**Interactive Storytelling and Emergent Gameplay using Dynamical Policy Models**

**Abstract:** Game developers are often faced with a double challenge: designing systems that foster player freedom and creativity, and allow the delivery of a compelling story[[1]](#footnote-2). Though each of these problems has been successfully addressed separately, there has been little success in building a system that can deal with both at the same time without losing balance. In this report we present the application of a new technique that supports player empowerment and story delivery by providing feedback mechanisms through which the player’s actions have an adjustable impact on the storyline and the storyline has an adjustable control over the gameplay offered to the player.

1. **Introduction**

**Defining the problem and its scope  
Rationale for solving the problem  
Building on Zambetta 08  
Promises**

1. **Background**

**Emergent gameplay  
Game as a storytelling media  
Reductionism vs. Constructivism**

1. **Related work**

**Emergence in video games  
Adam Russell Situationist Game AI  
GTA 3 & HL 2**

1. **Methodology**

**The model (Zambetta 08)   
Assessing the model in a realistic application  
The Clan Wars game structure  
Integrating the model in the game structure**

1. **Experimentation**

**General experimental framework  
Scalability in the light of the design problem  
Control over the gameplay  
Control over the story**

1. **Conclusion and future perspectives**

**How much has been achieved  
How much can we achieve from here  
What are the limitations and what can be done to overcome them**

1. **Introduction**

Though video game development is still a very young discipline in comparison to other engineering problems such as architectural design, it has been rapidly flourishing both in terms of economical growth and product diversification. Despite this diversity a common problem arises in many video games, it stems from the conflict of two main design goals: delivering a compelling story, and offering a high level of interactivity. Every video game embodies a design answer to that problem by offering particular tradeoff between allowing the player to have more control over what is happening in the game, and restricting the player to ensure compatibility with the story elements. Let’s examine each side of this tradeoff to better understand its nature. If we take the problem from a story making angle then our goal would be to ensure the good flow and structure of the game narrative. If we give control to the player over some elements of the story, such as the possibility to kill key characters or to enter some particular area that is not meant to be discovered at that point of the story, then there is a possibility for the narrative structure to lose its dramatic tension. One might then start to wonder why the player is given any control, but interactivity is the essence of video games. Giving the player a chance to be active is what sets video games apart from other story telling media such as books and movies, this interactivity is the very reason behind the success of video games.

It is now clear that interactivity and story delivery are by definition hard to accommodate, it is therefore legitimate to ask whether one should really try to attempt such an unnatural combination. By looking at the video games created so far we can easily notice that some of them are situated at the extremes of this design problem. Video games that rely on interactivity only generally belong to the “simulation” or “sandbox” games categories; this is the case of most sports games such as the “FIFA” series or puzzle games like “Tetris”, “Minesweeper”, or “The Incredible Machine”. Storyless games even appear in more mid-stream genres like real-time strategy games with the “Age of Empires” series or First-person shooter games with the “Counter-Strike” or “Unreal Tournament” series. At the other end of the spectrum, some video games rely heavily on their storyline while offering very limited interactivity. Some titles such as the “Myst”, “Sam & Max”, or “Monkey Island” series have been very successful at captivating the players over the years. Another story-based game type such as the Japanese dating simulation games has also been successful in Asia.

Despite the success of the games sitting at the extremes of the tradeoff, there has been a progressive shift in the market towards more balanced designs [Russel 08]. This change is not only visible through the overwhelming commercial success of the games offering both interactivity and compelling stories such as “Grand Theft Auto 3” or “Half Life 2”, but also through the erosion of the extremes. Most of the successful franchises discussed in the previous paragraph have indeed either ended or been modified to adopt a more balanced position. It is therefore of prime importance for game designers to be able to combine interactivity and story delivery if they wish to produce video games that will answer the market demand.

The work presented in this report demonstrates the use of a possible design solution to this problem. While our solution relies on previous work [Zambetta 08] it is the first time it has been applied to a realistic game structure such as the one we built. The aim of our work is not to focus on interactivity or story delivery only but to bring an answer to the question: “How can they be combined in a productive way?” The next section gives more in-depth background information to introduce the concepts we build upon. Section 3 presents how the problem has been addressed in related research and recent video games. Our methodology is described in section 4; and our experiments are detailed in section 5. Finally, we present our conclusions in section 6 and give an overview of future work.

