Piotr Kopalko Warszawa 13.06.2011

System obsługi teatru

Spis treści

Wymagania użytkownika

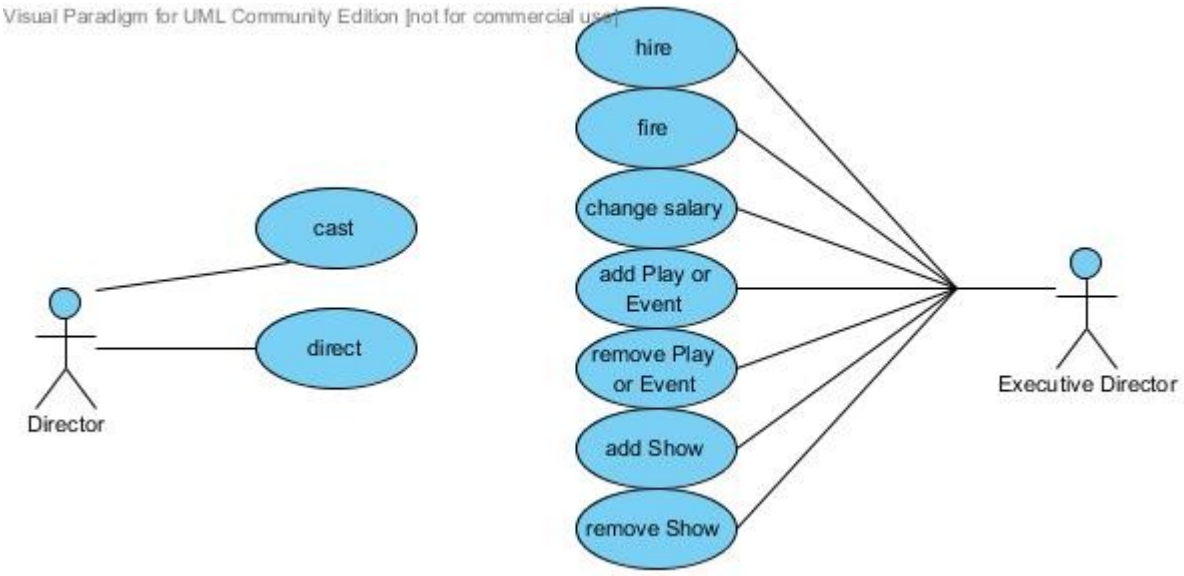
Istnieje wiele teatrów, którym brakuje sprawnego systemu zarządzania repertuarem, który pozwolił by gościom teatru na szybkie sprawdzenie repertuaru, a pracownikom w sprawnej organizacji pracy.

System taki powinien przechowywać informacje o pracownikach teatru w osobach dyrektora ,reżysera i aktora, spektaklach, sztukach i wydarzeniach i wykonawcach biorących w nich udział, salach i miejscach w nich i rezerwacjach.

