Automated run-time mission and dialog generation

Kelly, John David

Monterey, California. Naval Postgraduate School

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

AUTOMATED RUN-TIME MISSION AND DIALOG GENERATION

by

John David Kelly

March 2007

Thesis Advisor:	Chris Darken
Second Reader:	Perry McDowell

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE March 2007	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE Automated Run-Time Mission and Dialog Generation	5. FUNDING NUMBERS
6. AUTHOR(S) LT John D. Kelly	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

Current mission driven systems, be they games or training simulations, are generally restricted to using a set of training missions that are hard coded into the system. This has the unfortunate effect of limiting the number of times a person or team can be sent through a simulator before it begins to lose its training value or the number of times a person can replay a game without it becoming predictable and somewhat boring. The fact that all of the mission parameters must be hard coded also increases the time required for scenario development. This study defines an architecture for automating the creation of missions at run time allowing much larger variety in the number and content of missions in a given system. Our architecture also allows for the creation of varied and more believable dialog with minimal scenario creation time required. We also explore an alternate method for determining agents attitudes towards the users avatar which is more robust than the more commonly used system and which can be used as an input to dialog generation further improving the realism of the dialog. A commercial game, Neverwinter Nights by Bioware, has been used to produce a proof of concept.

14. SUBJECT TERMS Artificial Intelligence, Natural Language Processing, Social Network Analysis, Simulation, Automated Scenario Generation	15. NUMBER OF PAGES 75
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution unlimited

AUTOMATED RUN-TIME MISSION AND DIALOG GENERATION

John D. Kelly
Lieutenant, United States Navy
B.S., North Carolina State University, 2000

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MODELING, VIRTUAL ENVIRONMENTS AND SIMULATION (MOVES)

from the

NAVAL POSTGRADUATE SCHOOL
March 2007

Author: John D. Kelly

Approved by: Christian Darken
Thesis Advisor

Perry McDowell
Second Reader

Rudolph Darken
Chairman, MOVES Academic Committee

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Current mission driven systems, be they games or training simulations, are generally restricted to using a set of training missions that are hard coded into the system. This has the unfortunate effect of limiting the number of times a person or team can be sent through a simulator before it begins to lose its training value or the number of times a person can replay a game without it becoming predictable and somewhat boring. The fact that all of the mission parameters must be hard coded also increases the time required for scenario development. This study defines an architecture for automating the creation of missions at run time allowing much larger variety in the number and content of missions in a given system. Our architecture also allows for the creation of varied and more believable dialog with minimal scenario creation time required. We also explore an alternate method for determining agents attitudes towards the users avatar which is more robust than the more commonly used system and which can be used as an input to dialog generation further improving the realism of the dialog. A commercial game, Neverwinter Nights by Bioware, has been used to produce a proof of concept.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION....1
A. BACKGROUND INFORMATION....1
B. PROBLEM STATEMENT....3
C. TECHNICAL APPROACH....4
D. CONTRIBUTIONS....4

II. LITERATURE REVIEW....7
A. INTERNAL REPRESENTATION OF AGENT KNOWLEDGE....7
B. AUTOMATED MISSION GENERATION....7
C. AUTOMATED DIALOG GENERATION....9
D. SOCIAL NETWORKS....11
E. CHOICE OF IMPLEMENTATION....11

III. GENERAL SOLUTION....13
A. INTRODUCTION....13
B. MISSION AND DIALOG GENERATION....13
C. SOCIAL NETWORKS....15

IV. SPECIFIC APPLICATIONS....19
A. INTRODUCTION....19
B. NAVAL COMBAT SIMULATION....19
C. URBAN ENVIRONMENT SIMULATION....20

V. PROOF OF CONCEPT....23
A. MISSION AND DIALOG GENERATOR IMPLEMENTATION....23
1. Z-Dialog....25
2. Knowledge Representation....25
3. Threat and Item Generation....26
4. Mission and Dialog Generator....28
5. Implementation Advantages....33
B. SOCIAL NETWORKS IMPLEMENTATION....35
1. Trivial Case....36
2. Degenerate Case....38
3. Friend of a Friend Case....39
4. Hermit Case....40
5. Man in the Middle Case....42
6. Cult Leader Case....44
7. Dictator Case....45
8. Village Example....47

VI. CONCLUSIONS AND FUTURE WORK....53
A. CONCLUSIONS....53
B. FUTURE WORK....54

LIST OF REFERENCES....57

INITIAL DISTRIBUTION LIST 59

LIST OF FIGURES

Figure 1. Grammar Template Example 10
Figure 2. Basic Architecture Line Diagram 13
Figure 3. Influence Matrices 16
Figure 4. Weighting Equations 17
Figure 5. Solving for x Vector 17
Figure 6. Prototype Scenario 23
Figure 7. Village Scenario 24
Figure 8. Initial Welcome Screen 29
Figure 9. Soldier Conversation Part I 30
Figure 10. Soldier Conversation Part II 32
Figure 11. Test Case Events 35
Figure 12. A and B Matrices for Trivial Case 36
Figure 13. Derived Influence Matrix for Trivial Case 37
Figure 14. Attitude Change Vectors for Trivial Case 37
Figure 15. A and B Matrices for Degenerate Case 38
Figure 16. Derived Influence Matrix for Degenerate Case 38
Figure 17. Attitude Change Vectors for Degenerate Case 39
Figure 18. A and B Matrices for Friend of a Friend Case 39
Figure 19. Derived Influence Matrix for Friend of a Friend Case 40
Figure 20. Attitude Change Vectors for Friend of a Friend Case 40
Figure 21. A and B Matrices for Hermit Case 41
Figure 22. Derived Influence Matrix for Hermit Case 41
Figure 23. Attitude Change Vectors for Hermit Case 42
Figure 24. A and B Matrices for Man in the Middle Case 42
Figure 25. Derived Influence Matrix for Man in the Middle Case 43
Figure 26. Attitude Change Vectors for Man in the Middle Case 43
Figure 27. A and B Matrices for Cult Leader Case 44
Figure 28. Derived Influence Matrix for Cult Leader Case 44
Figure 29. Attitude Change Vectors for Cult Leader Case 45
Figure 30. A and B Matrices for Dictator Case 46
Figure 31. Derived Influence Matrix for Dictator Case 46
Figure 32. Attitude Change Vectors for Dictator Case 47
Figure 33. Family Matrix 48
Figure 34. Work Matrix 48
Figure 35. Friend and Foe Matrix 49
Figure 36. Weighted Matrix 50
Figure 37. A Matrix 50
Figure 38. B Matrix 51
Figure 39. Derived Matrix: $(I-A)^{-1}B$ 51
Figure 40. Event and Attitude Change Vectors 52

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

Table 1. Generic List of Role Playing Game Mission Types 33

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

This thesis would not have been completed without the help and support of several people who I'd like to take this time to thank:

My advisor, Chris Darken, who accepted my mercurial changes in topic with good grace and enthusiasm and kept me working even when my brain told me it was fed up.

My parents, David and Janice Kelly, for their unwavering support and unending questions that made me reevaluate things I thought were obvious.

My first AI teacher, Michael Young, for getting me hooked on the topic.

My fellow MOVES cubicle dwellers, for all the comic relief and WAY too much alcohol when it was needed.

The Sons of Wackonia, for just being them.

And finally Sarah Sybert, for keeping me looking forward and distracting me when I needed it.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND INFORMATION

Simulations are used for many purposes in the military including that of training. In any simulation that requires interaction between units and or echelons of command some modeling of communication becomes necessary. For many simulations human trainers do this and if it is modeled by a computer, it is canned text which is the same every time the simulation is run. This same problem is seen in the electronic entertainment industry, especially in the area of role playing games. Role-playing or adventure games are those where the player controls an avatar in the game world and interacts with people and objects in that world to achieve particular goals. The game world can be realistic or fantastic but the type of interactions can be directly mapped to military simulations of various types. Since these games share the same problem of predictable interactions with simulations it makes sense to leverage the accessibility and scripting capabilities of these games to develop a generic architecture to address these problems that can be easily adapted to both program types.

Currently missions for conventional simulations and role playing games have to be hard coded into the system making subsequent iterations predictable thus reducing the training value and enjoyment factor of replaying the scenario multiple times. As of now this is the standard approach in the game industry making most role-playing games good for only a single time through, twice if the player goes back to complete the side quests. This approach also consumes huge amounts of the programmers' time, as they have to script each of the possible actions and story line branches. A better alternative would be a system that takes information about the environment and produces possible missions for the user during run time. This would allow a greatly expanded number of missions for the game or simulation and make the number of iterations without repeating a specific mission much greater.

Examples of this are military simulations built using plug-ins to SSDS and AEGIS systems for the USN ships. The systems can be put into a training mode and have simulated land masses and contacts piped directly to the operators screens but these

simulations are either entered manually ahead of time or they must be entered at runtime by an operator working behind the scenes who more often than not needs the training just as much as the trainee. Even when an operator runs the scenario, in order to ensure all of the training goals are met, these 'red forces' are often working off of a script that is just as rigid as when it is all automated. This predictability in many cases causes the training value of the exercise to degrade because the trainees know what is going to happen and when it will, thus making the training session more akin to rehearsing for a play than anything else. It does little to prepare the trainees for what they must do when faced with a truly novel situation.

In games, this problem occurs with action and role-playing games such as the Final Fantasy series or Grand Theft Auto. Despite having increasingly large virtual worlds to roam in, the character still is restricted to certain storylines with that require users to complete very specific actions in order to progress. The Elder Scrolls series does a slightly better job but even they simply give the player more possible storylines allowing the player to ignore the primary ones if desired. Once a player has completed the set storylines often there is no further use for the game other than building up characters for the bragging rights involved.

Massively Multiplayer Online Games (MMOG) are notorious for their repetitive storylines. Examples of this are EverQuest, Asheron's Call, and the extremely popular World of Warcraft. In these games there is a set group of missions hard coded into the system and while certain missions are available to different character types, those never change. Once a player completes all current missions, there is nothing new for the player to do, until the developers release a patch with new content. Significant work has been done on the automated creation of testing scenarios for various computerized systems as well as the creation of interactive narratives. Few of these approaches have progressed past the academic stage though as they are too large and unwieldy for commercial use.

One of the main problems with having the various computer controlled non-player characters (NPCs) create new missions as they go is that it requires that the agents hold updated knowledge about the state of the world in order to avoid possible contradictions. The scalability problems related to trying to give an agent a

comprehensive knowledge of the problem space is immense. To have every agent in the world carry a list of every possible threat would become space prohibitive very quickly and the idea of expanding that across an arbitrarily large virtual world brings new meaning to the phrase 'bloat code'.

Another factor that reduces the believability is the formulaic manner of NPC speach that results from most pattern based templates for the generation of dialog. An average person told to relate a piece of information to several people is unlikely to use the exact same syntax in every instance as a computer generally will. Thus a certain amount of randomization or simply a greater vocabulary from which to work from is necessary. This is also true of speaking styles among people of various social, vocational, and educational strata. Consequently, varying styles of speech should be available for use in order to make the computer-controlled agents more believable and provide a greater sense of presence in the user.

A second point of contention that we have with current simulations and games is the way the attitudes of NPCs towards the player are modeled. Many current simulations and computer based roleplaying games such as Neverwinter Nights and World of Warcraft use a faction system to model the attitudes of the NPCs towards the player. Under this system each person is assigned an allegiance to a faction that exists in the game world and every person that belongs to that faction maintains the exact same attitude as every other person in the faction. Some games, such as Eve Online have extended the idea of factions and assign multiple factions to each NPC and use a simple equation to calculate the agents overall attitude towards the player. While this approach is more robust, it still uses a single set function to determine attitude for the NPC so any agent that has matching demographics (race, corporation, etc) will have the exact same attitude across the board and again this fails to take personal differences into account. We find this global uniformity in attitude to go against common sense.

B. PROBLEM STATEMENT

Systems currently have to make the tradeoff between the number of unique scenarios that can be built and the time it takes to produce them and therefore the money spent for development. This same problem occurs when trying to update a system to

include new threats or mission objectives after it is already in use. If these additions are performed automatically, assuming no outside human intervention, the simulation or game must inform the player of any new requirements. This requires that the system can create comprehensible dialog at run time. Simulations and games for the most part also lack the ability to robustly model and react to changing attitudes on the part of the agents in the system and consequently are unable to react to user actions in a sufficiently complex manner. We are attempting to address these problems.

C. TECHNICAL APPROACH

We created an architecture that uses sentence templates and libraries of sentence fragments differentiated by NPC type to create dialog at run-time that is appropriate for whatever interaction is currently taking place between the player and the NPC. By defining a sentence template for each of the types of interactions required to take place between the player and the NPCs, we allow the system to mix and match sentence fragments to create and communicate new information and missions on the fly while keeping the system small enough to work within an established video game system.

We also developed an alternate method for determining attitude and modifying it at run time with minimal computational overhead resulting in very individualized responses to the various events that take place within the game world. To do this we use a set of social networks that represent the relationships between the various NPCs in the game world. We first show that this is possible using a small example with only five people in our game world and then expand it to deal with a village of twenty-eight people to show that the system still works on a larger scale. Development time issues come up as the number of agents increases but this will be addressed when we discuss the system itself.

The program Neverwinter Nights (NWN) produced by Bioware will be used for proof of concept.

D. CONTRIBUTIONS

We did not attempt to solve the whole problem of believable narrative creation as that is well beyond the scope of this research but we do present a usable architecture that can serve as a middle ground between that and the conventional approach of hard-coding

all dialog. Our approach to modeling social interactions among NPCs is relatively new and though we are aware of several projects moving in this direction particularly dealing with attempts to more accurately model counterinsurgency operations, we have been unable to find any work that has been published in this area. We believe that both of these could greatly improve the believability and replay value of simulations and games such as those currently available on the market.

