Rozszerzenia składni C *vs.* C++

Stałe wyliczeniowe w C:

umożliwiają definiowanie typów, wraz ze zmiennymi do których można przypisać wartości

```
enum dzien {PON, WT, SR, CZW, PT, SOB, NIE};
enum srodekTransportu {SAMOCHOD, TRAMWAJ, AUTOBUS,
ROWER, NOGI};
```

Przykład:

```
enum KOLOR { czerwony,zielony,niebieski};
enum KOLOR ulubionyKolor;
ulubionyKolor = czerwony;
```


16

Rozszerzenia składni C *vs.* C++

Stałe wyliczeniowe w C++:

Nie wymagane już jest pisanie słowa `enum`, np.:

```
enum KOLOR ulubionyKolor = czerwony;  // OK. w C i C++
KOLOR nielubianyKolor = seledynowy;   // OK. tylko w C++
```

`enum` już nie jest tożsamy z typem `int` (brak konwersji), np.:

```
ulubionyKolor  = 2;                   // OK. tylko w C
```


17

Różne właściwości

Funkcje tak samo jak zmienne mają swoje miejsce w pamięci, gdzie są zapisane. Można więc uzyskać ich adres.

Podobnie jak adres tablicy jest zwracany przez jej nazwę, podaną bez nawiasu kwadratowego, adres funkcji uzyskuje się za pomocą nazwy funkcji pozbawionej listy argumentów.

Można tworzyć tablice wskaźników do funkcji i wywoływać funkcje nie po nazwie, a po numerze w tablicy

18

Różne właściwości

Wskaźniki do funkcji - przykład:

```
int suma (int a, int b) {
  return a+b;
}

int main () {
  int (*wsk_suma)(int, int);      // zmienna wskaźnikowa
  wsk_suma = suma;          // przypisanie adresu funkcji
  printf("4+5=%d\n", (*wsk_suma)(4,5));
  return 0;
}
```

gdzie:

```
   nazwa zmiennej:   wsk_suma
   typ wskazywany:   int  …  (int, int)
```


19

Różne właściwości

Tablice wskaźników do funkcji - przykład:

```
void ErrMsg() { printf("Blad\n"); };
void OKMsg()  { printf("OK\n");   };
void YesMsg() { printf("Tak\n");  };
void NoMsg()  { printf("Nie\n");  };
void (*func_table[])() = {ErrMsg, OKMsg, YesMsg, NoMsg };

int main () {
  srand( (unsigned)time( NULL ) );
  double r = rand();
  int i = floor(4*r/RAND_MAX);
  (*func_table[i])();
  return 0;
}            void fun(int i, void (*func_table[])() );
```


20

OBIEKTOWE METODY INŻYNIERII OPROGRAMOWANIA

21

Obiektowe metody inżynierii oprogramowania

1. Analiza obiektowa (OOA – Object Oriented Analysis)
2. Projektowanie obiektowe (OOD – Object Oriented Design)
3. Programowanie obiektowe (OOP – Object Oriented Programming)

Peter Coad, Edward Yourdon

Analiza obiektowa

Analiza obiektowa
służy zbudowaniu *modelu rzeczywistości* wg pewnych zasad.

Trzymanie się tych zasad:
- daje spójną reprezentację stanowiącą podstawę analizy (co budować) i projektowania (jak budować),
- pozwala na identyfikację wspólnych cech atrybutów i usług,
- pozwala na budowę specyfikacji poddających się zmianom,
- pozwala na powtórne wykorzystanie wyników analizy dla rodzin systemów,
- pozwala na lepsze wzajemne zrozumienie analityka i eksperta w danej dziedzinie zastosowania.

Analiza obiektowa

Zarządzanie złożonością w analizie:

- **Przyjęcie powszechnie stosowanych metod organizacji**
  – tworzenie klas obiektów i rozróżnianie ich; rozróżnienie między obiektem a jego atrybutami; rozróżnienie między całym obiektem a jego składowymi.
  Abstrakcja proceduralna
  – pomijanie niektórych szczegółów rzeczy lub procesów wybierając tylko pewną istotną część na danym poziomie uogólnienia; definiujemy atrybuty oraz usługi, które mają wyłączność na manipulowanie tymi atrybutami.

