**Modelowanie i analiza systemów informatycznych:**

Systemy agentowe

Dokumentacja gry „Wojownicy”

Autor:

Krystian Świdurski

# Założenia i opis gry

Celem projektu jest zaprojektowanie gry Wojownicy w postaci systemu wieloagentowego przy użyciu metody O-MaSE oraz jej implementacja w środowisku JADE.

Zasady gry:

## 1.1 Wojownicy

Zadaniem wojowników poruszanie się po labiryncie korytarzy i walka z innymi wojownikami, bądź też potworami w celu zdobycia skarbu. Skarb można zdobyć na dwa sposoby – pokonując przeciwnika, lub pokonując potwora strzegącego skarbu. Należy zaznaczyć, że wojownicy nie współpracują ze sobą, każdy działa na własną rękę.

Wojownicy mogą różnić się między sobą poniższymi parametrami:

* **życie** – definiuje liczbę punktów życia, z którą wojownik rozpoczyna grę. Punkty życia wojownika regenerują się automatycznie - prędkość regeneracji wynosi 1 punkt życia na sekundę.
* **maksymalne obrażenia** – definiuje maksymalną liczbę punktów życia odbieranych przeciwnikowi w trakcie jednej tury walki. Wojownik zadaje obrażenia maksymalne z prawdopodobieństwem 40%, oraz połowę obrażeń maksymalnych z prawdopodobieństwem 60%. Ilość zadawanych obrażeń jest odejmowana od aktualnego stanu punktów życia przeciwnika.
* **prędkość** – definiuje odstęp czasowy między kolejnymi ruchami wojownika. Prędkość jest wartością z przedziału 1 – 60, gdzie 60 oznacza jedną sekundę, a 1 aż 60 sekund przerwy między kolejnymi ruchami.
* **położenie ­­**– zdefiniowane jest jako para liczb x oraz y. Obie te liczby muszą należeć do przedziału 0 – 9. Położenie definiuje aktualną pozycję agenta na mapie.

Użytkownik określa wszystkie te parametry poprzez przyjazny interfejs użytkownika pojawiający się w trakcie dodawania nowego agenta.

Wojownik podejmuje decyzje o kolejnych akcjach bazując na informacjach o otaczającym go świecie – w tym celu wykorzystuje otrzymywaną co krok od zarządcy mapy listę z informacjami o sąsiadujących z nim polach oraz listę odwiedzonych wcześniej pól plus. W podejmowaniu decyzji uczestniczy również procentowy wskaźnik posiadanego zdrowia obliczany z poniższego wzoru:

**s = hp / hp\_max**

**s** - szansa na wygraną

**hp** – ilość punktów życia,

**hp\_max** – maksymalna (startowa) ilość punktów życia

Jeżeli wynik jest wyższy od 0.4 to szanse na wygraną są wysokie i wojownik kontynuuje walkę, bądź poszukiwania skarbu, jeżeli wynik jest w przedziale 0.4 - 0.2 to wojownik podejmuje decyzję o ucieczce, lub nie podejmuje walki. Przy wyniku niższym od 0.2 wojownik rozpoczyna poszukiwania apteczki. Apteczka regeneruje 100 punktów, lecz maksymalnie do poziomu **hp\_max**.

## 1.2 Potwory

Potwory – agenci pilnujący skarbów. W odróżnieniu od wojowników potwory nie poruszają się po labiryncie, nie walczą ze sobą nawzajem i nie uciekają – strzegą skarbu aż do śmierci. Potwór podobnie jak wojownik opisany jest parametrami maksymalnych obrażeń i życia, które mają dokładnie takie samo znaczenie. Ze względu na brak możliwości ucieczki i wykorzystywania apteczek punkty życia potwora regenerują się trzy razy szybciej niż ma to miejsce w przypadku wojownika.

Użytkownik dodając agenta dodaje również skarb w danej lokalizacji – wartość skarbu wynosi zawsze 10.

