
</tr>
<tr>
<td>8.</td>
<td>During and After Action Reviews</td>
</tr>
<tr>
<td>9.</td>
<td>Quick composability to link subject matter experts to the game platform</td>
</tr>
<tr>
<td>10.</td>
<td>Ability to port community vetted assets to other or emerging platforms</td>
</tr>
</tbody>
</table>

Each of these criteria is discussed below.

1. **Database-driven culturally appropriate character, group and community development with nonverbal and verbal communication:** Emphasis is on being able to represent characters in culturally appropriate emotions and for the characters to be able to read the emotions as expressed through conversation, actions, facial and body expression. Characters should be able to express a language that is appropriate for the culture in question. See Figure 1 on the following page.

2. **Behavioral, psychological and cultural representation:** The characters should not only act and sound culturally appropriate, but they should act both individually and as groups as representations of the culture.
Figure 1. The game unfolds through conversations with characters to discover intelligence-like information in the community. Through these interactions the player can rate the characters. A key development objective of FPCT is to present game characters capable of showing moods like neutral, angry, fearful and glad through nonverbal communication.

3. Ability to link to OneSAF, JNEM and other similar models and the respective data repositories that feed them: Various other data sets are available that provide information that is used in the cultural game. This includes cooperation level, mood, golden nuggets or critical pieces of information, and other factors stored in these repositories. These repositories are key because they integrate information created by subject matter experts into the game, they provide centrally vetted information, and they effectively allow game players to work with the data…for example trying to change community mood…and put this data back into the repository to study further affects.

Figure 2. In HI2NT spiral two, underway this year, story elements for FPCT Core will be input from OneSAF and JNEM.
4. Well-developed conversation system for story telling that calls other interactive story development elements such as animations. Essentially, the cultural game play unfolds through the story of the environment told through conversations that the players meet and sound, nonverbal communication and other reactions to the characters of the story. The story includes the golden nuggets of the environment.

The Conscript™ Conversation System

Figure 3. Conscript™ is a conversation system developed for FPCT Core that integrates many game elements to tell stories that lead to golden nuggets.

5. Emphasis on ongoing development of layered and nuanced realism to include multi-layered sound. Adding realistic layers and nuance to the environment is an ongoing task. For example, currently sound is represented on four layers in the environment to include conversation, local, regional and global. In addition, the physical representation of the area should represent a geotypical environment that would be seen in an actual area. Nuanced realism does not only refer to the physicality of the game-based cultural trainer. Behavior representations and enhancements are also ongoing. So, for example, social network models can be added to the behavioral, psychological and cultural representations to make the character and group interactions, and therefore the quest for golden nuggets, more realistic.

6. Speech-to-text/text-to-speech capability: The game play of cultural training needs to be as realistically nuanced as possible. This is because it is so deeply entrenched in cultural nuance, golden nugget discovery, mood determination and other similar factors, and because it is basically technology that helps the war fighter or other subject matter expert learn how to work with live people and cultures. Therefore, the ability to speak through verbal communication directly to characters in the game, or through an interpreter is advantageous.

7. Nonlinear gameplay: Life does not unfold linearly, whether in a war theater or otherwise. Further, compelling gameplay demands that successful participation is not predictable or obvious. Particularly in a culturally driven environment, a slightly different action could have a totally different gameplay outcome.
8. **During and After Action Reviews:** The during and after action reviews are where the players receive feedback on their effective navigation of the culture to accomplish missions. The information needs to occur during and after gameplay. Players can win the game based on their successful identification of particular cultures, their ability to identify and affect moods in the community, and their ability to uncover golden nuggets and accomplish missions. One common requirement of the reviews is to be able to determine cause and effect of the above outcomes.

9. **Quick composability to link subject matter experts to the game platform:** Cultural representations are usually very nuanced and can also be different depending on the situation. They also can be a reflection of the point of view of the subject matter expert (SME)…which can also differ from SME to SME. What, for example, is a “typical” American? For this reason, it is important that the game engine platform offer the capability for the SME to quickly and easily adjust the cultural representation of the game to represent the teaching point or particular point of view.

**Composable Behavioral, Psychological and Cultural Representation in FPCT**

![Diagram](image)

Figure 4. FPCT includes behavioral/psychological/cultural modeling tools as shown on the right. The culture design tool allows SMEs to create specific cultures, as warranted. As part of the HI2NT Federation, next-phase development of these moods will be defined from JNEM, which will allow the creation of real-time immersive environments.

10. **Ability to port community vetted assets to other or emerging platforms:** Typically, we think of game-based engines or culture or anything else for that matter as solely pc or console (i.e. XBox, PS3). This is increasingly not the case. Mobile platforms such as Android or iPhone or tablet type hardware such as iPad are also viable gaming platforms for training. Further, it usually takes substantial investment to develop tools that can be agreed upon as accurate representations. Once a model, asset or another type of development is successfully completed, it is important to be able to reuse the asset in other situations, or perhaps in conjunction with other situations. So for example, an accepted way of representing social networks could be used on a variety of platforms and enhanced by the community over time.
The nature of communication and information dissemination has evolved dramatically over the last fifteen years because of the web. While social network communication such as that featured by the current popular tools such as Facebook, Twitter or texting gets much attention, these trends toward socially networked media and information dissemination have been occurring since the inception of the web, and are really not new phenomena although the overall popularity of the tools is increasing. This trend toward social-networked communication does have implications for overall competitive advantage. Communication and information dissemination projects that follow a social network paradigm tend to be organic community-driven systems that facilitate quick to market recursive development. Security also needs to be balanced in this increasingly open environment. The chart below illustrates these characteristics with their implications for Virtual culture gaming engines.

**Table 2. Social Media-Based Development Construct Dimensions and their Implication for Virtual Game-based Cultural Development**

<table>
<thead>
<tr>
<th>Social Media-Based Development Construct Dimension</th>
<th>Description</th>
<th>Game Engine Implication</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Organic Community</strong></td>
<td>The development is rapidly changing and driven by the collective community of end users and developers. To this end, it offers a spectrum of transparency to the layers of community down to the lowest echelons and up to the highest echelons. A dynamic innovation process is nurtured. Ideas flow freely throughout the network.</td>
<td>Crowdsourcing or open development by community contributes to a grass roots perspective and a non-hierarchical development organization. Expert developers specialize in user interface, high-level, and <em>new</em> development concepts and turn content development over to community as soon as possible. Mining user content for rules and themes becomes another function of the experts.</td>
</tr>
<tr>
<td><strong>Quick to Market</strong></td>
<td>Hand to warfighter and other users as needed. Low barrier to entry.</td>
<td>The non-hierarchical user community defined above can create faster development because it allows users to develop and reserves development experts for the activities described above.</td>
</tr>
<tr>
<td><strong>Secure</strong></td>
<td>Security balanced with transparency and low barrier to entry.</td>
<td>Nature of social network development puts new stresses on security and new paradigms must be created.</td>
</tr>
<tr>
<td><strong>Affordable</strong></td>
<td>Compensation model tied to end-user utility and adoption.</td>
<td>Usage models integrated into compensation schemes.</td>
</tr>
<tr>
<td><strong>Recursive</strong></td>
<td>Organic nature of design supports rule and datamining of both social media business and game construct.</td>
<td>New ideas for game development occur from studying the usage and patterns of the community.</td>
</tr>
</tbody>
</table>

Source: Portions Adopted from DARPA Transformative Apps Call for Proposals, March 2010.
Other Types of Platforms: The Need for Cross-Platform Development

Our non-platform-specific strategy depends on looking at current industry development trends and anticipating the direction in which those platforms will next develop. Staying abreast of these trends allows us to keep pace with technology and to develop in such a manner that we can adapt to shifts in the target platform as needed, and not have to begin all development from scratch.

The idea of cross-platform development depends on analysis of currently available platforms and distilling their features into those that would directly affect FPCT development should we need to port assets to the new platform. Through use of a central repository of vetted assets, we are able to quickly adapt to changes in the platform needs of our client.

Following is a list of several relevant platforms, what new features they would bring to development, and how they would enhance FPCT’s core gameplay experience.

CAVE: CAVE technology is designed around a room-sized, multi-screen, multi-person immersive operating environment. This type of user experience is vastly different from the type of user experience currently offered by FPCT.

CAVE changes the user experience from the current single-screen setup that FPCT uses to a multi-screen setup, with each screen presenting part of the game world. This fundamentally changes the way FPCT would have to display its graphics, as they would now have to map to a 3D cube rather than to a classic single screen. In addition to screen display changes, several gameplay changes would take place, and would need to be evaluated as to how fundamentally they alter the gameplay experience. An example of this would be an NPC approaching the player character as they do currently, but the close proximity of the conversation would clip through the lower (floor-level) screen of the CAVE environment, causing the NPC’s feet to warp and stretch across the floor. The user interface of FPCT would also have to be redone.

CAVE would add some potential benefits in the form of increased player immersion and a more realistic representation of the operational environment.

VIRTSIM: While direct sources of information about Motion Reality’s VIRTSIM are scarce, we do have some knowledge about its capabilities. VIRTSIM uses a system of motion capture cameras and suits in tandem with 3D head-mounted displays to provide real-time motion capture and movement input into a simulation space. This system allows for a group of users to be placed into a simulated environment and can even provide some amount of tactile feedback. Users can then interact with each other, as well as AI characters in the environment.

VIRTSIM combines some of the benefits of technologies like CAVE but with a larger degree of immersion. By tracking a player’s head and updating their vision in their goggles, the player actually feels like they are “inside” a particular operating environment. They are physically moving through this environment, and their in-game avatar reflects their actual physical movements.

VIRTSIM supports the ability for a trainer or another player to “possess” an NPC, driving their motions and dialogue in real-time through their own bodysuit and microphone. This approach allows a gameplay session to change on-the-fly as the trainers see fit.