Analiza obiektowa

Zarządzanie złożonością w analizie:

- **Hermetyzacja** – zasada używana przy budowie całościowej struktury programu:
  - każda składowa programu powinna zamykać w sobie jedną decyzję projektową;
  - styk z każdym modułem powinien być zdefiniowany tak, aby odkrywać możliwie mało ze swej wewnętrznej struktury.
- **Dziedziczenie**
  - uproszczenie definicji klas podobnych do już definiowanych przez wykorzystanie tych zdefiniowanych w definicji nowych;
  - opisuje generalizację i specjalizację czyniąc wspólne atrybuty i usługi jawnymi wewnątrz hierarchii klas.

Analiza obiektowa

Zarządzanie złożonością w analizie:

- **Skojarzenie**
  – wiązanie ze sobą idei podobnych lub rzeczy które zdarzają się w tym samym czasie lub w podobnych okolicznościach.
- **Komunikaty**
  – porozumiewanie się między obiektami.
- **Skala**
  – pokazanie relacji między obiektami (części do całości) dla wyobrażenia sobie np. dużego lub bardzo małego obiektu.

Analiza obiektowa

Zarządzanie złożonością w analizie:

- **Kategorie zachowania**
  - reakcja na zdarzenie,
  - zachowanie ze względu na podobieństwo ewolucji (zmiany w czasie),
  - zachowanie ze względu na podobieństwo funkcji.

Analiza obiektowa

Metody analizy:

1. inne niż obiektowe
   - Rozkład funkcjonalny
   - Metoda przepływu danych (*Data Flow Diagrams*)
   - Modelowanie informacji (*Entity Relationship Diagrams*)
2. podejście obiektowe
   … (o tym będzie dalej)

Analiza obiektowa

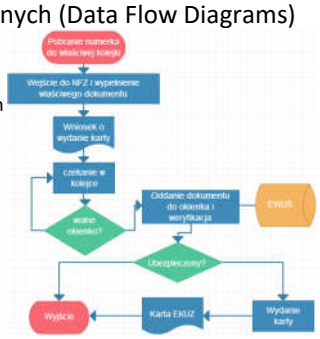
Rozkład funkcjonalny (*nieobiektowy*)

Dekompozycja algorytmiczna –
podział algorytmu na odrębne czynności: każdy projektowany moduł programowy systemu określa znaczący krok w procesie przetwarzania.

Analiza obiektowa

Metoda przepływu danych (Data Flow Diagrams) (*nieobiektowy*)

Graficzna prezentacja, tzw. diagram przepływu danych zawiera następujące rodzaje elementów:
- funkcje (procesy),
- magazyny danych,
- źródła i odbiorcy,
- przepływy (znaki pokazujące kierunek przesyłu danych)



Analiza obiektowa

Modelowanie informacji (Entity Relationship Diagrams) (*nieobiektowy*)

Graficzna prezentacja logicznej struktury danych w bazie danych.

Występujące pojęcia: **encje** (grupy lub kategorie danych) oraz **atrybuty** (podgrupy wewnątrz encji)).

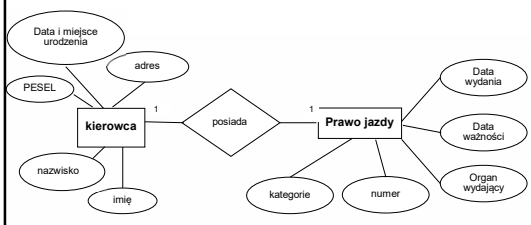
Relacje wewnątrz modelu:

1. Obowiązkowe
2. Opcjonalne
3. Wielu-do-wielu
4. Jeden-do-wielu
5. Jeden-do-jeden
6. Rekurencyjne

Analiza obiektowa

Modelowanie informacji (Entity Relationship Diagrams) (*nieobiektowy*)

Graficzna prezentacja logicznej struktury danych w bazie danych:



Analiza obiektowa

Podejście obiektowe

Rodzaj dekompozycji, w której zamiast wyszczególniać czynności wyodrębnia się obiekty, na których dokonywane mogą być operacje.

Różnice:

- ❑ **Dekompozycja algorytmiczna** – opisuje kolejność czynności
- ❑ **dekompozycja obiektowa** – opisuje uczestniczące w czynnościach obiekty oraz wzajemne oddziaływanie obiektów na siebie.

Dekompozycja algorytmiczna i obiektowa są sobie przeciwstawne – nie można ich przeprowadzić jednocześnie.