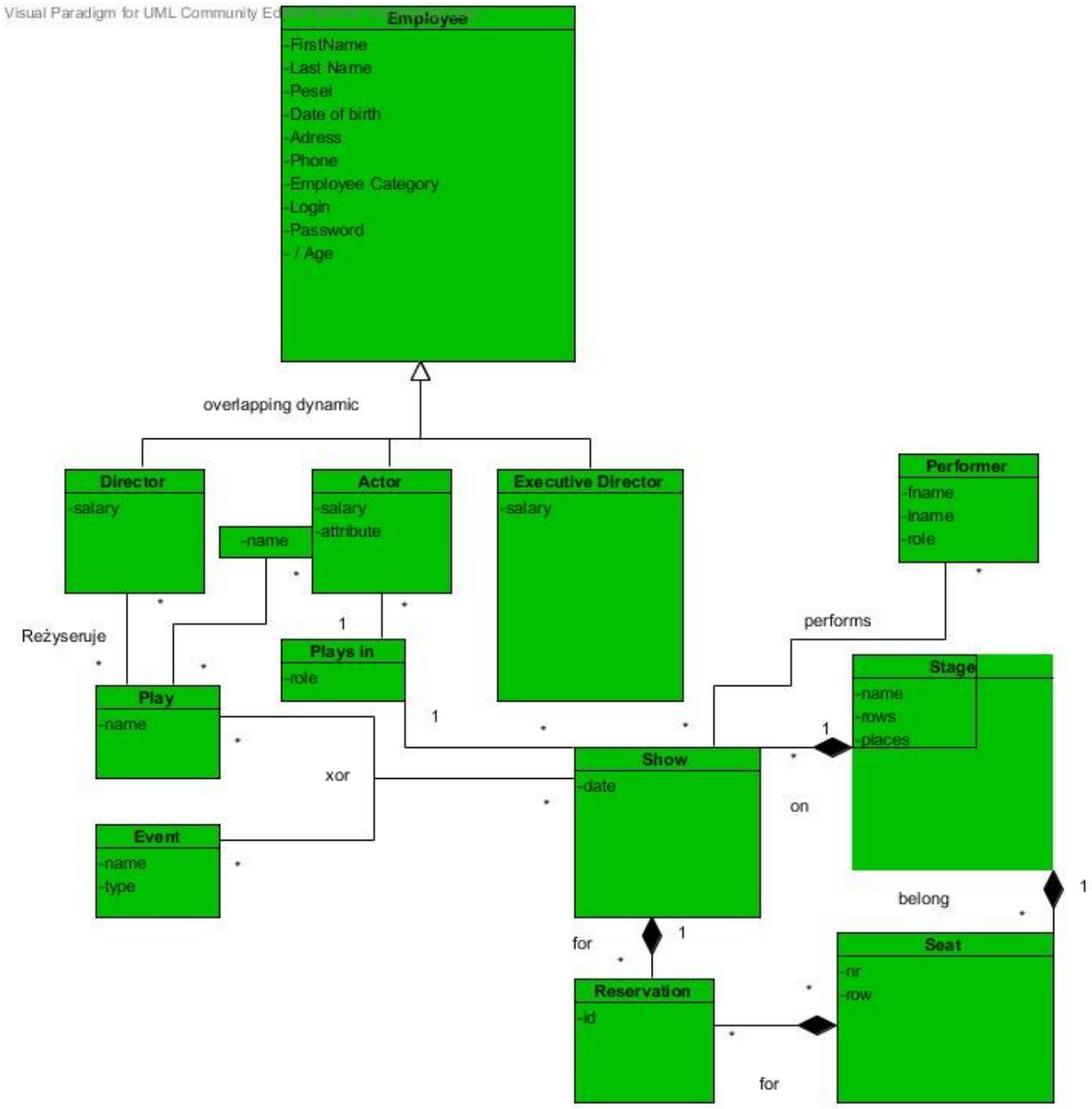
Dla pracownika powinnismy przechowywać dane osobowe, takie jak imię i nazwisko, data urodzenia, pesel, adres, telefon.Dla reżysera, aktora i dyrektora musimy przechowywać informacje o wysokości pensji. Pracownicy muszą także mieć unikalny login i hasło, aby móc zalogować się do systemu i sprawdzać informacje niedostępne dla gości.

Sztuka i wydarzenie zawierają informacje o nazwie, a dla wydarzenia powinnismy pamiętać informację o jego typie. Spektakl to konkretne wydarzenie, określonego dnia w określonym miejscu, dlatego musimy pamiętać datę, spektakl jest połączony kompozycją ze scena na której będzie się odbywał. System przewiduje też możliwość rezerwowania miejsc na konkretne spektakle. W rezerwacji musimy przechowywać informacje o osobie rezerwującej, zarezerwowanych miejscach i spektaklu, którego dotyczy. Wykonawca to osoba nie związana z teatrem, ale biorąca udział w wydarzeniu, które w tymże się odbędzie, bądź odbyło.Wydarzenie jest spektaklem nie organizowanym przez teatr, tylko zewnętrznie.

Diagram przypadków użycia



Podstawowy diagram klas.