THIS PAGE INTENTIONALLY LEFT BLANK

II. LITERATURE REVIEW

A. INTERNAL REPRESENTATION OF AGENT KNOWLEDGE

One of the first problems is finding a way to simply represent the world state to the agents and allow them to be cognizant of the threats that they would be both aware of and would be interested in. It is impractical to attempt to represent all objects in the world individually as this quickly becomes untenable as the system scales to deal with larger and larger groups of entities.

An approach for representing an agents internal knowledge of the state of the world was proposed using symbolic preconditions in order to facilitate planning (Orkin). In this case planning means that the computer attempts represents the world as a set of discrete states and actions that will change the current world state. Each action has a set of preconditions that must be met prior to it taking place and a set of results that happen as a consequence of that action. The agent is then able, given an initial world state, a set of possible actions it can perform, and a desired eventual world state to internally plan a string of actions that will cause the world to change from its current state to a state that meets with the agents goals. The method Orkin used was to allow integers to represent abstract states using a minimal number of symbols. Instead of representing all of the objects in the world and their status they used a single pair of variables to describe the current threats to the agent thus reducing the number of variables required to represent the states and make planning less computationally expensive. This approach could be easily modified for use in the NWN framework which only allows integers, floats, and strings in its agent specific and global variables and therefore is useful when trying to represent the world states for our agents.

B. AUTOMATED MISSION GENERATION

In the process of reviewing the available literature concerning automated mission generation for simulations we quickly realized that very little had been published on this particular area. Most games use largely static missions to drive the storyline

One game that currently uses a system that demonstrates some of the behavior that this paper proposes is the MMOG, EVE-Online. The mission system as it has been

implemented takes a set of mission types, ranging from the infamous “FEDEX” quest that consists of go get X and bring it to Y location, to missions involving gathering materials, and search and destroy missions targeting possible threats from groups that are considered hostile to the mission giver. The missions each have several possible templates where the location, item, enemy type and force size, etc are variables that are entered into pre-generated blocks of text to provide a basic narrative thread. They also include “story-line” quests that are designed to give the player a look at the back story to the game but unlike most games they are completely voluntary. While this allows for much greater replayability, after a while the player begins to see the same mission texts again and again and it becomes repetitive. This is an improvement on the traditional systems but not a solution.

Historically, story graphs could be used where all possible branches the story might take are mapped out and the player must move among these predefined states as they progress through the narrative. This is somewhat analogous to the ‘choose your own adventure’ books that were popular on the ‘80s and while it provided a decent level of interactivity there were relatively few different storylines possible. There is also the problem that in many situations that occur in a story there are multiple ways of achieving a given goal which causes even more possible choices, expanding the graph even more so. To demonstrate the problem, assume that every decision point in a story graph is simply binary, with thirty choices in the storyline (a deceptively small number) the total number of nodes necessary to map out the story space would be 2^{30} or 1,073,741,824 distinct story states that would need to be coded. This would obviously become quickly untenable. (Shaw)(Szilas & Rety)

Another approach that is linked to AI planning techniques is the idea of ‘story management’ where the graph nodes are created and the system takes the past events into account before determining the next possible actions for the player. While this allows more freedom than traditional stories it still requires the system to have a story goal, or a conceptual idea of what the player should be doing rather than allowing the character to freely move through the environment (Mateas)(Mateas & Stern 2003)(Mateas & Stern

2005). This style of approach still requires an overarching storyline and is therefore too constrained for the purposes of this research.

Osborn approaches the problem by setting no boundaries on the interactions and giving his actors instinctual responses to stimuli. He uses a multi-agent system to interact and change the world as the story progresses. While this would produce the same results as the system we are proposing, the approach seems like overkill, tantamount to using a thermonuclear device to eliminate an anthill (Osborn).

One final approach is to represent the storyline of a game as a probabilistic model (Khan & Ward). Khan and Ward propose that a Bayesian model can be used, creating a probabilistic graph where the nodes are the various worlds states and the arcs are possible player actions that are weighted with the probability of a player choosing to take that particular action as opposed to the other possibilities. These probabilities would then allow the game itself to react and present options that it reasons are more in line with the players goals. This is an interesting approach but we don't see it as necessary to providing the basic replayability we are interested in. Especially in a military simulation when as often as not, the mission givers being modeled are the officers in command of the player and really would not particularly care if the mission was to the liking of the player as long as it got completed successfully. This approach also suffers from the requirement of an overarching storyline and is therefore not viable.

C. AUTOMATED DIALOG GENERATION

Quite a bit of research has been conducted on automated dialog generation. The most common way I've seen that this is approached is by providing templates, often called grammars, with blanks the computer can fill in from its knowledge base not unlike the idea of a MadLib. (Beaubouef)(Lonneker)(Shaw). This becomes even more complex by replacing these blanks in turn with templates creating a theoretically infinite set of choices. Figure 1 shows an example of this type of multiple level grammar.

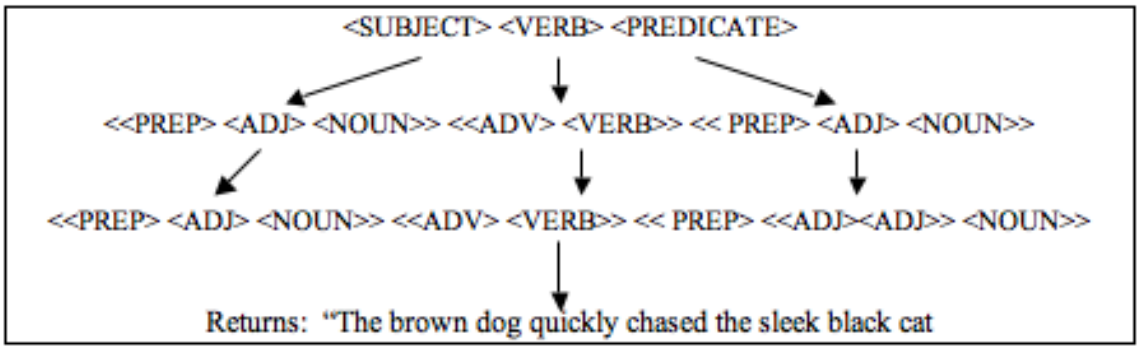

Figure 1. Grammar Template Example

This research proposes to use a similar approach matching the dialog first to the character type and then to the mission type and the threat at hand. Another suggested approach is to take previously created stories and adapt them to a given set of initial constraints using a case-based-reasoning approach (Peinado & Gervas).

A possible way to improve on this system is examined in Reiter and Dale's look and natural language generation (NLG) systems (Reiter & Dale). They propose a "corpus-based approach" where the designers of the system compile a set of all of the required output documents which they call an "initial corpus" along with any appropriate inputs and then analyze the resulting documents to try and identify any information present that is not readily available to the NLG. This should leave the designers with several types of information; text that does not change between iterations; text that can be directly taken from the knowledge base; text that can be derived from information in the knowledge base; and information that is not available in the knowledge base. The first three can be easily identified in the proposed system and since it is totally automated the last case will have to be carefully omitted to avoid invalid or incomplete text strings when producing the NPCs dialog. The authors also explore the necessary components in a NLG including modules for ordering the words, phrases, and sentences and ensuring that the proper rules of grammar are imposed. While some of this is necessary, in some cases, such as subcultures or uneducated characters conversing with the PC, certain rules of grammar and the available vocabulary will need to be modified to fit the character that is speaking. This could also be the case in a military simulation if the mission dialog is being passed to the user by various agencies that use different phrases or acronyms or

even in the case of combined missions where a unit is receiving information from a foreign national. In the proposed system the NLG in our scripts will have to be able to differentiate between various mission givers, whom may not all communicate in the same manner depending on the simulation. This differentiation will require more effort up front but the results should be more than worth it in character believability.

D. SOCIAL NETWORKS

Our first exposure to the idea of social networks was in fact anecdotal in the idea of "Six Degrees of Kevin Bacon". This is a game where an actor is named and the players must connect that actor to Kevin Bacon through people he has starred in movies with. This is based on an experiment conducted by Stanley Milgram in 1967 from which he concluded that any person in the world is connected to any other person in the world by way of a string of personal acquaintances. The idea the person A knows person B who knows person C and so on until you reaching person X. In smaller populations, such as actors for example, approximately six jumps are needed to go between any two people in a population (Wasserman & Faust). This led us to the idea of using social networks to model the interactions between agents. Despite our best efforts we have found no research indicating that anyone else has attempted to use social networks to model social interactions inside either simulations or video games.

E. CHOICE OF IMPLEMENTATION

Our choice to use NWN to implement our system was not arbitrary. A fair amount of academic research has been conducted using the program prior to our choosing to use it. This is mainly due to the well supported scripting capabilities that are provided with the game and the large coding community that has grown up over the years allowing for a good deal of help when learning the scripting language and also when it comes to specific problems when producing code. Several large collections of code exist free for use by the public and numerous forums exist where programmers can submit questions to be answered by both their peers and the developers of the game itself (Bioware 2004)(Bioware 2007). Other than game research, NWN has been used by the Air Force and Army to conduct studies on both the feasibility of using COTS video games to build simulations and studying the in personal interactions across various cultures (Warren et al 2004)(Warren et al 2005).

THIS PAGE INTENTIONALLY LEFT BLANK

III. GENERAL SOLUTION

A. INTRODUCTION

In this chapter we will present the theoretical architecture for both our mission and dialog generation system and our work towards using social networks to model complex social interactions in the simulated environment. We discuss the general method in which we propose to automate communication between the agents and the users. We also show the mathematical basis behind our approach to the use of social networks. The actual implementation of these ideas are not presented until Chapter V.

B. MISSION AND DIALOG GENERATION

The general architecture proposed to automate the generation of both missions and dialog is made up of two integrated parts. The first is the automated registering of objects so they are capable of being aware of each other without extensive work on the part of the developer. The second is the mission and dialog generator that takes inputs from the entities and the world to produce tasking and information for the user. A basic line diagram of the architecture is provided in Figure 2 below.

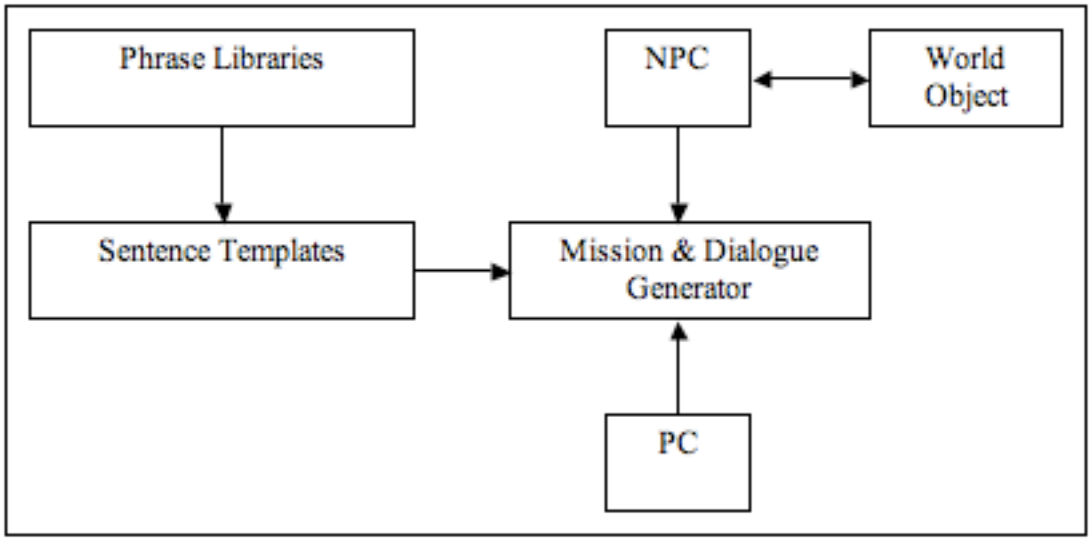

Figure 2. Basic Architecture Line Diagram

The registering of the various objects, be they actors or items of interest, is done programmatically. This however requires that a strict naming convention be imposed on

all objects in the system. If the simulation is done in an object oriented programming language then the use of inheritance can be leveraged to allow a single function to register any object type. During registration the object provides the world object with its type, unique tag, spawn location, and its notoriety, a measure of how obvious it is to the rest of the world. When an object is removed from the world it runs a script that removes its data from the worlds database. This clean up is necessary in order to allow the system to create new objects in the world after others are removed.

The agents that interact with the user are each given a scenario specific name, type, list of areas of interest, list of threats of interest, and knowledge level when they are created. In the mission and dialog generator the phrases that are used to fill in the sentence templates are drawn from a phrase library that is divided up by agent type.

Both missions and simple informative dialog can be generalized into the idea of sentence templates since in either case the system is required to communicate with the user. For example, “There is a soldier in the woods” and “Go kill the soldier in the woods” both use the same pieces of information from the world state and can be generated by our architecture in the same manner but the first is simply informing the player and the second is issuing a command, assigning a mission if you will. The mission and dialog generator works by taking as inputs the agents type and knowledge level, accessing the appropriate sentence template and filling in each part of the template with an appropriate phrase from the phrase library. If multiple equivalent phrases are programmed into the library then the draw from the library can be randomized so even if asking the same agent the same question, the user will get the same information but the response will be worded in a slightly different fashion each time. This provides more variety in the conversation in keeping with the goal of making the dialog more believable.

In picking the type of sentence that is appropriate the agent type is taken into account and from the set of all possible sentences, only types that the agent has access to are considered. A heuristic of some sort can be used to decide which sentences are used or it could be as simple as a random draw.