While VIRTSIM’s technology has been primarily aimed at training and review of tactical movement and combat, we acknowledge that it could provide for a more rich and realistic user experience if used as an interface to...
FPCT. However, this more realistic user experience does come at a cost. The hardware requirements to setup a training space is more expensive and is not as portable as a desktop application, and therefore, is not as easily widely deployed.

It is unknown at this time the full extent of what FPCT would have to change to integrate with VIRT SIM, but even conservative estimates place this task as very long-term. Integration with VIRT SIM would fundamentally change the way that FPCT operates, and AI, pathing, UI, custom development, and environments would likely have to be re-written from scratch. Some assets such as sound and character meshes may stay largely intact with few modifications.

**Gaming Platforms (PS3 and XBOX360):** Current gaming platforms such as the PS3 and XBOX360 are other possible platforms for FPCT development. The strengths of commercially-available gaming platforms are multifold: they provide a consistent hardware on which to run code, which far simplifies (or even eliminates altogether) distribution and compatibility problems between users. If a user has the current OS version of the platform, then it will run the code in the same way as any other version of the platform running that same OS version.

A secondary consideration of using commercially-available gaming platforms is cost. Current versions of the PS3 cost $299, and the XBOX360 can run from $199 to $299, depending on hard drive size and other configurations. This pricing structure is very attractive considering that even a bare-bones laptop can run $400, and these laptops typically do not have the graphical muscle needed to run 3D applications.

The PS3 is both Wi-fi and Ethernet capable out of the box, and the XBOX360 is Ethernet capable out of the box and Wi-fi capability is added with a separately-purchased adapter.

Another benefit of these platforms is that many major game engines already support them. Unreal Engine 3 currently runs on both PS3 and XBOX360, and Unity support appears to be incoming.

As Unreal Engine 3 already supports these platforms, and aside from control and hardware changes inherent to the platform, current FPCT development has the possibility of porting over without too much trouble.

Commercially-available gaming platforms are limited in a few ways: they draw a lot of power, which requires the appropriate infrastructure, and a consistent Wi-fi signal to download content updates from the internet. Disc and USB thumbstick-based updates are possible, however, if a signal is not available.

Both Microsoft and Sony require developers to be licensed, and content has to be approved by them. Microsoft has a more stringent approval process than Sony, but Sony’s development fees are higher.

**iPad:** The iPad is a relatively new device that aims for the consumer space between the workflow ability of a desktop or laptop computer and the ease of use and portability of a smartphone. While the iPad is larger than a smartphone by a large margin, it is thin and light (1.5 or 1.6 lbs, depending on the model), and is meant to be a portable solution for graphic design, reading text, and connecting to Apple’s app store to download more functionality as the need arises.

The iPad, and other “tablet computers” by extension, offer a unique environment for interaction with a simulation. The iPad features touch and multi-touch inputs, a large 9.7-inch (diagonal) LED-backlit glossy widescreen, and a high display resolution of 1024 by 768 pixels at 132 pixels per inch. This large screen and high resolution open up developing strategies not available to mobile devices, and the high-fidelity 3D nature of FPCT could possibly be displayed as originally intended for desktop computers. Touch interface could be an
intuitive way to navigate through FPCT, or the software keyboard could be used to take notes and record data.

The iPad is location aware as it supports GPS, it contains a digital compass, and can assist GPS by connecting to wi-fi or cellular networks. Within the context of FPCT, location-awareness can translate into position-directed navigation through a virtual environment (the player might turn, and the view within the game would adjust accordingly) or even the addition or subtraction of location-based data as the situation demands.

Wi-fi and cellular networks are available for the iPad, as well as Bluetooth technology. These tools could allow instances of an iPad optimized FPCT to search for and connect to squadmates, instructors, or it could download more current gameplay data such as patches or content updates.

**iPhone:** The iPhone is one of several smartphones currently available in the market. The architecture allows the development of software-based “apps”, which would be hosted by Apple’s iTunes store. A drawback about this style of development is that Apple products (including the iPad) require developed apps to be approved by them, and are all publicly hosted, which may lead to security concerns.

Another way to develop for Apple products may be to use the iPhone Developer Enterprise Program. Open to companies of at least 500 employees, this program requires the developer to have a D-U-N-S number to participate. The benefit of enrolling in this program is the ability to distribute your applications to employees or members of your organization through Ad Hoc distribution, bypassing the App Store entirely. This approach would be preferable in light of security concerns. Furthermore, Apple’s approval is not needed, and the developer won’t have to wait to host their App; they could simply distribute it to the parties that need it through the company’s distribution channels as needed.

The iPhone features, like the iPad, touch interface and GPS access. However, the screen is much smaller than the iPad, and development for this platform would require significant changes to the gameplay and structure of FPCT.

A mobile version of FPCT would need to be slimmed down, as the processor on the iPhone is not as robust as a desktop or laptop computer. Rather than trying to cram the current version of FPCT into a mobile platform, a better strategy would be to create a version that would play to the strengths of the mobile platforms, such as portability, touch interface, location awareness, voice input, and ad-hoc networking through Bluetooth.

The latest unreleased iteration of the iPhone is rumored to have a front-facing camera in addition to a back-facing one. This is presumably to allow video-chatting. Such a camera orientation may allow for additional development leveraging camera-based face-tracking and recognition not dissimilar from current-generation digital cameras.

**Android:** Android is a more developmentally open counterpart to Apple’s current offerings. Android is already featured in several smartphones, such as the HTC Incredible, and these phones offer feature-sets similar in function to the iPhone, including touch interface, access to wi-fi and cellular networks, GPS, and an “app”-based development architecture. The gameplay elements of a mobile version of FPCT could theoretically run on either Android or iPhone, though the hardware specs would vary by phone, and the SDKs are different and would require different developmental paths.

The main difference between developing with Android versus developing for the iPhone or iPad is that there is no content review process, no cost for development tools, and a secure content distribution system could theoretically be created to distribute mobile version of FPCT to any approved Android-capable device. There is
no need to apply for a special developer’s license or to obtain permission from a third party before distributing your work. Developing for Android is an attractive option because it is possible to create custom apps quickly, easily, and cheaply, without having to submit your development for approval and regulation by a third party.

Table 3. Major Mobile Devices (iPhone, iPad, Android) and Their Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>iPhone</th>
<th>iPad</th>
<th>Android</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coding Platform</td>
<td>Objective C, C++ or C</td>
<td>Objective C, C++ or C</td>
<td>Java programming language, C or C++</td>
</tr>
<tr>
<td>Graphical Capility</td>
<td>A single unified platform, capable of running 3D with a screen resolution of 480x320</td>
<td>A single unified platform, capable of running 3D and boasting the largest screen resolution and screen size of the mobile platforms (1024x768)</td>
<td>Multiple levels of hardware, ranging from budget hardware to performance hardware. Capable of running 3D, but may display differently depending on which device is used. Screen resolution ranges depending on which device is used.</td>
</tr>
<tr>
<td>Developer Kit</td>
<td>Developing on iPhone requires an iPhone dev kit, which currently costs $99/yr for independent developers and $299/yr for companies.</td>
<td>Developing on iPhone requires an iPhone dev kit, which currently costs $99/yr for independent developers and $299/yr for companies.</td>
<td>Developing on Android is free for both independent and corporate developers</td>
</tr>
<tr>
<td>Approval Process</td>
<td>Requires approval through the iPhone App store, where Apple has complete control over which apps get accepted and which apps get rejected</td>
<td>Requires approval through the iPhone App store, where Apple has complete control over which apps get accepted and which apps get rejected</td>
<td>No approval process for the Android app store</td>
</tr>
<tr>
<td>Developer Support</td>
<td>Extensive documentation, large community, and official training resources released frequently</td>
<td>Extensive documentation, large community, and official training resources released frequently</td>
<td>Extensive documentation, small community, and limited training resources</td>
</tr>
<tr>
<td>Community</td>
<td>As of April 2010, iPhone has sold 50 million units, with 185,000 available apps through the app store</td>
<td>As of April 2010 (one month since release) iPad has sold 1 million units. Most iPhone apps can be run on the iPad, with over 1000 iPad-specific apps available through the app store</td>
<td>As of April 2010, Android has sold 1.5 million units, with 38,000 available apps through the app store</td>
</tr>
<tr>
<td>Interface</td>
<td>Touch screen, with a touch-screen keyboard</td>
<td>Touch screen, with a touch-screen keyboard</td>
<td>Depending on the phone model, there are options for a touch screen keyboard, hardware keyboard, or both</td>
</tr>
<tr>
<td>Capability for Speech Recognition</td>
<td>Yes, built-in mic</td>
<td>Yes, built-in mic</td>
<td>Yes, built-in mic</td>
</tr>
<tr>
<td>Location Awareness/GPS</td>
<td>Yes, built-in GPS based on satellite data</td>
<td>Yes, but not as accurate, as it is based off of cell/WIFI network, rather than satellite data</td>
<td>Yes, built-in GPS based on satellite data</td>
</tr>
</tbody>
</table>

Game Engine Analysis and Ranking

Introduction: This section is an analysis of four of the most widely used commercial videogame engines that have been identified in industry rankings and by expert observers as the standard bearers for game developers. The four game engines – Unreal Engine 3, Gamebryo Lightspeed, CryENGINE3 and Unity3D – are consistently viewed as the top engines for facilitating varied genres of game development across multiple platforms. VBS2 and its upgraded version, Real Virtuality 3, are not considered among the most popular engines used by game developers, but are included for comparison.

In addition to the industry analysis, each game engine (Table 4) has been scored based on nine key criteria necessary for the development of a virtual cultural trainer. Those nine components are: access to source code; quality of training, documentation and community support; robustness of development tools; access to updates; platform portability; accommodation of body and facial animations; fidelity that accommodates visualizations and sound requirements for cultural representation; quality of pathfinding and Artificial Intelligence (AI); and User Interface customization. Each criteria was measured on a 0 – 5 scale, in which 0 indicates that an engine does not currently support the criterion and 5 represents the ideal option. The findings of this analysis can be found in Figure 5. Further detail is also provided in Appendix A.

