1 Omówić pojęcie informatyki i jej działy

Informatyka – dyscyplina nauki zaliczana do nauk ścisłych oraz techniki zajmująca się przetwarzaniem informacji, w tym również technologiami przetwarzania informacji oraz technologiami wytwarzania systemów przetwarzających informację. Początkowo stanowiła część matematyki, później rozwinęła się do odrębnej dyscypliny – pozostaje jednak nadal w ścisłej relacji z matematyką, która dostarcza informatyce podstaw teoretycznych.

Bardziej znane i popularne działy informatyki to przede wszystkim:

- administracja sieciowa – zarządzanie siecią komputerową;
- administracja systemem – zarządzanie systemem informatycznym;
- algorytmika – tworzenie i analizowanie algorytmów. Podstawowa, najstarsza dyscyplina informatyki;
- architektura procesorów – projektowanie procesorów, bez których nie byłoby komputerów;
- bezpieczeństwo komputerowe – dyscyplina łącząca informatykę z telekomunikacją w celu zapewnienia poufności i bezpieczeństwa danych;
- grafika komputerowa – wykorzystuje technikę komputerową w celu wizualizacji rzeczywistości;
- inżynieria oprogramowania – produkcja oprogramowania;
- języki programowania – tworzenie języków programowania. Wyróżniająca się, podstawowa dyscyplina informatyki;
- programowanie – czyli tworzenie kodu źródłowego programów komputerowych. Najpopularniejsza dyscyplina informatyki;
- sprzęt komputerowy – komputery i ich urządzenia peryferyjne;
- symulacja komputerowa – komputerowa symulacja z wykorzystaniem modelowania matematycznego;
- systemy informatyczne – tworzenie systemów informatycznych w celach użytkowych;
- sztuczna inteligencja – komputerowe symulowanie inteligencji;
- teoria informacji – dyscyplina zajmująca się problematyką informacji, w tym teorią przetwarzania i przesyłania informacji;
- webmastering – projektowanie, programowanie i publikacja serwisów internetowych.

2 Przedstawić cechy programowania strukturalnego

Wikipedia :: Programowanie strukturalne

- hierarchiczne dzielenie kodu
- jeden punkt wejścia
- jeden lub wiele punktów wyjścia
- nieużywanie lub ograniczenie instrukcji skoku “goto”

Dobre struktury:

- instrukcje warunkowe (if, if...else)
- pętle (while, repeat)
- instrukcje wyboru (case, ale nie switch z C i potomnych)

Strukturalność zakłócają:

- break
- switch
- continue

3 Scharakteryzować paradygmat programowania obiektowego

Abstrakcja

Każdy obiekt w systemie służy jako model abstrakcyjnego "wykonawcy", który może wykonywać pracę, opisywać i zmieniać swój stan oraz komunikować się z innymi obiektami w systemie bez ujawniania, w jaki sposób zaimplementowano dane cechy. Procesy, funkcje lub metody mogą być również abstrahowane, a kiedy tak się dzieje, konieczne są rozmaite techniki rozszerzania abstrakcji.

Hermetyzacja

Czyli ukrywanie implementacji, enkapsulacja. Zapewnia, że obiekt nie może zmieniać stanu wewnętrznego innych obiektów w nieoczekiwany sposób. Tylko własne metody obiektu są uprawnione do zmiany jego stanu. Każdy typ obiektu prezentuje innym obiektom swój interfejs, który określa dopuszczalne metody współpracy. Pewne języki osłabiają to założenie, dopuszczając pewien poziom bezpośredniego (kontrolowanego) dostępu do "wnętrzności" obiektu. Ograniczają w ten sposób poziom abstrakcji. Przykładowo w niektórych kompilatorach języka C++ istnieje możliwość tymczasowego wyłączenia mechanizmu enkapsulacji; otwiera to dostęp do wszystkich pól i metod prywatnych, ułatwiając programistom pracę nad pośrednimi etapami tworzenia kodu i znajdowaniem błędów.

Polimorfizm

Referencje i kolekcje obiektów mogą dotyczyć obiektów różnego typu, a wywołanie metody dla referencji spowoduje zachowanie odpowiednie dla pełnego typu obiektu wywoływanego. Jeśli dzieje się to w czasie działania programu, to nazywa się to późnym wiązaniem lub wiązaniem dynamicznym. Niektóre języki udostępniają bardziej statyczne (w trakcie kompilacji) rozwiązania polimorfizmu – na przykład szablony i przeciążanie operatorów w C++.

