12-1-2008

Protecting Critical Infrastructure with Games Technology

Adrian Boeing
Edith Cowan University

Martin Masek
Edith Cowan University

Bill Bailey
Edith Cowan University, b.bailey@ecu.edu.au

Recommended Citation

Boeing, A., Masek, M., & Bailey, B. (2008). Protecting Critical Infrastructure with Games Technology. DOI: https://doi.org/10.4225/75/57a827f3aa0db

DOI: 10.4225/75/57a827f3aa0db
9th Australian Information Warfare and Security Conference, Edith Cowan University, Perth Western Australia, 1st December, 2008

Protecting Critical Infrastructure with Games Technology

Adrian Boeing[1], Martin Masek[1], Bill Bailey[2]
[1] – School of Computer and Information Science
[2] – School of Engineering
Edith Cowan University
{a.boeing,m.masek,b.bailey}@ecu.edu.au

Abstract

It is widely recognised that there is a considerable gap in the protection of the national infrastructure. Trying to identify what is in fact 'critical' is proving to be very difficult as threats constantly evolve. An interactive prototyping tool is useful in playing out scenarios and simulating the effect of change, however existing simulators in the critical infrastructure area are typically limited in the visual representation and interactivity. To remedy this we propose the use of games technology. Through its use, critical infrastructure scenarios can be rapidly constructed, tested, and refined. In this paper, we highlight the features of games technology and associated tools to creating immersive simulations of critical infrastructure and present our implementation.

Keywords

Critical infrastructure, simulation, protection, vulnerability, games technology

INTRODUCTION

Modern society relies upon various networks of infrastructure. Critical infrastructure are assets that when disrupted cause a large scale impact on society and the economy. Such infrastructure includes agriculture, water, energy, communications, transport, health, government, and defence. Each infrastructure network is complex, and each network is often interrelated (Rinaldi, 2001). A disruption in one network can propagate into others causing a set of interdependent failures. For example, a gas pipeline explosion that occurred in Western Australia on the 3rd June 2008, reduced the states gas supply by 35% (Gosch, 2008). The effect of this loss propagated through to affect the transport, building, hospitality, health, and mining industries leading to a total estimated economic loss to the state of AU$6.7 billion (Sydney Morning Herald, 2008).

There are several reasons why critical infrastructure can fail. In the example above, the rupture of a pipeline was a technical failure. The risk of such technical failures can be determined through standard risk engineering, where risk is measured as the probability of failure multiplied by the severity of the consequences. Such risks can be mitigated by analysing the probability of failure and building in safety margins, redundancies, and maintenance processes to lower the risk to an acceptable level.

Another cause for failure is deliberate interference and sabotage. Although a classical probabilistic-based risk approach is at times used to determine an appropriate level of protection, its validity is questioned. Manunta (2002) identifies potential shortcomings that challenge some of the assumptions behind the use of classical risk based approaches. If a security specialist uses the probability of an attack to identify areas to defend, an attacker can utilise the same probabilities to identify areas that will not be defended. Such threats are not static and can evolve when they encounter counter-measures.

Many organisations perform security risk assessments only once a year, although since September 2001, some organisations undertake reviews on a more frequent basis (Gibson, 2003). Manunta (2002) questions the relevance of such ad hock reviews of risk, because by the time the results arrive, they tend to be outdated. The context will dictate the how quickly such results become invalid. But in the current security climate, the terrorist threat is dynamic, evolutionary, and able to adapt and learn.

If elements in the risk analysis are seen to be at fault, so too is the method by which the risk assessment has been carried out. In the majority of the cases, those that have been conducting the risk analysis have not always been the required experts in their field who can truly identify what is deemed 'critical'. Due to the interdependent nature of critical infrastructure, this often requires a number of multiple experts to be involved in the process, which is often problematic. If their expertise could be harnessed in a way that allows security mangers to understand the complexities of the infrastructure involved, then it may be possible to develop improved risk assessments.

