 ISBN: 978-989-20-2190-4

Ancient and Medieval Battle Simulator

Pedro Moraes Vaz, Pedro A. Santos, Rui Prada

Instituto Superior Técnico

pedromvaz@gmail.com, pasantos@math.ist.utl.pt, rui.prada@gaips.inesc-id.pt

Resumo

O objectivo deste trabalho foi desenvolver uma *framework* que permitisse aos jogadores passarem pelo papel de um general em batalhas da antiguidade ou medievais. A *framework* permite aos jogadores criarem planos tácticos que descrevem como os seus exércitos vão ser colocados no campo de batalha, como se vão comportar, e depois permite que os jogadores vejam as batalhas nas quais os seus exércitos participaram.

Palavras-chave

Batalha, Medieval, Antiga, Táctica, Simulação

Abstract

The objective of this work was to develop a framework that allows players to play the role of a general in ancient and medieval battles. Players create tactical plans describing how their armies will be placed in the battlefield, how they will behave, and then players watch the battles in which their armies participated.

Keywords

Battle, Medieval, Ancient, Tactics, Simulation

Introduction

Various games have been made where they try to reproduce ancient and medieval battles, both videogames and board games. We can divide these games into three categories: games where players can control their soldiers during the battle by moving them around and issuing orders, games where players can choose where to place their soldiers and issue one order each before the battle starts, having few or no choices during that battle, and games where players can only decide what types of soldiers they have in their armies, and how many, while having no control during the battle, because it is processed automatically.

Our main goal was to create a framework where players could create their own tactical plans independently (i.e. without attaching them to a specific army), and then use them in any battle, with whatever army was available. These tactical plans would define where to place specific categories (e.g. Infantry, Cavalry and Artillery) of soldiers in the formation, where to place the strongest or weakest, and which orders they would follow during the battle. Apart from the tactical plans, players would have no control over a battle, so battle simulations would have to rely on autonomous agents to control each contingent (i.e. group of soldiers). These battles would then be shown to the players involved.

Before creating this framework, we analyzed seven different games to learn what kinds of choices players had in order to control their armies in the battlefield, before and during the battle. We also read about some historical ancient and medieval battles described in books, because we wanted our framework to be able to reproduce most of these battles. Our research showed that these games either gave players too much control during a battle, or too few options when planning their tactics for the upcoming battle. We wanted our framework to be a middle ground between the two: allowing players to plan their tactics before a battle took place, while giving them a wide array of options.

Framework Architecture

Figure 1 presents our framework's architecture, to give the reader an idea of its components and how they are connected to each other. Figures 2 and 3 show the two Graphical User Interfaces (or GUIs) that we designed to help players use our framework.