Dziedziczenie

Porządkuje i wspomaga polimorfizm i enkapsulację dzięki umożliwieniu definiowania i tworzenia specjalizowanych obiektów na podstawie bardziej ogólnych. Dla obiektów specjalizowanych nie trzeba redefiniować całej funkcjonalności, lecz tylko tę, której nie ma obiekt ogólniejszy. W typowym przypadku powstają grupy obiektów zwane klasami, oraz grupy klas zwane drzewami. Odzwierciedlają one wspólne cechy obiektów.

4 Omówić elementy klasy w programie

Wikipedia :: Klasa (programowanie obiektowe)

Metody określają możliwe zachowania, jake na obiekcie można wykonać, a ich definicje znajdują się w klasie danego obiektu. Metoda jest rodzajem podprogramu języka programowania z dodatkową właściwością, th. dostępem do atrybutów obiektu, na którym została wywołana. Zbiór metod nazywany jest interfejsem.

Atrubuty, zwane także polami lub właściwościami, są to elementy składające się na strukturę danych przechowywanych w obiektach. Zbiór wartości wszystkich atrybutów tworzy stan obiektu, który przechowywany jest w pamięciu lub innym nośniku danych pod określonym adresem, dzięki czemu możliwy jest dostęp do niego za pośrednictwem referencji.

Niezmienniki to elementy stałe zachowywane przez wszystkie metody klasy. Niezmiennik jest pewnym wyrażeniem odnoszącym się do atrybutu, które musi być zawsze spełnione aby stan obiektu był prawidłowy, niezależnie od tego, jakie operacje na nim wykonamy. Niezmienniki mogą być implementowane poprzez zabronienie bezpośredniego dostępu do atrybutów obiektu i utworzenie dodatkowych metod dostepowych, które oprócz ich stawiania, sprawdzają także niezmienniki. W niektórych językach niezmienniki można definiować bezpośrednio jako część specyfikacji klasy.

Elementy statyczne nie są związane z żadnym konkretnym obiektem klasy, lecz tworzą globalny stan oraz globalnie dostepne operacje, które można wywoływać nawet wtedy, gdy nie posiadamy żadnej instancji klasy.

5 Scharakteryzować cechy algorytmów, w tym złożoność obliczeniową

Algorytm – w matematyce oraz informatyce to skończony, uporządkowany ciąg jasno zdefiniowanych czynności, koniecznych do wykonania pewnego zadania.

Cechy algorytmów

- poprawność — algorytm daje dobre wyniki,
- jednoznaczność — daje takie same wyniki przy takich samych danych,
- skończoność — wykonuje się w skończonej ilości kroków,
- sprawność — czasowa; szybkość działania i pamięciowa; "zasobożerność"

Złożoność obliczeniowa

Ilość zasobów niezbędnych do wykonania algorytmu można rozumieć jako jego złożoność. W zależności od rozważanego zasobu mówimy o złożoności czasowej czy też pamięciowej.

Oczywiście w większości wypadków ilość potrzebnych zasobów będzie się różnić w zależności od danych wejściowych z zakresu danego zagadnienia. Złożoność algorytmu jest funkcją rozmiaru danych wejściowych.

Kolejnym problemem jest fakt, iż złożoność zwykle nie zależy wyłącznie od rozmiaru danych, ale może się znacznie różnić dla danych wejściowych o identycznym rozmiarze. Dwoma często stosowanymi sposobami podejścia są: rozpatrywanie przypadków najgorszych (złożoność pesymistyczna) oraz zastosowanie określonego sposobu uśrednienia wszystkich możliwych przypadków (złożoność oczekiwana).

Klasa złożoności

W teorii obliczeń klasa złożoności to zbiór problemów obliczeniowych o podobnej złożoności obliczeniowej.

Na przykład klasa P to zbiór problemów decyzyjnych, które można rozwiązać na maszynie Turinga w czasie wielomianowym, natomiast klasa NP to zbiór problemów decyzyjnych, które można rozwiązać na niedeterministycznej maszynie Turinga w czasie wielomianowym.

6 Wyjaśnić pojęcie algorytmu i dokonać klasyfikacji algorytmów wg wybranych kryteriów

Wikipedia :: Algorytm Wikipedia (ENG) :: Algorytm

Algorytm skończony ciąg jasno zdefiniowanych czynności, koniecznych do wykonania pewnego rodzaju zadań. Algorytm ma przeprowadzić system z pewnego stanu początkowego do pożądanego stanu końcowego.