SIMULATION FOR THE ANALYSIS OF CRITICAL INFRASTRUCTURE

Our proposed solution to the problems outlined above is an interactive, high fidelity simulation tool built by adapting existing games technology. Simulation tools help bridge the gap in understanding between experts from different disciplines by determining the effect of changes to the system whilst hiding its complexity (Talend, 2008). This makes each expert area accessible to the whole team. Furthermore, the online nature of modern computer games gives greater opportunities for remote experts to perform security reviews and exercises.

Several tools exist for the simulation of critical infrastructure. Pederson et al. (2006) identified 30 simulation systems in a survey of research into critical infrastructure interdependency modelling. These simulations range in maturity from research to commercial systems and model a variety of infrastructure using various simulation types.

Most existing critical infrastructure simulations provide a simple single-user graphical visualization of the infrastructures and their interdependencies. This typically consists of a graph display, with infrastructure nodes connected by edges to show interdependencies. Some simulations are integrated with a Geographic Information Systems (GIS) application and the graph is overlaid over a top-down 2D map of the area. Such displays are useful for traditional risk-based analysis in the design and analysis of critical infrastructure systems, but not for identifying local security vulnerabilities.

THE ROLE OF GAMES TECHNOLOGY

Coupling a critical infrastructure simulator to a 3D environment provides several possibilities for security analysis. For example, accurate views from existing and proposed security cameras can be analysed, as can views from locations where infrastructure could be observed by an attacker. These visibility studies can be performed under various simulated environmental conditions, as demonstrated in Figure 1. Adding a 3D positional sound system would allow the evaluation of audible systems, such as the effectiveness of alarm networks in an emergency.

A simulator that lets users take the roles of a person on the ground allows for the enactment of realistic scenarios, allowing response times and behaviour patterns to be determined. Finally, the inclusion of artificial intelligence (AI) controlled characters in the simulator allows for the simulation of human activities. This is useful for automatic testing of the infrastructure and simulating the behaviour of large crowds. A major benefit of simulating the infrastructure is that security tests can be applied before infrastructure is build, allowing for lower cost design alterations to minimise vulnerabilities.

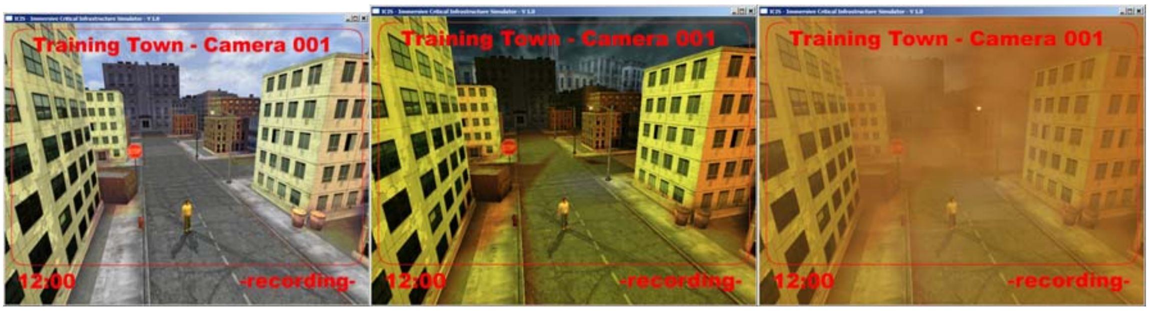

Training Town - Camera 001
12:00
-recording-
Training Town - Camera 001
12:00
-recording-
Training Town - Camera 001
12:00
-recording-

Figure 1: Simulating different environmental conditions.

By leveraging games technology there is the ability to present the ‘intruder’ into the visual concept thus allowing for a greater assessment of the potential capability of actually inflicting damage. Many times this is a perceived threat that is not borne out by the reality of the known situation. Moreover, once a target has been identified there is the added ability to develop variations of target hardening, which can be tested prior to implementation. This has substantial cost benefits as it is not necessary to actually build in the suggested changes, in order to test them against the ‘intruder’. An example scenario of an adversary seeking to sabotage the power infrastructure is shown in Figure 2.