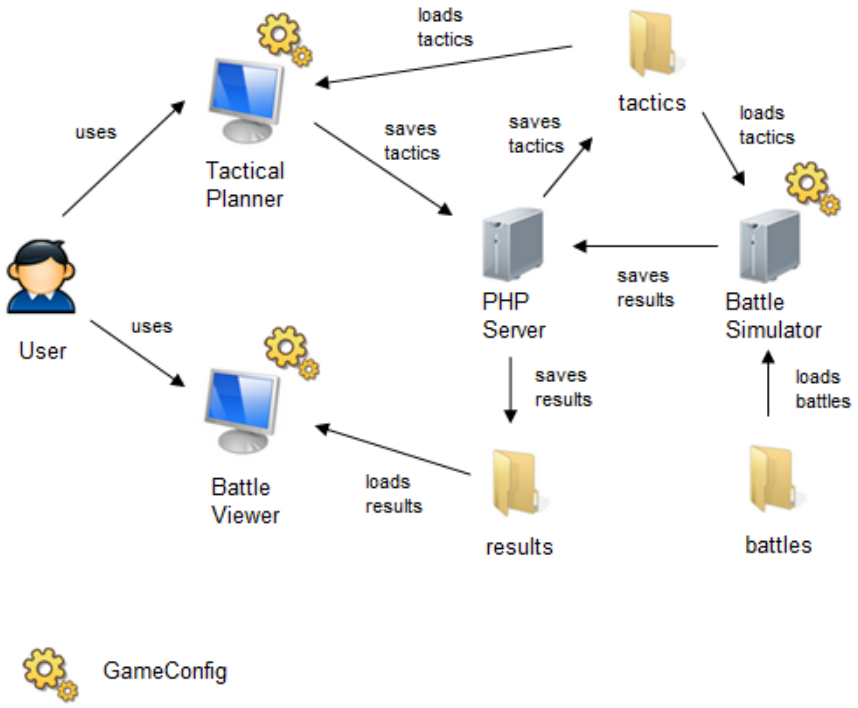

Figure 1 - The Framework architecture

The Tactical Planner is a GUI that allows players to create and save new tactical plans for their armies, as well as load existing ones. We created various prototypes until we concluded that our players could create rich and varied tactical plans with ease. The Battle Viewer is a GUI that allows players to view the battles in which their armies participated,

battles that have already taken place (i.e. that have already been simulated). The Battle Simulator simulates every battle between two armies, and then stores it so players can watch it afterwards. Each army has a specific tactical plan in that battle, which the Simulator must load. The GameConfig loads all the game configurations. This way, we can change the game elements (weapons, contingent categories, contingent types, damage types) in one place and it affects the whole framework.

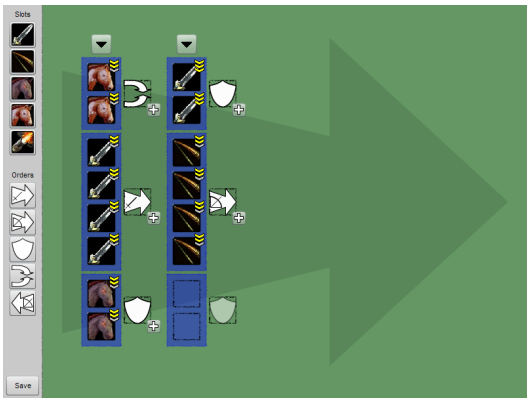

Figure 2 - The Tactical Planner

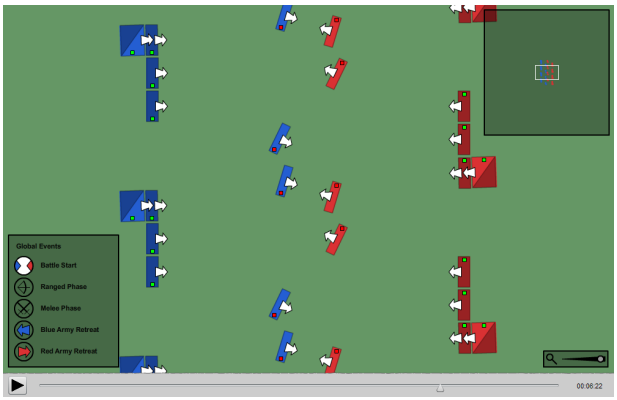

Figure 3 - The Battle Viewer

The PHP Server is needed because this whole framework was developed in Adobe Flash, and storing any information created in Flash requires connecting to a server, which then handles the necessary files.

Battle Conceptual Model for the Game

This conceptual model is based on concepts present in (Devries et al., 2006) and (Keegan, 1976). The player plays the role of a **General**, who is in charge of commanding his **Armies** into **Battle**. A battle is a confrontation between two armies, each lead by a different general. Each army is made of one or more Army **Contingents**, which are groups of soldiers with the same weapon, armor and experience. Contingents are grouped into **Categories** that define their role in the battlefield.

In our framework, the way the general (i.e. the player) commands his armies is by creating tactical plans (or **Tactics)** that can be used in any battle and with any army. Players define the army's **Formation** in each tactical plan, which determines where any available contingents will be placed in the battlefield. A formation is divided into **Sectors** (Center, Left Flank and Right Flank), each having one or more **Sector Lines**. Sector lines in the center have four **Slots**, while in the flanks they only have two, as seen in Figure 2). Each slot may be

assigned a contingent category and a **Rank**. Ranks allow the general (i.e. the player) to determine where to place the strongest or the weakest contingents, inside the formation.