The use of templates and agent type specific phrase libraries makes the entire system modular allowing a developer to quickly insert new agents and even agent types into the system. Rather than having to reconstruct entire conversations for each agent in the system a developer can simply assign a type and knowledge level to the agent and the conversation is created programmatically at run-time. It is true that a similar time savings could be obtained in many systems simply by cutting and pasting a conversation from one agent file to another but this would provide identical static conversations. The proposed system creates dynamic conversations that would differ from agent to agent and between conversation instances. This modular approach makes it easy to change the language used by the agents as well. The same agent information could be used but by switching the sentence templates and phrase libraries it is possible to switch from English speaking agents to Arabic speaking agents fairly simply. If the grammar is similar enough then even the sentence templates could conceivably remain the same, consequently it becomes simply a matter of changing the phrase libraries to switch the dialect the agents are using. A possible example of this would be switching the simulation from speaking Mandarin to Cantonese for instance. Assuming that both dialects of Chinese have the same basic grammar then the only differences should be the vocabulary.

C. SOCIAL NETWORKS

We also propose an alternate method for determining an agent's attitude towards the user and modifying it at run time with minimal computational overhead resulting in very individualized responses to the various events that take place within the game world. To do this we develop a set of social networks that each represent the relationships between the various NPCs in the game world.

Initially we considered representing each of the social networks as a bidirectional graph where the nodes each represent an agent in the world and each arc shows the effect that agent has on the agents it comes in contact with and visa versa. An event would have an effect on a particular agent and then the effect would diffuse outward from the initial agent. Unfortunately this approach makes second and higher order effects and

loops in the graph difficult to deal with. While we found that these difficulties could be overcome, the solutions would be computationally expensive.

Instead we came up with an alternate solution. Each of the social networks can be represented as a fully connected bidirectional graph where arcs that are not in the non-fully connected graph simply have a weight of zero. This can then be represented as an n by n matrix designated as A_k; each element (a_{xy}) represents the influence agent x has upon agent y and where n is the number of entities in the game world and k is the number of networks described by the game world developer. In this case influence is defined as the effect a change in agent x's attitude towards the user has upon the attitude of agent y towards the user. Each entry in the matrix has been set as a value between -1 and 1 inclusive. This bounding ensures that the value of each subsequent iteration through the network will have less effect than the iteration before it. Examples of these networks would be family connections, networks of friends, coworkers, etc. These matrices are then combined into a single matrix using a weighting function and then normalized. Matrix B is the amount of influence the personal effects of an event has on the agent in question and as such only has entries on the diagonal. This is done such that the sum of the absolute value of each entry in any particular row of the sum of matrices A and B is equal to one as shown in Figure 3.

$$A_k = \begin{pmatrix} 0 & a_{12} & \cdots & a_{1n} \\ a_{21} & & \ddots & \vdots \\ \vdots & & & a_{(n-1)n} \\ a_{n1} & \cdots & a_{n(n-1)} & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} b_{11} & 0 & \cdots & 0 \\ 0 & & \ddots & \vdots \\ \vdots & & & 0 \\ 0 & \cdots & 0 & b_{nn} \end{pmatrix}$$

$$\sum_{i=0}^{n} |a_{ix}| + \sum_{i=0}^{n} |b_{ix}| = 1$$

Figure 3. Influence Matrices

We chose to use a simple linear weighting function where the influence of each network on each person was determined individually but a more complicated function for modeling the contribution of each network could easily be substituted. This is shown in Figure 4. The normalization of each network is accomplished by dividing each entry by the total of the summation of the absolute value of each entry in that row.

$$A = w_1 A_1 + w_2 A_2 + \ldots + w_k A_k$$
$$w_{n1} + w_{n2} + \ldots + w_{nk} + w_{nB} = 1$$
$$B = w_B I$$

Figure 4. Weighting Equations

In the initial equation the LHS x vector is the new change in attitude and the RHS x vector is the change at an earlier iteration through the equation. The u vector represents the initial effects that a given event would have on each of the entities in the system. In both the x and the u vectors, each row represents the effect on an entity and corresponds to the same entity-row pairs in the A and B matrices. Given this equation we could then iterate through, constantly updating the x vector as it changed over time but this leads to the problem of when to stop iterating. Rather than doing multiple iterations and not necessarily knowing when another iteration would no longer be significant, we solved the equation for x as shown in Figure 5.

$$x = Ax + Bu$$
$$(I - A)x = Bu$$
$$x = (I - A)^{-1} Bu$$

Figure 5. Solving for x Vector

By developing a layered set of social networks offline for the various agents in the game world prior to run time we can calculate the change in attitude of any particular agent (x) in the game world for a given event (u) through what turns out to be a simple vector-matrix multiplication. If the events are predetermined then all of the attitude changes can be precalculated and then simply added to the agent's attitude value at run

time. This information could then be fed into the dialog generator described above and the tone or even the content of the communications from the agent could be modified at run time dependent on the agent's attitude towards the user.

The main difficulty with this approach is that it requires a large initial investment of time and effort during development. Rather than simply assigning a faction and walking away, our approach requires the developer to take the time to develop at least a basic background story for each and every agent in the simulation. Even in our relatively small example scenario of twenty-eight agents it took several hours to create networks to represent friends, family, and co-workers. In a larger simulation this time would increase dramatically unless tools are first developed to aid the developer. The basic requirements for such tools are described in the future work section of this paper.

IV. SPECIFIC APPLICATIONS

A. INTRODUCTION

We identified two specific military applications for our architecture in addition to the entertainment based proof of concept that we implemented, though we expect there to be many others that we did not address. We will describe each application with respect to its user, interactive agent types, threat and location types, and the applicability of using our social networks system to augment the basic architecture.

B. NAVAL COMBAT SIMULATION

The naval combat environment was our original impetus for this research and as such the architecture is well suited for this particular application. The user in this case would be a single ship operating either on its own or as part of a group of ships. This allows for all forms of communications between actors, including everything from flashing light to radio communication systems such as Bridge to Bridge, RF, UHF, SHF, etc. and computer networks using email or chat.

Interactions would take place between the user's ship and other aircraft and vessels in its vicinity as well as its subordinate and superior commands and component warfare commanders. Agent types would include own force, allied, and hostile aircraft and vessels, commercial aircraft and shipping, and orders passed from the various warfare commanders such as the Air Warfare Commander (AWC), Surface Combatant Comander (SCC), etc. Each agent type would then have its own set of applicable sentence templates and phrase library so the mission and dialog generator

Threats would be classified as any other vessel or unit in the battlespace and location types would be defined according to geographic, political, or even organizational boundaries. For vessels a threat may be any contact they can detect using their sensor systems. The dialog that could be generated would be anything from simple advisories telling other vessels to maintain situational awareness to calls for aid if the vessel is under attack or has suffered some sort of casualty. For warfare commanders the locations of interest would be a type of battle space, for example, anything with an elevation greater than ground or sea level would fall under the purview of the AWC. This would also feed

into which sentence template to use as the AWC may simply inform the user when an aircraft is detected leaving an airbase in a hostile country, it might also task the unit to fire upon that unit in the case that it is approaching a friendly vessel in international waters.

At this level of simulation the social networks system we have described is not particularly useful in a simulation that takes place over a short time period. In a system that included a campaign style of play such that the actions a player takes in one scenario have lasting effects in subsequent scenarios, then a network could be described to show the attitudes of both allied and hostile forces and commanders towards the users unit. This would cause allied commanders to task a unit with more important or more sensitive missions depending upon the unit's track record or even cause enemy forces to specifically target a unit if it had caused them damage in the past.

C. URBAN ENVIRONMENT SIMULATION

A very similar approach can be taken when dealing with a simulator that represents an urban environment. In this case the user would take the part of a single soldier or possibly a small group, and the forms of communication modeled would range from electronic or radio communications with their home base to interpersonal communications with the various agents within the city as they move through the simulated environment.

Interactions would take place between the user and the chain of command, friendly or hostile forces, or civilian entities present in the game world. Because of the wide range of possible entity types, this scenario provides the largest variety of sentence templates considering that in many scenarios that are relevant to current operations. Additionally, there is no guarantee that all of the agents in the system would speak in the same language. Interactions would range from receiving new objectives mid-mission, to interrogating or otherwise eliciting information from an agent, to directing allied forces in a combined operation. Due to the sheer number of possibilities for interaction, the richness of the scenario would be limited only by the number of possibilities the developers wish to explore and the availability of subject matter experts in aiding them in such a scenarios creation.

Threats would be people and or items of interest in the battlespace to include insurgents, IEDs, weapons caches, etc. Location types could be anything from specific types of buildings such as government offices, police stations, etc., to geographic regions like neighborhoods or a unit area of responsibility, to convoy routes, to complexes such as military bases and airports. The dialog generated could be the results of an interrogation that reveals the location of more insurgents or orders from higher command tasking the user to lead a team to capture or kill the insurgents. Due to the fact that these various threats are created at run time, this provides a constantly changing environment in which to train in.

A social network that takes into account both allied, neutral, and hostile forces in the environment would go a long way towards modeling the fluid environment that soldiers find themselves in when dealing with a counterinsurgency mission in an urban environment. In such multi-faceted environment, how the user interacts with various groups will effect how willing they are to deal with the user later. It could determine if they willingly provide intelligence or try to kill the user on sight. This can also change over time as the line between insurgent and civilian continues to shift through both conciliatory and inflammatory actions on the part of the user. Networks describing the agents' families, work relationships, insurgent groups, and relationships to the various military forces in the simulation would all be useful in determining their attitude towards the user at any specific time and this would greatly enhance the simulation over the static forces that are more commonly modeled.

THIS PAGE INTENTIONALLY LEFT BLANK

V. PROOF OF CONCEPT

A. MISSION AND DIALOG GENERATOR IMPLEMENTATION

We chose to implement the mission and dialog generator in the context of the commercial video game Neverwinter Nights (NWN). Initially we created a small proof of concept example that consisted of three interactive agents each of a different type, two items of interest and three threat entities.

Figure 6. Prototype Scenario

The agents were hard coded into the system but the items and threats were randomly created upon scenario initialization and then replaced after a six second wait time (one system 'heartbeat') as the user removed them from the world. Each agent type had varying location and threat types that they were interested in and a set awareness of their surroundings ranging from close to oblivious to perfect knowledge. This demonstrated that the agents could create dialog at run time that accurately reflected the

items and threats in the world and that the system could refresh itself over time allowing for a continuous flow of random threats and items to constantly challenge the user.

After the prototype system was proven to function as expected, we expanded the system to a scenario that includes a village and the surrounding countryside. We increased the number of interactive agents to twenty-eight and included three item locations and nine threat entities that again were randomly created at runtime. This larger scenario was created to show that the system was scalable. It also was developed to show the time advantage that came from multiple agents of the same type. In the initial scenario it was not obvious because each of the agents were of a different type so there was no reuse of the phrase libraries though they all used the same sentence templates. In the larger scenario our seven agent types were, Peasant, Boy, Girl, Priest, Noble, Servant, and Soldier. There are at least two of every type and as many as ten of the peasant type. The system handled this larger scenario without any noticeable degradation in performance.

Figure 7. Village Scenario

1. Z-Dialog

In order to implement the system we took advantage of a third party module Z-Dialog which was published on the NWN Code Vault web site as freeware. This code bypasses the normal conversation creation system where each line of dialog and user response is hard-coded into the system during scenario development. While this system is relatively flexible allowing for various flags to control how conversations change over the course of a scenario they still were limited to the options programmed in during development. The Z-Dialog module allows the developer to call a function that programmatically produces a conversation at run-time. The Z-Dialog system requires two variables to be set in order to function. First, a single string variable called 'dialog' with the dialog function set as its value to be included in the variable space of the interactive agent. Second, that the conversation tab of the agents properties be set to 'zdlg_converse'. The functionality provided by this module was essential to our implementation of the architecture in NWN and we would have had to either create a similar infrastructure to support our project or find an alternate simulation/game to work with had it not been available.

2. Knowledge Representation

Each agent the user can interact with has two variables stored in its variable space during development. The first variable was the agent type. This allows the program to choose which sentence templates to use, which phrase library to draw from, and the threat and location types that the agent is concerned with. For our system we chose to represent this with a simple integer ranging from zero to six as we had seven agent types in use. The second variable was the knowledge level, which represents how aware of the world space the agent is. This number was used when determining if the agent was aware of any particular threat and was stored as an integer. The range of this variable was arbitrarily chosen to be between zero and one hundred where zero is completely oblivious and one hundred was perfect knowledge of the world space. Upon scenario initialization, each agent was queried for its agent type and the appropriate list of sentence templates, location concerns, and threat concerns were stored in the agent's variable space.

3. Threat and Item Generation

The NWN game engine treats both monsters, which we chose as the threats for our scenario, and items as objects that can be spawned at run time. The function however requires a specific location that must be set up during scenario creation. This necessitated that we designate stationary waypoints that could be used by the function when the various objects are created. The waypoints used to spawn threats were designated as 'spawn points' and named 'SPAWNPOINTXXX' where XXX is a three-digit number that ranges from zero to the number of threats required minus one. Similarly, the waypoints used to spawn items were designated as 'item points' and named 'ITEMPOINTXXX' where XXX is a three-digit number that ranges from zero to the number of items required minus one. Because of the pseudo object-oriented approach of the NWN game engine this segregation of the two location types is artificial and was simply done to ensure a certain number of threats and items were available for testing at all times. Upon scenario initialization, each spawn and item point was populated with the appropriate object type. The threats and items were chosen randomly from a list that was hard-coded into the system by the developer during scenario creation.

It is necessary to note that it would have been possible to write the script to look for any specific naming convention and create any set of objects, even objects of differing types, at the waypoints designated. For example, we could have emulated the numbering system above and named the waypoints 'LOCATIONXXX' and randomly chosen an item or threat to be placed at that location as drawn from a single combined list. Because of the engine we are still limited to a finite predetermined set of locations however. This restriction is peculiar to the engine however and is not mandated by our architecture. In fact the original system we envisioned would randomize the location, timing, and type of any spawned threat in order to avoid the predictability issues that we identified as problematic initially.

