

Object Oriented Programming

Lecture No. 7

Ing. Lukáš Slánský

Univerzita Pardubice

8. 12. 2008



Univerzita
Pardubice
Fakulta elektrotechniky
a informatiky

Contents

- 1 Use Case Diagrams
- 2 State Diagrams
- 3 What has not been (but should be) said
- 4 Some Notes to Semester Work



Use Case Diagrams

Use Case diagrams

- are not showing structure of system,



Use Case Diagrams

Use Case diagrams

- are not showing structure of system,
- are not showing behavior of system,



Use Case Diagrams

Use Case diagrams

- are not showing structure of system,
- are not showing behavior of system,
- shows interaction between system and outer participants (people, other systems).



Use Case Diagrams

Use Case diagrams

- are not showing structure of system,
- are not showing behavior of system,
- shows interaction between system and outer participants (people, other systems).
- Use Case diagrams helps analytic to communicate with customer.

Use Case Diagrams

Use Case diagrams

- are not showing structure of system,
- are not showing behavior of system,
- shows interaction between system and outer participants (people, other systems).
- Use Case diagrams helps analytic to communicate with customer.
 - ▶ What roles users have?
 - ▶ How will users use system?

What Are Use Case Diagrams?

- Use Cases describe system usage scenarios.
- They describe user roles.
- They describe system from users point of view.

Example

Example: We want to design coffee machine.

Example

Example: We want to design coffee machine.

- 1 We talk with potential users about possible usage of the machine.

Example

Example: We want to design coffee machine.

- ① We talk with potential users about possible usage of the machine.
 - ▶ Most users said that they want to buy the coffee ;-)

Example

Example: We want to design coffee machine.

- ① We talk with potential users about possible usage of the machine.
 - ▶ Most users said that they want to buy the coffee ;-) – we have scenario = Use Case (č. případ užití)
- ② We identify all participants (actors) that play its role in the scenario.

Example

Example: We want to design coffee machine.

- ① We talk with potential users about possible usage of the machine.
 - ▶ Most users said that they want to buy the coffee ;-) – we have scenario = Use Case (č. případ užití)
- ② We identify all participants (actors) that play its role in the scenario.
 - ▶ In this use case is the only actor the customer that wants to buy coffee.

Example

Example: We want to design coffee machine.

- ① We talk with potential users about possible usage of the machine.
 - ▶ Most users said that they want to buy the coffee ;-) – we have scenario = Use Case (č. případ užití)
- ② We identify all participants (actors) that play its role in the scenario.
 - ▶ In this use case is the only actor the customer that wants to buy coffee.
- ③ How the scenario looks like? What are preconditions of the scenario usage? What are scenario consequences?

Example

Example: We want to design coffee machine.

- ① We talk with potential users about possible usage of the machine.
 - ▶ Most users said that they want to buy the coffee ;-) – we have scenario = Use Case (č. případ užití)
- ② We identify all participants (actors) that play its role in the scenario.
 - ▶ In this use case is the only actor the customer that wants to buy coffee.
- ③ How the scenario looks like? What are preconditions of the scenario usage? What are scenario consequences?
- ④ What other scenarios for the participant are possible?

Example

Example: We want to design coffee machine.

- ① We talk with potential users about possible usage of the machine.
 - ▶ Most users said that they want to buy the coffee ;-) – we have scenario = Use Case (č. případ užití)
- ② We identify all participants (actors) that play its role in the scenario.
 - ▶ In this use case is the only actor the customer that wants to buy coffee.
- ③ How the scenario looks like? What are preconditions of the scenario usage? What are scenario consequences?
- ④ What other scenarios for the participant are possible?
- ⑤ How to modify scenario when customer realises that he/she has not enough money?

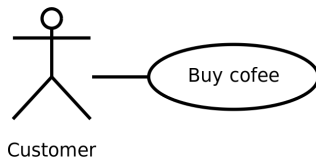
Example

Example: We want to design coffee machine.