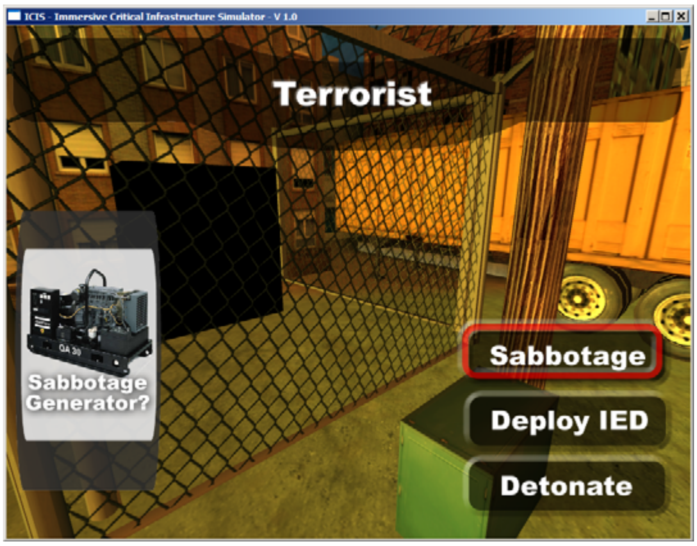


Figure 2: Introducing adversaries into the simulation allows for interactive testing and refinement of countermeasures.

GAME ENGINE TECHNOLOGY REVIEW

We now give an overview of the tools and technology used to create modern games to demonstrate their applicability in creating virtual worlds for infrastructure modelling and simulation. The majority of computer games are not created 'from scratch', but rather use and extend on existing components. The sum of these components that are used to create a game is called a game engine. Game engine functionality includes computer code that implements features such as audio, graphics, network, physics, and AI components. In addition, most game engines provide visual interactive tools for using these systems and populating them with the actual content. For example, the game engine includes the code that will draw objects in 3D perspective with various effects such as lighting and fog, whilst the tools are used to construct the actual objects that will be drawn.

In creating an immersive environment for critical infrastructure simulation there are many game engines to choose from. DevMaster.net is a database of game engines, listing features and including user reviews (DevMaster.net), at time of writing containing information on 287 game engines. Some engines are free using various open source licences, whilst others may cost hundreds of thousands of dollars to licence. Engine quality, ease of use and fidelity of the environment produced also vary with game engines.

The cost of the technology license is only one factor to consider. The other major factors are support costs, which hinge on the quality of the tools provided with the game engine. This includes the quality of the world editor, script development environment, and the tools that are used to import the 3D models from their various formats to the one the game engine understands.

Finally, a major influence on the cost of developing a simulator is the level of support that is available from the vendor. Open source and independent games often have less support from the developers than professional offerings, however they often have a large community that is willing to provide support. Additionally, these platforms often contain books, websites, or research publications that can greatly assist developers, unlike other more traditional offerings.

Audio

Game engines typically support background audio, used to play music and menu sounds, along with 3D positional sound. Positional sound is emitted from a specific location in the world. Its perceived loudness is affected by distance, velocity, and occlusion by objects in the world. In an immersive critical infrastructure simulation, positional sound can be used to answer questions related to audibility of sound emitted by intruders, alarms, and the infrastructure itself. The simulation can be used to test different alarm locations, placement of barriers that can only be breached by making noise, and designing the layout to place personnel in locations where sound from malfunctioning infrastructure can be detected.