1. **Background**

The apparent incompatibilities we described at the design level between compelling stories and player empowerment are also present at the technical level. Since the techniques that support story delivery and interactivity are directly relevant to our work we describe them here to provide the necessary background information.

* 1. **Reductionism**

Video games, as any other type of game, define some rules for the players to abide. Rules in video games are not something the players can choose to ignore; they are enforced through the game mechanics by the developers to frame the gameplay which defines the space of possibilities in the game by setting boundaries. Over the years game designers have been progressively pushing these boundaries further to offer an always greater level of interactivity. Most of the video game-related research has been focusing on this challenge and have come to create an ensemble of techniques that allows more interactivity, greater scalability, and supports player creativity. These techniques are generally referred to as **reductionist** because they tend to organize large and complex systems by using a small set of general rules. Reductionist approaches typically involve a small number of different types of entities with each type potentially covering many entities; the set of rules then dictate how the different types of entities interact locally. A classic example of such reductionist techniques is the way games physics are handled in action and adventure games such as “Half-Life 2”: Some entities such as explosive barrels and non-playing characters (NPCs) are sensitive to damage while weapon projectiles and explosions are damaging entities. This allows emergent gameplay features such as the possibility of targeting groups of explosive barrels located near enemy NPCs thus inflicting damage to one of the barrels which will explode and damage the other barrels to trigger their explosion which, in turn, will injure or kill all enemies nearby. Such chains of interactions are very efficiently implemented through reductionist techniques because there is no need to implement specific weapon projectile-to-barrel, barrel-to-barrel, and barrel-to-NPCs interactions but only a damaging object type-to-damage-sensitive object type interaction. Not only do reductionist techniques such as the one just described offer a high level of interactivity but they also allow high productivity and open-endedness which makes them all the more appealing. Despite all these advantages reductionist techniques also have a few disadvantages. The gameplay features provided through a reductionist approach, once mastered, rarely evolve to renew the player’s interest and therefore tends to make the game repetitive. Emergent dynamics are also a source of bugs or flaws in the gameplay because they are sometimes hard to predict. Such unintended mechanics can sometimes be benefic and allow players to solve problems in a creative way e.g. securely opening doors with remote mines in the game “Deus Ex” or the rocket-jumping techniques in “Quake 3 Arena” where players use the blow of their own rockets to jump further. Most of the time however, unintended emergent mechanics have negative consequences: in “Half-Life” where players could skip entire sections of the game by making ladders out of wall-mines.

* 1. **Constructivism**

Though reductionist techniques can be very powerful they are not suited to the creation and organization of the unique elements and processes of a video game. If the narrative of a game requires a sequence with particular animations of the characters, dialogs, special camera movements, and a tailor-made music, they will be implemented in the game through the use of a particular script made exclusively for that sequence because it is faster to produce and more efficient. Everything that can be considered unique inside the game is generally handled using constructivist techniques. The constructivist approach, which is more specific to video games and therefore less in touch with academic research, is on many levels the opposite of the reductionist approach. A constructivist technique typically requires the definition of many different types of entities or processes with very few occurrences of each type; each type obeys to specific rules that often have a global impact on the game. Constructivist techniques allow developers to fine tune sections of the game to give a unique experience to the player and support the narrative; they also contribute to the richness of the game by adding more content. Constructivism is commonly used in story telling where the normal gameplay is deactivated but it also omnipresent through any content that is not generic such as world design, special characters, quests, unique objects weapons or spells are all good candidates for this. Because all of this content is often used only once or in a fixed way, constructivist content is poorly scalable and limits replayability because fixed elements will offer the exact same experience each time the game is played.

1. **Related work**

In section 1 we have discussed the possibility of designing video games that rely massively on either interactivity or story delivery, and identified a growing need for more balanced games. In this section we examine what has already been achieved and studied that is directly relevant to this goal of balanced design. The first place to look when faced with a video game design problem is the very large collection of games that have been released in the past. We are here going to discuss the design choices made in two critically acclaimed video games: the first-person shooter “Half-Life 2”, also referred to as HL2, developed by Valve Corporation and released in November 2004, and the role-playing game “The Elder Scrolls IV: Oblivion”, generally referred to as Oblivion, developed by Bethesda Softworks and released in march 2006.