Each sector line may be assigned one or more **Orders**. Orders are associated to Battle **Events**, and are only executed when those events occur. Events can either be **Global** (concerning the whole army) or **Local** (concerning specific sectors). Every Event-Order pair is called a **Rule**, and each sector line may have up to five rules with different **Priorities**. Rules with a higher priority will be executed first. The whole Conceptual Map is shown in Figure 4.

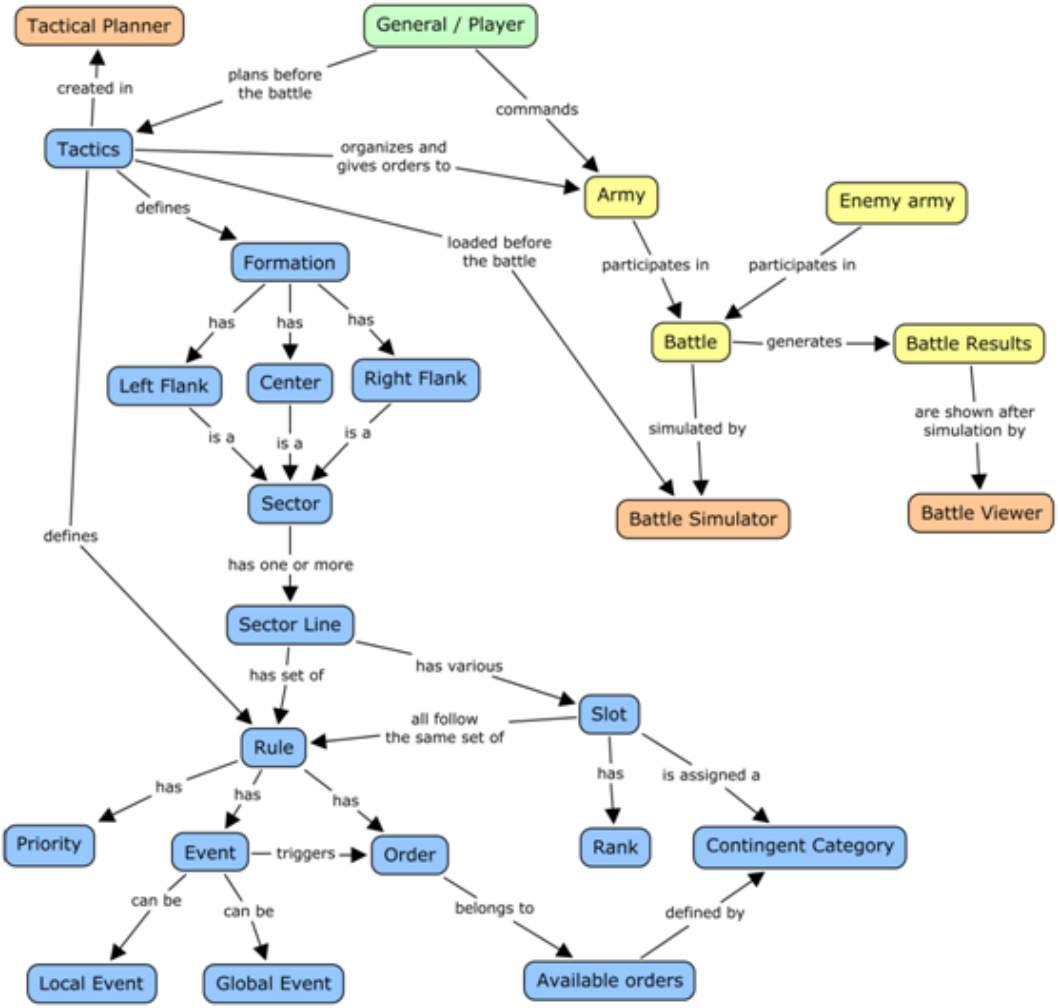

Figure 4 - The Conceptual Map

Orders and Events

Currently, there are six Orders and five Global Events implemented in our framework. The orders are as follows:

- **Defend** - agents stay in their position and engage the enemy when in range

- **Melee Attack** - agents move towards the enemy and engage in melee combat
- **Ranged Attack** - agents move towards the enemy and engage in ranged combat
- **Skirmish** - agents engage the enemy in ranged combat but keep a safe distance
- **Flank Attack** - agents move around the battlefield and attack the enemy from its side
- **Retreat** - agents run away from the battlefield

The global events are the following:

- **Battle Start** - triggered when the battle starts
- **Ranged Phase** - triggered when the first arrow is fired
- **Melee Phase** - triggered on the first shock between contingents
- **Friendly Retreat** - triggered when a friendly contingent runs away
- **Enemy Retreat** - triggered when an enemy contingent runs away

Battle Simulator

In order for a battle to be simulated, the Battle Simulator must load three files: the battle itself (containing information about every battle contingent on both sides) and two tactical plans (one for each side). After all the files are loaded, the battle contingents are placed on the battlefield according to the formation in each tactical plan. Currently, all battlefields resemble a grassy plain, with no roads, debris, hills, mountains or rivers.

During the battle, the simulator is basically a cycle that covers all the agents (i.e. contingents) from both sides and gives them a chance to analyze their environment (i.e. the battlefield) and decide what to do, according to the orders assigned to them in their tactical plan. The battle state -- consisting of the contingents' positions, actions, targets and status -- is stored after each step of the cycle is complete, for later viewing on the Battle Viewer.

A battle ends when all the contingents from one side are either destroyed or outside the battlefield. When the battle ends, the cycle stops and the battle is stored, so the Battle Viewer can then show that battle to the players involved.

Placing the contingents on the battlefield

Since we wanted all tactical plans to be independent from any army, we needed an algorithm to place any specific army on the battlefield, according to the formation defined in its tactical plan for that battle. This algorithm consisted of three steps: distributing all army contingents into the category-assigned slots in the formation, calculating how many blocks were needed for each slot, and aligning the slots on the battlefield.

A block is a 50x50 meter area where only one battle contingent can fit, and the battlefield is a grid of these blocks. When an army contingent is distributed into the category-assigned slots, it must then be split into one or more blocks in the battlefield, depending on its size, because blocks have a maximum number of soldiers allowed. Each part of that now-split army contingent is called a Battle Contingent, and is controlled by an autonomous agent during the battle.

Assigning the army contingents to the category-assigned slots consists of three sub-steps. Consider we have an army with 2000 Archers, 3000 Skirmishers and 6000 Swordsmen, and a formation where the slots in both flanks are assigned the Ranged Infantry category. The slots in the left flank have the highest rank, and the ones on the right flank have the lowest rank. Finally, the slots in the center are assigned the Melee Infantry category.

The first sub-step is to group all the army contingents by categories: this means we will group the Archers and the Skirmishers as Ranged Infantry, and the Swordsmen as Melee Infantry. The same grouping will be applied to the slots in the formation, so the slots in the flanks will be grouped as Ranged Infantry, and the slots in the center as Melee Infantry.

The second sub-step is to order the army contingents in each group from strongest to weakest, and order the slots in each group from highest rank to lowest rank. For this example, we will consider that the Archers are stronger than the Skirmishers. The Ranged Infantry slots in the left flank will come first (because they have the highest rank) and the slots on the right flank will come last (because they have the lowest rank).

The third sub-step is to equally distribute the army contingents from each category into the slots of the same category, in their current order, so that each slot ends with roughly the same number of soldiers assigned. Figure 5 shows the distribution for the Ranged Infantry category. The distribution for the Melee Infantry category is straightforward.

Consider that each block on the battlefield can have a maximum of 1000 soldiers. Figure 5 shows how many blocks each slot needs, and how many soldiers stay in each block. Since the slots in the tactical plan are shown side by side, the blocks in the battlefield will be side by side as well, so the formation in the battlefield remains true to the tactical plan. In the end, the formation in the battlefield will look like this (from left to right): 2 blocks of 625 archers, 1 block of 750 archers, 1 block of 500 skirmishers, 8 blocks of 750 swordsmen, and 4 blocks of 625 skirmishers.