Scenariusz przypadku użycia

Nazwa przypadku użycia	Dodanie aktora do spektaklu
Autor	Piotr Kopalko
Priorytet	Wysoki
Typ	Szczegółowy
Opis	Przypadek dotyczy obsadzenia aktora w spektaklu.Aktor nie może jednocześnie występować w innym spektaklu w tym samym czasie.
Aktorzy	Reżyser ,Dyrektor
Warunek początkowy	Spektakl musi być w systemie.
Warunek końcowy	Aktor jest obsadzony w sztuce.
Przepływ główny	1.Sprawdzenie czy aktor istnieje w systemie. 2.Sprawdzenie czy aktor nie ma innego spektaklu w tym samym dniu. 3.Obsadzenie aktora w spektaklu.
Przepływ alternatywny	1a. Aktor nie istnieje w systemie. • System prosi o wprowadzenie danych • Dyrektor wprowadza dane aktora 2a.Aktor występuje w innym spektaklu tego samego dnia. Kończy to przypadek użycia.
Dodatkowe uwagi	Brak

Diagram aktywności

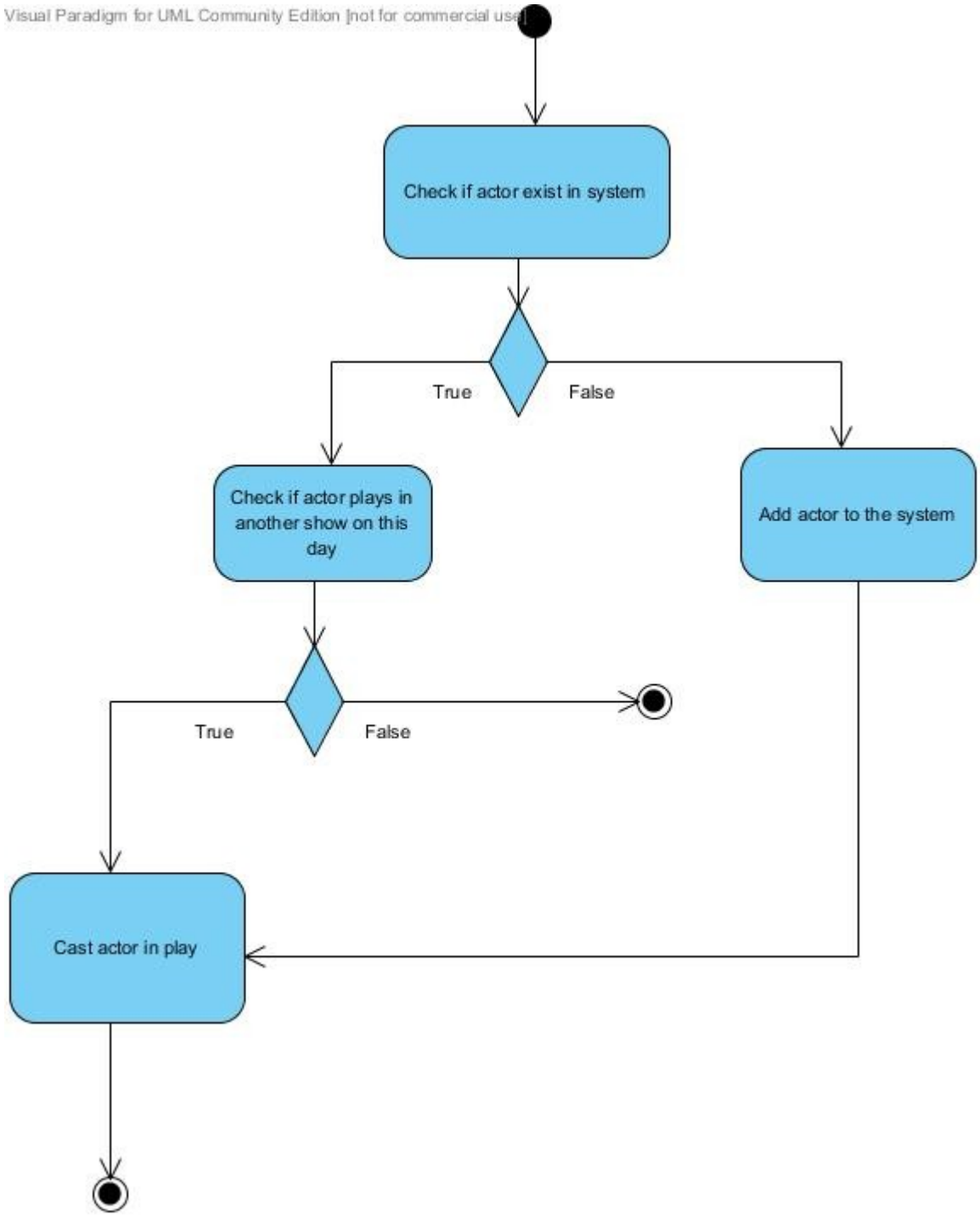
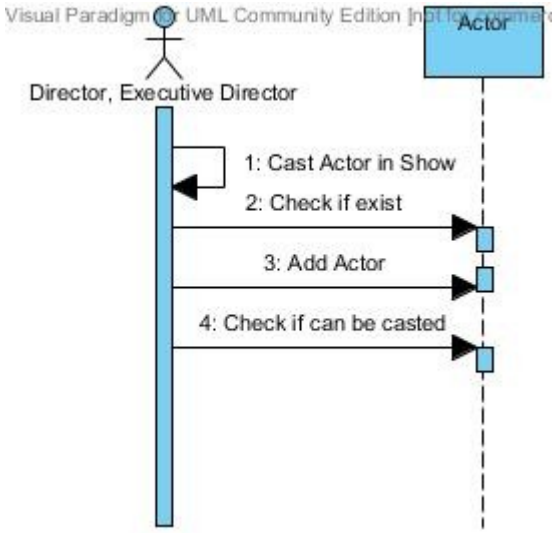