In HL2 the story is delivered to the player through carefully scripted sequences where non-playing characters move, talk, and interact with the environment. During these phases the degree of interactivity offered to the player is reduced to ensure that the player will not do anything that would be inconsistent with the story. The normal gameplay which relies on reductionist techniques is momentarily truncated to dispense a chunk of the story in a constructivist manner. The whole game therefore consists of a succession of mutually-exclusive reductionist highly interactive gameplay and constructivist story sequences. While this technique can be used to achieve an overall balance between story delivery and interactivity, it fails to unite them in a mutually profitable fashion: the rich and dynamic storyline doesn’t prevent the gameplay from being homogeneous, and though the general level of interactivity is high the player remains prisoner of the same predefined narrative at every playthrough. Through the work presented in this report our aim is not to offer a design that accommodates interactivity and story delivery by keeping them separate from each other but to enable a positive synergy between the two. Half-Life 2 is therefore a good example of a game design that addresses the same basic problem: bringing interactivity and story delivery together, but chooses a solution that is opposite to ours.

The structure of the story in the game Oblivion is composed of one main quest and many optional quests that have no direct impact on the main one. Though all of this content is constructivist in nature it does not go against but rather promotes player empowerment. This very positive dynamic is enabled through careful design choices. The constructivist content is seamlessly embedded in the reductionist framework of the game. Storyline elements in Oblivion are mostly delivered through dialogues between the player and NPCs (non-playing characters), though this medium is generally not the best to buildup dramatic tension, it does not require the “normal” (reductionist) gameplay to be deactivated, and therefore leaves the player with a certain amount of control over these story element. By deciding whether or not to talk to some NPCs, what to tell them, and how to interact with them, the player can choose to continue a particular quest in different ways, leave it uncompleted for the time being, or start new quests. By providing ways to interact with the storyline elements it is therefore possible to give the player more freedom and control. This however requires presenting the player with enough different choices to ensure his satisfaction, which is exactly what is achieved through the over-abundance of optional quests. This overall structure has the effect of letting the player be responsible for his or her choices. Though the player may come to regret some of its choices, a bad choice is always better accepted than an enforced restriction which is very frustrating, and good choices are all the more rewarding when the player can take full credit for it instead of just following the predefined path laid out by the game designer.  
Though the design strategy used in Oblivion allows a successful reunion of interactivity and story delivery, it suffers from very poor scalability which has considerable impact on productivity. To ensure enough diversity for the player to be satisfied this design requires the creation of an extremely big amount of storyline content; every game development studios might not be able to withstand such a lengthy and costly process. The developers of Oblivion have succeeded in giving the player a sensation of endless possibilities, but anybody playing the game twice would soon come to realize that a second playthrough leads the player to repeat the same actions. Relying on a massive amount of constructivist content inevitably leads to poor scalability and replayability. Similarly to Oblivion, our approach is directed towards the reunion of interactivity and story-delivery, it does not however connect the many possibilities offered by the reductionist mechanics of the game to a many chunks of constructivist story content, but gives a high-level semantic interpretation of the game mechanics that can be used to support both constructivist and reductionist story delivery techniques.

Though video game development is still in its infancy there is an ever-growing interest in academic research for game-related topics and a progressive tendency to organize the accumulated body of domain knowledge in the more academic form of conferences and publications. In the second part of this section we examine two publications that are relevant to our work, one [Sweetser08] stemming from academic research, and the other [Russell08] from field experience.

In [Sweetser08] the author studies the relation between player empowerment and story delivery in her chapter “Emergent Narrative”, where she defines a “Storytelling Continuum” at the extremes of which we can find at one end a monolithic fully predefined story with no way for the player to have an impact on it, and at the other end a fully emergent story composed of many small predefined elements that are sensitive to the player’s actions. To allow partially interactive or fully interactive storytelling to take place in a game, and therefore allow the desired combination of interactivity and story delivery, the author stresses the need for a technique to connect the player’s actions and the story:

“If you look at a game narrative across multiple […] levels, you can see the player’s low-level actions at the bottom and the game’s high-level story at the top. How to map these things together is of key importance.”