The waypoints themselves have several types of information attached to them at design time to allow for use later. First, it has a Boolean that designates whether or not it is currently in use by the system. Second there is a timer value that tells how many

system heartbeats need to pass before creating an object at this location. This allows for a simple check when seeing if a waypoint needs to have an object spawned at its location. This check is done by a for loop that checks each location's value and calls the creation function if the value is false and the timer value is equal to zero. Initially these are set to False and zero respectively allowing the same code that checks the locations to be used to populate the world with objects during scenario initialization. Third it has a location type which is represented as a global integer constant assigned to it that corresponds to the location concerns encoded on the various agents according to their agent type. Finally, each waypoint has multiple name values. We use two in our proof of concept but it could easily be more if the scenario developers desired more variation in how the locations are described in the agent's dialog. These name values are normally a proper name and a generic name though this varies in some cases. For example, a waypoint in the middle of a forest could be described as 'the Black Forest' and 'the forest west of town'. The point of this was not so much to standardize the naming of areas in the game but to avoid having to specify two or more names for every location for every agent type which quickly becomes a large task as more agent types and waypoints are added. The number would be *kn* as opposed to *knm* where *k* is the number of name variations, *n* is the number of agent types, and *m* is the number of waypoints in the scenario. This would be eighteen (two variations multiplied by nine waypoints) as opposed to one hundred twenty-six (two variations multiplied by nine waypoints multiplied by seven agent types), and that's just in our limited scenario.

When the creation function is called, be it for a threat or item, it performs the same actions but draws from different lists of unique 'tags' or keywords reserved by the engine to refer to specific instances of threat or item objects. When created, the object writes several things to the variable space of the waypoint at which it is created. The variables are the object's tag, its plain name, its type, and its notoriety level. An example of a tag is 'NW_ORC001' obviously this is not something which would occurin normal conversation with the agents; therefore, the plain name is designated ('Orc' in this case). The type is used when determining if agents are concerned with that type of object. Various object types were enumerated in the setup and are treated as integer constants by

the scripts and are referred to exclusively by their variable name in the scripts. The orc is listed as a MONSTER so any agent whose list of threat types includes MONSTER will be concerned with the orc's existence in the world. The notoriety level is a simple method for modeling how likely anyone is to know of the objects existence. In the case of a threat object it is a measure of how obvious this object is in the environment, how much a threat it is to those around it, and how important it is in the grand scheme of things all rolled into one. This value has an arbitrarily assigned value that ranges between zero and one hundred. In the case of the aforementioned orc, the notoriety of such a creature would be moderate to low, possibly in the low 40s, since such creatures while belligerent are relatively easy to dispose of and somewhat common actors in fantasy games like NWN.

When an object is removed from the game world, by being picked up if it is an item or killed if it is a threat, then a removal script is run. This removal script replaces all of the values assigned to the waypoint upon object creation with zeroes, in the case of integers, or null strings in the case of strings. It assigns a value of False to the waypoints 'in use' Boolean and randomly assigns an integer value to the timer value, between one and four for items and between one and six for threats. A script is run at every heartbeat that first decrements the timer value by one and then checks if a waypoint is ready to have an object created at it. Since the system heartbeat occurs once every six seconds, this gives the player time to move away from the waypoint before it refreshes. Any value could be assigned to the timer value upon object removal. We chose random values from ranges selected to allow some variation while keeping the time period short. Our goal was to be able to observe the both that an object of random type was being created and that the agents were able to register this fact and respond accordingly.

4. Mission and Dialog Generator

When the user initiates a conversation with an agent in the game world, the z-dialog module causes our dialog function to be called. The first screen that is called up shows a greeting that eventually was based on the agent's attitude towards the user and offered several possible selections for the user, such as 'Any threats?', 'Any rumors?',

and ‘Goodbye’. The attitude measure and its effect was implemented later to show our work with the social networks and is further covered below, in social networks section of this chapter.

The ‘Any threats?’ option calls a function that searches the world object for any threats in existence and then subjects it to a series of tests to see if the agent is not only aware of the object but actually cares about its existence.

Figure 8. Initial Welcome Screen

To demonstrate both spatial and type concerns, we compare the lists of threat and location concerns of the agent to the values of the corresponding variables as stored on each of the waypoints. Finally if the agent’s concerns align with the threat type and location then the creature’s notoriety value is compared to the agent’s knowledge level. Since both have values from zero to one hundred, this comparison is implemented as a

value comparison where the check is successful if the agent's knowledge level is higher than the object's notoriety level. This is admittedly a naïve model, but it is enough for our purpose.

In Figures nine and ten, two different conversations with agents of the same type are shown. In both conversations line two says "2. Fails on Location." This is included to show that while there is a threat at that location it is not one that the soldier agent type is concerned with. The knowledge level of the agent in Figure 9 is higher than that of the agent in Figure 10. This can be seen in lines eight and nine where the agent in Figure 10 fails its knowledge check.

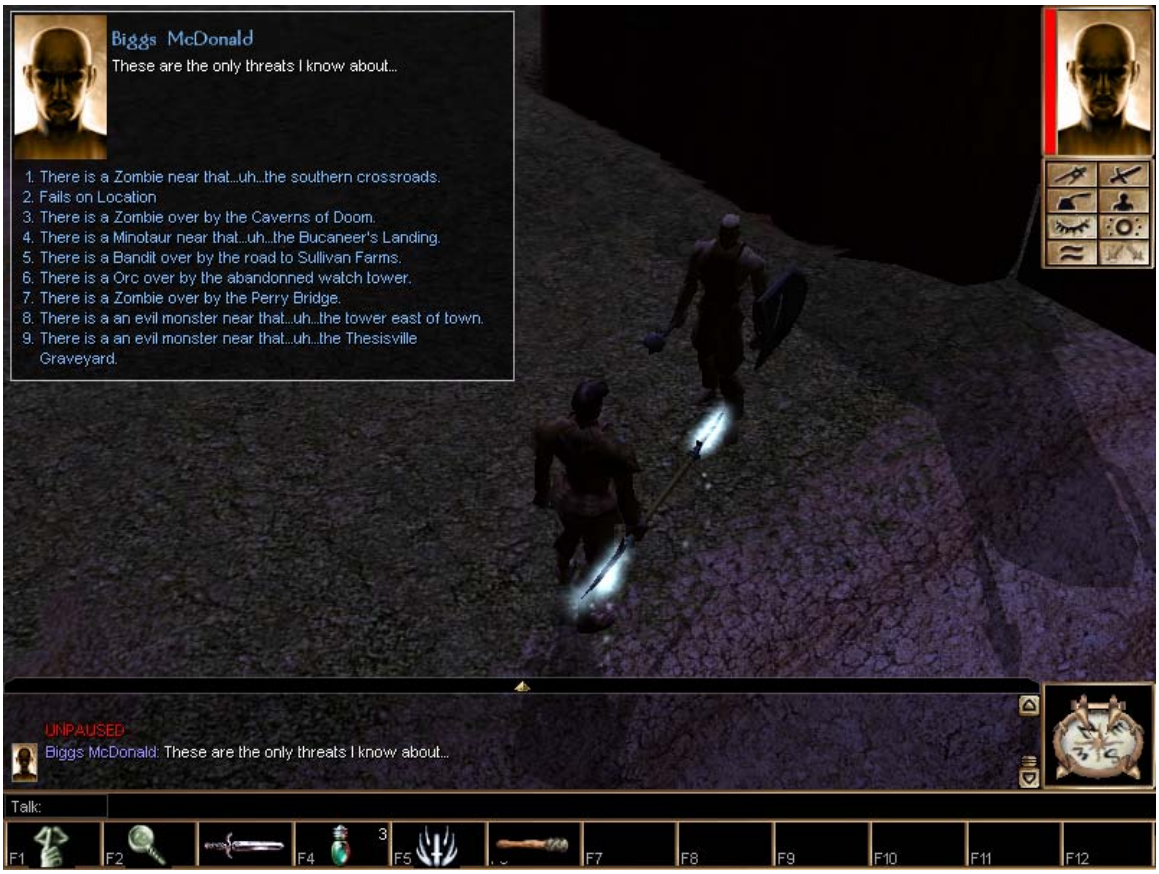

Figure 9. Soldier Conversation Part I

The sentence templates used here are simple informative dialog. The template is of the form <INTRO><THREAT><LOCATION> where:

<INTRO> = "There is "

<THREAT> = <ACTUAL NAME> or “a dangerous creature” or “something evil”

<LOCATION>= <LOCATION1> or <LOCATION2>

The <ACTUAL NAME> variable is drawn from the threat object. If the knowledge level of the agent is not high enough then one of the ‘vague’ phrases stored on the agent is used. The <LOCATION1> and <LOCATION2> variables are drawn from the threat’s spawn point. This template results in one of the following sentences:

“There is a Zombie near that…uh…the southern crossroads.”

“There is a Zombie near the crossroads.”

“There is a dangerous creature near that…uh…the southern crossroads.”

“There is a dangerous creature near the crossroads.”

“There is something evil near that…uh…the southern crossroads.”

“There is something evil near the crossroads.”

With a set agent, location, and creature type there are still six possible ways for the agent to respond to a query for information.

Figure 10. Soldier Conversation Part II

Once we have determined that the agent is both concerned about and aware of the threat we then determine which sentence template to use. This is done by checking which sentence templates are available to the agent type and then, if more than one is available, a random template is chosen. Many of these sentence templates are analogous to the various mission types that appear in many video games. An abbreviated list and short description is provided in Table 1 below.

Name	Description
Defend	In these missions the player is required to protect a specified entity, item, or location from some sort of threat. This may occur at a fixed location or possibly while in transit between two locations.
Recon	In these missions the player is required to gather information on a specified entity or location.
Kill	In these missions the player is required to eliminate a specified entity or possibly group of entities.
“Talk To”	In these missions the player is required to communicate with a specified entity or group of entities.
Transport	In these missions the player is required to move one or more entities or

	items from one location in the game space to another.
Gather	In these missions the player is required to gather a certain number of items or entities.
Complex	Any of the above mission types may be combined in multiple-phase missions.

Table 1. Generic List of Role Playing Game Mission Types

This list is not exhaustive but it covers the vast majority of missions or quests that a player is likely to receive in the course of a game of this type. In our implementation we only created three sentence templates to choose from. The first simply informs the player of a threat saying something along the lines of "There is a orc at the crossroads." This is simply the agent providing information to the player and does not correspond to one of the enumerated mission types. The second goes as follows, "Kill the orc at the crossroads." This is a simple version of the Kill mission type. The third follows the following lines, "Go get the sword at the shrine." This is an example of the Gather mission type where the number of required items is one. As each sentence template is only available to certain agent types, the phrase libraries required for those templates need only be built for those agent types that are capable of accessing the templates in question.

The 'Any rumors' option performs essentially the same actions for item type objects that the threats option does for creature type objects. The two different triggering phrases were chosen to allow the two types to be differentiated during testing. As both the items and threats are internally represented as objects, a generic function could have been used to deal with both types at once.

5. Implementation Advantages

This implementation also allows for the rapid addition of new agents, agent types, threats, items, or even missions to the scenario. This implementation really pays off when adding further functionality to a system since in most cases the costs for adding a second new agent, item, or conversation is close to double that of the first, triple for adding a third, and so on. Under our methodology the cumulative cost for adding new objects becomes less when adding larger numbers. Specific examples are as follows.

To add a new agent to the scenario requires simply adding an agent avatar and setting the conversation function in the agent's attributes settings and assigning the knowledge, type, and dialog variables. To add the agent avatar is simple, using the NWN toolset: the designer selects an avatar type and then click on the location on the map where the avatar is to be placed. Using the standard method for adding agents in NWN the developer would still have to add the agent avatar but then he would have to hardcode any conversations, including both the agent and player dialog, for all possible situations. This would then remain static throughout the scenario. The developer could bypass this step by cutting and pasting a previously created conversation from another agent but this either results in an identical conversation or time must then be taken to alter the conversation appropriately. Either way, given large numbers of agents of a specific type, our implementation provides dynamically created conversations complete with real time agent knowledge at a fraction of the time required using conventional means.

To add a new agent type using the standard NWN method would be exactly the same as adding a new agent. Using our methodology, it involves creating a new phrase library with the required phrases for any available sentence templates and setting up the list of threats, locations, and sentence templates that the agent type is concerned with. This overhead is more work than is required in the standard method but it quickly becomes worth the effort as more and more agents of that type are created in the scenario.

The addition of new threats or items to the scenario using the standard methodology is done in much the same way as adding an agent. The developer selects a threat or item and places it in a static location. Any references to the new threat or item must then be manually added to the conversations of any agent the developer wishes the player to be able to gather information from. With our implementation this process becomes trivial. The developer adds the name, tag, notoriety, and type values to the creation functions. Creation and placement of threats and items and their inclusion in interactions with agents is now taken care of by the system itself with no further work done by the developer.

In the standard methodology, to add new mission types or other interactions the developer must manually update each agent's conversation. Our implementation

significantly reduces the time necessary to do this in a large system. In the conversation creation function the developer adds a new sentence template and then adds the ability to access that template to the appropriate agent types. This effectually modifies the conversations of every agent of the required type with a single change to the code.

B. SOCIAL NETWORKS IMPLEMENTATION

Initially we developed a basic three-person environment to test our model using a Python script to perform the required calculations. This implementation had two basic flaws: the matrix and event vectors were both hard-coded and memory-less and the three-person model was too small for all the most basic scenarios. We decided to try an implementation using a spreadsheet instead. The spreadsheet approach allowed us to solve all of the problems mentioned above with minimal programming effort.

