Звіт

Навчальний проект, Лабораторні роботи 2, 3.b

Тема проекту: моделювання битв. Робота з програмою відбувається через зручний GUI, написаний з допомогою Qt. Моделювання відбувається у дискретному часі, із симуляцією неперервного за допомогою технологій багато поточності. На основі результату моделювання будується графік. Проведенні симуляції дають результат наближений до очікуваного у неперервному часі. Проведено benchmarking. Виявлено, що при використанні 1 потоку(число однакове для к-ті потоків для юнітів і для тиаів юнітів) симуляція для 2 армій розміром близько 300 юнітів 4 різних типів працює в середньому 0.38с, при використанні 2 потоків 0.2с, і 4 потоків 0.4с, що пояснюється високими затратами на обробку такої кількості потоків. Найбільш раціональним в даному випадку є дати програмі самостійно вирішувати, яку кількість потоків їй створювати.

При розробці було використано такі бібліотеки:

* Boost::ut – для unit-tests.
* freeglut-3.2.2­ – необхідна для роботи бібліотеки з будування графіків.
* easy\_plot\_cpp-master - для будування графіків.
* Intel TBB – для роботи з багато поточністю.
* Qt creator 12.0.2 – для написання GUI.

Аналогами розробленого застосунку є спеціалізовані військові програми для моделювання подій на полі бою, які суттєво випереджають розроблену програму, програми для моделювання битв в іграх (Heroes of Might and Magic III, Medieval: Total War, Crusader Kings II, тощо), які серед переваг зазвичай мають красиву візуалізацію та працюють більш ефективно через високий рівень спеціалізації, проте оскільки вони використовуються у іграх, то працюють над раніше визначеними типами юнітів. У наведеній реалізації є можливість додавати нових юнітів, змінювати їхні властивості додаючи до них items, та впливати на хід битви задаючи на початку спеціальні події (наприклад, туман, дощ...). Також присутня можливість додавання до армій підкріплень, та поставок припасів, чого, зазвичай, немає в інших моделюваннях. Також серед аналогів можна визначити деякі інші учбові системи, які написані для одного(декількох) типу юнітів із вказаним(и) коефіцієнтом скількох противників вони перемагають за одиницю часу, і готовим рішенням систем диференціальних рівнянь для нього. Перевагою таких систем є більша точність, недоліками є необхідність кожного разу складати та вирішувати нові системи диференціальних рівнянь при зміні кількості типів юнітів, причому чим більше типів, тим складніші системи утворюються.

Паттерни

1) Реалізовано паттерн singleton для класу BattleModeling.

2) Реалізовано паттерн builder для класів BattleModeling, Unit, MoralUnit.

3) Реалізовано паттерн factory method у класі FileManager для зчитування юнітів з файлу в залежності від типу.

4) Реалізовано паттерн prototype у вигляді інтерфейсів Cloneable і Prototype. Для класів Unit і MoralUnit написана реалізація, у якій шаблонний метод clone повертає поліморфний покажчик Unit\* на новостворену точну копію об’єкта, а метод create повертає поліморфний покажчик Unit\* на новостворений об’єкт із там же типом об’єкту та значеннями unitType та priorityTarget як і початковий об’єкт.

5) Реалізовано паттерн iterator для класу Army.

6) Реалізовано паттерн memento для класу Army.

7) Реалізовано паттерн template method для класів Unit, MoralUnit.