She then goes on to propose two ways to perform this mapping in order to enable emergent narrative. She first proposes the use of a story-graph mechanism where the player’s actions decide which node of the graph will be visited next, this technique is similar in nature to the one used in Oblivion and therefore shares the same limitations. In the second alternative proposed by the author all the player’s actions have a weighted impact on storyline variables which would drive the plot away and towards particular elements of the narrative: “Once the culmination of the player’s actions, or the weighted sum, surpasses a plot threshold, the story would be propelled forwards in a given direction”. This last technique is similar in nature to the one we propose in this report in that it enables a constant synergy between the player and the storyline. The approach we propose is however wider in scope in that it can take input not only from the player’s interactions but also from the NPCs, and maps back a story-level abstraction of these inputs onto the game mechanics, and therefore the player.

In [Russell08] the author argues for the need of a hybrid approach between reductionism and constructivism in order to better deal with the challenge of balanced game design. He proposes the concept of “Situationist AI” in which the actions and perceptions of NPCs are not task or situation-independent. Our work follows this general school of thought and implements this approach through the use of an evolution of Richardson’s model of Arms Race, as detailed in the next section.

1. **Methodology**

In this section we describe the nature and theoretical foundations of our model and present the general experimental framework through which we assess its validity.

* 1. **The foundations of the model**

In order to connect the interactive mechanics of the game which are reductionist in nature, to the more constructivist story components, we make use of a model that provides a high-level abstraction and control of all the low-level interactions. Our work builds on [Zambetta08] where the original model conceived by Lewis Fry Richardson has been improved to handle interactive scenarios. Richardson, a renowned English mathematician and psychologist, modeled armament build-up between two nations using a system of two linear ordinary differential equations:



Where and are the level of armament of two nations, and are the mutual fear constants, and are the restraint constants representing internal opposition against arms expenditure, and and are independent factors which can be interpreted as grievance between rivals.  
This version presented in [Zambetta08] allows runtime modification of the parameters thereby making the model interactive as outlined above. The role of the resulting model inside a game architecture is presented in figure 1. The diagram shows how the model takes input from the low-level mechanics through events which have a variable impact on the overall behavior of the model. The global behavior of the model is then used by the gameplay and storyline control unit as a high-level abstraction of the game’s state and dynamic. If the game mechanics need to be affected for *gameplay* or *narrative* reasons feedback will be provided to the model which will impact the game mechanics that are sensitive to the model. Note that this diagram only depicts the new possibilities offered by the use of our model; direct interactions between the game mechanics and the story components are therefore possible but often ill-advised since they take away the flexibility of either the game mechanics or the storyline.



Model

Game mechanics

Gameplay and storyline control

Low-level events

High-level interpretation

Situational awareness

High-level feedback

Player

***Figure 1. The model establishes a bridge between the low-level game mechanics and the high-level gameplay and story elements.***

The model we use is an evolution of [Zambetta08] and preserves its theoretical foundations. To demonstrate the qualities of this model and therefore show how Interactive storytelling and emergent gameplay can be brought together, it is crucial to test the model in a real world scenario. To that end we developed a game structure for a prototypical video game we refer to as “Clan Wars”, which we describe in the next section.

* 1. **The Clan Wars game structure**

In this section we present the Clan Wars game and its internal structure. We first give a brief explanation of the game’s concepts and then describe in more detail the gameplay offered to the player.