Using the spreadsheet we developed a five-person environment with persons A, B, C, D, and E. Using the same group of eight events we examined their effects on seven test cases that were variations of the A and B matrices of this five-person environment. After this analysis, we then expanded our matrix to reflect the twenty-eight people that populated the village in our scenario and their relationships. Notice that the events are designed to show that the effects are symmetrical across the test matrices given positive and negative events in the same and diametrically opposed positions. In most cases we used our maximum and minimum values to show the largest possible values in the resulting attitude change vectors.

$$Event1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, Event2 = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, Event3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, Event4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix},$$

$$Event5 = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}, Event6 = \begin{bmatrix} -0.5 \\ -0.5 \\ -0.5 \\ -0.5 \\ -0.5 \end{bmatrix}, Event7 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, Event8 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix}$$

Figure 11. Test Case Events

Since the values in our event vectors are bounded between negative and positive one, positive one represents an extremely good effect such as the person winning millions in the lottery the same day as they marry their high school sweetheart, while a negative one would be having their dog run over by their wife as she leaves him for his best friend…or living through the events described in any country song. As such, event one shows an event that has a dramatic good effect on entity A. Event two is a dramatic negative effect on entity A. Events three and four are the same except the effect is on entity E. Events five and six show an average good or bad effect on the entire population, for example the entire office getting a Christmas bonus or the whole office coming down with a nasty strain of the flu. Event seven has dramatic positive effect on both entities A and E. Finally, event eight has a dramatic positive effect on entity A and a dramatic negative effect on event E.

1. Trivial Case

The first test case was a trivial one, which consisted of fully connected graph in which all of the entities had equal influences on each other and themselves. The matrices are shown below in Figure 12.

$$A = \begin{bmatrix} 0 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0 & 0.1 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.6 & 0 & 0 & 0 & 0 \\ 0 & 0.6 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0 \\ 0 & 0 & 0 & 0.6 & 0 \\ 0 & 0 & 0 & 0 & 0.6 \end{bmatrix}$$

Figure 12. A and B Matrices for Trivial Case

Using the equations we derived in Chapter III we come up with the matrix in Figure 13.

$$(I-A)^{-1}B=\begin{bmatrix}0.636 & 0.090 & 0.090 & 0.090 & 0.090\\ 0.090 & 0.636 & 0.090 & 0.090 & 0.090\\ 0.090 & 0.090 & 0.636 & 0.090 & 0.090\\ 0.090 & 0.090 & 0.090 & 0.636 & 0.090\\ 0.090 & 0.090 & 0.090 & 0.090 & 0.636\end{bmatrix}$$

Figure 13. Derived Influence Matrix for Trivial Case

When this matrix is multiplied by each event vector, the attitude change vectors shown in Figure 14 are produced.

$$Event1=\begin{bmatrix}0.636\\0.090\\0.090\\0.090\\0.090\end{bmatrix}, Event2=\begin{bmatrix}-0.636\\-0.090\\-0.090\\-0.090\\-0.090\end{bmatrix}, Event3=\begin{bmatrix}0.090\\0.090\\0.090\\0.090\\0.636\end{bmatrix}, Event4=\begin{bmatrix}-0.090\\-0.090\\-0.090\\-0.090\\-0.636\end{bmatrix},$$

$$Event5=\begin{bmatrix}0.5\\0.5\\0.5\\0.5\\0.5\end{bmatrix}, Event6=\begin{bmatrix}-0.5\\-0.5\\-0.5\\-0.5\\-0.5\end{bmatrix}, Event7=\begin{bmatrix}0.727\\0.181\\0.181\\0.181\\0.727\end{bmatrix}, Event8=\begin{bmatrix}0.545\\-1.388e-17\\-1.388e-17\\-2.776e-17\\-0.545\end{bmatrix}$$

Figure 14. Attitude Change Vectors for Trivial Case

This case was used to show that our equations passed a basic commonsense test. As expected, in events one, two, three, and four the person who was directly affected had the most change in their attitude. Since all relationships in the matrix are equal, events one and three and two and four show mirror image patterns. In the same way, events one and two and three and four are opposites since the events are opposites. Event seven shows that the initial effects reinforce each other when an event directly affects two of the entities in a positive manner. Event eight shows that when two entities are affected in an opposing fashion the effects on the non-effected entities cancel each other out. The

effects on entities B, C, and D are miniscule and we believe that the variations between D and the other two are simply the results of rounding errors in the process.

2. Degenerate Case

The second test case consisted of a degenerate matrix that consisted two separate groups in which all of the entities within a group had equal influences on each other and themselves, but zero effect on the other group. The matrices are shown below in Figure 15.

$$A = \begin{bmatrix} 0 & 0.4 & 0 & 0 & 0 \\ 0.4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.2 & 0.2 \\ 0 & 0 & 0.2 & 0 & 0.2 \\ 0 & 0 & 0.2 & 0.2 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.6 & 0 & 0 & 0 & 0 \\ 0 & 0.6 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0 \\ 0 & 0 & 0 & 0.6 & 0 \\ 0 & 0 & 0 & 0 & 0.6 \end{bmatrix}$$

Figure 15. A and B Matrices for Degenerate Case

Using the equations we derived in Chapter III we come up with the matrix in Figure 16.

$$(I - A)^{-1} B = \begin{bmatrix} 0.714 & 0.286 & 0 & 0 & 0 \\ 0.286 & 0.714 & 0 & 0 & 0 \\ 0 & 0 & 0.666 & 0.167 & 0.167 \\ 0 & 0 & 0.167 & 0.666 & 0.167 \\ 0 & 0 & 0.167 & 0.167 & 0.666 \end{bmatrix}$$

Figure 16. Derived Influence Matrix for Degenerate Case

When this matrix is multiplied by each event vector, the attitude change vectors shown in Figure 17 are produced.

$$Event1 = \begin{bmatrix} 0.714 \\ 0.285 \\ 0 \\ 0 \\ 0 \end{bmatrix}, Event2 = \begin{bmatrix} -0.714 \\ -0.285 \\ 0 \\ 0 \\ 0 \end{bmatrix}, Event3 = \begin{bmatrix} 0 \\ 0 \\ 0.166 \\ 0.166 \\ 0.666 \end{bmatrix}, Event4 = \begin{bmatrix} 0 \\ 0 \\ -0.166 \\ -0.166 \\ -0.666 \end{bmatrix},$$

$$Event5 = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}, Event6 = \begin{bmatrix} -0.5 \\ -0.5 \\ -0.5 \\ -0.5 \\ -0.5 \end{bmatrix}, Event7 = \begin{bmatrix} 0.714 \\ 0.285 \\ 0.166 \\ 0.166 \\ 0.666 \end{bmatrix}, Event8 = \begin{bmatrix} 0.714 \\ 0.285 \\ -0.166 \\ -0.166 \\ -0.666 \end{bmatrix}$$

Figure 17. Attitude Change Vectors for Degenerate Case

This case was used to show that when groups of entities had no connection to each other, the influence did not spread from one group to another. This is specifically shown by events one through four and seven and eight.

3. Friend of a Friend Case

The third test case consisted of a chain of connected entities such that entity A was influenced by entity B, entity B is influenced by entities A and C, entity C is influenced by entities B and D, entity D is influenced by entities C and E, and entity E is influenced by entity D. The matrices are shown below in Figure 18.

$$A = \begin{bmatrix} 0 & 0.4 & 0 & 0 & 0 \\ 0.2 & 0 & 0.2 & 0 & 0 \\ 0 & 0.2 & 0 & 0.2 & 0 \\ 0 & 0 & 0.2 & 0 & 0.2 \\ 0 & 0 & 0 & 0.4 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.6 & 0 & 0 & 0 & 0 \\ 0 & 0.6 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0 \\ 0 & 0 & 0 & 0.6 & 0 \\ 0 & 0 & 0 & 0 & 0.6 \end{bmatrix}$$

Figure 18. A and B Matrices for Friend of a Friend Case

Using the equations we derived in Chapter III we come up with the matrix in Figure 19.

$$(I-A)^{-1}B=\begin{bmatrix}0.654 & 0.273 & 0.057 & 0.012 & 0.002\\ 0.136 & 0.683 & 0.142 & 0.031 & 0.006\\ 0.028 & 0.142 & 0.657 & 0.142 & 0.028\\ 0.006 & 0.031 & 0.142 & 0.683 & 0.136\\ 0.002 & 0.012 & 0.057 & 0.273 & 0.654\end{bmatrix}$$

Figure 19. Derived Influence Matrix for Friend of a Friend Case

When this matrix is multiplied by each event vector, the attitude change vectors shown in Figure 20 are produced.

$$Event1=\begin{bmatrix}0.654\\ 0.136\\ 0.028\\ 0.006\\ 0.002\end{bmatrix}, Event2=\begin{bmatrix}-0.654\\ -0.136\\ -0.028\\ -0.006\\ -0.002\end{bmatrix}, Event3=\begin{bmatrix}0.002\\ 0.006\\ 0.028\\ 0.136\\ 0.654\end{bmatrix}, Event4=\begin{bmatrix}-0.002\\ -0.006\\ -0.028\\ -0.136\\ -0.654\end{bmatrix},$$

$$Event5=\begin{bmatrix}0.5\\ 0.5\\ 0.5\\ 0.5\\ 0.5\end{bmatrix}, Event6=\begin{bmatrix}-0.5\\ -0.5\\ -0.5\\ -0.5\\ -0.5\end{bmatrix}, Event7=\begin{bmatrix}0.657\\ 0.142\\ 0.057\\ 0.142\\ 0.657\end{bmatrix}, Event8=\begin{bmatrix}0.652\\ 0.130\\ -3.469e-18\\ -0.130\\ -0.652\end{bmatrix}$$

Figure 20. Attitude Change Vectors for Friend of a Friend Case

This case shows the effects of an event as it gets progressively further from the entity that is directly affected. As expected, with only a single chain of influences connecting entities A and E, by way of all of the others, an event that directly affects entity A causes a progressively smaller change in attitude as you move further from the directly affected entity. The symmetry of the matrix causes similar results to those shown in the trivial case in events seven and eight.

4. Hermit Case

The fourth test case consisted of four entities making up a fully connected graph and a single almost isolated entity. The isolated entity, entity A, is influenced by one of

the entities in the larger matrix, namely B, but no one, including B is influenced by A. This would simulate an unknown observer who is concerned about how events affect another entity. The matrices are shown below in Figure 21.

$$A = \begin{bmatrix} 0 & 0.4 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0.1 & 0.1 \\ 0 & 0.1 & 0 & 0.1 & 0.1 \\ 0 & 0.1 & 0.1 & 0 & 0.1 \\ 0 & 0.1 & 0.1 & 0.1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.6 & 0 & 0 & 0 & 0 \\ 0 & 0.7 & 0 & 0 & 0 \\ 0 & 0 & 0.7 & 0 & 0 \\ 0 & 0 & 0 & 0.7 & 0 \\ 0 & 0 & 0 & 0 & 0.7 \end{bmatrix}$$

Figure 21. A and B Matrices for Hermit Case

Using the equations we derived in Chapter III we come up with the matrix in Figure 22.

$$(I - A)^{-1} B = \begin{bmatrix} 0.6 & 0.290 & 0.036 & 0.036 & 0.036 \\ 0 & 0.727 & 0.090 & 0.090 & 0.090 \\ 0 & 0.090 & 0.727 & 0.090 & 0.090 \\ 0 & 0.090 & 0.090 & 0.727 & 0.090 \\ 0 & 0.090 & 0.090 & 0.090 & 0.727 \end{bmatrix}$$

Figure 22. Derived Influence Matrix for Hermit Case

When this matrix is multiplied by each event vector, the attitude change vectors shown in Figure 23 are produced.

$$Event1 = \begin{bmatrix} 0.6 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, Event2 = \begin{bmatrix} -0.6 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, Event3 = \begin{bmatrix} 0.036 \\ 0.090 \\ 0.090 \\ 0.090 \\ 0.727 \end{bmatrix}, Event4 = \begin{bmatrix} -0.036 \\ -0.090 \\ -0.090 \\ -0.090 \\ -0.727 \end{bmatrix},$$

$$Event5 = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}, Event6 = \begin{bmatrix} -0.5 \\ -0.5 \\ -0.5 \\ -0.5 \\ -0.5 \end{bmatrix}, Event7 = \begin{bmatrix} 0.636 \\ 0.090 \\ 0.090 \\ 0.090 \\ 0.727 \end{bmatrix}, Event8 = \begin{bmatrix} 0.563 \\ -0.090 \\ -0.090 \\ -0.090 \\ -0.727 \end{bmatrix}$$

Figure 23. Attitude Change Vectors for Hermit Case

This case shows the effects of a one way connection to a larger matrix. This is shown by the fact that only entity A is affected in events one and two yet an event affecting any of the other entities also affects entity A, albeit to a lesser extent. Events seven and eight show the same effect as both have a positive direct effect on entity A but that effect is reinforced or partially negated by the event's affects on entity E.