Algorytm to jednoznaczny przepis obliczenia w skończonym czasie pewnych danych wejściowych do pewnych danych wynikowych.

Klasyfikacja, ze względu na:

- implementację

  rekursywne, iteracyjne algorytm rekursywny wywołuje sam siebie dopóki nie zostaną spełnione pewne warunki; algorytm iteracyjny wykorzystuje pętle i czasem dodatkowe struktury danych aby rozwiązać dany problem

  logiczne

  szeregowe, równoległe, rozproszone

  deterministyczne, niestederministyczne

  dokładne, przybliżeniowe

  kwantowe?

- metodę działania
- obszar problemu
- złożoność

7 Scharakteryzować język programowania i podać wybrane klasyfikacje języków

Język programowania

Podobnie jak języki naturalne, język programowania składa się ze zbiorów reguł syntaktycznych oraz semantyki, które opisują, jak należy budować poprawne wyrażenia oraz jak komputer ma je rozumieć. Język programowania pozwala na precyzyjny zapis algorytmów oraz innych zadań, jakie komputer ma wykonać.

Klasyfikacja pod względem wykorzystanych paradygmatów

- Programowanie proceduralne — zaleca dzielenie kodu na procedury, czyli fragmenty wykonujące ściśle określone operacje,
- Programowanie strukturalne — zaleca hierarchiczne dzielenie kodu na bloki, z jednym punktem wejścia i jednym lub wieloma punktami wyjścia. Chodzi przede wszystkim o nieużywanie (lub ograniczenie) instrukcji skoku (goto),
- Programowanie funkcyjne — funkcje należą do wartości podstawowych, a nacisk kładzie się na wartościowanie (często rekurencyjnych) funkcji, a nie na wykonywanie poleceń,
- Programowanie imperatywne — opisuje proces wykonywania jako sekwencję instrukcji zmieniających stan programu; programy imperatywne składają się z ciągu komend do wykonania przez komputer,
- Programowanie obiektowe — programy definiuje się za pomocą obiektów — elementów łączących stan (czyli dane, nazywane najczęściej polami) i zachowanie (czyli procedury, tu: metody); obiektowy program komputerowy wyrażony jest jako zbiór takich obiektów, komunikujących się pomiędzy sobą w celu wykonywania zadań,
- Programowanie uogólnione (generyczne np. C++, Scala) — pozwala na pisanie kodu programu bez wcześniejszej znajomości typów danych, na których kod ten będzie pracował,
- Programowanie zdarzeniowe — program jest cały czas bombardowany zdarzeniami (events), na które musi odpowiedzieć; przepływ sterowania w programie jest całkowicie niemożliwy do przewidzenia z góry
- Programowanie logiczne (np. Prolog) — odmiana programowania deklaratywnego, w której program podawany jest jako pewien zestaw zależności, a obliczenia są dowodem pewnego twierdzenia w oparciu o te zależności,
- Programowanie aspektowe (np. AspectJ) — wspomaga separację zagadnień i rozdzielenie programu na części w jak największym stopniu niezwiązane funkcjonalnie; oddziela fizycznie kod każdego zagadnienia poprzez umieszczenie ich w oddzielnych aspektach i logiczne zdefiniowanie punktów interakcji pomiędzy nimi,
- Programowanie deklaratywne — w przeciwieństwie do programów napisanych imperatywnie, programista opisuje warunki, jakie musi spełniać końcowe rozwiązanie (co chcemy osiągnąć), a nie szczegółową sekwencję kroków, które do niego prowadzą (jak to zrobić),
- Programowanie agentowe — wyższy od abstrakcji programowania obiektowego; polega on na tworzeniu agentów, które muszą być przygotowane na otrzymanie błędnych danych od innego agenta, lub nieotrzymanie ich w ogóle,
- Programowanie modularne — zaleca stosowanie nadrzędności modułów w stosunku do procedur i bloków tworzących program; moduł grupuje funkcjonalnie związane ze sobą dane oraz procedury i jest reprezentacją obiektu jednokrotnie występującego w programie

8 Porównać język Java oraz C++

Wikipedia (ENG) :: Comparison of Java and C++

- Brak refleksji w C++
- Kompilacja do natywnego kodu a maszyna wirtualna i bajtkod
- C++ wymaga ręcznego zarządzania pamięcią a w Javie jest GC
- Java wymaga stosowania JNI do dostępu do natywnych bibliotek
- ...