Expert Overview and Rankings: Selecting the best game engine is a subjective task, one that depends on multiple factors, particularly what type of game is being developed. Any number of game engines can successfully produce varied types of games for multiple platforms with arguably varied levels of fidelity. Gamasutra.com writer and game developer Mark DeLoura recently asked producers and technologists what factors they considered when choosing an engine. The most frequent responses were cost and time first. But, DeLoura wrote, “In third position to money and time came genre relevancy, or making sure that the game engine works properly for the type of game you are creating. This emphasizes the importance of doing a thorough evaluation of any game engine you’re looking at, and knowing if a particular engine was already used for a game technologically similar to what you are planning to create (a first-person shooter, for example).” This is particularly important in analyzing game engines for the development of a virtual cultural trainer.

Develop-online.net, a game-development industry news website, analyzed and ranked the ten most widely used engines in June 2009. The website’s top ten included:

1. Unreal Engine 3
2. Gamebryo Lightspeed
3. CryENGINE 3
4. Unity 3D
5. BlitzTech
6. Infernal Engine
7. Vision Engine 7.5
8. Bigworld Technology Suite
9. Vicious Engine 2
10. Torque 3D

Source: http://www.develop-online.net/news/32250/The-top-10-game-engines-revealed
The top four engines – Unreal Engine 3, Gamebryo Lightspeed, CryENGINE3 and Unity3D – consistently rank among the most-used engines and concur with observations from other industry websites, such as Gamasutra.com and DevMaster.net. Each has distinct strengths and weaknesses, and each has been used successfully by developers of Triple A commercial videogames. Still, none provides the best solution for every one of the nine criteria identified in Table 4. What follows are observations of industry experts and an analysis of the strengths and weaknesses of each engine.

**Unreal Engine 3**

**Developer:** Epic Games, Inc. Based in Cary, N.C., with studios in Shanghai, Korea, and Tokyo.  
**Platforms:** PC, Mac, Xbox 360, PS3. **Browser support:** No. **Published titles:** Mass Effect (Bioware), The Last Remnant (Square Enix), Lost Odyssey (Mistwalker), The Wheelman (Midway). **Middleware integrations:** Fonix, SpeedTree, GameSpy, Scaleform GFx, PhysX, Illuminate Labs, Umbra, morpheme, nFringe, HumanIK, Kynapse, Bink, ProFX, AI.implant, Quazal, DigiMask, Game-Link, Wwise, Enlighten.


“To be honest, Epic’s monster behemoth doesn’t really need any introduction at all. Now almost an industry standard in its third incarnation, Unreal Engine 3’s domination of the full-engine middleware sector is hard to argue against. Still, the main criticism leveled against it – usually by its competitors, we should say – is that the engine is geared towards first- (or third-) person shooters. Nevertheless, customers have managed to extend and rip apart the engine to power everything from Japanese-style RPGs to open-world action-racing games like The Wheelman.”


**UE3’s strengths:** As state earlier, Unreal Engine 3 (UE3) is one of the most-used commercial engines, partly because it has been around longer than most engines but also because Epic Games regularly adds new features and improvements. In terms of visual fidelity, Unreal Engine 3 is on par with other top Triple A game engines. It is very adept at creating the realism needed in a 3D virtual environment designed for cultural training.

UE3 comes with a range of custom and third-party tools that already are integrated into the engine. These tools can be leveraged to accomplish most of the tasks required by game developers and are current with industry trends. For example, many developers were using a third-party tool, ScaleForm, to create user interfaces; Epic noticed this and included ScaleForm with the release of UE3. Epic also has integrated Steam, a popular digital distribution game store and online platform, which will improve UE3’s capability for publishing online games. For unsupported development tasks, Unreal licenses include access to the source code, allowing for the development of custom tools and the integration of other existing tools.

In recent months, Epic created the Unreal Developer Kit (UDK), a free version of the engine that can be downloaded from Epic’s website. While the UDK does not include the source code, it does open the engine to a much larger independent and educational developer audience and gives some insight into the community that the company wishes to foster. Epic also recently announced that it is working on a version of the engine for the iPhone.

**UE3’s Weaknesses:** One drawback to the UE3 is that the engine was built to create small- to medium-sized environments for use in First Person Shooter (FPS) “death-match” style games. Typically there are at most 48
characters in an environment. Also, because UE3 was designed for FPS games, different styles of games could require more work than with another engine designed for the specified style. That said, access to the source code and clever design can make overcoming these hurdles simply a matter of time.

**Pricing Strategy:** As with the other top commercial game engines, the cost for the engine is only available by request and most likely varies on use. UE3’s cost is considered comparable to others in the Triple A game development space. The creation of the free UDK does lower the entry barrier for a broader group of developers, allowing commercial games to developed and released with lower pricing models.

**Culture:** Being one of the most-used game engines, UE3 has a large online support community. Epic has compiled a collection of various tutorials and documentation for each component of the engine. There are also official mailing lists for each component of development (programming, art, etc.) that are very active. Questions submitted to the mailing lists are frequently responded to within a few hours. Epic has dedicated staff for each component to respond to problems or questions. In terms of community access, there are separate documentation and support channels for UDK developers and licensed developers. Epic has demonstrated support for the mod/user created content community by including editors and tools for end users to make their own levels. The release of the UDK is an extension of this interest in allowing open access to tools.

**Conclusion:** Although it was designed for FPS games, UE3’s high fidelity graphics and customization capabilities, via access to source code, make it a good choice for many styles of games. The large developer community that Epic has created and the company’s responsiveness to changes in the industry indicate a willingness to stay a current, competitive option in an ever-changing game development space.

**CryENGINE 3**

**Developer:** Crytek, based in Frankfurt, Germany, with studios in Kiev, Budapest, Nottingham and Sophia, South Korea. **Platforms:** Xbox 360, PS3, PC, ‘MMO and next-gen ready’ **Browser support:** No. **Published titles:** Crysis (CryTek), Crysis Warhead (Crytek), Aion (NCsoft) Middleware integrations: Scaleform GFx, CRI, FMOD.


“This is CryENGINE, so what you really care about is the visuals. Crytek has seen fit to add a huge number of high-end graphical features to the engine, including a real-time dynamic global illumination solution fully optimized for current-generation machines. There’s also a new real-time soft particle system, which can be affected by object collisions, forces such as wind and gravity, and lighting and shadows; volumetric light beams; screen-space ambient occlusion support, a unique deferred shading solution and a high-level platform-agnostic shader scripting technology.”


**CryENGINE 3’s Strengths:** Crytek’s newest version of their 3D game engine, CryENGINE 3, is considered by many sources as the premier game engine in terms of high fidelity graphics, characters and development tools. The quality of textures, materials, lighting and animation is superb. A robust animation system allows for the use of facial and full-body motion capture, as well as in-game animation blending, syncing, layering and retargeting. The engine has its own lip-sync solution for accurate animation to audio.
Although CryENGINE 3 is best known for its graphical fidelity, its development tools for environment creation, animation, material editing and scripting, among others, are user friendly and powerful. Developers are able to create, place and modify assets in the game space using the real-time editor, which saves time over other engine editors that must compile to load a preview.

CryENGINE 3 has a quality AI and path-finding system, which is particularly useful in constructing realistic human and cultural behaviors with minimal effort. Much of the AI functionality can be modified through the engine's visual scripting system. The scripting system does not require programming knowledge.

CryENGINE 3 is designed to create large environments with a high level of detail involving varying landscapes as well as many NPC’s and player characters. The engine has good support for both outdoor and indoor environments.

CryENGINE 3 supports development on PC, Xbox 360, and PlayStation 3 concurrently. This means that with some initial set up FPCT could be developed for multiple platforms with minimal additional work.

**CryENGINE 3’s Weaknesses:** CryENGINE 3 has steep system requirements. The hardware required to power the high fidelity graphics that makes CryENGINE 3 stand out is fairly substantial.

**Pricing Strategy:** According to its website, www.mycryengine.com, Crytek offers multiple licensing options for CryENGINE 3. These licensing options include:

- Free educational licenses that come with the basic editor and no source code access
- Commercial licenses designed for triple A development studios. A commercial license can be purchased with source code or without. Prices are not listed on the website.
- A simulation license with or without source code access. The price is not listed on the website.

**Community:** The CryENGINE community is fairly substantial. There is a large modding community for Crysis, the game that Crytek made using CryENGINE, as well as a substantial developer forum. A large number of tutorials are available to developers on Crytek’s main website. Crytek has shown a interest in getting into the simulation market, which may create new opportunities for features that are more common in simulations, such as built-in after action reviews.

**Conclusion:** CryENGINE 3 is a high-fidelity 3D game engine with some of the best visual graphics on the commercial market right now. It has excellent development tools, AI systems and the potential for easy cross platform development. For creating a high fidelity virtual cultural trainer, FPCT would be hard pressed to find a better engine for its purposes and goals.

**Virtual Battle Space 2/Real Virtuality 3**

Neither the VBS2 game engine nor the upgraded version, Real Virtuality 3, both designed by Bohemia Interactive, were ranked by develop-online.net nor mentioned as a leading commercial game engine by Gamasutra.com.

**VBS2/Real Virtuality 3’s Strengths:** Virtual Battle Space 2 (VBS2) is a simulation engine designed by Bohemia Interactive for the purpose of creating large, open environment battle spaces for modern war-fighting scenarios. VBS2 has the ability to create large environments that can be populated by many non-player characters (NPCs). VBS2 also contains a large amount of military hardware already in the game,
including firearms, ordinance and vehicles. These included models can help cut down on development time if they are items that would already be used in the game or simulation being created.

Another strength of VBS2 is its advanced physics and ballistics models for small arms, ordinance, missiles and other military hardware. These physics simulations are accurate to real life data and allow the engine to be used as a realistic virtual live-fire training exercise without the real-life danger. The physics models also include the vehicles that are included with the engine, making their performance believable and useful. The VBS2 upgrade, Real Virtuality 3, includes an improved graphics component, multi-core support and the addition of a facial animation system.