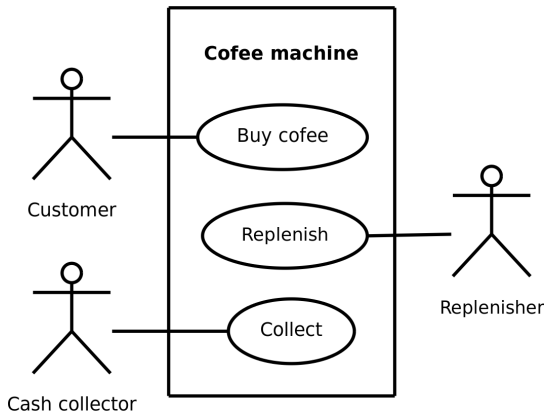
- ① We talk with potential users about possible usage of the machine.
 - ▶ Most users said that they want to buy the coffee ;-) – we have scenario = Use Case (č. případ užití)
- ② We identify all participants (actors) that play its role in the scenario.
 - ▶ In this use case is the only actor the customer that wants to buy coffee.
- ③ How the scenario looks like? What are preconditions of the scenario usage? What are scenario consequences?
- ④ What other scenarios for the participant are possible?
- ⑤ How to modify scenario when customer realises that he/she has not enough money?
- ⑥ Are there other participants that interact with coffee machine?



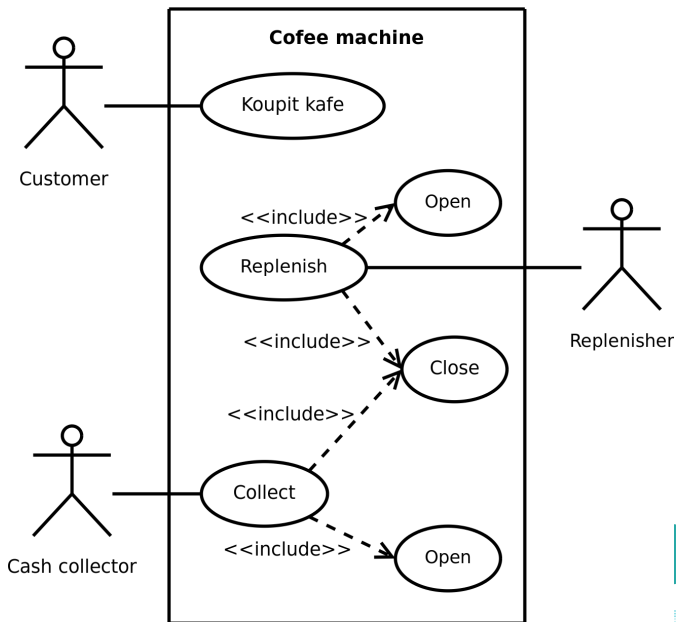
Use Cases in UML – I.



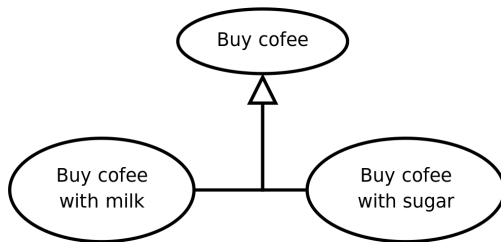
Use Cases in UML – II.



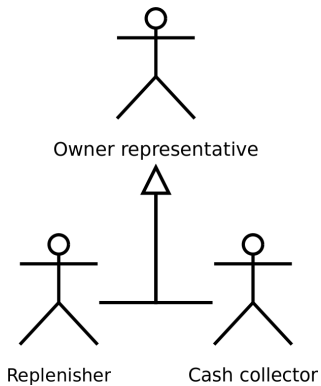
Use Cases in UML – III.



Use Cases in UML – IV.



Use Cases in UML – V.



State Diagrams

- Object is often in few distinct states during its' life.
 - ▶ Creation, work (multiple states), destruction, ...