Diagram sekwencji



Dla klasy reżyser(director), niezbędna będzie metoda Cast(), która umożliwi obsadzenie aktora zgodnie z powyższymi diagramami, natomiast klasa dyrektor(ExecutiveDirector) potrzebuje metody Hire(), która umożliwi wprowadzanie do systemu nowych pracowników.

Diagram stanów dla klasy Sztuka

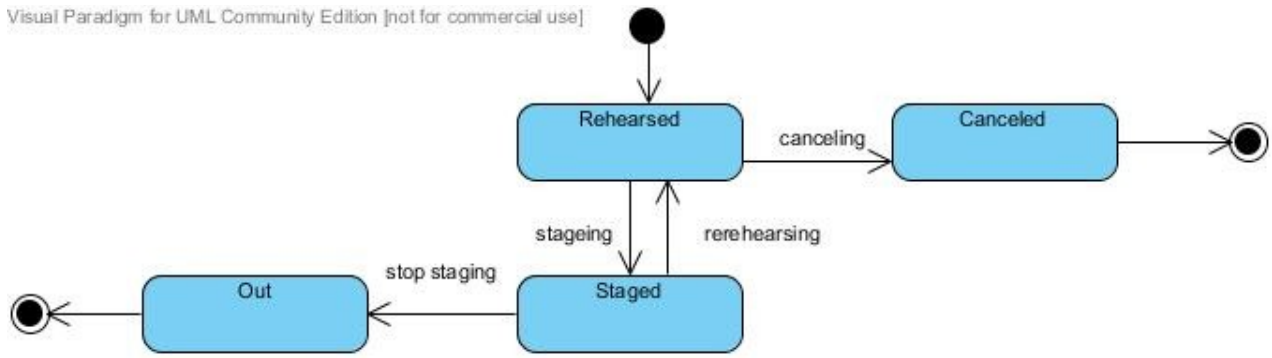
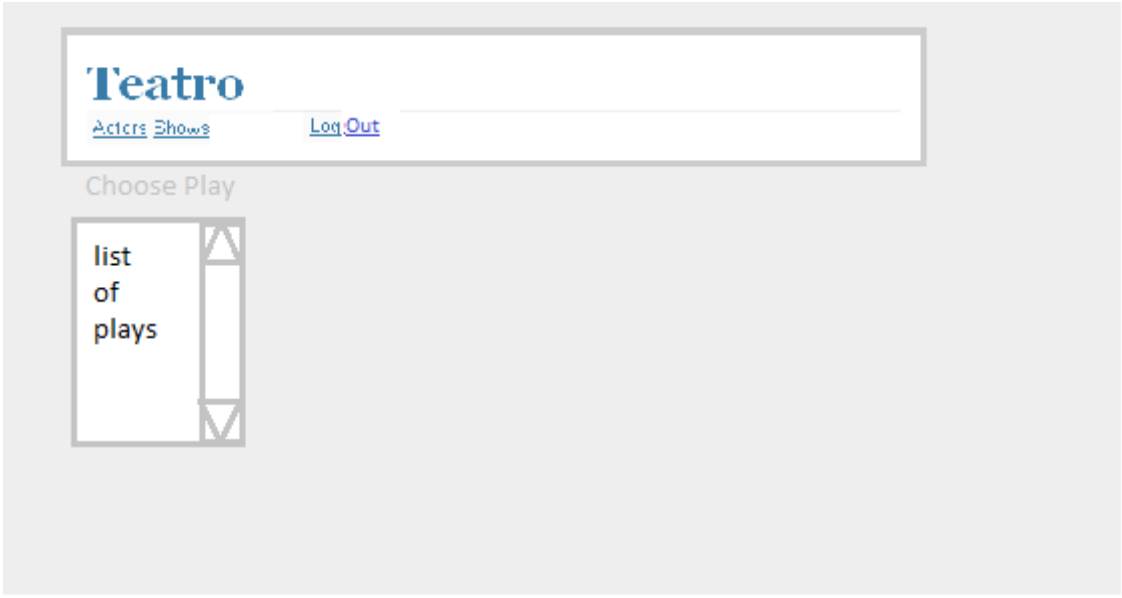
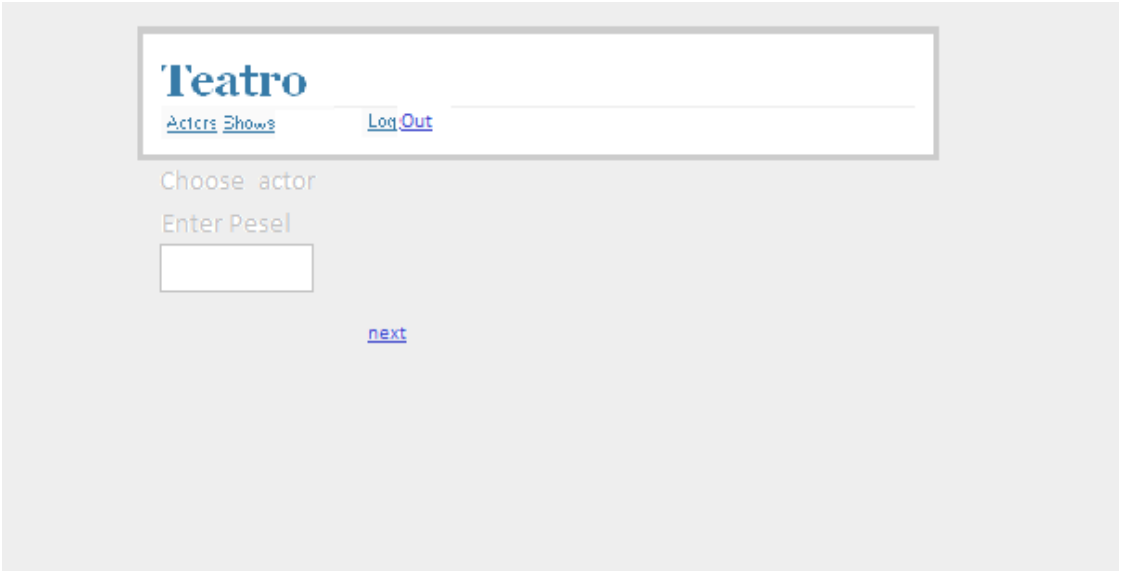
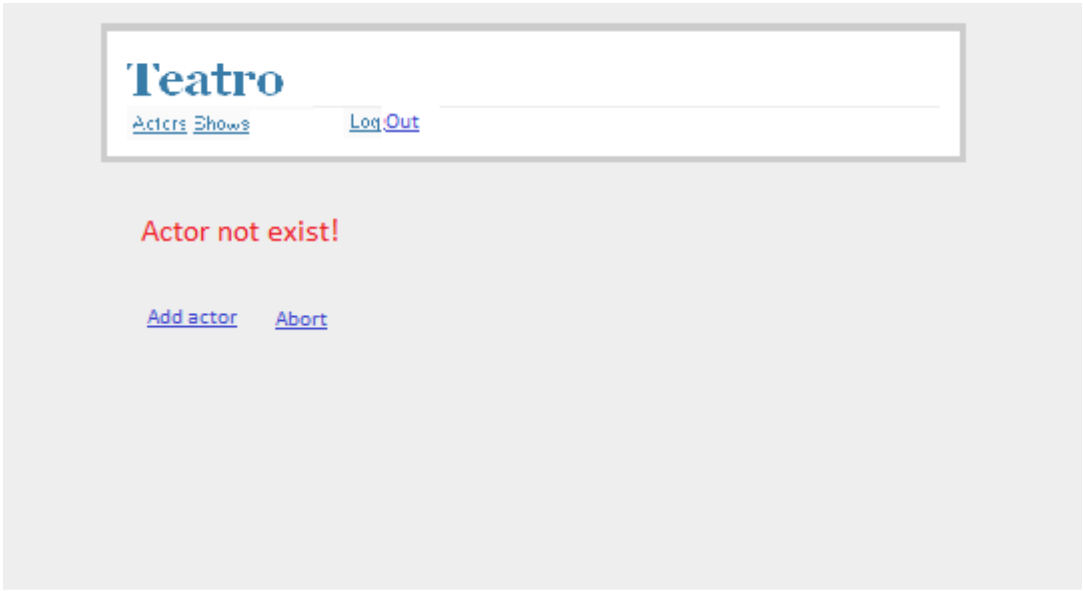