Graphics

The component of the game engine responsible for displaying the content to the screen is the graphics engine. The main functionality of the graphics engine is processing and displaying of 3D information in the world. This includes displaying and animating 3D computer controlled characters, lighting effects (e.g.: day and night), 3D representations of buildings, foliage, and all other content. The graphics engine also has the ability to display 2D information overlayed over the 3D view. An example of this is the graphical user interface that allows the user to press buttons and receive information from the simulator, as shown in Figure 3. The graphics is the primary means for providing information to the user of a critical infrastructure simulation, and is crucial for the simulation of visually based devices, such as security cameras. The graphical display can be used to provide information regarding the location of intruders, points of failure, the area monitored by security systems, etc.

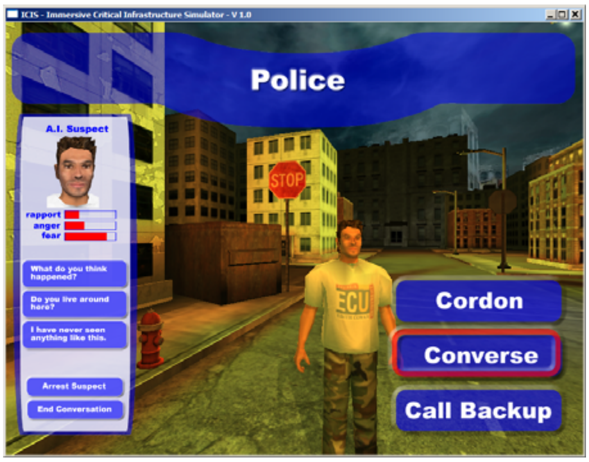

ICIS - Immersive Critical Infrastructure Simulator - V 1.0
Police
A.I. Suspect
rapport
anger
fear
What do you think happened?
Do you live around here?
I have never seen anything like this.
Arrest Suspect
End Conversation
STOP
ECU
Cordon
Converse
Call Backup

Figure 3: A Game Engine typically provides support to view graphics with 3D perspective and also 2D graphics for menus.

The 3D world can be viewed from several cameras, representing the view of different persons or actual cameras in the world. The top down view, much like the one in traditional 'big-picture' simulators can also be gained by positioning a camera above the scene looking down, as shown in Figure 4. This kind of view can be used to gain a general overview of the scenario, and is also useful in a de-briefing situation offering a convenient perspective for re-playing a scenario that had been run previously.

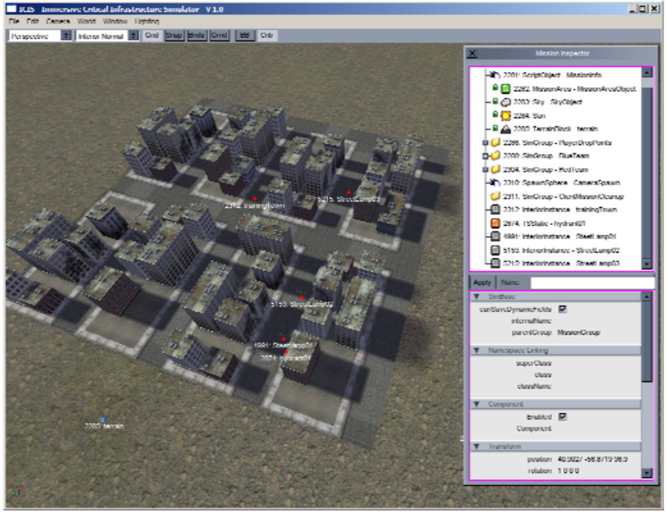

ICIS - Immersive Critical Infrastructure Simulator - V 1.0
File Edit Camera World Window Lighting
Perspective
Interior Normal
Grid
Snap
Bnds
Ornd
Orb
Mission Inspector
2281: ScriptObject - MissionInfo
2282: MissionArea - MissionAreaObject
2283: Sky - SkyObject
2284: Sun
2285: TerrainBlock - terrain
2286: SimGroup - PlayerDropPoints
2288: SimGroup - BlueTeam
2304: SimGroup - RedTeam
2310: SpawnSphere - CameraSpawn
2311: SimGroup - ClientMissionCleanup
2312: InteriorInstance - trainingTown
2874: TSStatic - hydrant01
4881: InteriorInstance - StreetLamp01
5150: InteriorInstance - StreetLamp02
5215: InteriorInstance - StreetLamp03
Apply
Name
SimBase
canSaveDynamicFields ☑
internalName
parentGroup MissionGroup
Namespace Linking
superClass
class
className
Component
enabled ☑
Component
Transform
position 40.9027 -56.8719 98.9
rotation 1 0 0 0
2312: trainingTown
5215: StreetLamp03
5150: StreetLamp02
4881: StreetLamp01
2874: hydrant01
2285: terrain