**VBS2/Real Virtuality3’s Weaknesses:** The lack of access to source code keeps developers from being able to make significant changes to the engine’s core systems. This means that it is often difficult to create any kind of simulation other than a large battle space environment with accurate physics. The inability to access source code limits the development of accurate cultural environments, despite the improved visual capabilities of the Real Virtuality 3 engine.

Another weakness is the inefficient process for importing resources, which requires several extra steps when compared to other commercial game engines. Also, the engine’s development tools require some knowledge of scripting, thereby requiring programmers to work alongside artists or artists to learn scripting. Also, VBS2 lacks many development tools found in many commercial engines, such as those for animation blending and custom rigging. The upgraded Real Virtuality 3 engine also lacks tools for importing custom skeletons and animation blending. The VBS2 engine is also not cross platform capable.

VBS 2 lacks the visual fidelity of other commercial engines; although the Real Virtuality 3 upgrade is improved, it does not compare to the fidelity levels of either Unreal Engine 3 or CryENGINE 3. VBS2 has a limited animation system because there is no way to create custom skeletons for characters. VBS2 does not have the capability of creating complex facial expressions, nor does the engine support blendshapes or morph targets that are commonly used for facial animation.

**Pricing Strategy:** Bohemia Interactive’s pricing strategy is a one-time fee that allows the developers access to the current version of the engine. This does not include source code, nor does it include updates to the engine. Source code is not available in any licensing plan and developers have to pay extra for engine updates.

**Community:** The community for VBS2 is small and unorganized. The unofficial community is mostly based on mod communities for the commercial version of the engine, which is shipped with Bohemia Interactive’s game, ArmA. The official community for the engine is even smaller and more unorganized than the mod community. Over all there is a general feeling of detachment from other users of the engine, as well as from the engine developers at Bohemia Interactive.

**Conclusion:** VBS2/Real Virtuality 3 are engines that were designed to create large open-space war-fighting environments. The tools that ship with the engine, the lack of access to source code and the detachment of the creators of the engine make turning VBS2 into a cultural trainer more difficult.
**Gamebryo Lightspeed**

**Developer:** Emergent Game Technologies, based in Calabasas, Calif., with offices in North Carolina, as well as China, Korea, Japan and the U.K. (Some of investors include Worldview, Jerusalem Venture Partners and Cisco Ventures.) **Platforms:** Xbox 360, PS3, Wii, PC. **Browser support:** No. **Published titles:** Fallout 3 (Bethesda), Civilization Revolution (Firaxis), Warhammer Online (EA Mythic). **Middleware integrations:** Nvidia PhysX and APEX, Scaleform GFx, Wwise, Speedtree, Illuminate Labs, NaturalMotion, Lightsprint, Aristen, Umbra, xaitment, memoraze, Allegorithmic, RAD.


“Gamebryo LightSpeed (3.0) was launched at GDC 2009, bringing greater real-time feedback and rapid prototyping capabilities to Gamebryo 2.6. Since GDC 2009, Emergent has been working on a significant number of enhancements focusing around rapid development. Working tightly with external developers has enabled Emergent to focus on what is most important to developers, and has also given them some improved prototypes and demos as part of their development kit. …

“One of Gamebryo’s strong features has always been its close partnerships with middleware companies. Over the past year, increased partnership with Audiokinetic, Xaitment, Pixelux, and Fork Particle (among others) has increased the capabilities of Gamebryo LightSpeed, allowing developers to use tools they are already familiar with.”

**Gamebryo Lightspeed’s Strengths:** Gamebryo Lightspeed hits a wide market of videogame genres, from Wheel of Fortune to the blockbuster, Fallout 3. The primary claim to fame in the newest iteration of the Gamebryo engine is the incorporation of several middleware providers in order to provide a rapid application development framework. Instead of treating the development of a game engine as a monolithic endeavor, Emergent draws most of its functionality from developers who are specializing in a particular area of game engine design.

For example, Gamebryo incorporates Autodesk Kynapse, a pathfinding middleware product that separates the need of providing NPC pathfinding from the engine. Also, instead of designing its own user interface tool, Emergent incorporated ScaleForm, another middleware tool that allows Adobe Flash interfaces in a videogame. The decoupling of middleware elements from the core functionality of Gamebryo is important for the customization of games such as FPCT, which vary greatly from most kinetic commercial games.

Gamebryo also supports a wide range of technologies that deal with sound, graphics, physics and multiplayer game play, all essential for realistic, immersive environments. Gamebryo has visual fidelity on par with the best engines in the industry. Having been used to produce games with vast expansive landscapes as well as intricate facial detail such as Fallout 3 and Oblivion, it would be well suited to producing the subtle cultural/behavioral cues in FPCT. Gamebryo also supports cross-platform development.

According to Emergent, the Gamebryo engine has been used to develop U.S. government and military projects, as well as for firefighting and surgical simulations.

**Gamebryo Lightspeed’s Weaknesses:** Gamebryo Lightspeed’s reliance on industry technologies has a potential downside should a company go out of business or their product is dropped by Gamebryo. Development in that product could easily be lost. Also, more interfaces in a system create the potential for more breakdowns.
**Pricing Strategy:** The pricing for Gamebryo is only available upon request but Emergent Technologies offers both educational and commercial licenses and does promote evaluation of the product before beginning commercial production.

**Community:** Emergent provides a forum on its website, called Pulse, for the discussion of its technology. Gamebryo’s middleware partners also provide robust online communities.

**Conclusion:** Gamebryo has been used on blockbuster games where characters exhibit high fidelity emotional responses with convincing sound and animations, making it an appropriate engine for cultural trainer games.

**Unity 3D**

**Developer:** Unity Technologies, based in San Francisco with development offices in Denmark, Lithuania and the UK. **Platforms:** PC, Mac, iPhone, Wii. **Browser support:** Yes. **Published titles:** FusionFall (Cartoon Network), Zombieve USA (mikamobile). **Middleware integrations:** PhysX, Mono.


“The community is something that Unity Technologies considers a big selling point of the engine: the low entry point means that many hobbyist and indie developers are ardent supporters. As they rationalize it, when big triple-As like Funcom and EA embark on development, they’ve got a wide (and educated) support base and also a pool of ready-trained talent to recruit from. Parts of Cartoon Network’s MMO FusionFall were developed by community members, for example. It’s also got arguably the best iPhone support, with the actual device acting as an input method to the editor to fine-tune accelerometer controls.”


“Perhaps the marquee title for Unity this past year has been EA’s Tiger Woods PGA Tour Online. Pushing the Tiger Woods console/PC experience to the web was no small task, and seeing EA adopt Unity for the title adds even more credibility to the use of Unity for a modern 3D gameplay experience on the web. On the iPhone side, games such as Tumbledrop (written by an artist with no prior programming experience) and Star Wars: Trench Run show off the capabilities and ease of use of the platform, while Max and the Magic Marker really shows off its cross-platform nature, shipping for PC, Mac, WiiWare, and web.


**Unity3D’s Strengths:** Unity3D distinguishes itself as a solution for cross-platform development. The 3D engine offers portability port to PC, browser, Android, iPhone, iPad, Xbox 360, PS3 and Wii. No other engine offers as many platforms. This feature, coupled with its strong community and minimal entry costs, has allowed Unity3D to emerge as a leader in the mobile and independent developer market while still pulling in many developers for PC and console.
To support efficient development, Unity3D streamlines most tools. All 3D assets can be brought into Unity3D in their original raw formats, eliminating the task of exporting into the engine-specific format. Preserving the raw formats of the assets allows for non-destructive workflows, which dramatically increases the efficiency of the pipeline.

Unity3D’s engine is growing almost as rapidly as its popularity, introducing frequent major overhauls and improvements to their already established tools. Every few months, Unity Technologies releases new major content updates, which usually include cross-platform support for even more platforms.

**Unity3D’s Weaknesses:** Unity3D still has not integrated a friendly, visual animation system that does not require the oversight of programmers. Unity3D’s support of user interface is also fairly rudimentary and developed through code; it does not compare to the visual complexity of ScaleForm or other UI programs that allow the artist to take visual control. Additionally, there are no pathing or AI systems built into Unity3D. Although there are some community-made solutions, they are not officially supported or integrated into the engine, meaning that they may not be guaranteed to work on newer iterations of the engine.

Unity3D’s technology for visual and audio fidelity is also not as good as other commercial engines, but this is partially because the company primarily targets a mobile audience. Even though Unity3D supports the basic necessary tools for 3D (animation, high-resolution specular and bump maps and terrain generation), their tools are rudimentary and do not come with the many extras, such as lip-sync, audio mixing, or blend-shape. Unity3D would not be the ideal solution for high-fidelity development on non-mobile platforms.

**Pricing Strategy:** Unity3D runs on a pricing strategy that is not typical of other engines, which typically require a yearly fee to maintain ownership of a license. Unity3D requires a one-time price per seat. Additionally, their per-seat license price is inexpensive, making the engine popular among independent game developers and hobbyists.

A free version is available to students and independents and has almost the same functionality as the paid version. The only things missing are graphical aspects, such as real-time shadows and post-processing render effects. This allows anyone to dive into the engine with minimal risk to see whether or not it will fit his or her needs, as well as establishing a diverse community of independents as well as big developers.

**Community:** One of the key traits of Unity3D is their focus on community, which is supported by the pricing strategy. The company has tech blogs, official training resources and question/answer wiki sites for developers. Unity developers update their blog at least once a week (sometimes even more) to post new tutorials, announce new features implemented into the next iteration of Unity3D or just post their general views on the direction of their company as it pertains to current technological trends. Users are in constant contact with the developers, and always know the next features coming down the Unity pipeline.