9 Scharakteryzować wybrane środowisko programistyczne języka C++

Borland

Eclipse

Visual Studio

DEV C++

- wbudowany kompilator
- wbudowany debuger
- kontrola wersji
- edytor kodu
  - podświetlenie składni
  - podpowiedzi
- tworzenie zasobów programu
- tworzenie baz danych
- tworzenie komponentów

10 Scharakteryzować wybrane środowisko programistyczne języka Java

Eclipse bądź Netbeans. Tak samo jak w 9.
Tylko nie wiem co tu opowiadać :/

11 Omówić dwa wybrane algorytmy sortowania

Sortowanie przez scalanie

Wyróżnić można trzy podstawowe kroki:

1. Podziel zestaw danych na dwie równe części
2. Zastosuj sortowanie przez scalanie dla każdej z nich oddzielnie, chyba że pozostał już tylko jeden element;
3. Połącz posortowane podciągi w jeden.

Dobra implementacja w C jest tylko 20% wolniejsza od quicksort.

Sortowanie bąbelkowe

Polega na porównywaniu dwóch kolejnych elementów i zamianie ich kolejności, jeżeli zaburza ona porządek, w jakim się sortuje tablicę. Sortowanie kończy się, gdy podczas kolejnego przejścia nie dokonano żadnej zmiany.

12 Wyjaśnić zasady działania instrukcji warunkowej i instrukcji pętli w wybranym języku programowania

Wikipedia :: Instrukcja warunkowa Wikipedia :: Pętla

Podstawowym rodzajem instrukcji warunkowej jest If-Then. Jest spotykany w większości języków programowania i umożliwia warunkowe wykonanie określonego bloku kodu, a jeśli warunek nie jest spełniony – alternatywnego bloku. Pomiędzy językami występują nieznaczne różnice składniowe, ale ogólny schemat w pseudokodzie zawsze wygląda następująco:

```
if warunek then
właściwy blok kodu
else
alternatywny blok kodu
end if
```

Na początku wykonywana jest ewaluacja warunku podanego w postaci wyrażenia logicznego. Jeśli wynikiem jest true, wykonywany jest właściwy blok kodu, a jeśli false – alternatywny. Następnie program kontynuuje od pierwszej komendy po end if.

Wiele języków programowania umożliwia zdefiniowanie więcej niż jednego warunku do sprawdzenia przy pomocy opcjonalnego bloku else-if:

```
if warunek 1 then
pierwszy blok kodu
else-if warunek 2 then
drugi blok kodu
else-if warunek 3 then
trzeci blok kodu
else
alternatywny blok kodu
end if
```

W tym wypadku warunki ewaluowane są po kolei do momentu, gdy któryś z nich nie da wartości true – wykonywany jest wtedy przypisany mu blok kodu. Jeśli żaden z warunków nie będzie prawdziwy, wykonywany jest blok alternatywny. Bez względu na ilość prawdziwych warunków, zawsze wykona się tylko pierwszy z nich, a pozostałe zostaną pominięte.

13 Scharakteryzować wybrany język skryptowy

Ruby

Ruby to nowoczesny i dojrzały język programowania. Zapożycza wiele elementów z języków programowania takich jak Perl, Smalltalk, Eiffel, Ada czy Lisp. Łączy w sobie programowanie obiektowe, funkcjonalne oraz imperatywne.

Został opracowany przez Yukihiro "matz" Matsumoto jako narzędzie dla administratorów systemów uniksowych, które miało zastąpić język Perl. Jednak dzięki powstaniu frameworku do tworzenia aplikacji internetowych Ruby on Rails (RoR) Ruby ogromnie zyskał na popularności.

Cechy szczególne języka Ruby

Oto kilka charakterystycznych cech języka Ruby, dzięki którym jest on wygodnym i potężnym narzędziem:

Bloki i lambdy

Bloki i lambdy umożliwiające tak zwane domknięcia pozwalają na zamykanie bloku kodu oraz zmiennych z otaczającego kontekstu i przekazywanie ich innym obiektom. Są to bardzo wygodne i łatwe w użyciu narzędzia do tworzenia wywołań zwrotnych, filtrów, bloków iteracji a nawet elementów własnego języka domenowego (Domain-specific Language[1] — DSL). W odróżnieniu od anonimowych funkcji czy klas znanych z innych języków, dedykowana składnia i możliwość odwoływania się do zmiennych z otaczającego kontekstu umożliwia pisanie w sposób funkcjonalny, niemal w tak elastyczny sposób jak w języku Lisp.