Figure 4: A 'Big-Picture' overview can be obtained using an overhead camera. This can be used by those acting in an observer role, or for re-playing missions as part of a de-briefing.

Network

In a computer game, playing against human opponents over a network provides extra challenge. Most 3D game engines offer networking components to facilitate this. In an immersive simulation, network communication allows geographically dispersed teams to evaluate scenarios. When interdisciplinary teams are concerned, as in the case of critical infrastructure simulation, the expertise required to analyse infrastructure interdependencies and determine vulnerabilities may be dispersed across continents. Providing a networked simulator allows the selection of team members based on expertise, rather than geographic proximity.

Physics

A physics engine is responsible for calculating how objects move and interact within the environment. The physics engine detects collisions between interacting objects, calculates the response forces, and computing the updated locations of the simulation objects. This allows the simulation of objects falling over (such as trees falling onto power lines, or across roads) and the effects of an attack on critical infrastructure (such as a vehicle crashing into a complex). This greatly increases the fidelity of the simulation. The physics engine can also be used to update the audio subsystem in a physically realistic manner, such as providing different sounds based on the contents on a container. This information can be used to educate security personal about how something should sound.

Artificial Intelligence

The artificial intelligence system contains functionality for automatically generating the behaviour of intelligent entities within the game. This functionality typically depends on the type of application the game engine targets, such as a first-person shooter or a real time strategy game. Common components of an AI system are path finding, behaviour based systems, and planning. Path finding enables a computer controlled character to navigate from one location to another. This allows the simulation of an intruder in a given scenario and allows the situation to be replayed and respond to modification of the environment (e.g. creating obstacles such as fences).

The behaviour subsystem of an AI engine allows different behaviours to be assigned to a character. Typical examples are adding an avoidance behaviour for one character, and a chase or attacking behaviour for another. This technology can be used to simulate large crowds of people, allowing the evaluation of evacuation plans or crowd dispersal during a terrorist attack, as illustrated in Figure 5.