Everyone has full access the forums and blogs, whether you pay for the license or not, and all training resources are available to anyone who wants them (and there are a lot of training resources, both official and unofficial). Their entire scripting API is also available on the website, with examples for most functions on how they might be implemented using a short example piece of code. In general, Unity3D is all about making everything available to everyone, in the best way they can.

**Conclusion:** Unity3D is in the forefront in the battle for cross-platform supremacy due to its multi-platform support, mobile graphical fidelity, community-focused business structure and inexpensive entry costs. Overall, Unity3D is the best solution for high-fidelity 3D for mobile platforms.
**Scoring the Engines:** An analysis was compiled of researched game engines with ratings based upon criteria for the engine to meet the needs of development like FPCT as well as cross-platform capabilities. Cross-platform is defined as an engine’s ability to develop games for PC, Mac, PS3, Xbox, mobile, etc.

Table 4 below outlines development criteria used to research and score a list of development engines. As mentioned above, scoring criteria include:

- Ease of development
- Access to training, documentation and quality of user community
- Access to latest technology or software versions
- Flexibility of engine to support requirements
- Accommodation of body and facial animations
- Complex pathfinding and Artificial Intelligence capability
- Ability to support multi-platform development
- Fidelity that accommodates cultural representation requirements for visualization and sound
- User Interface customization

These criteria are scored as follows:

0 = does not currently support capability
1 = not very capable or ideal, but possible
3 = average, or meets the basic industry standard, but not exceptional
4 = notably better than the average expectation for this criteria
5 = the ideal option for this criteria.

These criteria become the basis of the Y axis (meets requirements) in Figure 5 on the next page. Figure 5 demonstrates a summary of our research in the engines’ overall ability to meet the “requirements” of the FPCT project and their ability to be developed across multiple platforms.

As Figure 5 shows, for the meets requirements criteria, Unreal Engine 3, CryENGINE and Gamebryo Lightspeed are all very close in terms of meeting requirements. Overall, CryENGINE wins the category, based on graphical fidelity and animation capability, although, as mentioned all three engines are very close.

The x axis in Figure 5 scores “cross platformness.” For the cross-platform scoring, the “ease of platform conversion” and the “number of platforms available” was scored on a 0 – 5 scale. The weight for the “number of platforms available” was then given a multiplier to reflect the greater importance of this criterion to a project similar to FPCT. On the X axis (cross platformness), Unity exceeds other engines with support for mobile and iPad capability. For cross-platform development (iPhone, iPad, Android), Unity is the only engine that provides the necessary support. None of the other engines studied support mobile at this time. According to company announcements, Unity 3 will include support for the Android platform this summer. However, Unity does not score as highly on the Y axis (meets requirements) because of lower visual fidelity and therefore is not a substitute for the quality of the Unreal Engine 3.

VBS2/Real Virtuality 3 does not seem to include any multi-platform capability, has limited documentation and limited developer community access and therefore does not rank well in the analysis on either Y or X axis criteria.
In summary, Unreal Engine 3, CryENGINE 3 and Gamebryo ranked high requirements largely due to their high fidelity, which is important to cultural training models such as FPCT. For all except mobile and web or browser-based applications, we would recommend continuing to develop with Unreal Engine 3 because of our experience and the extreme closeness of the engines’ rankings. For mobile and web applications, Unity is the clear and only choice currently.

**Figure 5.** Game Ranked on Cross Platformness vs. Meeting Requirements

**Scoring Legend:**

**X Axis (crossplatformness):** For the cross-platform scoring, the ‘ease of platform conversion’ and the ‘number of platforms available’ was scored on a 0 – 5 scale. The weight for the ‘number of platforms available’ was then given a multiplier to reflect the greater importance of this criterion to a project similar to FPCT.

**Y Axis (meets requirements):** For the criteria scoring total, each criteria point was measured on a 0 – 5 scale:

- 0 = does not currently support capability
- 1 = not very capable or ideal, but possible
- 3 = average, or meets the basic industry standard, but not exceptional
- 4 = notably better than the average expectation for this criteria
- 5 = the ideal option for this criteria
<table>
<thead>
<tr>
<th>Development Criteria</th>
<th>Unreal Engine 3</th>
<th>Unity3D</th>
<th>VBS2 / Real Virtuality 3</th>
<th>CryENGINE</th>
<th>Gamebryo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ease of development</td>
<td>Rank: 4 Supports a wide range of efficient, artist-friendly, visual tools. Many tools are separated so that there is as little dependency between artists and programmers as possible.</td>
<td>Rank: 3 More dependent on coders. Tools inside the editor are easy to use, though some require programmer oversight, possibly slowing down workflow.</td>
<td>Rank: 1 Very dependent on coders. Developer tools would need to be rewritten for a program like FPCT. Most tools are outside of the editor and segmented, meaning that artists have to run 3 separate tools to accomplish even simple tasks.</td>
<td>Rank: 5 Includes the most interactive and artist friendly tools. Tools are inside the editor, visual and easy to use, allowing for a quick and efficient pipeline for the artists and programmers.</td>
<td>Rank: 4 Because the tools are built in a modular way, there may be more opportunity to incorporate alternate tools.</td>
</tr>
<tr>
<td>Access to training, documentation and quality of user community</td>
<td>Rank: 5 Largest user community. Well documented, resources available on their site, as well as newsletters and notification of updates. Training resources available through UDN’s website and over 100 free training videos.</td>
<td>Rank: 4 Community is moderate-sized. Well documented, their entire scripting API is available on the site. Training materials available on their blog, both documents and videos.</td>
<td>Rank: 1 Community appears to be less active than other engines. Developers must request access to view documentation. Most interactions take place between the user and a representative of VBS2. Paid training is available through Bohemia Interactive. There are a few specialized free training videos available on the forums.</td>
<td>Rank: 4 Community is moderate-sized. A large database of clear documentation is available upon gaining a login. CryENGINE trainers assist on-site after the purchase of a development license. Additional training is available through the community forums.</td>
<td>Community offers tutorials, videos, articles and discussion. Tutorials and extensive documentation available, which is cross-referenced for ease of topic. Formal training is unclear.</td>
</tr>
<tr>
<td>Development Criteria</td>
<td>Unreal Engine 3</td>
<td>Unity3D</td>
<td>VBS2 / Real Virtuality 3</td>
<td>CryENGINE</td>
<td>Gamebryo</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>---------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Access to latest technology or software versions | Rank: 5
Developers are notified whenever a new version of Unreal is released, as well as what was changed and upgrades are free. | Rank: 3
Minor updates are free, but major releases may require an upgrade fee per license. Upgrading is optional. | Rank: 1
Upgrades may require a fee. Real Virtuality 3 provides multicore support. | Rank: 3
All upgrades are free with the license purchase and the engine is constantly updated with new features. | Rank: 3
Assume updates will be released to align with industry standards. |
| Flexibility of engine to support requirements | Rank: 4
Provides access to source code that allows transformation of Unreal's engine (designed for FPS games) into a cultural training tool. | Rank: 3
Designed to be an open-ended mobile platform. Not tied to a particular mode of gameplay, allows design and implementation of game without having to work around predefined constraints. | Rank: 1
Does not provide access to source code, challenges creating cultural training (designed to be kinetic simulation.) Developers are dependent on Bohemia to implement major changes. Real Virtuality 3 does include built-in dynamic conversation. | Rank: 4
Provides access to source code, which allows transformation of CryENGINE (designed for FPS games) into a cultural training tool. | Rank: 5
Provides access to source code, base engine allows functionality to be built on. |

Table 4 is continued on the next page.
<table>
<thead>
<tr>
<th>Development Criteria</th>
<th>Unreal Engine 3</th>
<th>Unity3D</th>
<th>VBS2 / Real Virtuality 3</th>
<th>CryENGINE</th>
<th>Gamebryo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accommodation of body and facial animations</td>
<td>Rank: 4 Strong body and facial animation system, including artist-driven tools that allow the animators to customize appearances within the engine editor.</td>
<td>Rank: 2 Capable of body and facial animation, no tools to support immediate in-editor animation editing and blending</td>
<td>Rank: 1 Custom facial animations are not possible in VBS2; Real Virtuality 3 does include limited facial animation. Modifications to animations would often require programmer assistance and current art tools are not compatible.</td>
<td>Rank: 4 Supports complex, dynamic pathfinding. NPCs can navigate a level and change their paths based on new changes in environment structure.</td>
<td>Rank: 4 Integrated Autodesk Kynapse provides real-time dynamic pathfinding and other AI crowd systems.</td>
</tr>
<tr>
<td>Complex pathfinding and Artificial Intelligence capability</td>
<td>Rank: 4 Supports a complex pathfinding system and frequently updates it in monthly builds</td>
<td>Rank: 4 Company does not support any pathfinding system; it can be added with a custom add on</td>
<td>Rank: 4 Both engines include limited pathfinding capabilities. Real Virtuality 3 supports combat-focused AI, including gestures, suppressive fire and centimeter precision.</td>
<td>Rank: 4 Supports complex, dynamic pathfinding. NPCs can navigate a level and change their paths based on new changes in environment structure.</td>
<td>Rank: 4 Integrated Autodesk Kynapse provides real-time dynamic pathfinding and other AI crowd systems.</td>
</tr>
</tbody>
</table>

Table 4 is continued on the next page.
<table>
<thead>
<tr>
<th>Development Criteria</th>
<th>Unreal Engine 3</th>
<th>Unity3D</th>
<th>VBS2 / Real Virtuality 3</th>
<th>CryENGINE</th>
<th>Gamebryo</th>
</tr>
</thead>
</table>
| Ability to support multi-platform development | Rank: 3
Supports multi-platform authoring across PC, Xbox 360 and PS3. System does not port immediately and requires some setup. Unreal is developing a mobile platform, but no official information has been announced. | Rank: 5
Supports multi-platform authoring and visualization across PC, Xbox 360, PS3, iPhone, iPad, and Android. | Rank: 0
Does not support multi-platform authoring or mobile platforms. | Supports multi-platform authoring and visualization across PC, Xbox360 and PS3. Does not support mobile platforms. |
| Fidelity that accommodates cultural representation requirements for visualization and sound | Rank: 4
The industry standard in current-generation graphical fidelity and audio quality. | Rank: 3
Supports the highest quality graphical fidelity for mobile platforms. Does have a sound system, but it offers only basic functions. | Rank: 3
VBS2 offers basic tools for mid-level graphical fidelity. Does include many useful tools and options for sound. Real Virtuality 3 offers DirectX 9 Shader Model 3 support, which improves graphical fidelity. | Rank: 5
Renowned for providing the highest level of visual fidelity and most engine improvements are focused on this aspect. | Rank: 4
Top notch sound, graphics tools enable development of an immersive experience. |