5. Man in the Middle Case

The fifth test case consisted of a fully connected graph where although all of the entities have positive feelings towards entity B, entity A and the entities C, D, and E dislike each other. The matrices are shown below in Figure 24.

$$A = \begin{bmatrix} 0 & 0.1 & -0.1 & -0.1 & -0.1 \\ 0.1 & 0 & 0.1 & 0.1 & 0.1 \\ -0.1 & 0.1 & 0 & 0.1 & 0.1 \\ -0.1 & 0.1 & 0.1 & 0 & 0.1 \\ -0.1 & 0.1 & 0.1 & 0.1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.6 & 0 & 0 & 0 & 0 \\ 0 & 0.6 & 0 & 0 & 0 \\ 0 & 0 & 0.6 & 0 & 0 \\ 0 & 0 & 0 & 0.6 & 0 \\ 0 & 0 & 0 & 0 & 0.6 \end{bmatrix}$$

Figure 24. A and B Matrices for Man in the Middle Case

Using the equations we derived in chapter three we come up with the matrix in Figure 25.

$$(I-A)^{-1}B=\begin{bmatrix} 0.626 & 0.040 & -0.073 & -0.073 & -0.073 \\ 0.040 & 0.626 & 0.073 & 0.073 & 0.073 \\ -0.073 & 0.073 & 0.631 & 0.086 & 0.086 \\ -0.073 & 0.073 & 0.086 & 0.631 & 0.086 \\ -0.073 & 0.073 & 0.086 & 0.086 & 0.631 \end{bmatrix}$$

Figure 25. Derived Influence Matrix for Man in the Middle Case

When this matrix is multiplied by each event vector, the attitude change vectors shown in Figure 26 are produced.

$$Event1=\begin{bmatrix} 0.626 \\ 0.040 \\ -0.073 \\ -0.073 \\ -0.073 \end{bmatrix}, Event2=\begin{bmatrix} -0.626 \\ -0.040 \\ 0.073 \\ 0.073 \\ 0.073 \end{bmatrix}, Event3=\begin{bmatrix} -0.073 \\ 0.073 \\ 0.086 \\ 0.086 \\ 0.631 \end{bmatrix}, Event4=\begin{bmatrix} 0.073 \\ -0.073 \\ -0.086 \\ -0.086 \\ -0.631 \end{bmatrix},$$

$$Event5=\begin{bmatrix} 0.223 \\ 0.443 \\ 0.402 \\ 0.402 \\ 0.402 \end{bmatrix}, Event6=\begin{bmatrix} -0.223 \\ -0.443 \\ -0.402 \\ -0.402 \\ -0.402 \end{bmatrix}, Event7=\begin{bmatrix} 0.552 \\ 0.113 \\ 0.013 \\ 0.013 \\ 0.558 \end{bmatrix}, Event8=\begin{bmatrix} 0.699 \\ -0.032 \\ -0.159 \\ -0.159 \\ -0.705 \end{bmatrix}$$

Figure 26. Attitude Change Vectors for Man in the Middle Case

This case is the first that shows entities with conflicting loyalties. This is shown by the fact that any event that has a good affect for an entity other than entity B will cause at least one other entity to demonstrate a negative attitude change. Similarly, if the affect is negative then at least one other entity will demonstrate a positive attitude change. It should also be noted that in event seven the change in attitude for entity B is greater than for a event with a good result for either entity A or entity E alone while all of the other entities had less of an attitude increase due to their dislike for each other.

6. Cult Leader Case

The sixth test case consisted of a fully connected graph where all of the entities have strong positive feelings towards entity A. The other entities also have low B matrix values, which make them more strongly affected by outside forces than by direct effects on themselves. The matrices are shown below in Figure 27.

$$A = \begin{bmatrix} 0 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.5 & 0 & 0.1 & 0.1 & 0.1 \\ 0.5 & 0.1 & 0 & 0.1 & 0.1 \\ 0.5 & 0.1 & 0.1 & 0 & 0.1 \\ 0.5 & 0.1 & 0.1 & 0.1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.6 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & 0 & 0 & 0 \\ 0 & 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0 & 0.2 \end{bmatrix}$$

Figure 27. A and B Matrices for Cult Leader Case

Using the equations we derived in Chapter III we come up with the matrix in Figure 28.

$$(I - A)^{-1} B = \begin{bmatrix} 0.840 & 0.040 & 0.040 & 0.040 & 0.040 \\ 0.600 & 0.236 & 0.054 & 0.054 & 0.054 \\ 0.600 & 0.054 & 0.236 & 0.054 & 0.054 \\ 0.600 & 0.054 & 0.054 & 0.236 & 0.054 \\ 0.600 & 0.054 & 0.054 & 0.054 & 0.236 \end{bmatrix}$$

Figure 28. Derived Influence Matrix for Cult Leader Case

When this matrix is multiplied by each event vector, the attitude change vectors shown in Figure 29 are produced.

$$Event1 = \begin{bmatrix} 0.84 \\ 0.6 \\ 0.6 \\ 0.6 \\ 0.6 \end{bmatrix}, Event2 = \begin{bmatrix} -0.84 \\ -0.6 \\ -0.6 \\ -0.6 \\ -0.6 \end{bmatrix}, Event3 = \begin{bmatrix} 0.04 \\ 0.054 \\ 0.054 \\ 0.054 \\ 0.236 \end{bmatrix}, Event4 = \begin{bmatrix} -0.04 \\ -0.054 \\ -0.054 \\ -0.054 \\ -0.236 \end{bmatrix},$$

$$Event5 = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}, Event6 = \begin{bmatrix} -0.5 \\ -0.5 \\ -0.5 \\ -0.5 \\ -0.5 \end{bmatrix}, Event7 = \begin{bmatrix} 0.88 \\ 0.654 \\ 0.654 \\ 0.654 \\ 0.836 \end{bmatrix}, Event8 = \begin{bmatrix} 0.8 \\ 0.545 \\ 0.545 \\ 0.545 \\ 0.363 \end{bmatrix}$$

Figure 29. Attitude Change Vectors for Cult Leader Case

This case shows the effects of having low values in the B matrix. Entity A is the only one with a high value in the B matrix so any event that affects entity A has extreme effects on all of the entities. Conversely, an event that affects entity E will have a minimal effect on the others. This is best seen in event eight where the event has a negative immediate effect on entity E but that is overridden by the immediate positive effect on entity A. Even entity E has a positive attitude change despite the negative effect on them personally.

7. Dictator Case

The final test case consisted of a fully connected graph where all of the entities have strong negative feelings towards entity A. At the same time, entity A has positive feelings towards the other entities. The other entities have low B matrix values, which make them more strongly affected by outside forces than by their own personal effects while entity A has a large B matrix value which causes it to be relatively resistant to an events effects on another entity. The matrices are shown below in Figure 30.

$$A = \begin{bmatrix} 0 & 0.1 & 0.1 & 0.1 & 0.1 \\ -0.5 & 0 & 0.1 & 0.1 & 0.1 \\ -0.5 & 0.1 & 0 & 0.1 & 0.1 \\ -0.5 & 0.1 & 0.1 & 0 & 0.1 \\ -0.5 & 0.1 & 0.1 & 0.1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.6 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & 0 & 0 & 0 \\ 0 & 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0 & 0.2 & 0 \\ 0 & 0 & 0 & 0 & 0.2 \end{bmatrix}$$

Figure 30. A and B Matrices for Dictator Case

Using the equations we derived in chapter three we come up with the matrix in Figure 31.

$$(I - A)^{-1} B = \begin{bmatrix} 0.466 & 0.022 & 0.022 & 0.022 & 0.022 \\ -0.333 & 0.191 & 0.010 & 0.010 & 0.010 \\ -0.333 & 0.010 & 0.191 & 0.010 & 0.010 \\ -0.333 & 0.010 & 0.010 & 0.191 & 0.010 \\ -0.333 & 0.010 & 0.010 & 0.010 & 0.191 \end{bmatrix}$$

Figure 31. Derived Influence Matrix for Dictator Case

When this matrix is multiplied by each event vector, the attitude change vectors shown in Figure 32 are produced.

$$Event1 = \begin{bmatrix} 0.466 \\ -0.333 \\ -0.333 \\ -0.333 \\ -0.333 \end{bmatrix}, Event2 = \begin{bmatrix} -0.466 \\ 0.333 \\ 0.333 \\ 0.333 \\ 0.333 \end{bmatrix}, Event3 = \begin{bmatrix} 0.022 \\ 0.010 \\ 0.010 \\ 0.010 \\ 0.191 \end{bmatrix}, Event4 = \begin{bmatrix} -0.022 \\ -0.010 \\ -0.010 \\ -0.010 \\ -0.191 \end{bmatrix},$$

$$Event5 = \begin{bmatrix} 0.277 \\ -0.055 \\ -0.055 \\ -0.055 \\ -0.055 \end{bmatrix}, Event6 = \begin{bmatrix} -0.277 \\ 0.055 \\ 0.055 \\ 0.055 \\ 0.055 \end{bmatrix}, Event7 = \begin{bmatrix} 0.488 \\ -0.323 \\ -0.323 \\ -0.323 \\ -0.141 \end{bmatrix}, Event8 = \begin{bmatrix} 0.444 \\ -0.343 \\ -0.343 \\ -0.343 \\ -0.525 \end{bmatrix}$$

Figure 32. Attitude Change Vectors for Dictator Case

This case also shows the effects of having low values in the B matrix. Entity A is the only one with a high value in the B matrix so any event that affects entity A has extreme effects on all of the entities. It represents a case where entity A cares about the populace but the populace does not particularly like entity A. This is shown in event seven where despite a positive initial effect on entity E there is still a net negative effect on entities B, C, D, and E. Event five and six show this same set of opposing effects, where entity A is effected positively the others are effected negatively and visa versa.

8. Village Example

To investigate a larger environment we used the twenty-eight agents we had built as part of the mission and dialog generator and formulated three layers of social networks that we then combined to create a consolidated network we could use to verify that our model was scalable. The three matrices we created represented the entities family relationships, their work relationships, and their friend and foe relationships. The matrix showing the family relationships is shown in Figure 33.

	Stog, Trevor	Stog, Lucy	Moses, Grammy	Moses, Grampa	Moses, Sally	Moses, George	Moses, Eugene	Longfeather, Remy	Young, Greg	Young, Nancy	McDonald, Biggs	Cabiness, Wedge	Beavers, Han	Deleon, Luke	Horton, Tim	Horton, Becca	Horton, Dan	Horton, Brandon	Horton, Susan	Pennyworth, Giles	Pennyworth, Jasmine	Darken, Rudy	Darken, Chris	Darken, Patricia	Dulaney, Kate	Dulaney, Jared	Dulaney, Lana
Stog, Trevor	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Stog, Lucy	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, Grammy	0.00	0.00	0.00	0.33	0.33	0.17	0.17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, Grampa	0.00	0.00	0.25	0.00	0.25	0.25	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, Sally	0.00	0.00	0.25	0.25	0.00	0.25	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, George	0.00	0.00	0.10	0.10	0.40	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, Eugene	0.00	0.00	-0.40	-0.10	0.30	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Longfeather, Remy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Young, Greg	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Young, Nancy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
McDonald, Biggs	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cabiness, Wedge	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Beavers, Han	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Deleon, Luke	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Tim	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.20	0.20	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Becca	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.30	0.30	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Dan	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.20	0.00	0.20	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Brandon	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.20	0.20	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Susan	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.60	0.05	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Pennyworth, Giles	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Pennyworth, Jasmine	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Darken, Rudy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.50	0.00	0.00	0.00	0.00
Darken, Chris	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.50	0.00	0.00	0.00
Darken, Patricia	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	-0.50	0.00	0.00	0.00	0.00
Dulaney, Kate	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.70
Dulaney, Jared	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.50
Dulaney, Lana	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.50	0.00

Figure 33. Family Matrix

The family matrix shows the influence the various members of each entity's family have on the entity that corresponds to the row in the matrix. The matrix showing the entities work relationships is shown in Figure 34.

	Stog, Trevor	Stog, Lucy	Moses, Grammy	Moses, Grampa	Moses, Sally	Moses, George	Moses, Eugene	Longfeather, Remy	Young, Greg	Young, Nancy	McDonald, Biggs	Cabiness, Wedge	Beavers, Han	Deleon, Luke	Horton, Tim	Horton, Becca	Horton, Dan	Horton, Brandon	Horton, Susan	Pennyworth, Giles	Pennyworth, Jasmine	Darken, Rudy	Darken, Chris	Darken, Patricia	Dulaney, Kate	Dulaney, Jared	Dulaney, Lana
Stog, Trevor	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.40	0.00
Stog, Lucy	0.00	0.00	-0.30	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.10	0.10	0.00	0.00	0.00	0.30	0.00
Moses, Grammy	0.00	0.20	0.00	0.00	0.00	0.00	-0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	0.00	0.00	0.00	-0.20	-0.10	0.00	0.00	0.00	-0.10	0.00
Moses, Grampa	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00
Moses, Sally	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	-0.10	0.20	0.00	0.00	0.00	0.30	0.00	0.00	0.30
Moses, George	-0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.00
Moses, Eugene	0.00	0.05	-0.75	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.00	0.05	0.05	0.00	0.00	0.05	0.00	0.00
Longfeather, Remy	0.00	0.00	0.00	-0.20	0.00	0.00	0.00	0.00	0.00	-0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00
Young, Greg	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.10	0.10	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.50	0.00	0.00	0.00
Young, Nancy	0.00	0.00	0.00	0.30	0.00	0.00	0.00	-0.60	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	0.00	0.00	0.00	0.00
McDonald, Biggs	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.10	0.10	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	0.00	0.00	0.00
Cabiness, Wedge	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.10	0.00	0.40	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	0.00	0.00	0.00
Beavers, Han	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.05	0.40	0.00	-0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	0.00	0.00	0.00
Deleon, Luke	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	0.00	0.50	0.15	0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	0.00	0.00	0.00
Horton, Tim	-0.40	0.00	0.00	0.00	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00
Horton, Becca	0.00	0.20	-0.30	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	-0.10	0.00	0.00	0.00	0.10	0.00
Horton, Dan	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.20	0.00	0.00	0.00	0.00	0.10	0.00	0.30
Horton, Brandon	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Susan	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	-0.10	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.30
Pennyworth, Giles	0.00	0.10	0.30	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.10	0.00
Pennyworth, Jasmine	0.00	0.10	-0.20	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.10	0.00
Darken, Rudy	0.00	0.00	0.00	0.60	0.00	0.00	0.00	-0.30	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Darken, Chris	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.40	0.00	-0.15	-0.15	-0.15	-0.15	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Darken, Patricia	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20
Dulaney, Kate	0.00	0.30	-0.20	0.00	0.00	0.00	-0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.05	-0.05	0.00	0.00	0.00	0.00	0.00
Dulaney, Jared	-0.80	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dulaney, Lana	0.00	0.00	0.00	0.00	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	0.30	0.20	0.00	0.00	0.00	0.00	0.05	0.00	0.00

Figure 34. Work Matrix

The work matrix shows the how each entity is related to each of the others concerning their work environment. We created groups to represent soldiers, priests,

children and their teachers, nobility, etc. The matrix we developed to represent the entities friend and foe relationships is shown in Figure 35.