* + 1. **The Clan Wars game concepts**

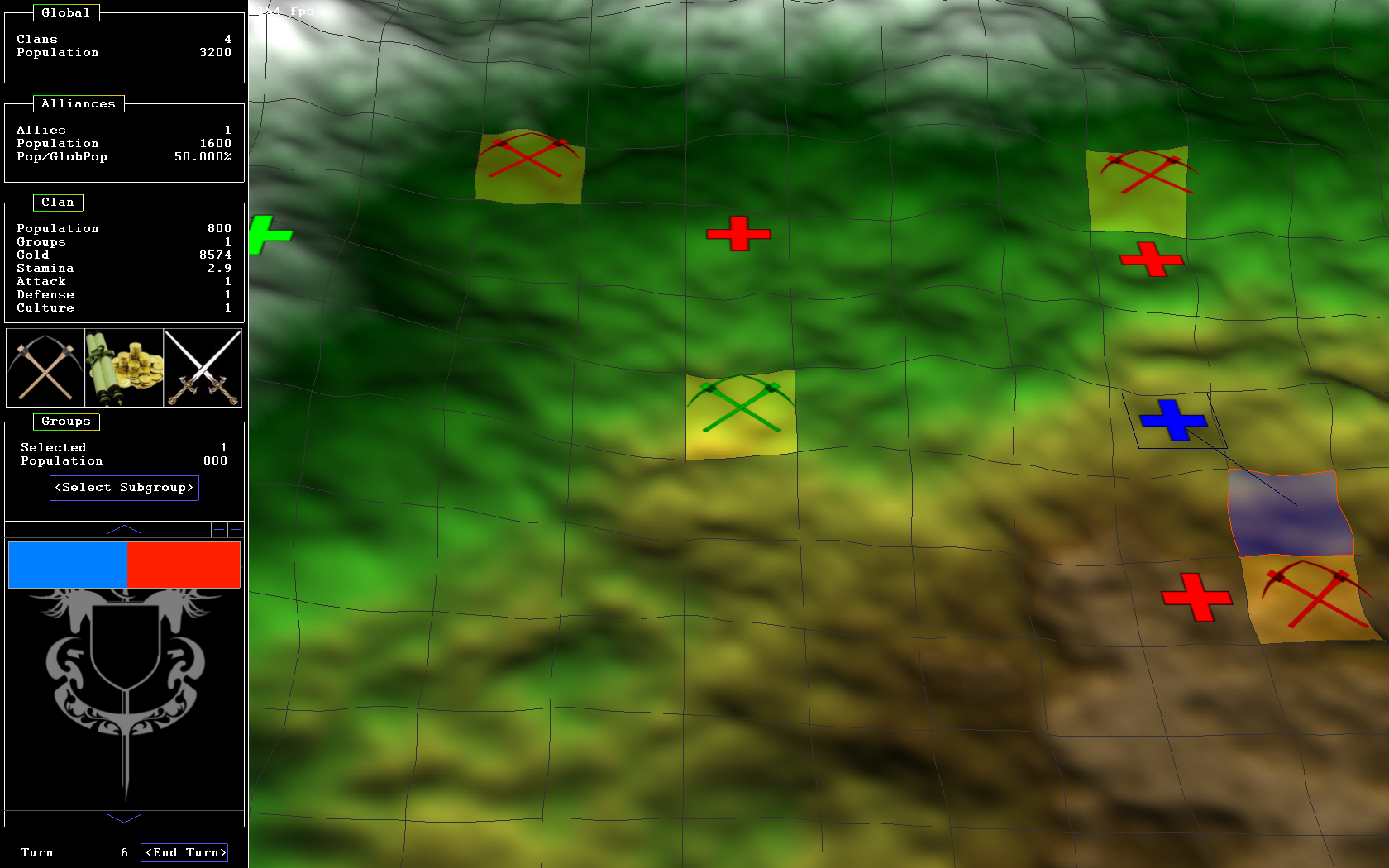
Clan Wars is a turn-based strategy game. Each player controls a clan which consists of up to a thousand individuals that can be divided in one or more groups. The goal of each player is to keep his population to a maximum throughout the turns. The potential causes for death are starvation and external aggression. Starvation happens when the number of gold units is inferior to the population number in the clan at the end of a turn. Aggression happens when a group is attacked by another. To prevent starvation one needs to increase the amount of gold own by his clan, this can be done by sending groups to mine gold which is randomly spread out in the map, or by killing enemies. Gold is also given from a clan to another as a sign of good disposition and often as a bribe to form an alliance. Alliances between clans can be expensive to keep going if the clans are naturally opposed but they ensure a mutual pact of non-aggression.  
The basic idea of the game is to give the player the power to decide how to play by giving him or her the tools to interact diplomatically or aggressively (or a mixture of both) with the other players.