## 1.3 Zarządca gry

Zarządca gry – dodatkowy agent służący do informowania wojowników i potworów o stanie mapy, pośredniczeniu w walce oraz wizualizacji przebiegu gry.

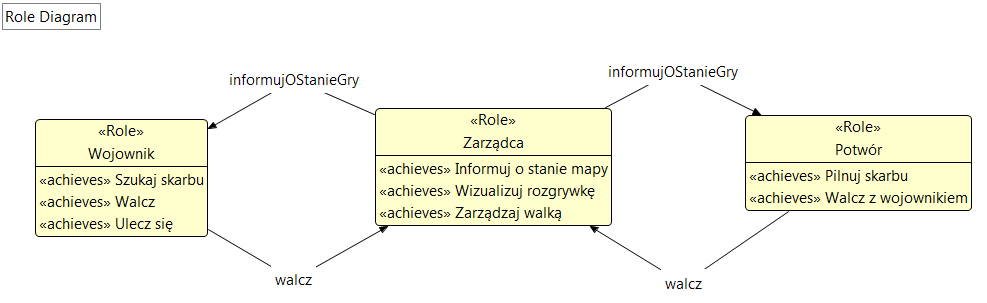
Zarządca gry oczekuje na wybór mapy przez użytkownika, a następnie przechodzi w tryb oczekiwania na dołączenie do gry kolejny agentów wojowników i potworów.

# Diagramy

***Rysunek 1.*** *Diagram celów.*

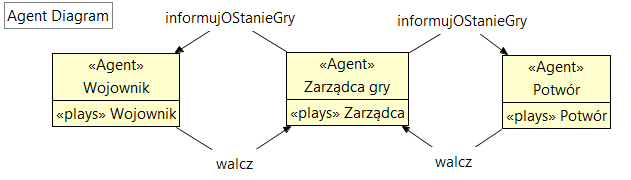
Diagram przedstawia cel nadrzędny, którym jest prowadzenie rozgrywki oraz trzy główne cele podrzędne dla trzech klas agentów.

Celem agenta-wojownika jest poszukiwanie skarbów oraz przeżycie, wojownik w trakcie poszukiwania skarbów może być zmuszony do walki i w związku z tym również do poszukiwania apteczki. Agent-potwór realizuje cel obrony skarbu tzn. strzeże go przed wojownikami, co może prowadzić do konieczności stoczenia walki. Ostatnią grupą są cele zarządcy gry, który wizualizuje jej przebieg, informuje pozostałych agentów o stanie gry oraz zarządza walką.