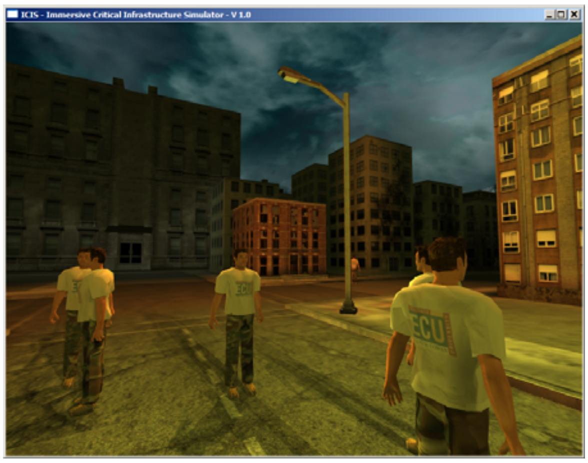
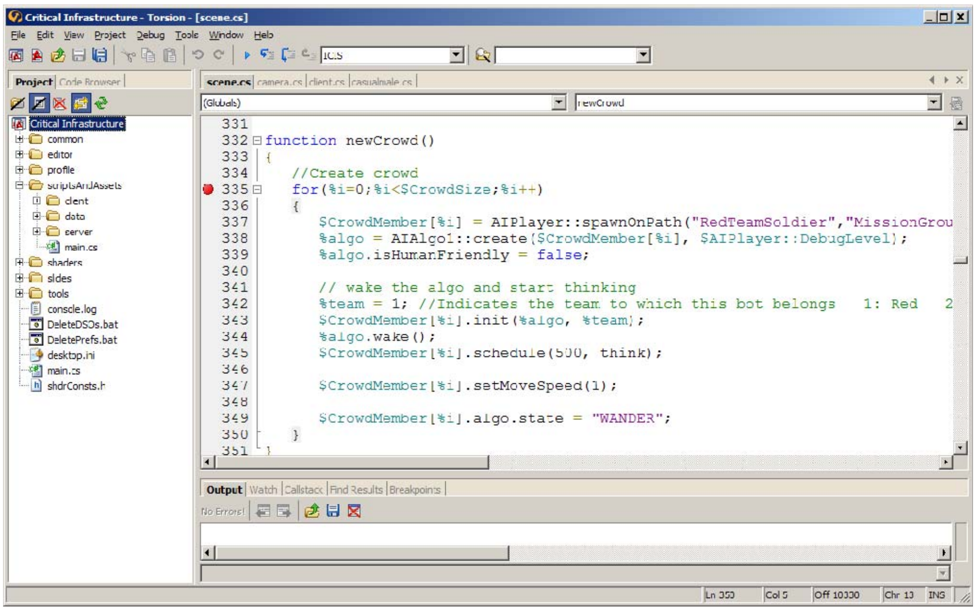


Figure 5: Computer controlled characters can be created using an AI engine.

Scripting

Each game has unique differences in mechanics and gameplay. In order to realize these using a generic game engine requires the setup of the game logic, or game rules. This is a programming task, however the size and complexity of the game engine program code makes it challenging to modify this code directly. To speed up development, and minimize the introduction of errors into the engine code, engines typically offer a simplified programming interface to the engine using a scripting language. Available scripting languages range from functional and object-oriented languages to flow-chart based graphical tools. An example for the Torque Game Engine Advanced is given in Figure 6, showing the script editor Torsion.

The logic for critical infrastructure simulations includes encoding their interdependencies and models for production, transportation, and storage. These can be implemented in script using suitable data structures and algorithms. Scripting languages offer an interface to the other modules of the game engine, and thus can be used to customise these. For example, custom artificial intelligence entities, such as for security patrols and intruders can be defined with specific behaviours and physical parameters.



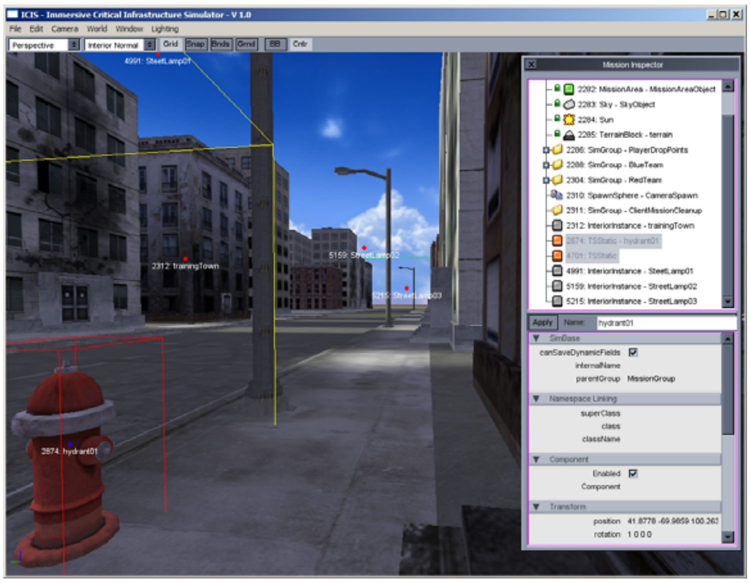
```
function newCrowd()
{
   //Create crowd
   for(%i=0;%i<$CrowdSize;%i++)
   {
      $CrowdMember[%i] = AIPlayer::spawnOnPath("RedTeamSoldier","MissionGrou
      %algo = AIAlgo1::create($CrowdMember[%i], $AIPlayer::DebugLevel);
      %algo.isHumanFriendly = false;

      // wake the algo and start thinking
      %team = 1; //Indicates the team to which this bot belongs   1: Red   2
      $CrowdMember[%i].init(%algo, %team);
      %algo.wake();
      $CrowdMember[%i].schedule(500, think);

      $CrowdMember[%i].setMoveSpeed(1);

      $CrowdMember[%i].algo.state = "WANDER";
   }
}
```