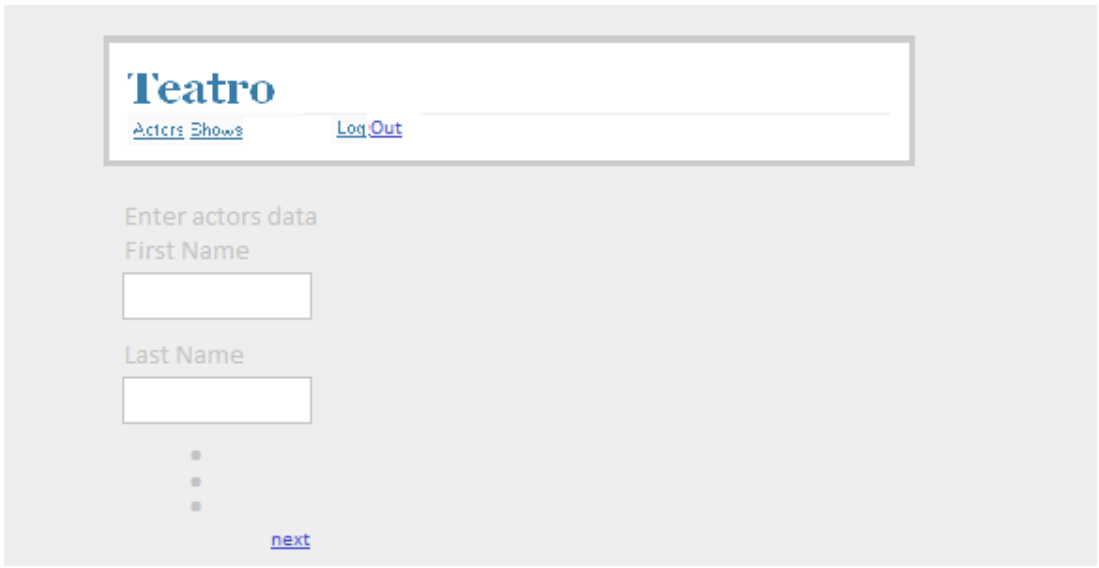
Diagram stanów stworzony dla klasy Sztuka(Play) unaocznił, potrzebę wprowadzenia do implementacji tej klasy atrybutu status i metody changeStatus(), aby było możliwe rozróżnienie między sztukami, które nie są już wystawiane, są w trakcie prób, są w aktulanym repertuarze lub zostały odwołane.