* + 1. **The gameplay mechanics**

To interact with the game the player uses the graphical interface presented in figure 2. Here is a list of what the player can do through that interface:

* Select a group that belongs to his clan by left-clicking on the group cross or pickaxes.
* Select a subgroup of the selected group by clicking the <Select Subgroup> button in the menu and entering the size of the desired subgroup selection.
* Move a selected group by right-clicking the desired destination of the selected group.
* Assign a selected group that is on a golden cell to mining by clicking the pickaxes icon in the menu.
* Attack a group adjacent to the selected group by clicking on the swords in the menu and left-clicking on the group to attack.
* Examine clan relations and offer gold, truths, or alliances to other clans by clicking on the scroll and gold icon in the menu and selecting the desired options.
* End the turn by clicking on the <End Turn> button in the menu.

***Figure 2. The Clan Wars interface with the menu on the left and the terrain grid. The colored crosses represent groups, and the pickaxes on the golden cells represent mining groups.***



All the actions of the players, human or artificial, that represent an interaction between two clans, have an impact on the relations between these two clans and possibly an impact on every player. The possible interactions between clans offered by the game are the following:

* Clan A makes a gold, peace, or alliance offer to clan B (positive interaction)
* Clan A turns down a gold, peace, or alliance offer from clan B (negative interaction)
* Clan A attacks clan B (negative interaction)
* Clan A attacks an enemy of clan B (positive interaction)
* Clan A attacks an ally of clan B (negative interaction)

In the game the relations between clans vary between -100 which corresponds to strong contention, and +100 which corresponds to total cooperation. Alliances require mutual stances equal or above to 60, they remain effective until one of the two allied clans has its stance towards the other clan go below 40. This has the effect of breaking the alliance; but a similar alliance can be renewed right away.

* 1. **The final model**

After describing the theoretical foundations of the model and the game structure it is integrated in, we can now present the internals of our model in greater detail.

**4.3.1. The model requirements**

The goal of our model, as explained in section 4.1, is to bridge the low-level mechanics of the game which are reductionist in nature with the more constructivist high-level elements such as global gameplay and story structure. To allow this connection our model needs to possess the following properties:

* Account for the **discrete game events** that can impact the model variables according to both their *type* and their *magnitude*.
* Simulate the *continuous* evolution of the model variables over *time*.
* Provide a *high-level* interpretation of the *state* and **dynamics** of the game.
* Provide a way to exert *high-level* control over the *low-level* state and dynamics of the game.

From these points two major themes emerge: the capacity of dealing with both discrete and continuous dynamics, and communication allowing bottom-up information propagation and top-down control.

**4.3.2. The elements of the model**

Let us now define the elements that the model deals with:

* **Events**: The game events we chose to use as a basis for our experiments are the interactions between clans as listed in section 4.2. We chose to focus on these events because they form a simple and cohesive group but also allow the kind of emergent gameplay we want the model to support. We associate each of these event types with an impact coefficient , corresponding to event.



* **Stances**: We define the stance of clan towards where represents hatred, represents neutrality, and represents love. We refer to these stances as a square matrix of size equal to the number of clans; for a game involving three clans would be:



Note that in our implementation stances are symmetric and so is (). In the above example the main diagonal comprises zeros only because we consider a clan to be neutral towards itself, but one could easily design a game scenario where these values have another role such as modeling the internal friction and resentment harbored inside a clan. Clearly, in such a case the matrix S would not be symmetric.



The role stnces play in our model is similar to the role of mutual fear and opposition against arms expenditure in the Richardson model, as outlined later in this section.

* **Tempers**: We use tempers similarly to the parameters defined in the Richardson model as the level of armament. We define the temper of clan where represents calm and represents a state of crisis. The tempers are grouped in a single vector of size equal to the number of clans; for a game involving three clans would be:



The vector, because it offers high-level information, is a good indicator of the state and dynamic of the game. By monitoring the temper of a clan it is possible to determine which gameplay alternative that particular clan more likely to choose. Performing calculations on the vector as a whole can also reveal useful information about the game: for instance, an average of the values can summarize the general tendency of the game towards warfare or diplomacy.



* **Relations**: Because we want the clans’ tempers to affect their relations with other clans we define the matrix similar to where represents the relation between clan and clan according to their stance and mutual tempers. Note that unlike stances which define a love or hate level the relations values are closer in practice to a “*line of conduct*” with representing competition and representing cooperation. In the right circumstances, such as a lack of resources, two clans that share a friendly stance may come to share a competitive relation. Similarly, if it is to their advantage, two clans that hate each-other might come to act cooperatively. Relations are in direct contact with the game mechanics because they influence the behaviors of the clans towards each other.