"Duck typing"

Ruby nie wymaga od programisty deklarowania typów zmiennych. Dzięki temu programista może skupić się na logice programu nie martwiąc się o interfejsy, typy, deklaracje i inne elementy kodu potrzebne w statycznie typowanych językach programowania. W Ruby "jak coś kwacze jak kaczka to jest kaczką" co oznacza, że wystarczy żeby przekazany obiekt miał potrzebne metody aby mógł być wykorzystany w danej roli.

Wadą takiego rozwiązania jest to, że kompilator nie prowadzi programisty za rękę. Powoduje to, że programy Ruby wymagają intensywnego automatycznego testowania, a zespoły pracujące przy pojedynczym module lub programie nie mogą być zbyt duże. Osobiście uważam, że w rezultacie uzyskujemy dobrze przetestowany i modułowy kod.

Automatyczne odśmiecanie pamięci

W Ruby programista nie musi się martwić alokacją i zwalnianiem pamięci. Implementacje Ruby wykorzystują nowoczesne implementacje Garbage Collector (GC)[2] takie jak Mark-and-Seweep.

Wszystko jest obiektem

Ruby jest językiem w pełni obiektowym. Każdy element tego języka reprezentowany jest przez obiekt. W wielu językach, liczby i inne typy podstawowe nie są obiektami. Ruby podąża za Smalltalkiem udostępniając metody i zmienne instancji wszystkim swoim typom. To ułatwia korzystanie z języka, ponieważ reguły mające zastosowanie do obiektów odnoszą się również do całego języka.

Wszystkie wyrażenia zwracają wartość

W Ruby każde wyrażenie — nawet takie jak wyrażenia warunkowe — zwraca wartość. W wyniku czego programy napisane w Ruby są bardziej ekspresyjne i treściwe.

Prawda w Ruby

Wszystko co nie jest wartością *nil* lub *false* jest prawdą. Ta prosta zasada powoduje że pisanie wyrażeń warunkowych staje się proste i bardzo intuicyjne.

Wyrażenia logiczne zapożyczone z języka Lisp

Tak jak w języku Lisp argumenty wyrażeń *or* czy *and* są ewaluowane tylko gdy jest to konieczne do ustalenia wartości logicznej danego wyrażenia. W następstwie tego za pomocą operatorów logicznych można pisać proste wyrażenia warunkowe.

Mixin-y

W Ruby niemożliwe jest wielokrotne dziedziczenie. Jest tak ze względu na to, że wielokrotne dziedziczenie bardzo komplikuje język programowania (dobrym przykładem jest język C++). W zamian Ruby oferuje dużo prostszy i wygodniejszy mechanizm zwany *mixin*. Polega to na tym, że do klas bądź pojedynczych obiektów można dołączyć wcześniej zdefiniowany zestaw metod. Umożliwia to rozszerzanie dowolnych klas i obiektów o zestaw nowych zachowań.

[1] http://en.wikipedia.org/wiki/Domain-specific_language
[2] http://en.wikipedia.org/wiki/Garbage_collection_(computer_science)

Meta programing

Ruby umożliwia definiowanie klas, obiektów i metod z poziomu samego języka. Daje to możliwość programowi na dostosowanie interfejsu klasy na przykład do zestawu kolumn obecnych w tabeli bazy danych z którą się komunikuje. Ta możliwość jest kluczem do sukcesu Ruby on Rails. Umożliwia wyeliminowanie potrzeby konfiguracji różnych aspektów programów wdrażając mantrę "konwencja ponad konfiguracją" (ang. convention over configuration).