Analiza obiektowa

Podejście obiektowe – składowe modelu:

- Pojęcia
- Obiekty
- Związki miedzy pojęciami
- Atrybuty obiektów
- Usługi obiektów
- Komunikaty między obiektami

Analiza obiektowa

Pojęcia – są środkiem służącym do rozpoznawania:

- Materialne – osoba, ołówek, samochód
- Niematerialne – czas, jakość, firma
- Role – doktor, pacjent, właściciel
- Opinie – wydajna praca, wysoka płaca, dobry przykład
- Relacyjne – małżeństwo, posiadanie
- Zdarzenia – sprzedaż, zakup, załamanie rynku
- Inne – zestaw, liczba, ikona, obraz, sygnał, proces

Pojęcia pozwalają nadać znaczenie obiektom w naszym świecie.

Analiza obiektowa

Trójka pojęciowa:

1. Symboliczna reprezentacja, np. : nazwa, ikonka, znaczek
2. Intensja – treść pojęcia, pełna definicja pojęcia
3. Ekstensja – zakres pojęcia, zbiór wszystkich rzeczy i wyobrażeń abstrakcyjnych, do których stosuje się dane pojęcie.

Analiza obiektowa

Typ obiektowy
– opis obiektu z jednolitym zbiorem atrybutów i usług, zawierający opis tworzenia nowych obiektów w klasie

Obiekty
– coś, do czego da się zastosować jakieś pojęcie. Obiekt jest egzemplarzem pojęcia (instancją).

Struktura i zachowanie obiektu są określone przez pojęcia, które się do niego odnoszą:

Pojęcie ↔ typ obiektowy

Obiekt to kapsułka ze zdefiniowanymi wartościami atrybutów i wyłącznie na nich działającymi usługami. Większość obiektów istnieje tylko przez pewien okres.

Analiza obiektowa

Struktury - związki między typami obiektowymi

Odwzorowania:

np. „zatrudnia” przypisuje obiekt typu „organizacja” związanym z nim obiektom typu „zatrudniony”. Odwzorowania mogą być jedno- lub wielowartościowe.

1. A jest zawsze związane z jednym B
2. A jest zawsze związane z jednym lub wieloma B
3. A jest zawsze związane z żadnym lub tylko jednym B
4. A jest zawsze związane z żadnym, jednym lub wieloma B

Analiza obiektowa

Struktury - związki między typami obiektowymi

Relacje:

Mogą obejmować kilka typów obiektów,

np. „umowa o prace” jest typem związków, którego *krotki* są niezmiennymi parami obiektów typu „osoba” i „organizacja”

Analiza obiektowa

Struktury - podtypy i nadtypy obiektów:

Podtyp – typ obiektowy, którego zbiór wszystkich elementów zawiera się w większym zbiorze, oraz definicja jest bardziej wyspecjalizowana niż definicja innego typu

Nadtyp - typ obiektowy, którego zbiór zawiera wszystkie elementy jednego lub więcej typów, oraz definicja jest ogólniejsza niż definicja innych typów

Np.: organizm – zwierzę – ssak - pies

Analiza obiektowa

Atrybuty – dane (stan systemu), dla których każdy obiekt ma swoją własną wartość

Każdy typ obiektowy jest opisany przez atrybuty.

Atrybuty są szczegółowo opisane w specyfikacji obiektu.

Atrybuty opisują wartości trzymane w obiekcie, aby wyłącznie usługi tego obiektu mogły nimi manipulować. Atrybuty i specjalne usługi na nich działające traktujemy jako nierozłączną całość.

Jeżeli inny obiekt chce odczytać wartości w obiekcie lub działać na nich w inny sposób, musi zrobić to poprzez specyfikację powiązania odpowiadającego komunikatowi, który spowoduje realizację usługi zdefiniowanej dla tego projektu.

Analiza obiektowa

Definiowanie usług

Usługa – zdefiniowane zachowanie obiektu, które jest on zobowiązany przejawić.

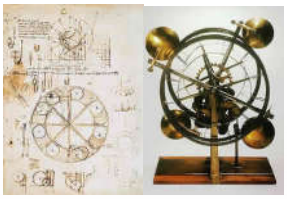
Usługi stosują się do atrybutów obiektu.