Figure 6: Aspects of the simulation can be customised using a scripting language. Torsion is an integrated development environment (IDE) for the Torque game engine, providing modern features such as syntax highlighting, word completion, and the ability to step through script code and view contents of variables.

Game Engine Tools

The difference between engines costing hundreds of thousands of dollars, and ones offered for free is often not so much in the graphics and realism possible, but in the quality of the supporting toolsets. This includes the quality of the world editor, script development environment, and the utilities that are used to import the 3D models from their various formats to the one the game engine understands.

The useability of the world editor is integral to the viability of a game engine. Through the world editor the user can construct and modify the environment. For example, a user may test placing lights at different positions to see how it affects the visible area. Alternatively the world editor can be used to interact with intelligent agents. If an "intruder" agent is attempting to enter a premises the user can experiment with different environmental configurations, such as placing additional fences around the building. A world editor will let the user perform the desired operations efficiently has the potential to significantly cut the development time for scenarios. The world editor for the Torque Game Engine Advanced is shown in Figure 7. This world editor is integrated into the game engine allowing interactive editing of the scenario whilst the simulation is running.



2202: MissionArea - MissionAreaObject
2283: Sky - SkyObject
2284: Sun
2285: TerrainBlock - terrain
2286: SimGroup - PlayerDropPoints
2288: SimGroup - BlueTeam
2304: SimGroup - RedTeam
2310: SpawnSphere - CameraSpawn
2311: SimGroup - ClientMissionCleanup
2312: InteriorInstance - trainingTown
2874: TSStatic - hydrant01
4701: TSStatic
4991: InteriorInstance - SteetLamp01
5159: InteriorInstance - StreetLamp02
5215: InteriorInstance - StreetLamp03

Name: hydrant01
canSaveDynamicFields ☑
internalName
parentGroup MissionGroup
superClass
class
className
Enabled ☑
Component
position 41.8778 -69.9859 100.263
rotation 1 0 0 0

Figure 7: A world editor allows the developer to graphically position objects in the world and modify their parameters.

Whilst the world editor allows for modification of the 3D world, the Graphics User Interface (GUI) editor allows for the creation of the 2D menu components. The GUI editor from the Torque Game Engine Advanced is shown in Figure 8, allowing various elements such as buttons and labels to be placed over the 3D scene.