	Stog, Trevor	Stog, Lucy	Moses, Grammy	Moses, Grampa	Moses, Sally	Moses, George	Moses, Eugene	Longfeather, Remy	Young, Greg	Young, Nancy	McDonald, Biggs	Cabiness, Wedge	Beavers, Han	Deleon, Luke	Horton, Tim	Horton, Becca	Horton, Dan	Horton, Brandon	Horton, Susan	Pennyworth, Giles	Pennyworth, Jasmine	Darken, Rudy	Darken, Chris	Darken, Patricia	Dulaney, Kate	Dulaney, Jared	Dulaney, Lana
Stog, Trevor	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.10	0.00	0.10	0.05	0.05	0.10	0.20	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	0.00	0.00	0.00	0.00	0.00
Stog, Lucy	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	0.00	0.00	0.30	0.00	0.00
Moses, Grammy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	0.10	0.10	0.20	0.00	0.00	0.00	0.00	0.00	-0.20	0.00	0.20
Moses, Grampa	0.05	0.05	0.05	0.00	0.05	0.05	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.10	0.05	0.05	0.05	0.05	0.05
Moses, Sally	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50
Moses, George	-0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	0.30	0.00	0.00	0.00	0.00
Moses, Eugene	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.20	0.00	0.00	0.00	0.00	0.00	-0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	0.00	0.00
Longfeather, Remy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.30	0.30	0.00	0.00	0.00	0.00
Young, Greg	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.20	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	0.30	0.00	0.00	0.00	0.00
Young, Nancy	0.00	0.30	-0.10	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.20	0.00
McDonald, Biggs	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.15	0.15	0.60	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cabiness, Wedge	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.30	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Beavers, Han	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.10	0.40	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	0.10	0.00	0.00	0.00	0.00
Deleon, Luke	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.20	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00
Horton, Tim	-0.05	0.00	0.00	0.10	0.00	0.10	0.00	0.10	0.10	0.00	-0.05	-0.02	-0.02	-0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.10	-0.01	0.00	0.00	0.30	0.00
Horton, Becca	0.10	0.10	0.00	0.00	0.00	0.10	0.10	0.00	0.10	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.10	0.10	0.10	0.00
Horton, Dan	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50
Horton, Brandon	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.70
Horton, Susan	0.00	0.00	0.00	0.00	0.70	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.30
Pennyworth, Giles	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.20	0.30
Pennyworth, Jasmine	0.00	-0.10	0.00	-0.10	0.00	0.00	-0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.10	0.40
Darken, Rudy	-0.01	-0.01	-0.01	0.15	-0.01	-0.01	-0.01	-0.16	-0.01	-0.01	-0.01	-0.01	-0.01	-0.01	0.10	-0.01	-0.10	-0.10	-0.01	-0.01	-0.01	0.00	0.10	0.10	-0.01	-0.01	-0.01
Darken, Chris	0.02	0.02	0.02	0.02	0.02	0.20	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.10	0.02	0.02	0.02	0.02	0.02	0.02	-0.04	0.00	0.14	0.02	0.10	0.02
Darken, Patricia	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.10	0.10	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.20	0.10	0.00	0.00	0.00	0.00
Dulaney, Kate	0.00	0.30	0.00	0.00	0.00	0.00	-0.20	0.00	0.10	0.10	0.00	0.00	0.00	0.00	0.10	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Dulaney, Jared	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.30	0.00	0.00	0.00	0.10	0.10	0.00	0.00	0.00	0.10	0.00	0.00
Dulaney, Lana	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.30	-0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Figure 35. Friend and Foe Matrix

The friend and foe matrix shows the relationships that the people in our village have with each other. This matrix covers everything that's not covered by the other two. Using the linear weighting algorithm we described in chapter three, the previous three matrices were combined to create a single matrix as shown in Figure 36. The weighting values for each entity are listed in the last four columns. Due to conflicting signs in the matrices that make up the weighted matrix some of the values cancel out. This made it necessary to normalize the values in the matrix so the summation of the absolute value of each of the values in each of the rows in our A matrix added up to one.

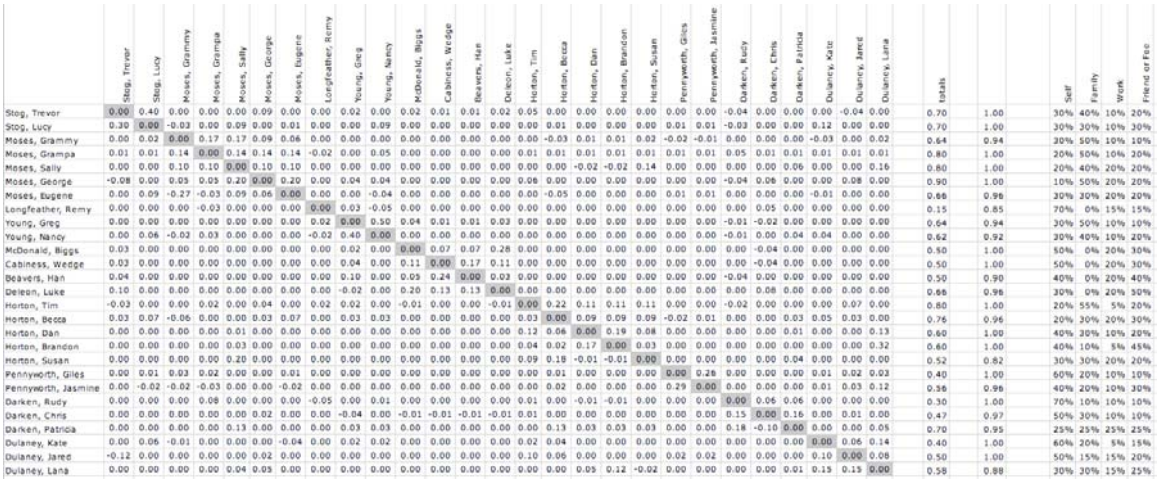

Figure 36. Weighted Matrix

The resulting A matrix is shown in Figure 37.

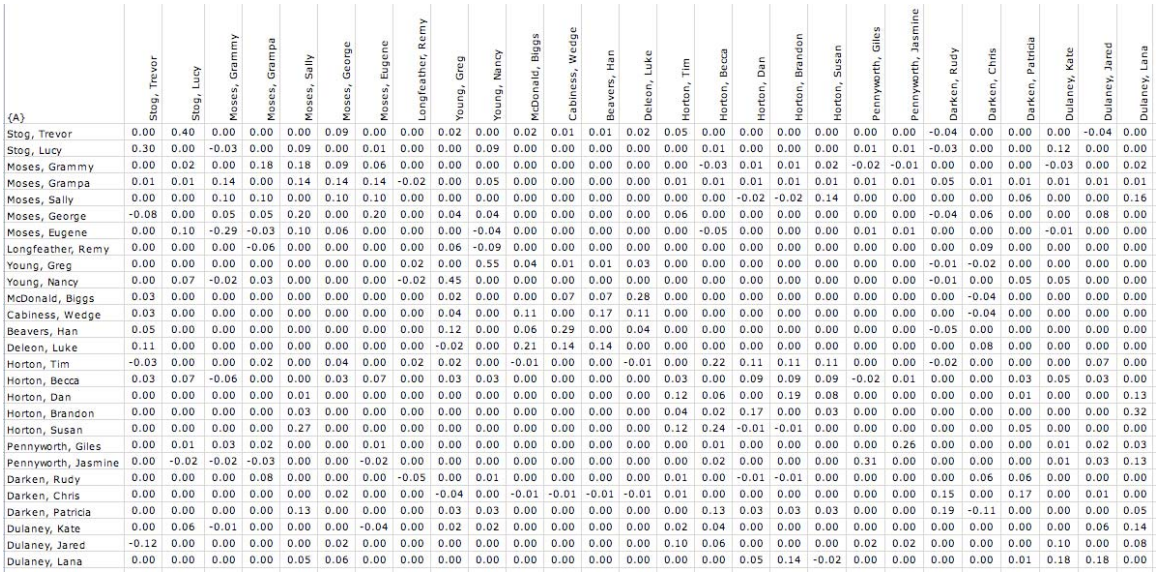

Figure 37. A Matrix

The resulting B matrix is shown in Figure 38.

{B}																												
Stog, Trevor	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Stog, Lucy	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, Grammy	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, Grampa	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, Sally	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, George	0.00	0.00	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Moses, Eugene	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Longfeather, Remy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.70	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Young, Greg	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Young, Nancy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
McDonald, Biggs	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cabiness, Wedge	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Beavers, Han	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Deleon, Luke	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Tim	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Becca	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.20	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Dan	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Brandon	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Horton, Susan	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Pennyworth, Giles	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.60	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Pennyworth, Jasmine	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Darken, Rudy	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.70	0.00	0.00	0.00	0.00	0.00	0.00
Darken, Chris	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.00	0.00
Darken, Patricia	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.25	0.00	0.00	0.00	0.00
Dulaney, Kate	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.60	0.00	0.00	0.00
Dulaney, Jared	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00
Dulaney, Lana	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30

Figure 38. B Matrix

Following the same procedure as we did with the five-person examples we derived the influence matrix shown in Figure 39.

INV{I-A}B																												
Stog, Trevor	0.34	0.14	0.00	0.00	0.02	0.01	0.01	0.00	0.02	0.03	0.02	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.00	0.00	-0.05	0.00	0.00	0.04	-0.01	0.01
Stog, Lucy	0.10	0.35	-0.01	0.00	0.02	0.00	0.01	0.00	0.03	0.05	0.01	0.01	0.00	0.00	0.01	0.01	0.00	0.01	0.01	0.01	0.01	-0.04	0.00	0.00	0.10	0.01	0.02
Moses, Grammy	0.00	0.01	0.31	0.04	0.06	0.02	0.05	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.01	0.02	-0.01	0.00	0.01	0.00	0.01	-0.01	0.02	0.02
Moses, Grampa	0.00	0.01	0.03	0.21	0.05	0.02	0.06	-0.02	0.01	0.03	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.02	0.01	0.01	0.04	0.01	0.01	0.02	0.02	0.02
Moses, Sally	0.00	0.01	0.03	0.03	0.23	0.02	0.05	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.01	0.01	0.00	0.01	0.05	0.00	0.00	0.01	0.00	0.02	0.03	0.03	0.06
Moses, George	-0.03	0.00	0.00	0.02	0.05	0.11	0.08	0.00	0.02	0.02	0.00	0.00	0.00	0.00	0.02	0.01	0.01	0.01	0.02	0.00	0.00	-0.02	0.03	0.01	0.02	0.06	0.02
Moses, Eugene	0.01	0.03	-0.09	-0.02	0.01	0.00	0.29	0.00	-0.01	-0.02	0.00	0.00	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	0.01	0.01	-0.01	0.00	0.00	0.00	0.00	0.00
Longfeather, Remy	0.00	0.00	0.00	-0.01	0.00	0.00	0.00	0.70	0.01	-0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.04	0.00	0.00	0.00	0.00
Young, Greg	0.01	0.02	0.00	0.00	0.00	0.00	0.00	0.01	0.40	0.22	0.04	0.02	0.02	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.02	-0.02	0.01	0.03	0.00	0.00
Young, Nancy	0.01	0.03	-0.01	0.01	0.01	0.00	0.00	-0.01	0.19	0.41	0.02	0.01	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.01	-0.01	0.01	0.05	0.01	0.01
McDonald, Biggs	0.03	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.01	0.55	0.08	0.06	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.01	-0.01	0.00	0.00	0.00	0.00
Cabiness, Wedge	0.02	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.02	0.09	0.55	0.09	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.02	-0.02	0.00	0.00	0.00	0.00
Beavers, Han	0.03	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.06	0.03	0.07	0.17	0.44	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	-0.04	-0.01	0.00	0.01	0.00	0.00
Deleon, Luke	0.05	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.14	0.12	0.08	0.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.04	0.00	0.01	0.00	0.00
Horton, Tim	-0.01	0.01	-0.01	0.01	0.02	0.01	0.01	0.02	0.02	0.01	-0.01	0.00	0.00	0.00	0.22	0.06	0.07	0.08	0.05	0.00	0.00	-0.01	0.00	0.01	0.03	0.06	0.04
Horton, Becca	0.02	0.04	-0.03	0.00	0.01	0.01	0.03	0.00	0.03	0.03	0.00	0.00	0.00	0.00	0.02	0.22	0.06	0.06	0.04	-0.01	0.00	0.00	0.00	0.01	0.06	0.03	0.03
Horton, Dan	0.00	0.00	0.00	0.00	0.02	0.00	0.01	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.03	0.03	0.44	0.11	0.04	0.00	0.00	0.00	0.00	0.01	0.04	0.04	0.08
Horton, Brandon	0.00	0.00	0.00	0.00	0.02	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.02	0.09	0.45	0.02	0.00	0.00	0.00	0.00	0.00	0.05	0.05	0.13
Horton, Susan	0.00	0.01	0.00	0.01	0.07	0.01	0.02	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.03	0.06	0.02	0.02	0.33	0.00	0.00	0.01	0.00	0.02	0.03	0.02	0.03
Pennyworth, Giles	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.00	0.65	0.11	0.00	0.00	0.00	0.02	0.03	0.03
Pennyworth, Jasmine	0.00	-0.01	0.00	-0.01	0.00	0.00	-0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.00	0.20	0.44	0.00	0.00	0.00	0.03	0.04	0.05
Darken, Rudy	0.00	0.00	0.00	0.02	0.01	0.00	0.01	-0.03	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.72	0.03	0.02	0.00	0.00	0.00
Darken, Chris	0.00	0.00	0.00	0.00	0.01	0.00	0.01	-0.01	-0.01	0.00	-0.01	-0.01	-0.01	-0.01	0.00	0.01	0.01	0.01	0.01	0.00	0.00	0.13	0.50	0.05	0.01	0.01	0.01
Darken, Patricia	0.00	0.01	0.00	0.01	0.04	0.00	0.01	-0.01	0.02	0.02	0.00	0.00	0.00	0.00	0.01	0.03	0.02	0.03	0.02	0.00	0.00	0.12	-0.05	0.25	0.02	0.02	0.04
Dulaney, Kate	0.00	0.02	0.00	0.00	0.00	0.00	-0.01	0.00	0.01	0.02	0.00	0.00	0.00	0.00	0.01	0.01	0.01	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.63	0.05	0.05
Dulaney, Jared	-0.04	-0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.02	0.01	0.02	0.01	0.02	0.01	0.00	0.00	0.00	0.07	0.52	0.04
Dulaney, Lana	-0.01	0.00	0.00	0.00	0.02	0.01	0.01	0.00	0.01	0.01	0.00	0.00	0.00	0.00	0.01	0.01	0.04	0.08	0.00	0.00	0.00	0.00	0.00	0.01	0.14	0.12	0.34

Figure 39. Derived Matrix: $(I-A)^{-1}B$

The matrix was then multiplied by our test events (u) resulting in the attitude change vectors (x) shown in Figure 40. By comparing the expectations we drew from the A matrix shown above to the resulting attitude change vectors we were able to determine whether our model predicted as we thought it should. This style of analysis is admittedly anecdotal and inherently imprecise but we simply wished to apply a common sense test to our initial findings before pursuing the research further.