State Diagrams

- Object is often in few distinct states during its' life.
 - ▶ Creation, work (multiple states), destruction, ...
- State diagram describes all states and *transitions* between them.

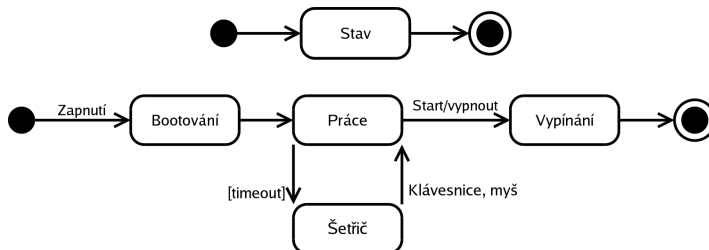
State Diagrams

- Object is often in few distinct states during its' life.
 - ▶ Creation, work (multiple states), destruction, ...
- State diagram describes all states and *transitions* between them.



State Diagrams

- Object is often in few distinct states during its' life.
 - ▶ Creation, work (multiple states), destruction, ...
- State diagram describes all states and *transitions* between them.



Procedure Fail

- It can be used only in constructor.



Procedure Fail

- It can be used only in constructor.
- It means that there is some error during constructor invocation and object creation should be undone.



Procedure Fail

- It can be used only in constructor.
- It means that there is some error during constructor invocation and object creation should be undone.
- The New function (or procedure) will release object in heap.



Procedure Fail

- It can be used only in constructor.
- It means that there is some error during constructor invocation and object creation should be undone.
- The New function (or procedure) will release object in heap.
- The pointer to object is set to *nil* and object is unusable.

Inherited



Inherited

- With keyword *Inherited* is invoked method from ancestor.



Inherited

- With keyword *Inherited* is invoked method from ancestor.
- It is used in inherited methods, constructors and destructors



Inherited

- With keyword *Inherited* is invoked method from ancestor.
- It is used in inherited methods, constructors and destructors

```
function CAncestor.Method(var1:Integer):string;  
begin  
    ...  
end;
```



Inherited

- With keyword *Inherited* is invoked method from ancestor.
- It is used in inherited methods, constructors and destructors

```
function CAncestor.Method(var1:Integer):string;
begin
    ...
end;
function CDescendant.Method(var:Integer):string;
var temp:string;
begin
    temp:=Inherited Method(var);
    ...
end;
```

Examples of Inherited

```
constructor CDescendant.Init;
```



Examples of Inherited

```
constructor CDescendant.Init;  
begin  
    inherited Init;  
    ... further constructors' code  
end;
```



Examples of Inherited

```
constructor CDescendant.Init;  
begin  
    inherited Init;  
    ... further constructors' code  
end;
```

```
destructor CDescendant.Done;
```



Examples of Inherited

```
constructor CDescendant.Init;  
begin  
    inherited Init;  
    ... further constructors' code  
end;
```

```
destructor CDescendant.Done;  
begin  
    ... destructors' code  
    inherited Done;  
end;
```

Pár poznámek k semestrální práci

- Minimální použití vícevrstvého modelu aplikace je oddělení datové vrstvy.

Pár poznámek k semestrální práci

- Minimální použití vícevrstvého modelu aplikace je oddělení datové vrstvy.
 - ▶ Mezi datovou a business vrstvou probíhá komunikace typu "Vytvoř nového zákazníka" nebo "Dej na fakturu ... položku ...".
 - ▶ Není to komunikace typu "Do souboru vlož řádek '13;Pavel Vomáčka'" nebo "INSERT INTO zakaznik. ...".