Table 4 is continued on the next page.
<table>
<thead>
<tr>
<th>Development Criteria</th>
<th>Unreal Engine 3</th>
<th>Unity3D</th>
<th>VBS2 / Real Virtuality 3</th>
<th>CryENGINE</th>
<th>Gamebryo</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Interface Customization</td>
<td>Rank: 5 Wide range of possibilities with interactive UI, as Unreal now officially supports ScaleForm for UI workflow. (ScaleForm is a flash-based UI system used in leading games.)</td>
<td>Rank: 3 UI is accomplished through basic button and window-based programming. Is very coder-dependent and straightforward.</td>
<td>Rank: 2 UI is very simple and straightforward. Almost entirely code-based. The system is not dynamic, but may satisfy requirements on a basic level.</td>
<td>Rank: 5 Wide range of possibilities with interactive UI. Officially supports ScaleForm for their UI workflow. (ScaleForm is a flash-based UI system used in leading games.)</td>
<td>Rank: 5 Officially supports ScaleForm for their UI workflow. (ScaleForm is a flash-based UI system used in leading games.)</td>
</tr>
<tr>
<td>Total Score out of 45</td>
<td>38</td>
<td>27</td>
<td>12</td>
<td>40</td>
<td>36</td>
</tr>
<tr>
<td>Cross Platform Score out of 15</td>
<td>9</td>
<td>14</td>
<td>0</td>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>


**The Way Ahead – FPCT Core – A Central Repository and Middleware**

Porting FPCT vetted cultural assets to a variety of platforms and devices would be advantageous to the warfighter because the training could be available in the most situation appropriate device and form. A wide spectrum of development strategies are available to port FPCT to the other engines and development platforms presented here. On one extreme, every port to a new platform or device can be a recreation of the game and near perfectly take advantage of all the unique elements of a particular platform of device. However, this strategy could be time consuming, expensive and impractical and take away resources from new development.

On the other hand would be a game that was so generic that it could be ported to anything with a click of a button. Because the specifics of every platform cannot be anticipated, the genericness loses its positive aspect because it has to cater to the lowest common denominator and may not adequately take advantage of the unique features of a platform or device.

Some key definitions, why it is important to be able to easily port FPCT, development issues, and how to overcome these challenges are some of the questions considered in this section.
**Definitions:** In order to propose appropriate ways to create a non-platform specific development strategy for FPCT, certain terms need to be defined. These include middleware, application programmer’s interfaces (APIs), and game assets.

- Middleware is a specific form of software that seeks to remove domain specific tasks into a uniformly accessible library. For example, computer player pathfinding, the ability for the computer to “know” how to navigate from Point A to Point B, has traditionally been unique to each game engine. Now the product Autodesk Kynapse provides common real-time pathing algorithms in their package. This has two primary benefits. One, it allows developers who use it on disparate engines to learn one tool set and reuse it. Secondly, it allows for reuse on disparate engines.

- Application Programmer’s Interfaces (APIs) are the main method by which a middleware solution (as well as other types of applications) provide functionality. For example, when developers purchase Kynapse from Autodesk, they are given a set of libraries that you can then integrate into their program. Integration of those libraries is provided by the API. APIs help to provide a uniform and (hopefully) non changing interface that programmers can rely on to do similar tasks in similar situations on different systems. So using Kynapse’s functionality from Unreal would require the same code as using the functionality from Gamebryo or VBS2. It is reflecting that code through the game that is engine-specific.

- Game Assets are anything a game engine needs to run a game. These assets can be almost anything the game uses besides core game engine code. Examples include mesh files for the graphics, animation data to make the meshes move, sound files to provide the aural experience, script files for game-specific logic, etc. This list is by no means exhaustive, and new types of assets can be envisioned for specific domains, for example, personality data sheets might be needed by a personality modeling library.

**Game Engine Specificity:** Before discussing the different sections of a game that require porting, it is worth noting that no matter what, certain games are easier to port to some game engines than others. This is because game engines tailor themselves to different genres. A sports simulator would not port well to a fighting game engine. This specificity of game engines means that porting to specific engines could entail a loss of fidelity if that engine does not readily facilitate particular aspects of the ported game. For example, FPCT models emotions, however, the VBS2 game engine has a very high fidelity physics model.

**Portability of Various Game Aspects:** Because a game is comprised of many aspects (audio, visual, logical), and each aspect has its own set of assets (sound files, meshes and animation, scripts), it is important consider each aspect individually as it relates to game portability. The following categories are non exhaustive and feature a rating for the ease of portability.

A five means that code can be ported from one system to another with no consideration for the system. A one means that code is completely intrinsic to that system.
FPCT Portability Ranking:

- AI/Pathing: AI Decision making and character personality models are intrinsically in the FPCT game logic domain and can be isolated pretty easily. Normally pathfinding is integrated into the engine proper, but Autodesk Kynapse is becoming a viable alternate middleware solution to this problem. Portability: 5/5.

- Animation: Animations are fairly engine agnostic as long as they can be converted to a file format that the engine accepts. Some of the animation re-targeting onto other characters would have to be redone in each engine as well as some of the animation blending and synchronization, but these tasks are not that time consuming compared to the creation of the animation. Portability: 4/5.

- Character Generation: The visual object that represents a character and its corresponding physics object are engine dependent. However, the model of a character is bound to the AI object model and thus can be isolated from the engine. The connector would be responsible for attaching these AI objects to their engine visual/physics counterparts. Portability: 4.5/5.

- Materials: The easiest way to create material assets for use in multi-platform development is by creating high resolution PSD files of each texture. This way the files can be batch converted from photoshop in any size or format required. Materials themselves will still need to be re-created in each engine, as shader codes and material assembly differs engine to engine. Portability: 3/5.

- Multiplayer: Most of the engines have some sort of multiplayer that it already supports, but they are closely tied to the architecture of the engine. It may be possible to look into some multiplayer and networking middleware, however this would still require significant work to integrate it into each individual game engine. Portability: 2/5.

- HI2NT Integration: Communicating with the other HI2NT Federates will be handled by implementing the ProtoCore API. This can be encapsulated in the main game logic library thus shielding it from the game engine. Portability: 5/5.

- Scripting: Scripting for game logic can be completely custom and isolated if necessary because of previously discussed reasons in the Middleware section. Portability: 5/5.

- Sound: All sound files should be cut, mixed and compressed outside of the engines even though the engines support built-in mixing, so that audio doesn’t have to be retuned when bringing it from one engine to the next. Sound streaming does not seem to be supported by all engines, so any sound streaming should be specific to mobile engines unless another solution is discovered. Portability: 3/5.

- User Interface: All button graphics should be created and sized outside of the engine, so that little to no time is wasted trying to sync up the UI’s to be the right size and color. The buttons would have to be coded into their respective engines using different methods, but if the graphics and resolution stay consistent outside, that may avoid trouble. Using Scaleform would also open up some possibilities for agnostic engine development between engines that support Scaleform, allowing us to transition complex UI structures between CryENGINE and Unreal fairly painlessly. Portability: 3/5.

- World Space: In terms of world space, to create engine agnostic assets would require the development team to use Maya and 3ds Max for conversion of assets. This assumes the fidelity of each asset is retained across all platforms. If a project platform requires a lower resolution asset, it may be useful in the future to develop different LODs for each asset. Portability: 4/5.
This list is not exhaustive. As game engines evolve, they may feature new aspects of game development that can’t be anticipated with assets that have not been considered. This list is just a cursory set of considerations for portability.

**Methods to Port FPCT:** A middleware solution that isolates the FPCT game logic in one section, the FPCT core emotional model in another section, and the FPCT assets in central repository would best facilitate development across multiple engines.

Because FPCT needs to be able to run on multiple game engines, the game logic needs to be isolated from the game engine itself. Currently much of the logic that determines behavior and other game logic is intrinsically tied to the Unreal 3 engine itself.

However, if FPCT is developed such that it provides an interface that the team decides, connectors can be written into almost every game engine that provides integration: either through direct source code access or through wrapping FPCT data structures into language extensions for the scripting language that engine uses.

For example if the FPCT game logic were isolated into a C++ library for efficiency, an API can be provided so that to use FPCT on Unreal Engine 3, the same functions are called, and they are the same regardless from where they are called.

This would essentially be creating two middleware solutions, one that encompasses FPCT game logic, and one that encompasses FPCT emotional modeling. In this way, the reuse of the FPCT emotional modeling software is not tied to the game logic of FPCT, and can be used for other applications.

This still leaves the issue of how assets could be ported from one platform or engine to another. A central art repository with the data could facilitate this.

**Central Repository:** Having a central repository has several key benefits to matching the platform independent strategy.

**Portability**
- A central repository can be retrieved from anywhere, and thus by any platform.
- Deployed on a mobile platform or a desktop, the platform-independent data can adapt as needed.

**“Develop Once, Port Many Times”**
- Different versions of assets such as low polygon count for mobile and high polygon count for desktop can be stored and thus retrieved as appropriate.

There are also two different needs to fulfill in this repository. The data can be stored in an engine specific form for every engine if speed is the primary concern when porting, or it could be stored in an independent form and translated as necessary for each engine. Keeping it in this independent state saves space, but could possible sacrifice art fidelity in the end product.