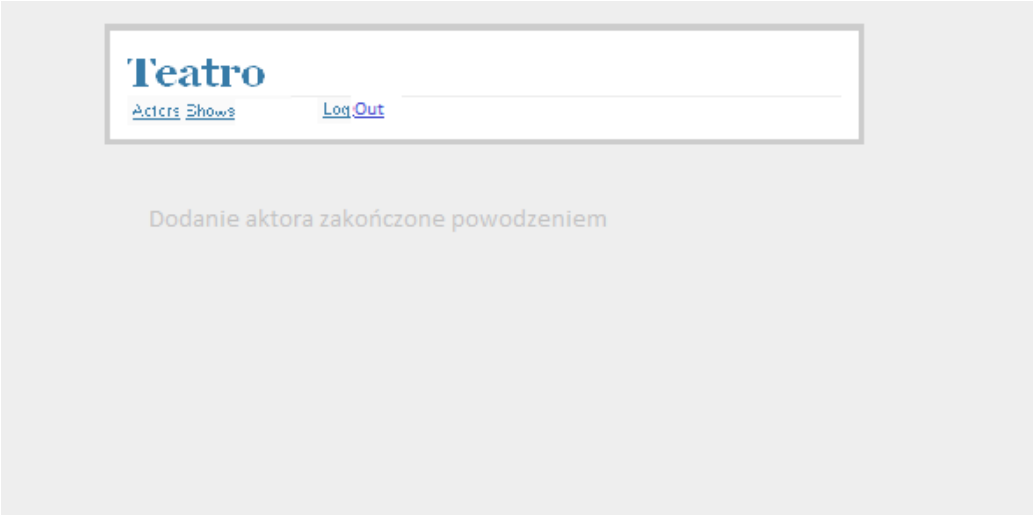
Projekt interfejsu użytkownika





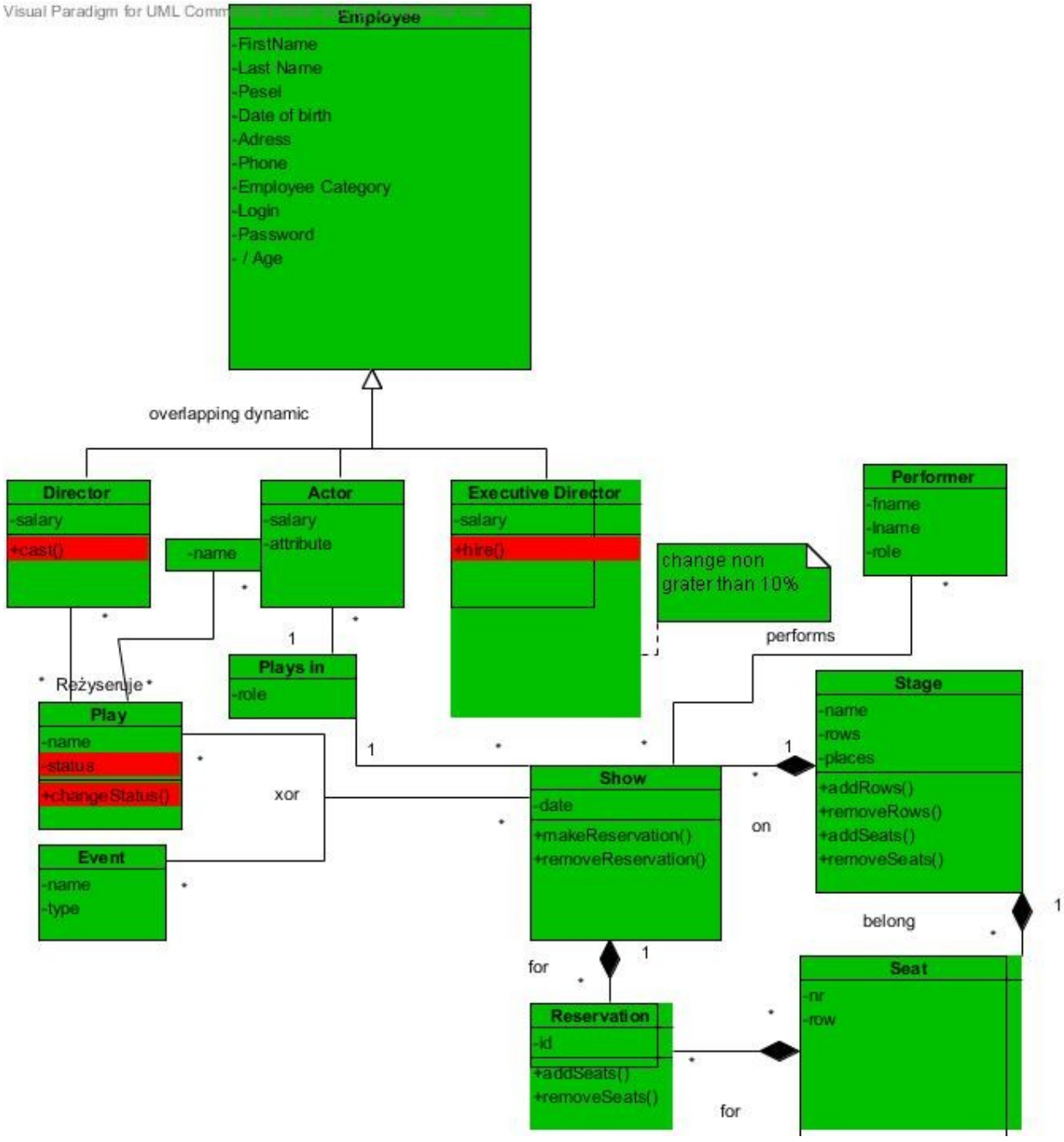

Teatro
Actors Shows
Log Out
Actor not exist!
Add actor
Abort


Teatro
Actors Shows
Log Out
Enter actors data
First Name
Last Name
next



Interfejs użytkownika, będzie dostępny przez dowolną przeglądarkę internetową. Po zalogowaniu się pracownik, będzie miał dostępne dodatkowe funkcje, zgodne ze swoim stanowiskiem w teatrze.

Diagram klas po analizie dynamicznej



Decyzje projektowe

1 Ekstensja będzie, realizowana za pomocą osobnej klasy Base wewnątrz, której znajduje się lista przechowująca wszelkiego rodzaju obiekty. Taka implementacja, jest umotywowana niewielkimi rozmiarami teatrów, (do kilkudziesięciu tysięcy obiektów), dzieki czemu lista obiektów zapewni odpowiednią wydajność i pozwoli na zaoszczedzenie czasu przy implementacji. Serializacja jednej klasy, również znacząco upraszcza sprawę, pozwalając na szybkie zapisywanie i odczytywanie ekstensji z pliku.

2 Dziedziczenie overlapping i dynamic reżysera, aktora i dyrektora z klasy pracownik zostanie rozwiązane kompozycją i dyskryminatorem w klasie pracownik. Dyskryminator będzie obiektem typu dictionary, kluczami nazwy ról a wartościami referencje do instancji tych ról.

3 Asocjacje wiele do wielu będa realizowane za pomocą list w klasach powiązanych asocjacją, w których będą przechowywane referencje do siebie nawzajem. Referencja jeden do wiele to z jednej strony pojedyncza referencja w atrybucie prostym a z drugiej lista referencji.

4 Atrybut pochodny w klasie osoba, będzie realizowany za pomocą metody obliczającej wiek.

5 Ograniczenie xor będzie zaimplementowane za pomocą metod w klsie spektakl, setEvent() i setPlay().

6 Kompozycja, będzie zaimplementowana, za pomocą listy referencji do części po stronie całości i atrybutu zawierającego referencję do całości. Dodawanie uswanie, części i całości będzie zapewnione przez metody po stronie całości.

Diagram implementacyjny