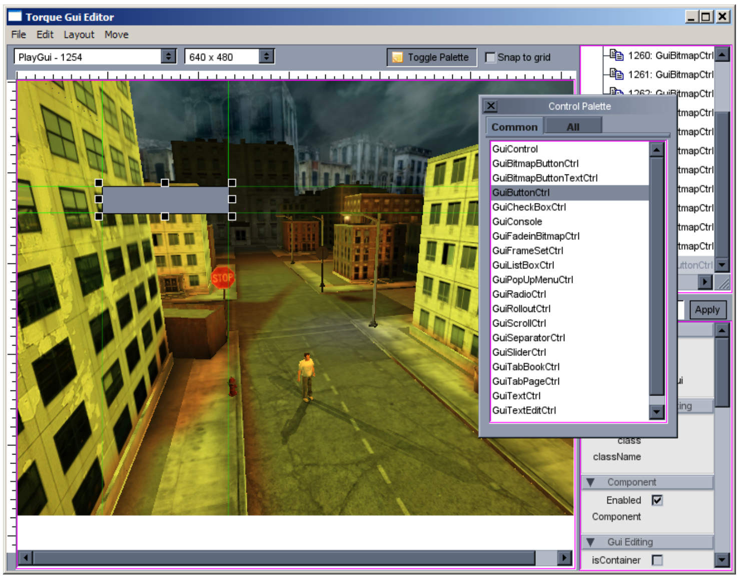


Figure 8: The GUI editor in the Torque Game Engine Advanced allows 2D menus and displays to be overlayed on the 3D visuals of the world.

IMMERSIVE CRITICAL INFRASTRUCTURE SIMULATOR

Our work-in-progress, the Immersive Critical Infrastructure Simulator (ICIS), has been undergoing development from 2007 using the Torque Game Engine and Torque Game Engine Advanced from Garage Games (2008).

ICIS consists of an interactive 3D world in which computer networks, along with power, water, and gas producers and consumers and their interdependencies are modelled as a set of nodes connected by edges. The nodes represent the producers and consumers (e.g. a node may consume gas and produce electricity) and the edges represent the flow of a resource (e.g. electricity from a producer node to a light consumer node). The nodes are visualised as 3D objects, intended to resemble their real-world counterparts, placed in the virtual world. ICIS has the ability to run on a server, allowing remote client computers to connect to and interact with the world. Users of these clients can assume a number of roles, including essential services staff and adversaries.

CONCLUSION

Threats to critical infrastructure are not passive. They constantly evolve, and so too must the ability to defend against such threats. An interactive high-fidelity simulation tool greatly adds to the ability of experts to conduct multiple risk assessments often without the need to physically deploy, providing a considerable cost saving on both time and travel.

Leveraging games technology allows simulation scenarios to be rapidly constructed at a very high fidelity and immediately tested. The networked gaming technology allows remote experts to interact in an intuitive environment and explore, identify and assess the critical components of the infrastructure. The scenario can be modified and different configurations can be examined and tested to ascertain the impact of the change on the risks to the critical infrastructure.

ACKNOWLEDGEMENTS The authors would like to acknowledge the work of Dion Beetson, Stephen Price, Tristan Franco, Peter Grundy, and Jarryd Pinto in the development of the Immersive Critical Infrastructure Simulator (ICIS).

REFERENCES

DevMaster.net, 3D engine database, Accessed: 27th August 2008 at http://www.devmaster.net/engines/

Garage Games (2008), Torque Game Engine, Accessed: 27th August 2008 at http://www.garagegames.com/

Gibson, S.D. (2003), The case for 'risk awareness', Security Journal, 16, 55-54.

Gosch, E. (2008), WA gas supply cut 30pc by blast at Varanus Island, The Australian, Retrieved 25th August 2008 from: http://www.theaustralian.news.com.au/story/0,25197,23824148-5006789,00.html.

Manunta, G. (2002). Risk and security: are they compatible concepts?, Security Journal, 15, 43-55.

Pederson, P., Dudenhoeffer, D., Hartley, S., and Permann, M. (2006), Critical infrastructure interdependency modeling: a survey of U.S. and International research, Idaho National Laboratory.

Rinaldi, S.M., Peerenboom, J.P., and Kelly, T.K. (2001), Identifying, understanding, and analysing critical infrastructure interdependencies, IEEE Control Systems Magazine, IEEE, 21, 11-25.

Sydney Morning Herald (2008), WA faces $6.7b gas bill, Business Day, The Sydney Morning Herald, Retrieved 25th August 2008 from: http://business.smh.com.au/business/wa-faces-67b-gas-bill-20080710-3cxn.html.

Talend, D. (2008), Prototyping of the virtual type, ControlDesign.com, Retrieved 26th August 2008 from: http://www.controldesign.com/articles/2008/168.html

COPYRIGHT