**Conclusions about Middleware:** What follows is a diagram that summarizes the development strategy. Creating a middleware solution with a central repository maximizes the code and art reuse possible, while optimizing the amount of “architecting” of generic code required.
Figure 6. Cross Platform Development with a Central Repository

Figure 6 illustrates the strategic architecture for FPCT to deploy on multiple platforms. Raw data flows out of sources like a digital content creation suite (Maya) or a HI2NT Federate like JNEM and is preconditioned so that it exists in a platform independent state. When it is required by a deployment plan, the art aspects will be pulled into a post conditioner so that it can be used on the particular game engine (if a copy was saved for each game engine, the post conditioner would do nothing). When data is needed, it is pulled in by the FPCT middleware layer and interpreted appropriately.
The Future of Game Engines

Game engines have overcome many of the graphical and memory limitations that were issues early in the industry’s development. Currently available commercial game engines such as Unreal Engine 3 and CryENGINE 3 are known for being able to design and display very high-fidelity 3D character models. We feel that game engines are moving away from a purely graphical update model, where each new iteration of the engine showcases vastly improved graphical capability, and that they are now moving more towards the key concepts of crowd-sourced development, connectedness and portability within multiple platforms.

Crowd-sourced development can be easily seen in the current iterations of the iTunes marketplace and “app”-style development seen in many smartphones. Instead of developing content themselves, the providers of these app stores rely on end-users to create and submit their own development, which is then audited and regulated by the provider company.

The future of crowd-sourced development could be in the formation of new content for existing games and simulations. Troops returning from an assignment could update their simulation with the most current information, and this content could in turn inform the next group to head out into the field. Story elements could be culled from user-submitted experiences, and could be utilized by a game engine to create wholly unique but topically relevant in-game stories.

The concept of connectedness can be seen in all current-generation game consoles (PS3, Xbox360, Wii) and even the portable ones (PSP, Nintendo DS), and it illustrates the customer’s growing assumption that they will be able to access their content whenever and wherever they want. For home consoles, the concept of connectedness allows the customer to easily locate and communicate with friends, share gameplay sessions online (multiplayer), and download full games and patches to disc-based game. For the portable consoles, connectedness refers to the ability to create quick ad-hoc connections with nearby friends for multiplayer sessions, and some utilize Wi-fi to play with friends around the country or worldwide. The future of this connectedness could manifest in an always-updated version of the game, where the patching process is completely invisible to the end user. Content could be added on the fly to adapt to on-the-ground truths, and then the version could be reverted remotely to a previous version once that content is no longer needed or relevant.

This connectedness also includes social networking. It is currently possible to gather a list of friends, see what they are playing, and add them quickly to your online gaming sessions. The future of social networking in context of gaming platforms is the idea of “emergent gameplay,” where your connected friends will be able to modify your game experience on-the-fly. An example of this may be a friend’s self-published indication that they accomplished a particularly difficult task. This notification would publish across that person’s friends list, informing them of this accomplishment and challenging them to beat it. The actions of your connected friends in this case would influence the way you play your game.

The concept of portability refers to the need for a software developer to deliver their product across many different platforms to fully access their audience. Fewer and fewer platform-specific games are being released, and ports or major game releases are often seen for all available platforms, including home game consoles and PC. Digital distribution is growing as more homes are broadband-capable, and game developers have begun to offer many of their new releases both online and in a more traditional disc-based format.

The rise of mobile gaming can be attributed to well-marketed devices such as the iPhone and Android-based phones, and, though these devices are not as graphically capable as consoles and personal computers, they excel at running programs that are social-media-capable and location aware, and those which leverage a
phone's hardware, such as voice and movement input. The future of portability may well include a consolidation of hardware solutions to a single, multi-use portable device.

Multi-platform portability is central to the new service Onlive, which will begin a beta within a few weeks. Onlive promises to bypass hardware entirely, processing the complex calculations in the cloud on external servers, and then using only a broadband connection to deliver gaming currently limited to high-end desktop computers to any broadband capable device. The future of this technology will see an end-user's smartphone, television, tablet computer, or any other broadband-capable device receiving high-end gaming experiences completely independent of the device's hardware specs. In the future, the platform will no longer be the hardware, but the provider of the streaming content. The competition between these providers will be the new “console wars.” This is consistent with the central repository strategy described here.

In summary, taking these different trends into account, we anticipate that future game engines will evolve along several paths:

- Crowdsourced development and testing:
  - This can be seen in the current iTunes app store and Android-capable phones. These platforms have an open-development design, where they host created apps.
  - The future could include user-submitted gameplay elements and story design

- Increased connectedness:
  - Invisible patching and on-the-fly content updates and modifications.

- Portability of development:
  - Currently seen in the Unity 3D engine. Development in Unity supports multiple platforms, including mobile and PC. http://unity3d.com/

- A move away from increasing hardware demands:
  - This can be seen in Onlive's business model (yet to be launched) of bypassing hardware limitations and streaming content directly to a monitor or TV. http://www.onlive.com/about.html

**Summary**

This paper offers both an analysis of the state-of-the-art for current virtual cultural game development and prospects for future development in a rapidly changing environment. Ongoing analysis will be necessary as trends in the expanding social-networked communications space, the increasing reliance on cloud computing, the crowd-sourcing of middleware solutions and the development of new mobile platforms will reshape strategic considerations. Indeed, even the general recommendation here to stay developing in the Unreal Engine 3 for core development and move to Unity3D for web and mobile development could change if UnReal Engine 3 develops more cross platform capability. Or, if the high-fidelity CryENGINE moved ahead of Unreal Engine 3 in crossplatform development this also could change the engine trade space landscape.

The criteria that we have established for virtual cultural training in a game-based environment still serve as a core means by which to analyze ongoing development strategies. The most powerful virtual cultural training will need to develop non-platform flexibility. At the same time, market forces will no doubt reshape the game engine space and the types of devices will surely continue to evolve. The iPad, for example, may create a catalyst to the tablet computer space as the iPhone did for smart phones. To this end, regardless of how the various engine competitors and devices shape up, the nonplatform specific development that is advocated in the paper should make the key assets and development of FPCT viable and enduring.
Appendix A: Game Engine Overview on Specific Technical Variables

The discussion below outlines other key features of the four engines discussed above: Unreal Engine 3, Unity 3D, VBS2, and the CryENGINE 3.

This document analyzes four major 3D commercial engines and analyzes their features. These engines are as follows: Unreal Engine 3, Unity 3D, VBS2, and the CryENGINE 3.

**Unreal Engine 3** (http://www.unrealtechnology.com/)

**Overview:** Unreal Engine 3, the current development engine for FPCT is the most popular commercial engine because of its high-fidelity, flexibility and wide array of tools.

This is a link to a video released on the launch of Unreal’s latest engine iteration, which showcases some of its graphical capability and editor improvements:

**Pricing:**
Unreal Engine 3 – (NYD)
UDK – Free (for independents and students) If a product is sold using it, then there is a $99 upfront charge, and a 25% charge on all revenue above $5,000

**AI/Pathing:** Unreal Engine 3’s AI is made for deathmatch-based gameplay so it is not ideally suitable to our needs. The FPCT team has written the vast majority of the AI. The pathfinding system in Unreal is based off of pylons and should be improved in the newer version of the engine.

**Animation:** Unreal Engine 3 has many high fidelity animation features that are flexible and easy to use. These features include the ability to import animations from Maya and Max. The engine is also capable of using full body and facial motion capture data for animations. Unreal supports the use of blend shapes which are useful for soft deformations of the body as well as facial animation. Programmers can create skeletal controls which generate dynamic actions such as keeping a person looking a specific object while they walk or move. The engine also is packaged with FaceFx, an automated lip-syncing program integrated into the engine, which allows for fast and accurate lip-syncing. Unreal 3 also has a large tool set that focuses on in-engine animation blending, re-targeting, and timing which allows animators to fine tune their animations for the game space. These tools are graphical in nature and are relatively easy to use. They allow for programmers to create custom nodes for specific actions or states in the Animation Tree, the structure that allows animators to blend and time animations. The animation system as a whole is flexible and robust.

**Character Generation:** Unreal Engine 3 does not contain an in-engine character customization and generation solution. However the programmers of FPCT have created a system for character customization and generation in our version of the engine. Our tools allow for the generation of random characters with different heads and articles of clothing.

**Materials:** Unreal Engine 3 uses TGA files for textures, and has a built in material editor with high customization of materials shaders and effects. Unreal Engine 3 uses a node based material editor that has the ability to write custom HLSL shaders within the editor.
**Multiplayer:** Unreal supports multiple players in the gamespace.

**Networking:** Unreal Engine 3 has a built-in networking system. This system could be used to communicate with the other federates involved with FPCT.

**Programming and Scripting:** Unreal Engine 3 supports C++ and Unreal Script, an Unreal-specific scripting language. The license for the engine also includes access to source code so that new tools and features can be created as well as allowing the editing of existing tools.

**Sound:** Has a wide array of sound options, including volumetric sound and sound occlusion.

**User Interface:** Unintuitive interface, and not generally supported by the developers. UT3 supports Scaleform, an artist-centric UI creation tool that is used in some of the most popular game releases because of its flexibility and its ability to take the weight off of the coders and place it on the artists.

**World Space:** Unreal Editor 3 supports both additive and subtractive BSP functions, static meshes, and has large terrain support. Lighting can be generated both dynamically and using static light maps. Terrains can be painted with layers using any material, and populated with foliage and static meshes using decoration layers. Decoration layers allow for randomized scaling and placement of foliage and static meshes. Terrains can be imported via heightmap data from other programs such as Terragen, or generated in editor via deforming tools.
Virtual Battlespace 2 (VBS2) (http://virtualbattlespace.vbs2.com)

**Overview:** VBS2 is an engine that focuses on kinetic simulation and training technology. Many features for army simulations are already integrated, such as a vehicle interaction interface and smoother terrain generation based on data. The Virtual Battle Space engine places an emphasis on the interactions of physics and dynamics, rather than the interactions between characters.