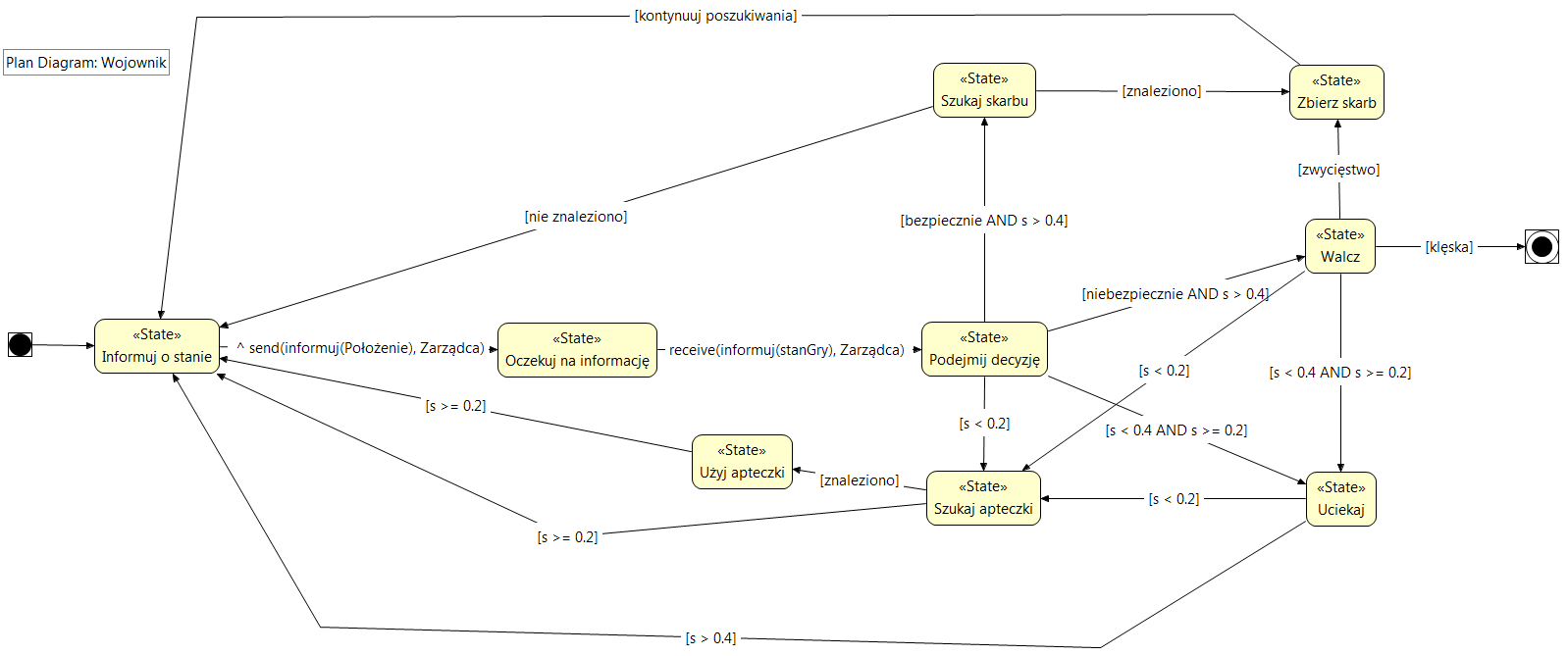


***Rysunek 2.*** *Diagram ról.*

Diagram ról przedstawia trzy role pełnione przez agentów wraz z ich celami szczegółowymi przedstawionymi na ***Ryskunku 1.*** Na diagramie przedstawiono również protokół związany z obsługą walki oraz protokół obsługujący wymianę informację o stanie gry.

**

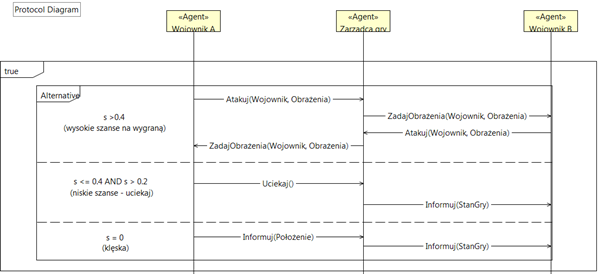
***Rysunek 3.*** *Diagram agentów przypisujący poszczególnych agentów do pełnionych ról.*

**

***Rysunek 4.*** *Diagram planu agenta wojownik.*

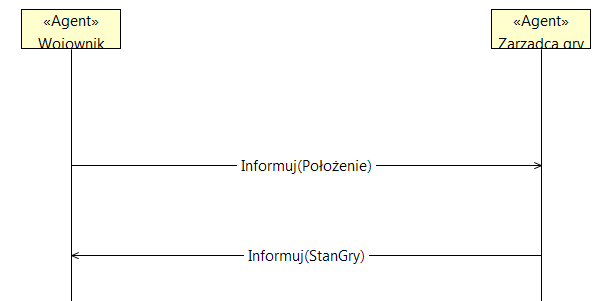
Na diagramie przedstawiono przejścia między stanami dla agenta wojownika. Agent rozpoczyna nową sekwencję od przesłania zarządcy mapy informacji o swoim wewnętrznym stanie (położeniu, punktach życia itp.), a następnie oczekuje na odpowiedź zwrotną zawierającą informacje o stanie mapy w jego sąsiedztwie. Agent dysponując danymi o stanie mapy w jego najbliższym otoczeniu podejmuje decyzję, do którego z kolejnych stanów przejść. W tym celu wykorzystywany jest m.in. wzór przedstawiony w punkcie 1.1 oraz informację, czy położenie, w którym się znajduje jest bezpieczne. Przez położenie bezpieczne rozumiemy brak innych wojowników i potworów w zasięgu wzroku wojownika.