Slots	Left Flank Slot 1 (Rank 3)	Left Flank Slot 2 (Rank 3)	Right Flank Slot 1 (Rank 2)	Right Flank Slot 2 (Rank 2)

Battle Contingents	1 250 Archers	750 Archers + 500 Skirmishers	1 250 Skirmishers	1 250 Skirmishers
Blocks	2 blocks of 625 Archers	1 block of 750 Archers + 1 block of 500 Skirmishers	2 blocks of 625 Skirmishers	2 blocks of 625 Skirmishers

Figure 5 – Distribution of 2000 Archers and 3000 Skirmishers into four Ranged Infantry slots

Agents

Every battle contingent on the battlefield is controlled by an Agent. For this work, we used state-based agents, because we needed them to know which orders they were assigned and what they were doing at any given moment, in order to have a coherent behavior as the battle progressed. These agents are defined in (Wooldridge, 2002).

Every battle contingent has a list of attributes that define its role on the battlefield, and how well it will do in battle: the number of soldiers in it, their speed, combat experience, cohesion (or morale), which weapons they use (and their damage), armor and tiredness.

Agents in a game usually act in one of a limited set of ways and they will carry on doing the same thing until some event or influence makes them change. According to (Millington, 2006), State Machines are the best technique for this purpose, as they were designed for it. Since we used state-based agents in our framework, we defined their behavior in a State Machine, which is shown in Figure 6.

Every order a player can issue in a tactical plan will only affect an agent's Moving State or send the agent into the Retreat State. The remaining states concern combat -- Ranged State, Shock State and Melee State -- and players cannot predict when their contingents will engage the enemy. Players can only control where their agents move to. Ranged combat is started when a contingent using ranged weapons gets in range with an enemy contingent. Shock and melee combat start when two enemy contingents collide against each other.

Every agent (i.e. battle contingent) avoids colliding against friendly contingents when moving, because contingents in ancient and medieval battles were in a tight formation (soldiers were close to each other), which left no space for another contingent to move through it. Also, friendly contingents keep a distance from each other, so they can reach their target without being blocked. The only exceptions are when a contingent is skirmishing (moving away from its target) or retreating from the battlefield, because in these situations the contingent is in a loose formation, meaning its soldiers have some space between them.

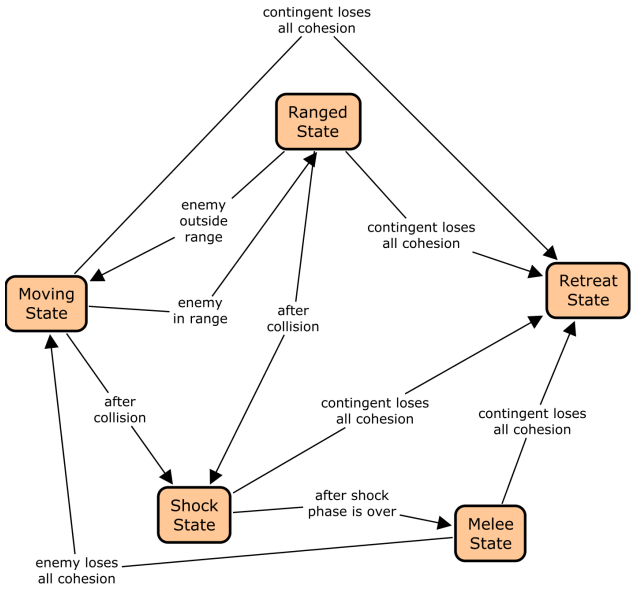

Figure 6 – The Agent's State Machine

When an agent chooses a target to engage, it takes various parameters into consideration: the distance to the target, if the target is obstructed by other agents (i.e. contingents), the angle of rotation needed to face the target, if the target is retreating from the battle, if the target is already attacking the agent, and if it was the agent's last chosen target.

There are two types of damage in combat: physical damage and cohesion (or morale) damage. Any form of physical damage inflicted on a contingent also inflicts cohesion damage. For instance, when two contingents are in melee (i.e. hand-to-hand) combat, soldiers will die in the process, and the remaining soldiers will start losing their will to fight, which affects their cohesion. When a contingent is retreating from the battlefield, it inflicts cohesion damage to nearby friendly contingents, because seeing a contingent flee will cause some panic in the remaining contingents.