This is a link that showcases some of the capabilities of VBS2:
http://www.youtube.com/watch?v=RYutmZ19Xso&feature=related

**Pricing:** VBS2 – (NYD)

**AI/Pathing:** The Virtual Battlespace 2 has the VBS2Fusion API. It is compatible with C++ and allows for multiplayer use. With VBS2Fusion triggers can be created, modified, deleted, activated and deactivated programmatically. Waypoints can be monitored, created, modified or deleted dynamically. Autodesk’s Kynapse AI middleware package is compatible with VBS2 and provides for agent perception, path-finding, and crowd simulation in agents. Presagis’ AI Implant provides similar functionality.

**Animation:** VBS2’s animation system is hands-off, and developers must use the skeletal rigs that come with VBS2 rather than integrating their own. VBS2’s supplied rigs do not contain facial animation, therefore it will be impossible to transition FPCT’s facial animations over unless there is a middleware supported by VBS2 to make this transition. This means that all of FPCT’s existing animations may be lost. If Bohemia Interactive would allow the team to send the animations for them to implement, it would take significantly more time, especially if they are difficult to edit animations. Similarly to Unity3D, all rigs and animations must be manipulated through code, so the programmer’s would need to create their own tool to work with VBS2 to make it easier for the artists to test their animations without a coder’s oversight.

**Character Generation:** Bohemia Interactive has stated that in the current version of VBS2, customized randomization of character models (clothes, appearances, etc.) is impossible. It might be possible if we work with their Fusion team to implement this feature for us, but we rely on them for development and they may not develop the tool how we want it.

**Materials:** VBS2 uses a proprietary PAA texture format. It supports creation of materials through the Oxygen2 modeling editor, which has a list of different shaders for use. Currently VBS2 supports normal and specular maps, but they do not showcase either in many of their demos.

**Multiplayer:** VBS2 supports multiple players in the gamespace, and the developers have cited that we should be able to get 256 NPCs in one area, easily. Bohemia Interactive also alluded to the possibility of developing in such a way that we could get over a thousand.

**Networking:** VBS2 has a built in networking system. This system could be used to communicate with the other federates involved with FPCT.

**Programming and Scripting:** VBS2 uses VBS script and can use a C++ interface. The VBS script, however, is very closed off and does not allow for the addition of new variables into the system. The license does not give access to source code, making it very difficult to implement custom features.
**Sound:** Has a standard set of sound options, with some special features such as sound occlusion and speed of sound. VBS2 contains a large sound package with realistic vehicle and weapon themed sounds.

**User Interface:** The editor includes some built-in interfaces, and claims that custom interfaces are possible.

**World Space:** Visitor 3 supports importation of large areas up to 100x100km using modified satellite imagery that contains information on placement of materials, roads, foliage, buildings, and other static objects. A dynamic lighting system simulates realistic lighting effects, as well as realistic shadow casting from objects and clouds. The modeling editor, Oxygen2 is used for editing the terrain for further tweaking, as well as static object placement.
**CryENGINE 3** (http://mycryengine.com/)

**Overview:** CryENGINE 3 is a well-known powerful engine that is used in current-gen games to achieve the highest level of graphical detail. It has more robust versions of all of Unreal’s toolset, including some that Unreal has yet to receive.

Here is a tech demo that shows Crytek’s capabilities:
http://www.gametrailers.com/user-movie/cryengine-3-gdc-2009-tech/312567

**Pricing:**
- CryENGINE 3 – (NYD)
- CryENGINE 3 Educational – Free

**AI/Pathing:** CryENGINE 3 has the Flow-Graph as a visual scripting system, allowing designers to create and control events. LUA scripting can be used to enhance the visual system, and is comparable to Unreal 3.0 (as far we know). Sight and hearing functionality for agents is included with the AI system. CryENGINE 3 contains automated mesh navigation which claims to be superior to waypoint-driven approaches. The navigation mesh system works both for indoor, multi-story, structures as well as outdoor environments. CryENGINE 3’s dynamic navigation makes it possible for agent paths to be created, modified, or destroyed in real-time.

**Animation:** CryENGINE 3 contains a very robust animation system. The engine allows for the importing of animations from Maya, Max, and SoftImage. The engine can also implement full body and facial motion capture from animation files, create facial animations from video footage and allows for automated lip-synching. CryENGINE 3 also has the ability to use blend shapes, which allows for soft deformations of the body and for facial animations. The tools inside the engine allow for animation blending, layering, re-targeting, and timing. These tools are graphical and relatively easy to use, flexible, and robust. The engine allows animators to create complex animation trees, without the need of a programmer, with various predefined states which tell the animations when to play during the game. The engine also has a built in tool for the synchronization of animations between multiple characters, such as shaking hands. CryENGINE 3 also has a dynamic inverse kinematics solver which programmers can use to dynamically generate animations such as keeping the foot from clipping through uneven terrain or keeping the head facing a certain direction while a character is moving.

**Character Generation:** CryENGINE 3 has a built in character customization and generation system that is similar to the system that the FPCT team has created for Unreal 3. In addition to allowing characters to be generated from different heads and articles of clothing the system also allows artists to create people that are taller, shorter, fatter, or skinnier.

**Materials:** CryENGINE 3 uses DDS files for textures, and has a built in material editor that has high customization of material shaders and effects. The CryENGINE 3 Sandbox editor has a menu based material editor that allows users to plug in different shader effects to each material.

**Multiplayer:** CryENGINE 3 supports multiple players in the gamespace.

**Networking:** CryENGINE 3 has networking code built. This could be used to communicate with the other federates involved with FPCT.
**Programming and Scripting:** CryENGINE 3 uses C++ and LUA scripting. There are licenses that have access to source code. This would allow programmers to create new tools and edit existing ones.

**Sound:** Has even more sound options than Unreal, along with a few added options such as sound moods and pre-defined sound mixes.

**User Interface:** Uses Scaleform, which is integrated and packaged with the engine.

**World Space:** Sandbox editor supports additive BSP, static meshes, and a robust terrain generation suite. CryENGINE 3 supports real time dynamic global illumination and deferred lighting resulting in natural lighting with dynamic soft shadows. CryENGINE 3 terrains can be imported via heightmap or generated in the editor, materials can be applied and then painted on in layers. The same holds true for foliage and static meshes, which can be populated and randomly scaled on the fly.
**Unity3D** ([http://unity3d.com](http://unity3d.com)):

**Overview:** A newer commercial engine that focuses on Web-based and mobile platform deployment. It is one of the leading 3D engines to be integrated onto the web and the iPhone. Currently they are using a development model that focuses on multi-platform development, citing that you can develop once and port with the push of a button across PC, browser, iPhone, Android, Xbox360, PS3 and Wii.

This is a link to their tech demo, which is navigable through the browser: [http://unity3d.com/gallery/live-demos/index.html#tropical-paradise](http://unity3d.com/gallery/live-demos/index.html#tropical-paradise)

**Pricing:**
- Unity Indie – Free
- Unity iPhone Indie – +$400
- Unity Pro – $1499 (Required for corporations making more than $100,000 a year)
- Unity iPhone Pro – +$1499 (addition to purchase of a $1499 Unity Pro License)
- Unity Educational License prices are negotiable, with no set price listed on the site.

**AI/Pathing:** Unity does not contain a built in AI and pathing system. Any AI functions would have to be coded by the FPCT team. There are some community plug-ins for AI and pathing, but these are not officially supported by Unity.

**Animation:** Unity3D’s animation system is less versatile than Unreal Engine 3 or CryENGINE 3. Unity allows for the importing of animations and motion capture data with animation files from Maya and MotionBuilder. Unity does not officially support any type of blend shape and the community tools that allow for blend shapes are not very well developed. While there is animation blending and timing within the engine, it is all done through scripting which takes significantly more time to set up than the graphical systems that exist within Unreal Engine 3 and CryENGINE 3. Overall, the animation tools look somewhat incomplete and are slightly difficult to use. Unity simply does not have the high-fidelity animation options as Unreal Engine 3 or CryENGINE 3.

**Character Generation:** Unity 3D does not contain a character customization or generation system. It may be possible to create such a system with the tools available in the engine.

**Materials:** Unity3d uses PSD, TIFF, JPG, TGA, PNG, GIF, BMP, IFF, and PICT files for textures. It has a built in material editor that uses pre-existing shaders to change material appearance. It also supports custom shader creation through a scripting language called ShaderLab.

**Multiplayer:** Unity supports multiple players in the game space.

**Networking:** Unity has a built in networking system. This system could be used to communicate with the other federates involved with FPCT.

**Programming and Scripting:** Unity uses Javascript and C# for scripting and programming. Source code is available with the purchase of an additional license.
**Sound:** Has limited sound options, such as a lack of sound occlusion and volumetric sound, but does support sound streaming from the web, where audio can be streamed directly from a website into the game interface. GUI does not currently show how far sound reaches in gamespace.

**User Interface:** No graphical UI editor (all UI editing is code-based).

**World Space:** Unity editor supports static mesh placement, as well as terrain generation within the editor. Unity3d supports dynamic hard and soft shadows in the pro version, and vertex shading in the free version. Unity terrains support generation of terrain in editor using deforming tools, and can import heightmap data from other programs. Terrain textures can be painted on using the same deforming tools via a layering system in the terrain options menu. Trees and rocks can be painted onto the terrain and randomly scaled using Detail meshes.
Bibliography

DARPA Transformative Apps Call for Proposals, March 2010

http://www.apple.com/ipad/specs/ for hardware specifics, and


http://www.develop-online.net/news/32250/The-top-10-game-engines-revealed


http://unity3d.com/

http://www.onlive.com/about.html

http://www.unrealtechnology.com/


http://virtualbattlespace.vbs2.com

http://www.youtube.com/watch?v=RYutmZ19Xso&feature=related

http://mycryengine.com/

http://unity3d.com/gallery/live-demos/index.html#tropical-paradise