Definiując nowe usługi wyróżniamy następujące działania:

- Identyfikacja stanów obiektu
- Identyfikacja wymaganych usług
- Identyfikacja powiązań odpowiadających komunikatom
- Specyfikacja usług
- Zebranie w jedną całość dokumentacji analizy

Analiza obiektowa

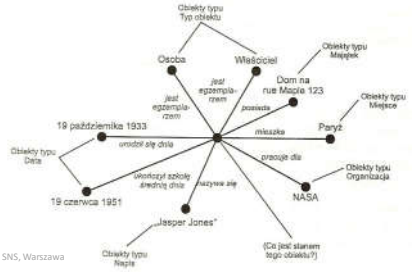
- Zbiór powiązanych ze sobą obiektów o pewnej funkcjonalności można nazywać systemem.
- Raz uruchomiony system działa – obiekty oddziałują na siebie, zmieniając swój stan i tworząc nowe obiekty powiązane z tym systemem, a także usuwając już istniejące.



Perpetuum mobile –
projekt: Leonardo da Vinci ,
Muzeum w Monachium, Niemcy
http://www.leifiphysik.de/themenbereiche/arbeit-energie-und-leistung/geschichte

Analiza obiektowa

- Stan obiektu – kolekcja powiązań obiektu z innymi obiektami i typami obiektowymi w pewnym okresie czasu



Analiza obiektowa

„życie" obiektu:

- W systemie obiekty powstają, istniejąc zmieniają wielokrotnie swój stan, oraz są usuwane
- Zmiana stanu jest zmianą powiązań (atrybutów i/lub związków) obiektu
- Istotna zmiana stanu obiektu nazywana jest **zdarzeniem**
- Rodzaje zdarzeń: tworzenie, kończenie, łączenie/rozłączanie, inne
- Analityk nie potrzebuje wiedzy o każdym możliwym zdarzeniu: klasyfikuje je tworząc typy zdarzeń, np. zamiast definiowania zdarzenia „wczoraj mój pies ugryzł nowego listonosza w lewą nogę" tworzy typ: „pies ugryzł osobę"
- Zdarzenia są historią obiektów

Analiza obiektowa

„życie" systemu:

- Zmienność systemu wyraża się przez procesy
- Procesy odczytują i zmieniają stany obiektów
- **Operacje** to jednostki przetwarzania, z których składają się **procesy**
- Każda operacja wymaga obiektów, na których może operować (jeden lub więcej)
- Operacje mogą zwraca nowe obiekty
- Operacje są też związane z typami zdarzeniowymi – efektem operacji mogą być zdarzenia
- Ograniczenia, przy spełnieniu których operacja wykona się poprawnie, to warunki wstępne
- Ograniczenia jakie musza zachodzić w wyniku zakończenia operacji to warunki końcowe
- Operacje zegarowe: emitują zdarzenia - tyknięcia zegara
- Specyfikacja sposobu wykonania operacji to **metoda**

Analiza obiektowa

Metody

- są izolowane od rozważań o przyczynach i skutkach, tj.:
  - Metoda działa tak samo niezależnie od tego, jakie zdarzenie ją wywołało i jaki jest stan obiektów (wystarczy że spełniają ograniczenia)
  - Metoda w swym działaniu nie uwzględnia tego, jakie następne metody wywoła zdarzenie jej zakończenia
- Można je specyfikować jako moduły zawierające różnorodne składniki: ciągi zdarzeń, wywołań i warunków sterujących; są to zagnieżdżone struktury operacji.

Analiza obiektowa

Wyzwalacz

- połączenie zdarzenia i wywoływanego procesu

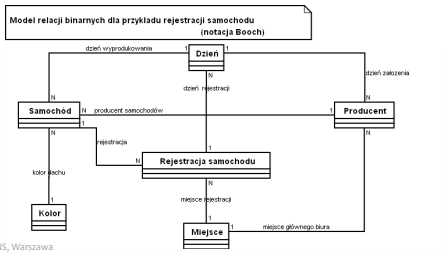
  Rodzaje wyzwalaczy == reguły wyzwalania

- Wyzwalacz może wyzwalać wiele procesów, które mogą być wykonywane sekwencyjnie lub równolegle

Analiza obiektowa

Reprezentowanie struktury obiektów

model relacji binarnych (diagram obiektowy) – wyraża związki między typami



Analiza obiektowa

Reprezentowanie zachowania obiektów

- Maszyny skończone: stany, przejścia między stanami (zależą od zdarzeń).
- Specyfikacje oparte na scenariuszach (diagramy sieci Petriego).
- Specyfikacje decyzyjne wyrażone za pomocą tabeli decyzyjnej i drzewa decyzyjnego.

