**Raditya Surya Pratama**

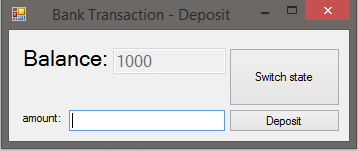
**Radu Alexandru Stoica**

Bank TRANSACTION

State Pattern

INTRODUCTION

The fifth assignment for DPR is to choose our own design pattern, and we chose State Design Pattern. We made a simple bank application that able to show the balance of the account, and then deposit a several amount of money to the account and then withdraw the money. Each activity represent a state (e.g. Deposit State, Withdraw State).



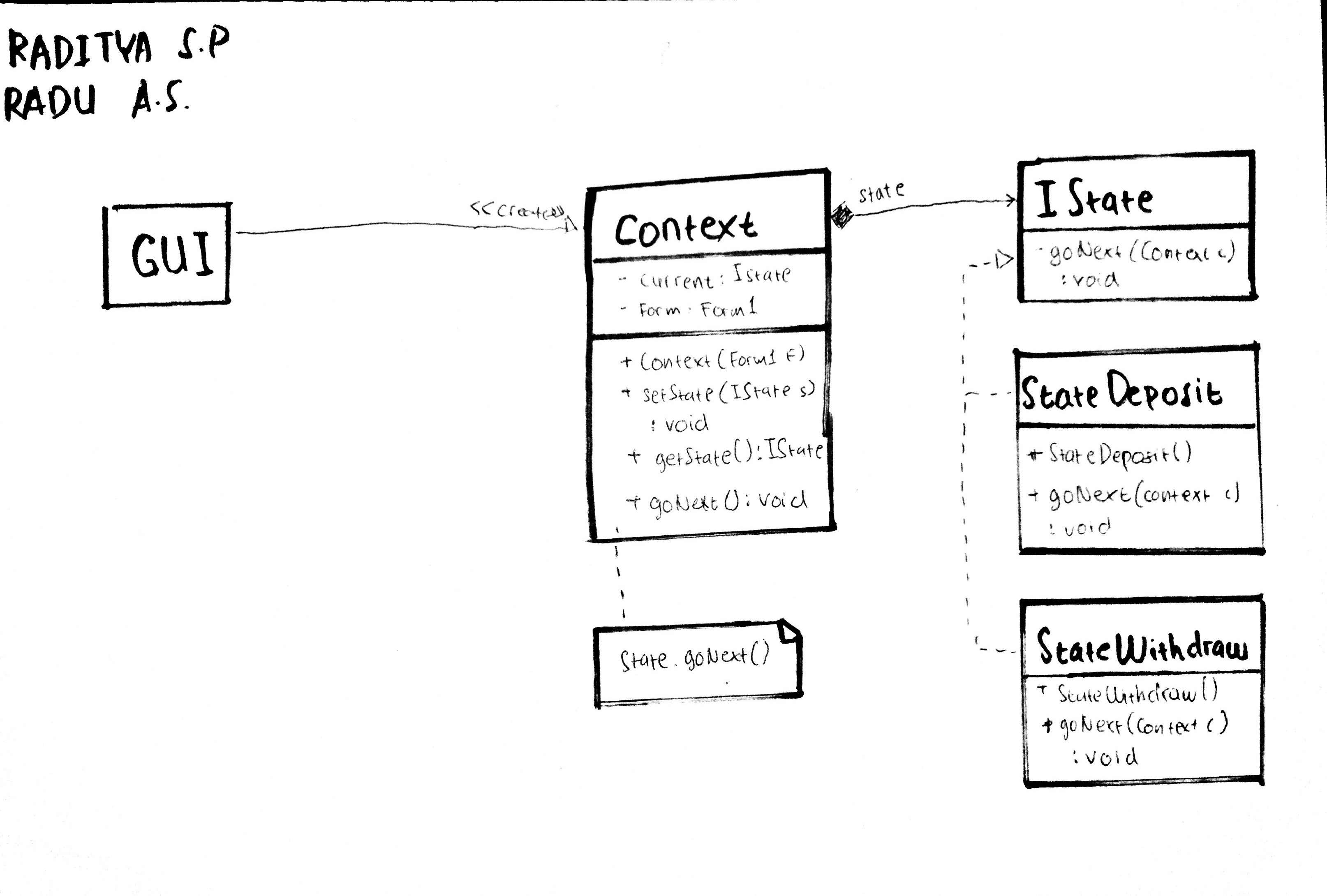
STATE PATTERN

The state pattern defines a manner for controlling communication among classes or entities. It is used to change the behavior of the object when the objects internal state is changed. This change of the state of the calss or object is hidden from the outer world with the use of a context. The state pattern can be used to implement complex decision-making algorithms represented as a flow chart or state diagram. A state pattern can also be implemente where various states exist and each state has a specific action.



Source: http://www.dofactory.com/net/state-design-pattern

UML DIAGRAM



REUSABILITY

The reusability of our code is represented by the classes that available in the application that could be used or reused by another component.

In this case, the Context class could be reused in the application for any state or conditional functionality.

EXTENSIBILITY

The extensibility of our code is represented by the implementation of the Context class that implementing the IState interface and the existence of other state that also implementing the IState interface (E.G. StateDeposit and StateWithdraw Class), so if there is any other state, new class could be added.

MAINTAINABILITY

For the maintainability matters in the application, the classes should be easily changed or updated. In this case, the maintainability is supported by the separation of state condition, and can be maintain from the classes that implementing the IState Interface.

PATTERN DOWNSIDE

There are a couple of disadvantages of using the Abstract Factory pattern:

* Difficulty in reviewing the structure of the state machine as its implementation is smeared across multiple classes
* The state interface is expensive to maintain when new event is introduced

Source: http://angry-architect.blogspot.nl/2006/08/problems-with-state-pattern.html

UNIT TEST

The unit test is included in the project solution. The unit test covers the type of the context object that created is implementing the IState interface, and when the state is change the Context object should change into StateDebosit or StateWithdraw object type.