Pár poznámek k semestrální práci

- Minimální použití vícevrstvého modelu aplikace je oddělení datové vrstvy.
 - ▶ Mezi datovou a business vrstvou probíhá komunikace typu "Vytvoř nového zákazníka" nebo "Dej na fakturu ... položku ...".
 - ▶ Není to komunikace typu "Do souboru vlož řádek '13;Pavel Vomáčka'" nebo "INSERT INTO zakaznik. ...".
- Programátorská dokumentace

Pár poznámek k semestrální práci

- Minimální použití vícevrstvého modelu aplikace je oddělení datové vrstvy.
 - ▶ Mezi datovou a business vrstvou probíhá komunikace typu "Vytvoř nového zákazníka" nebo "Dej na fakturu ... položku ...".
 - ▶ Není to komunikace typu "Do souboru vlož řádek '13;Pavel Vomáčka'" nebo "INSERT INTO zakaznik. ...".
- Programátorská dokumentace
 - ▶ Poskytuje pohled na strukturu programu – všechny procedury, třídy (metody i atributy), proměnné, ... (ne algoritmy)
 - ★ U všech procedur, funkcí a metod stručně uveden význam všech parametrů a návratových hodnot, stejně jako popis funkce dané entity.

Pár poznámek k semestrální práci

- Minimální použití vícevrstvého modelu aplikace je oddělení datové vrstvy.
 - ▶ Mezi datovou a business vrstvou probíhá komunikace typu "Vytvoř nového zákazníka" nebo "Dej na fakturu ... položku ...".
 - ▶ Není to komunikace typu "Do souboru vlož řádek '13;Pavel Vomáčka'" nebo "INSERT INTO zakaznik. ...".
- Programátorská dokumentace
 - ▶ Poskytuje pohled na strukturu programu – všechny procedury, třídy (metody i atributy), proměnné, ... (ne algoritmy)
 - ★ U všech procedur, funkcí a metod stručně uveden význam všech parametrů a návratových hodnot, stejně jako popis funkce dané entity.
 - ▶ Součástí musí být diagram tříd, který popisuje kompletní objektovou strukturu programu – všechny vytvořené a použité třídy.

Pár poznámek k semestrální práci

- Minimální použití vícevrstvého modelu aplikace je oddělení datové vrstvy.
 - ▶ Mezi datovou a business vrstvou probíhá komunikace typu "Vytvoř nového zákazníka" nebo "Dej na fakturu ... položku ...".
 - ▶ Není to komunikace typu "Do souboru vlož řádek '13;Pavel Vomáčka'" nebo "INSERT INTO zakaznik. ...".
- Programátorská dokumentace
 - ▶ Poskytuje pohled na strukturu programu – všechny procedury, třídy (metody i atributy), proměnné, ... (ne algoritmy)
 - ★ U všech procedur, funkcí a metod stručně uveden význam všech parametrů a návratových hodnot, stejně jako popis funkce dané entity.
 - ▶ Součástí musí být diagram tříd, který popisuje kompletní objektovou strukturu programu – všechny vytvořené a použité třídy.
- Všechny identifikátory volte čitelně – tj. ne Metoda1 nebo Form8.

Pár poznámek k semestrální práci

- Minimální použití vícevrstvého modelu aplikace je oddělení datové vrstvy.
 - ▶ Mezi datovou a business vrstvou probíhá komunikace typu "Vytvoř nového zákazníka" nebo "Dej na fakturu ... položku ...".
 - ▶ Není to komunikace typu "Do souboru vlož řádek '13;Pavel Vomáčka'" nebo "INSERT INTO zakaznik. ...".
- Programátorská dokumentace
 - ▶ Poskytuje pohled na strukturu programu – všechny procedury, třídy (metody i atributy), proměnné, ... (ne algoritmy)
 - ★ U všech procedur, funkcí a metod stručně uveden význam všech parametrů a návratových hodnot, stejně jako popis funkce dané entity.
 - ▶ Součástí musí být diagram tříd, který popisuje kompletní objektovou strukturu programu – všechny vytvořené a použité třídy.
- Všechny identifikátory volte čitelně – tj. ne Metoda1 nebo Form8.
- Všechny bloky programu (procedury, funkce, metody) dělejte snesitelně dlouhé – maximálně 30 řádků, raději méně.