Analiza obiektowa

Podejścia do analizy obiektowej - historia

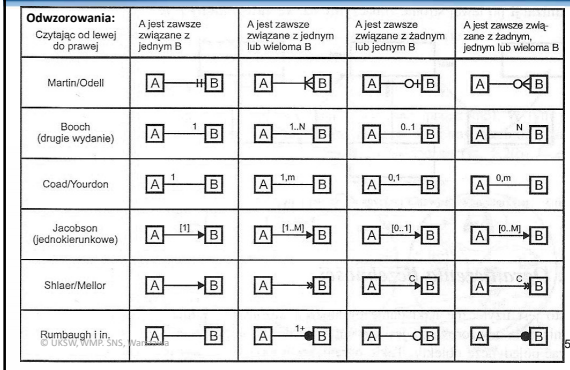
- Martin/Odell
- Booch
- Coad/Yourdon
- Jacobson
- Shlaer/Mellor
- Rumbaugh i in.

Źródło: „Podstawy metod obiektowych”
J. Martin, J.J. Odell, WNT, 1997

Analiza obiektowa

Martin/Odell	Booch	Coad/Yourdon	Jacobson	Shlaer/Mellor	Rumbaugh i in.
Typ obiektowy	Klasa	Klasa i obiekt	Typ obiektowy	Obiekt	Klasa
Obiekt	Obiekt	Obiekt	Obiekt, Egzemplarz	Egzemplarz	Obiekt
Typ związków	Używanie relacji	Związek obiektów		Związek	Powiązanie
Odwzorowanie	Rola		Związek znajomości	Związek, Odwzorowanie	Rola
Uogólnianie, Specjalizowa-nie	Dziedziczenie	Gen-Spec	Dziedziczy	Podtyp-Nadtyp	Uogólnianie
Składanie	Zawieranie	Część-Całość			Agregacja

Analiza obiektowa



Analiza obiektowa

Analizą obiektowa wg metody Coad/Yourdon:

5 głównych czynności:

1. znajdowanie klas i obiektów
2. identyfikacja struktur
3. identyfikacja tematów
4. definiowania atrybutów
5. definiowania usług

model analizy obiektowej zawiera 5 warstw:

1. warstwa tematów
2. warstwa klas i obiektów
3. warstwa struktury
4. warstwa atrybutów
5. warstwa usług

Analiza obiektowa

Analizą obiektowa wg metody OMT (Rumbaugh):

3 części składowe modelu, pokazujące różne jego aspekty:

Model Obiektów (OMT Object Model): statyczny obraz struktury modelu
- klasy
- atrybuty
- operacje
- relacje między klasami i instancjami

Model Dynamiczny (OMT Dynamic Model): współdziałanie obiektów (powiązania wyznaczone przez komunikaty).
Tu mieszczą się różne diagramy pokazujące przepływ sterowania, także ograniczenia i warunki na wartości atrybutów.

Model Funkcyjny (OMT Functional Model)
specyfikacja operacji jako funkcji przekształcających wejście na wyjście, warunki poprawności (asercje).

PROJEKTOWANIE OBIEKTOWE

Projektowanie obiektowe

- Za budowę modelu odpowiedzialni są analitycy, którzy mają kontakt z klientem
- Analitycy uzgadniają z inżynierami oprogramowania poprawność budowanego modelu, dostosowując go do możliwości narzędzia programistycznego, które zostanie użyte podczas fazy implementacji
- Rola inżyniera oprogramowania:
  - jednocześnie znać kilka metodyk projektowania lub przynajmniej zdawać sobie sprawę z różnic, jakie między nimi występują
  - potrafić przekładać niejasne żądania klientów na precyzyjne specyfikacje
  - umieć rozmawiać z zamawiającym oprogramowanie posługując się terminami z dziedziny aplikacji, a nie terminami informatycznymi

58

Projektowanie obiektowe

Obszary wiedzy inżyniera oprogramowania

- Inżynieria oprogramowania, informatyka
- Znajomość środowiska programistycznego (język programowania, platforma systemowa, środowisko projektowe)
- Znajomość organizacji (przedsiębiorstw, administracji publicznej)
- Znajomość dziedziny aplikacji (np. awioniki, operacji bankowych, logistyki, chemii, itp.)
- Psychologia i socjologia (etyka, komunikatywność, gospodarowanie czasem, itp.)