**4.3.3. The inner workings of the model**

We have introduced all the elements pertaining to the model, we now need to connect them in a way that offers the necessary capabilities described at the beginning of this section. We obtain the following equations:



Let us examine each of them in detail:



Equation (1) governs the continuous evolution of the tempers according the stances. For a game including three clans the equation takes the following form:



Which for clan gives:



The logic behind this equation is that a clan in a state of crisis (with a high) is more likely to attack its enemies and less likely to attack its friends. This puts in danger, and therefore increases the tempter of, the enemies of the clan (negative ) as well as decrease the temper of the friends of the clan (positive ) because they are not only less likely to be attacked but they can also rely on clan to battle with its enemies.



Equation (1) is an adapted version of the original Richardson equation presented in section 4.1, in our version the governing parameters of the equation are regrouped in one variable that can adopt both positive and negative values.



(2)



Equation (2), re-written above for better readability, defines how the relations values are calculated. The relation between two clans is defined as the product of the stance they share and the averaged sum of their current mood. One would indeed be more inclined to present a united front with one’s friends when in a state of crisis than in a more relaxed situation. Similarly, a bad stance between two clans would be worsened when in a situation of crisis.

(3)



Equation (3) defines the updating of stances in the occurrence of events. Whenever an event occurs, the stance between the initiator and the receiver of the event is adjusted according to the defined impact of the type of the event and the relevance of the event in its context. The equation is such that the stance between the two concerned clan is modified so that their resulting relation is incremented by the product of the event type impact and the relevance of the event. We chose this design because it appeared to be fairly intuitive and simple to manage. Note that the update of the stance can drive relations, once updated through equation (2), to adopt out of bounds values. When that is the case the stance is updated to the closest value that keeps the relation in. In the game semantics this translates as: “A clan can only hate or love another clan so much”.



**4.3.4. Satisfying the model requirements**

We have presented the elements of the model and the rules that bind them together. We can now review the requirements expressed at the beginning of this section and examine to which extent they are fulfilled by our model:

* Take into account the discrete game events that are relevant and impact the model variables according to both the type of these events and their magnitude:   
  The game events that we defined as relevant impact the model variables through the update of the stances as described in equation (3) and are then propagated to the relations and tempers through equations (2) and (1). The type of the events and their magnitude are represented by the parameters and in equation (3) which together define the proper impact the event occurrence has on the game.



* Simulate the continuous evolution of model variables over time:  
  Equation (1) defines the evolution of tempers over time, which is then reflected on the relation through equation (2).
* Provide a high-level interpretation of the state and dynamics of the game:  
  The vector offers a meaningful high-level view of the game state and dynamics, more detailed information is also easily accessible through relations and stances. These variables are not only much easier to interpret than series of low-level events but are also more meaningful in the game’s semantic domain.



* Provide a way to exert high-level control over the low-level state and dynamics of the game:   
  Because the players’ actions are partly based on the relations they share with other players, control over the low-level state and dynamics of the game can be achieved in various ways. Enforcing a particular line of conduct in a clan’s behavior is only a matter of restricting its relations variables to particular domains. Though this technique would suit most story-delivery requirements without compromising the mechanics of the game, our model also allows for the use of more subtle mechanics: By modifying the tempers or the events’ properties one can indirectly influence the relations and therefore steer the game towards or away from a particular dynamic.

**4.4. Connecting the model and the game structure together**

To enable the game structure to harness the power of our model we need to establish a connection between the two of them. In this section we present our implementation of this connection. We first detail how we connected the low-level mechanics to the model through game event, and then describe how the model data impacts NPCs game decisions.

**4.4.1. Game events**

Simple events follow the basic rules defined by equation (3) as described in section 4.3.3. In the ClanWars game the *Attack* event is fired when a clan, the event initiator, attacks another clan, the event receiver. The impact of this event on the stance of the receiver towards the initiator is determined by the predefined impact of the attack event type and the relevance of the event which in our implementation is the portion of the receivers’ total units killed by the initiator. This implementation is a good example of how the model can bring together design-time elements such as the event type impact and run-time data such as the fraction of the clan killed.