Wojownik poszukuje skarbu, gdy wartość parametru s jest większa od 0.4 i w polu, na którym się znajduje nie ma innego wojownika, bądź potwora. Gdy s jest większe od 0.4 i w pobliżu wojownika znajduje się przeciwnik wtedy wojownik rozpoczyna walkę. Walka toczy się do momentu, aż jeden z przeciwników przegra lub ucieknie. Wojownik ucieka, gdy parametr s znajduje się w przedziale 0.2 – 0.4, lub szuka apteczki gdy s spanie poniżej 0.2 – w takim przypadku agent będzie poszukiwał apteczki tak długo aż ją znajdzie, bądź jego punkty życia zregenerują się do ponad 20%.

**

***Rysunek 5.*** *Diagram przedstawiający protokół walki.*

Na diagramie przedstawiono kilka bardzo prostych protokołów będących de facto wymianą pojedynczych komunikatów – pozwoli to zobrazować zależności między nimi.



***Rysunek 6.*** *Diagram przedstawiający uzyskiwanie informacji o stanie gry.*

Na diagramie przedstawiono prosty protokół wymiany informacji o stanie gry. W tym przypadku wojownik przesyła zarządcy gry informacje o swoim położeniu. W odpowiedzi zarządca odsyła informacje o stanie gry w sąsiedztwie wojownika.

# Implementacja

Projekt wykonany został przy użyciu języka JAVA w wersji 1.8, frameworku graficznego JAVAFX oraz biblioteki JADE w wersji 4.3.

Agenci – klasy wszystkich agentów umieszczono w pakiecie *pl.swidurski.jade.agents.* Kod wspólny dla agentów wojowników i potworów umieszczono w osobnej klasie abstrakcyjnej FighterAgent, dzięki czemu niektóre akcje takie jak walka i wysyłanie informacji o własnym stanie można było obsłużyć za pomocą tych samych klas zarówno dla agenta wojownika – klasa WarriorAgent - jak i agenta potwora – klasa MonsterAgent.

Akcje – wszystkie akcje agentów umieszczono w pakiecie *pl.swidurski.jade.agents.behaviours.* Trzy najistotniejsze klasy w tym pakiecie to:

* WarriorBehaviour – opisująca zachowanie agenta wojownika,
* MonsterBehaviour – opisująca akcje agenta potwora,
* InformAboutMapStateBehaviour – informująca o agentów o stanie mapy

GUI – wszystkie klasy związane z prezentacją mapy i wyświetlaniem informacji o stanie agentów umieszczono w pakiecie *pl.swidurski.jade.gui*. Znalazły się w nim zarówno klasy pomocnicze (MapGUI, WarriorGUI i MonsterGUI), jak i klasy kontrolerów (wszystkie z sufixem Controller). Dodatkowo w folderze resources/gui umieszczono pliki .fxml będące xml’owym opisem interfejsu użytkownika.

Model – w pakiecie pl.swidurski.jade.model umieszczono sześć klas opisujących model agentów wykorzystywany przez klasy z pakietu GUI i AGENTS do wyświetlania przebiegu gry. Na model składa się klasa opisująca mapę, klasy opisujące każde z pól należących do mapy oraz typy obiektów znajdujących się w każdym polu.

Mapy – zaimplementowano obsługę wczytywania map z pliku. Dwie przykładowe mapy znajdują się w folderze resources/maps i mają rozszerzenie \*.map – są to tak naprawdę pliki tekstowe o bardzo prostej strukturze.

# Uruchomienie