59

Projektowanie obiektowe

- Podczas analizy wzajemne oddziaływania między obiektami są reprezentowane jako zdarzenia.
- Projektant ma za zadanie wybrać rodzaj przepływu sterowania, jaki będzie implementowany w projekcie.
- Zewnętrzne sterowanie:
  - Sekwencyjne proceduralne
  - Sekwencyjne zdarzeniowe
  - Współbieżne
- Wewnętrzne sterowanie

60

Projektowanie obiektowe

Systemy sterowane procedurami

- Sterowanie proceduralne umiejscowione jest w kodzie programu:
  - zaczyna się na wierzchołku hierarchii podprogramów i przez wywołania podprogramów przechodzi do niższych poziomów, albo
  - jeden z komponentów jest menedżerem systemu: steruje rozpoczynaniem, zatrzymywaniem i koordynacją innych

61

Projektowanie obiektowe

Systemy sterowane procedurami

- Procedury generują żądania wprowadzenia danych i czekają na ich pojawienie się. Po wprowadzeniu danej sterowanie przekazywane jest zwrotnie do procedury. Wartość licznika rozkazów, zmiennych lokalnych i stosu, na którym odkładane są wywołania procedur, definiują stan systemu.

- Można stosować tylko, gdy schemat zmiany stanów wykazuje regularne następstwo zdarzeń wejściowych i wyjściowych.
- Trudno jest stworzyć elastyczne interfejsy użytkownika.

62

Projektowanie obiektowe

Systemy sterowane zdarzeniami

- Sterowanie zdarzeniowe rezyduje wewnątrz programu koordynującego lub monitorującego.
- Procedury aplikacji są przywiązane do zdarzeń i są wywoływane, gdy pojawiają się odpowiadające im zdarzenia, tj. podsystemy same decydują, które zdarzenia są dla nich interesujące.
- Odwołania do programu koordynującego pozwalają na wprowadzanie danych wejściowych oraz wysyłanie danych wyjściowych (procedury nie zachowują sterowania np. czekając na dane).

63

Projektowanie obiektowe

Sterowanie wewnętrzne

Proceduralne

•przekazywanie sterowania (wywołania procedur, wywołania między zadaniowe) jest kontrolowane przez program i jest poza kontrolą użytkownika

Zdarzeniowe

•wzajemne oddziaływania między obiektami są takie same jak zewnętrzne, za wyjątkiem braku oczekiwań na zdarzenia: obiekty mogą wymuszać odpowiedzi innych obiektów

Sterowanie zewnętrzne

Proceduralne

•Jest kontrolowane przez użytkownika (trudne w realizacji)

Zdarzeniowe

•Oczekiwanie na zdarzenia, które są pod kontrolą użytkownika (dość intuicyjne)

64

Projektowanie obiektowe

- W procesie rozwoju oprogramowania:
- Faza analizy – opisuje wymagania
- Faza projektu – definiuje strategię rozwiązania, tj. projekt klas obiektów przez:
  - Zdefiniowanie interfejsów
  - Zdefiniowanie algorytmów użytych do zaimplementowania operacji
  - Optymalizowanie struktur danych i algorytmów

W fazie projektu wprowadza się również dodatkowe „wewnętrzne" klasy obiektów niezbędne dla implementacji

65

Projektowanie obiektowe

Kolejność tworzenia projektu obiektów

1. Wpisać do klas obiektów wszystkie operacje z modeli analitycznych.
2. Zaprojektować algorytmy implementujące operacje, biorąc pod uwagę złożoność obliczeniową, łatwość implementacji, zrozumiałość i elastyczność.
3. Zoptymalizować ścieżki dostępu do danych.
4. Zaimplementować sterowanie, które odwzoruje model dynamiczny (system sterowany procedurami lub zdarzeniami).
5. Zmienić strukturę klas aby rozszerzyć związki dziedziczenia.
6. Zaprojektować implementacje związków klas (nadmiarowe związki gmatwają analizę, ale na etapie implementacji mogą poprawić wydajność i uprościć realizację niektórych procesów).
7. Określić reprezentację obiektu.
8. Umieścić klasy i związki w modułach.

66