14 Omówić system stron WWW

15 Scharakteryzować zasady instalacji i administrowania serwerami internetowymi.

- bezpieczeństwo fizyczne komputera
  - zasilanie
  - chłodzenie
  - RAID
  - kontrola fizycznego dostępu
- izolacja procesów - user, chroot...
- minimalna ilość usług
- firewall
- monitoring
- logowanie

16 Porównać systemy operacyjne klasy Windows i Unix

Wywodzą się z innych środowisk i zastosowań.
- desktop PC
- serwery obsługujące wielu użytkowników na raz na sprzęcie "mainframe"

17 Scharakteryzować modele cyklu „życia" systemu informatycznego [3]

Życie systemu – uporządkowany szereg prac (wzajemnie spójnych etapów) wykonanych przy tworzeniu systemu informatycznego. Poprzez analogię do życia ludzkiego wyróżnić możemy trzy podstawowe fazy:

1. Faza narodzin – moment uświadamiania sobie (zgłoszenia) potrzeby istnienia systemu; początek fazy wzrostu
2. Faza wzrostu (eksploatacji) – kolejne działania pozwalające na pełne i skuteczne stworzenie, a następnie użytkowanie systemu
3. Faza śmierci – moment zaprzestania eksploatacji systemu.

W cyklu życia systemu informatycznego najczęściej jednak wyodrębnia się następujące fazy:

1. Analiza wymagań
2. Projekt
3. Implementacja (kodowanie)
4. Testowanie

[3] http://docs8.chomikuj.pl/196923976,PL,0,0,Cykl-zycia-systemu-informatycznego.doc

5. Instalacja
6. Eksploatacja
7. Wycofanie

Model kaskadowy

1. Analiza potrzeb
2. Specyfikacja systemu
3. Projektowanie
4. Programowanie
5. Testowanie
6. Integracja
7. Adaptacja i modyfikacja
8. Eksploatacja
9. Dezaktualizacja

Model Fry'ego

1. Projektowania – realizowane są
   (a) Formułowanie i analiza potrzeb
   (b) Modelowanie konceptualny (opis modelu danych, modelu procesu danych w systemie)
   (c) Projektowanie fizyczne
2. Eksploatacja
   (a) Wdrożenie
   (b) Eksploatacja
   (c) Kontrola
   (d) Modyfikacja i adaptacja.

Model z prototypem

Polega na budowaniu prototypu, który prezentowany jest użytkownikowi celem weryfikacji i na tej podstawie modyfikowany. Takie podejście ma na celu zredukowanie czasu oczekiwania na konkretne rezultaty, zapewnienie szybkiego sprzężenia zwrotnego pomiędzy użytkownikiem a projektantem, a także zaangażowanie użytkownika w projektowanie i analizę potrzeb.

1. Ogólne określenie potrzeb użytkownika
2. Konstruowanie prototypu
3. Użycie i weryfikacja prototypu
4. Modyfikacja prototypu (w zależności od wcześniejszej oceny klienta po wprowadzeniu zmian następuje powrót do punktu poprzedniego lub następnego)
5. Przekształcenie w ostatecznie funkcjonujący system
6. Eksploatacja i modyfikacja systemu

W zależności od wyników prezentacji możemy mieć do czynienia z jednym z dwóch modeli prototypów:

1. Tymczasowy — prototyp odrzucony; stworzony na potrzeby zrozumienie potrzeb klienta; może nigdy nie być przekształcony w ostatecznie funkcjonujący system,
2. Rozwojowy — prototyp przekształcany w ostatecznie funkcjonujący system.

Model spiralny

1. planowania
2. analizy ryzyka
3. konstruowania
4. weryfikacji.

Pracę rozpoczynamy od wstępnych wymagań i planowania projektu, następnie na ich podstawie analizujemy ryzyko i konstruujemy wstępny prototyp, który zostaje poddany weryfikacji przez użytkownika. W kolejnym cyklu planujemy zmiany i analizujemy ryzyko oparte na reakcji użytkownika, po czym tworzymy kolejny prototyp do weryfikacji. Po kolejnych kilku przebiegach należy spodziewać się, że system przyjmie oczekiwaną postać i zakończymy pracę konstrukcją systemu.

18 Omówić etapy tworzenia systemu informatycznego

1. Specyfikacja wymagań
2. Projektowanie
3. Implementacja
4. Testowanie
5. Wdrożenie i pielęgnacja

19 Wyjaśnij pojęcia: baza danych, system zarządzania bazami danych oraz integralność danych.

Baza danych[4]

Zbiór danych zapisanych zgodnie z określonymi regułami. W węższym znaczeniu obejmuje dane cyfrowe gromadzone zgodnie z zasadami przyjętymi dla danego programu komputerowego specjalizowanego do gromadzenia i przetwarzania tych danych.