Priority	Event	Order
1	Battle Start	Defend
2	Ranged Phase	Melee Attack
3	Friendly Retreat	Retreat

Figure 7 – Three rules assigned to an agent

Agents can react to the events that occur during a battle. Consider Figure 7, where an agent is assigned three Rules to follow. When the battle starts, the agent will defend its position, due to the first rule. The second and third rules do not apply, because the ranged

phase has not started and no one has retreated. When the ranged phase starts, the agent will start a melee attack, because the second rule can now be executed and it has a higher priority than the first. Finally, when a friendly contingent retreats, the agent will retreat as well, because of rule 3, which is of the highest priority.

Results

After completing the framework, we playtested it with fifteen subjects with ages ranging from 25 to 46. All the subjects were observed during the tests and filled a survey in the end, a usual playtesting process explained in (Schell, 2008). The subjects were divided into three groups: the first group was made of five inexperienced players in this genre, who were given no prior information about the framework, the second group was made of five experienced players in this genre, who were also given no prior information about the framework, and the last group was made of five inexperienced players who were told how the GUIs in the framework worked. This way we could compare the results from three different situations.

The test consisted of showing each subject a pre-simulated battle where both sides used the same tactical plan and had similar armies. The subject's army -- slightly smaller and weaker -- lost the battle and, after watching it, each subject was asked to find a new tactical plan that he thought would defeat the enemy army.

From the Battle Viewer surveys (related to the first battle), we found out that three of the subjects from Group 1 had difficulties recognizing the various contingents on the battlefield (swordsmen, archers and knights), while the remaining subjects from the three groups did not. Only two subjects from Group 1 had difficulties understanding some (or all) of the actions performed by the contingents during the battle. Thirteen of the subjects found the agents' behaviors acceptable, one from Group 3 found them realistic and one from Group 1found them unrealistic.

From the Tactical Planner surveys, we found out that eleven of the test subjects understood how every order worked in the battlefield (three from Group 1 and Group 2, and five from Group 3). Only one subject from Group 1 used the ranks available in the Tactical Planner, while Group 2 had three subjects who used it and Group 3 had another three. Group 1 had two subjects who used the global events, Group 2 had three and Group 3 had five. Still, we noticed that global events were not as straightforward to use as we hoped, since they cover

the whole battle (which may be unpredictable), and players usually prefer to make more localized tactical decisions.

From the Battle Viewer surveys (related to the second battle), we found out that twelve of the subjects felt the agents executed their tactical plan most (or all) of the time. Only two of the subjects (one from Group1 and one from Group 2) felt that the second battle did not have a better outcome than the first.

Conclusions

From the survey results, we concluded that the framework is easy for experienced players to use, since they are familiar with the concepts, and for inexperienced players who are taught how to play. Inexperienced players who are not taught how to play will have some problems understanding and using the concepts involved in ancient and medieval battles and tactics. Therefore, our intention is to create tutorials to help new players learn how to use our framework, by explaining the concepts and showing some practical examples.

We planned on having global (army-related) and local (sector-related) events available in the framework, but only the global events were implemented, which significantly limited the number of events our agents can react to in a battle. We are aiming to add local events in the future, since this is the most natural way for players to use events. We are also planning on adding terrain features to the battlefield, like rivers and hills, which will increase the level of tactical thinking. These features may be used by a weak army to its advantage; it all depends on the general's (i.e. the player's) experience.

References

Devries, K., Dougherty, M., Dickie, I., Jestice, P., & Jorgensen, C. (2006). *Battles of the Medieval World.* London: Amber Books.

Keegan, J. (1976). *The Face of Battle.* London: Penguin Books Ltd.

Millington, I. (2006). *Artificial Intelligence for Games.* San Francisco: Morgan Kaufmann Publishers.

Preece, J. (2007). *Interaction Design: beyond human-computer interaction.* John Wiley & Sons.

Schell, J. (2008). *The Art of Game Design: A Book of Lenses.* Burlington: Morgan Kaufmann.

Wooldridge, M. (2002). *An Introduction to MultiAgent Systems.* West Sussex: John Wiley & Sons Ltd.