	Hfamily		Uhate		Ulove		Sworld		Pdarken		Ffeud		Hsoldier		Skid (Broke)	
	{u1}	{x1}	{u2}	{x2}	{u3}	{x3}	{u4}	{x4}	{u5}	{x5}	{u6}	{x6}	{u7}	{x7}	{u8}	{x8}
Stog, Trevor	0.00	0.04	0.00	0.05	0.00	0.00	1.00	0.66	0.00	0.04	-1.00	-0.43	0.00	0.01	0.00	0.01
Stog, Lucy	0.00	0.03	0.00	0.04	0.00	0.00	1.00	0.71	0.00	0.03	-1.00	-0.38	0.00	0.00	0.00	0.02
Moses, Grammy	1.00	0.47	0.00	-0.01	0.00	0.04	1.00	0.57	0.00	-0.02	0.00	0.00	0.00	0.00	0.00	0.02
Moses, Grampa	1.00	0.38	0.00	-0.04	1.00	0.21	1.00	0.62	0.00	-0.06	0.00	0.03	0.00	0.00	0.00	0.02
Moses, Sally	1.00	0.35	0.00	-0.01	0.00	0.03	1.00	0.62	0.00	-0.03	0.00	0.01	0.00	0.00	0.00	0.06
Moses, George	1.00	0.25	0.00	0.02	0.00	0.02	1.00	0.44	0.00	-0.02	0.00	0.08	0.00	0.00	0.00	0.02
Moses, Eugene	1.00	0.20	0.00	0.01	0.00	-0.02	1.00	0.22	0.00	0.01	0.00	-0.06	0.00	0.00	0.00	0.00
Longfeather, Remy	0.00	-0.02	0.00	-0.01	0.00	-0.01	1.00	0.71	0.00	-0.06	0.00	-0.02	0.00	0.00	0.00	0.00
Young, Greg	0.00	0.00	0.00	0.02	0.00	0.00	1.00	0.79	0.00	0.03	1.00	0.60	0.00	0.02	0.00	0.00
Young, Nancy	0.00	0.01	0.00	0.01	0.00	0.01	1.00	0.75	0.00	0.01	1.00	0.55	0.00	0.01	0.00	0.01
McDonald, Biggs	0.00	0.00	0.00	0.01	0.00	0.00	1.00	0.84	0.00	0.02	0.00	-0.01	0.00	0.06	0.00	0.00
Cabiness, Wedge	0.00	0.00	0.00	0.02	0.00	0.00	1.00	0.84	0.00	0.04	0.00	0.01	0.00	0.09	0.00	0.00
Beavers, Han	0.00	0.00	0.00	0.04	0.00	0.00	1.00	0.81	0.00	0.05	0.00	0.05	1.00	0.44	0.00	0.00
Deleon, Luke	0.00	0.01	0.00	0.00	0.00	0.00	1.00	0.82	0.00	-0.04	0.00	-0.05	0.00	0.08	0.00	0.00
Horton, Tim	0.00	0.04	0.00	0.01	0.00	0.01	1.00	0.66	0.00	0.01	0.00	0.04	0.00	0.00	0.00	0.04
Horton, Becca	0.00	0.02	0.00	0.00	0.00	0.00	1.00	0.64	0.00	-0.01	0.00	0.00	0.00	0.00	0.00	0.03
Horton, Dan	0.00	0.03	0.00	0.00	0.00	0.00	1.00	0.86	0.00	-0.01	0.00	0.01	0.00	0.00	0.00	0.08
Horton, Brandon	0.00	0.03	0.00	0.00	0.00	0.00	1.00	0.87	0.00	-0.01	0.00	0.01	0.00	0.00	0.00	0.13
Horton, Susan	0.00	0.11	0.00	-0.01	0.00	0.01	1.00	0.73	0.00	-0.03	0.00	0.01	0.00	0.00	0.00	0.03
Pennyworth, Giles	0.00	0.02	0.00	0.00	0.00	0.00	1.00	0.89	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03
Pennyworth, Jasmine	0.00	-0.02	0.00	0.00	0.00	-0.01	1.00	0.77	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.05
Darken, Rudy	0.00	0.03	-1.00	-0.72	0.00	0.02	1.00	0.78	-1.00	-0.76	0.00	0.01	0.00	0.00	0.00	0.00
Darken, Chris	0.00	0.02	0.00	-0.13	0.00	0.00	1.00	0.70	-1.00	-0.68	0.00	-0.01	0.00	-0.01	0.00	0.01
Darken, Patricia	0.00	0.06	0.00	-0.12	0.00	0.01	1.00	0.64	-1.00	-0.33	0.00	0.04	0.00	0.00	0.00	0.04
Dulaney, Kate	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.84	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.05
Dulaney, Jared	0.00	0.01	0.00	0.00	0.00	0.00	1.00	0.70	0.00	-0.01	0.00	0.06	0.00	0.00	0.00	0.04
Dulaney, Lana	0.00	0.04	0.00	0.00	0.00	0.00	1.00	0.79	0.00	-0.01	0.00	0.02	0.00	0.00	1.00	0.34

Figure 40. Event and Attitude Change Vectors

In the first seven cases the model seems to have performed according to our expectations. In each of the events the resulting attitude changes fit our expectations and any variations could be explained by examining the original A matrix we developed. The eighth case however broke our model. The “Skid” or ‘Save the Kid’ scenario has a large direct positive effect on Lana Dulaney, a child in the Dulaney family. The narrative behind this case was that the user saved the child’s life in the face of some peril. The inconsistency we found was that a greater attitude change occurred in Dan Horton and Brandon Horton, two of Lana’s friends, than in Lana’s parents. We believe this to be the result of two factors. First since the summation of the absolute value of any rows elements must equal to one, the effect on the adults is lessened since the parents have more connections in their various networks and the smaller social circles of the children cause larger effects in them. An alternative explanation that does not fault the model is that the event may have been badly formed. An event like saving the child’s life should perhaps have a strong direct effect on the parents as well.

VI. CONCLUSIONS AND FUTURE WORK

A. CONCLUSIONS

Our mission and dialog generator is a generic architecture that can be used to extend the number of repetitions that a simulation or game can be used before it becomes predictable. By randomizing or even better yet creating a probability distribution to govern the time and location of emerging threats, the number of unique repetitions possible is staggering. Given agents of various types and even a limited number of sentence templates representing various mission types the possibilities increase manifold. The proposed architecture also increases the amount of code reuse and therefore reduces the time and monetary costs required for development or the addition of new functionality. This is especially true in large systems since the savings that come from our architecture are even more so in larger systems. With smaller systems our architecture loses these advantages but remains highly versatile and less predictable than more conventional approaches.

We have identified what we believe to be a major limitation of our system; it suffers from a lack of individuality among the agents. Other than preprogrammed names the system has no way to distinguish between individuals and especially in the urban environment this would be useful in discussing items that are person specific. We suggest that a combination of our approach and the standard methodology might be useful in solving this problem.

As far as our analysis of our social networks model went, we have shown that it could plausibly be used to model complex social interactions in a simulation. We proposed it to augment our dialog generator but other applications could be imagined. The main problem with our solution is the complexity required to fully implement it. As opposed to the current practice of using overarching factions our system requires the developer to put much greater thought into the creation of the simulation or game environment.

Both the social networks and the event creation requires the developer to consider the complex social and psychological interactions involved and in a simulation requiring truly realistic models, domain experts would need to be brought in to aid in this aspect of model development. To aid in this, we believe that the development of a software tool to allow non-technical personnel to help build these systems is essential. Our social networks model appears to be valid and pass our common sense tests, but the model still requires a large amount of work to fully validate it.

B. FUTURE WORK

We propose the following extensions of our research as possible avenues for future work.

- The mission and dialog generator was created to be modular and therefore allow for different styles of speaking in the same language. This works because in most cases the same language uses similar grammars. It would be interesting to extend the system for use with multiple grammars simultaneously.
- The knowledge level and notoriety level variables are currently assigned manually and in a largely arbitrary manner. Creating functions to automate the process would be useful.
    - The knowledge variable could also be divided into various knowledge areas to represent the fact that agents would tend to have their own areas of expertise.
    - Notoriety could be based on location, intent, past actions, and even have the value rise over time to model the fact that more people become aware of its presence in the game space.
- Modify the system to allow for the death and replacement of an interactive agent. This could probably be extended to allow for the creation of agents on the fly, further automating the scenario authoring process.
- A model representing misinformation, either intentional or unintentional would be interesting when paired with the dialog generation system. This

would allow for more realism as certain missions may be based on bad intelligence. A model for information corruption as it moves further away from the source of the information (i.e. telephone game) might also be interesting to implement.

- The agent's internal representation of the world state is updated instantaneously, which is highly unrealistic. The system should be revised to allow for information transit time to model the fact that it would take time to learn of a new threat or realize that an old threat has been eliminated.
- Addition of a temporal element to our social networks model would also be interesting.
- Experimental validation of the social network model would be interesting but may incur several, if not all of the difficulties described above.
- Expanding the network to hundreds, perhaps thousands of entities would be interesting especially if it required spreading the simulation across a network.
- As suggested above a graphical interface for social network creation would be useful, if not necessary for working in large scale simulations and games.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Beaubouef, T., and R. Lang. "Rough Set Techniques for Uncertainty Management in Automated Story Generation." Proceedings of the 36th Annual Southeast Regional Conference. Marietta, GA, USA, April 1-3, 1998. December 2006 <http://portal.acm.org/citation.cfm?id=275295.275379>.

Bioware Corp. "Neverwinter Nights Vault." Bioware Corp. 2007. <http://nwvault.ign.com>.

Bioware Corp. Neverwinter Nights - Diamond Edition. Vol. 1.68. New York:, 2005.

Khan, M., and R. Ward. "Automatic Scenario Generation in Adventure Games." 2nd International Conference on Application and Development of Computer Games (ADCOG 2003). Hong Kong, China, 6-7 January.

Loonneker, B., and J. Meister. ""Dream on": Designing the ideal Story Generator Algorithm." 2005. December 2006 <http://mustard.tapor.uvic.ca:8080/cocoon/ach_abstracts/proof/session_105_meister.pdf>.

Mateas, M., and A. Stern. "Facade: An Experiment in Building a Fully-Realized Interactive Drama." Game Developers Conference (GDC '03). San Jose, CA, USA, March 4-8, 2003.

---. "Procedural Authorship: A Case-Study of the Interactive Drama Façade." Digital Arts and Culture (DAC) Conference. Copenhagen, Denmark, December 1, 2005.

Mateas, M. "Beyond Story Graphs: Story Management in Game Worlds." 2005. December 2006 <http://mustard.tapor.uvic.ca:8080/cocoon/ach_abstracts/proof/session_105_meister.pdf>.

Orkin, J. "Symbolic Representation of Game World State: Towards Real-Time Planning in Games." AAAI-04 Workshop on Challenges in Game AI. San Jose, CA, USA, July 25-26.

Osborn, B. A. An Agent-Based Architecture for Generating Interactive Stories. PHD Naval Postgraduate School, 2002.

Peinado, F., and P. Gervás. "A generative and case-based implementation of Proppian morphology." 2005. December 2006 <http://mustard.tapor.uvic.ca:8080/cocoon/ach_abstracts/proof/session_105_meister.pdf>.

Reiter, E., and R. Dale. "Building Applied Natural Language Generation Systems." Journal of Natural Language Engineering 3.1 (1997): 57-87. . December 2006.

Shaw, D. Aspects of Interactive Storytelling Systems. MS University of Melbourne, 2004 Melbourne, Australia.

Szilas, N., and J. Rety. "Minimal Structures for Stories." Proceedings of the 1st ACM Workshop on Story Representation, Mechanism and Context. New York, NY, USA, December 2006.

Warren, R., et al. "Simulating Scenarios for Research on Culture & Cognition using a Commercial Role-Play Game." Proceedings of the 37th Conference on Winter Simulation. Orlando, Florida, December 2006 <http://portal.acm.org/citation.cfm?id=1162708.1162903>.

Warren, R., et al. "A Game-Based Testbed for Culture & Personality Research." Proceedings of the NATO Modeling and Simulation Group. Hague, The Netherlands, December 2006 <http://www.dodsbir.net/sitis/view_pdf.asp?id=NATOWarren.doc>.

Wasserman, S., and K. Faust. Social Networks Analysis: Methods and Applications. Cambridge: Cambridge University Press, 1994.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Temasek Defence Systems Institute
   National University of Singapore
   Singapore