This simple mechanism can also be used to design cascading events which are naturally useful in emergent games. The event *ThirdPartyAttack* in the game represents the attack of a third party as perceived by a clan that is neither the attacker nor the attacked. With three clans A, B, and C, then if clan A attacks clan B a third party attack event will be fired from A to C. This is implemented by defining the *Attack* event previously described as the starting point of the *ThirdPartyAttack* events, this way, whenever an *Attack* event is fired the corresponding *ThirdPartyAttack* events are automatically fired as well. For the *ThirdPartyAttack* events to have a proper impact we defined the relation between the attacked clan and each the receivers of the events as the relevance of each event. By doing this we build on the basic semantics of the model to enable new gameplay mechanics: When a clan attacks another it provokes a degradation of its relations with the friends of the attacked clan but also an amelioration of its relations with the enemies of the attacked clan.

Finally some events can have an effect that supersedes the basic impact mechanism. This is the case of events that represents changes about the model variables themselves. In the Clan Wars game the *AcceptAllianceOffer* event represents the establishment of an alliance as a mutual agreement between two clans; this implicitly requires the relation variables of these two clans to be updated to match their expected behavior as allied clans. To fulfill this requirement the stances between the clans are increased to an adequate level and the corresponding relation updated.

**4.4.2. NPCs decisions**

To enable a feedback mechanism through the model as depicted in figure 1 of section 4.1, we connected the AIs decision making system to the model variables. This gives the AIs the capability of acting according to both general rules and the model which itself can be easily influenced by the designed storyline.

We have used this balanced approach in the AIs decision making system because AIs’ behaviors are the primary feedback channel to the player in Clan Wars. AI-controlled clans choose which clans to ally with and which groups to attack using relations variables and general “reductionist” rules based purely strategic considerations such as the power of the other clans and the disposition of the resources on the terrain. The importance give to either type of information can be adjusted at runtime to allow the storyline components of the game to tighten or loosen their grip on the game behavior on specific points.

1. **Experimentation**

In this section we present the results obtained through our experimental framework.

* 1. **General experimental framework**

We list in table 1 the main parameters of our experimental framework.

|  |  |
| --- | --- |
| **Parameters** | **Role** |
| Number of clans | The number of clans initially present at the start of the simulation |
| Resources | The amount of resources available per clan |
| Maximum alliances | The number of alliances a clan can initiate |
| Events impacts | The particular impact of each event on the model |

**Table 1. The basic parameters of our experimental framework.**

The Clan Wars game as a whole also includes many other parameters, such as the size of the terrain or the speed of movement given to groups. Though they can have an indirect impact on the emergent patterns that may arise from the game we decided to keep these secondary parameters fixed and to focus on the four primary parameters listed in table 1.

It is a common methodology when testing a data structure or algorithm to present a statistical analysis of the results based on enough independent runs to ensure that the derived figures are meaningful. While this methodology is the method of choice when assessing performances of any kind, it is not suited to measure the emergent dynamics and high-level interpretation and control we aim to achieve. The results presented in this section are therefore not intended as a benchmark of our model but as practical evidence of what our model in its current implementation can offer in the light of the original design problem: enabling a reunion of the reductionist and constructivist techniques in the engineering of more balanced video games.

* 1. **Gameplay emergence and high-level semantics**

Fix the number of clans (n=4 or n=5, for instance)

Foreach parameter

Show a few cases (one screenshot per case)

Summarize the subset of screenshots

* 1. **Scenarios of use**

Changing parameters: resources, alliances, events (Analysis)

Design-time interpretation (Because we can analyze then we cam design based on the observations. A tool to facilitate designers in the decision making process.)

Run-time emergent storytelling (Generating results in game: stability vs. new game play trade-off).

**6. Conclusion**

The storyline does not take away the reductionist mechanisms of the game but rather steers their behavior with an adjustable force towards the desired situation or dynamics.

Limitations in the model and in the achieved work

Need for a way to test for emergence automatically in order to perform statistical analysis.

1. In this report we refer to as “story” elements everything that is related to how the story is orchestrated and delivered to the player. It is therefore very close to the concept of script in cinema. [↑](#footnote-ref-2)