System zarządzania bazą danych[5]

System zarządzania bazą danych, SZBD (ang. Database Management System, DBMS) – oprogramowanie bądź system informatyczny służący do zarządzania bazą danych. System zarządzania bazą danych może być również serwerem bazy danych (SBD) lub też może udostępniać bazę danych lokalnie – na określonym komputerze.

[4] http://pl.wikipedia.org/wiki/Baza_danych
[5] http://pl.wikipedia.org/wiki/System_zarządzania_bazą_danych

Integralność danych[6]

Funkcja bezpieczeństwa polegająca na tym, że dane nie zostały zmienione, dodane lub usunięte w nieautoryzowany sposób.

W odniesieniu do relacyjnych baz danych integralność definiowana jest jako połączenie trzech koncepcji:

- dokładność (ang. accuracy),
- prawdziwość (ang. correctness),
- oraz aktualność (ang. validity).

[7]Integralność to ograniczenie nakładane na bazę danych przez model relacyjny. Dwie podstawowe reguły integralności to integralność encji (wartość klucza głównego nie może być wartością NULL) oraz integralność odwołań (nie mogą istnieć niedopasowane wartości klucza obcego).

20 Język baz danych SQL

Użycie SQL, zgodnie z jego nazwą, polega na zadawaniu zapytań do bazy danych. Zapytania można zaliczyć do jednego z czterech głównych podzbiorów:

SQL DML (ang. Data Manipulation Language – „język manipulacji danymi"),
SQL DDL (ang. Data Definition Language – „język definicji danych"),
SQL DCL (ang. Data Control Language – „język kontroli nad danymi").
SQL DQL (ang. Data Query Language – „język definiowania zapytań").

Instrukcje SQL w obrębie zapytań tradycyjnie zapisywane są wielkimi literami, jednak nie jest to wymóg. Każde zapytanie w SQL-u musi kończyć się znakiem średnika (;).

Dodatkowo, niektóre programy do łączenia się z silnikiem bazy danych (np. psql w przypadku PostgreSQL), używają swoich własnych instrukcji, spoza standardu SQL, które służą np. do połączenia się z bazą, wyświetlenia dokumentacji itp.

20.1 DML

DML (Data Manipulation Language) służy do wykonywania operacji na danych – do ich umieszczania w bazie, kasowania, przeglądania oraz dokonywania zmian. Najważniejsze polecenia z tego zbioru to:

INSERT – umieszczenie danych w bazie,
UPDATE – zmiana danych,
DELETE – usunięcie danych z bazy.

Dane tekstowe muszą być zawsze ujęte w znaki pojedynczego cudzysłowu (').

20.2 DDL

Dzięki DDL (Data Definition Language) można operować na strukturach, w których dane są przechowywane – czyli np. dodawać, zmieniać i kasować tabele lub bazy. Najważniejsze polecenia tej grupy to:

CREATE (np. CREATE TABLE, CREATE DATABASE, ...) – utworzenie struktury (bazy, tabeli, indeksu itp.),
DROP (np. DROP TABLE, DROP DATABASE, ...) – usunięcie struktury,
ALTER (np. ALTER TABLE ADD COLUMN ...) – zmiana struktury (dodanie kolumny do tabeli, zmiana typu danych w kolumnie tabeli).

20.3 DCL

DCL (Data Control Language) ma zastosowanie do nadawania uprawnień do obiektów bazodanowych. Najważniejsze polecenia w tej grupie to:

GRANT - służące do nadawania uprawnień do pojedynczych obiektów lub globalnie konkretnemu użytkownikowi (np. GRANT ALL PRIVILEGES ON EMPLOYEE TO PIOTR WITH GRANT OPTION – przyznanie wszystkich praw do tabeli EMPLOYEE użytkownikowi PIOTR z opcją pozwalającą mu nadawać prawa do tej tabeli).

[6]http://pl.wikipedia.org/wiki/Integralność_danych
[7]http://pl.wikipedia.org/wiki/Model_relacyjny#Integralno.C5.9B.C4.87

REVOKE – służące do odbierania wskazanych uprawnień konkretnemu użytkownikowi (np. REVOKE ALL PRIVILEGES ON EMPLOYEE FROM PIOTR - odebranie użytkownikowi wszystkich praw do tabeli EMPLOYEE).

DENY.

20.4 DQL

DQL (Data Query Language) to język formułowania zapytań do bazy danych. W zakres tego języka wchodzi jedno polecenie - SELECT. Często SELECT traktuje się jako część języka DML, ale to podejście nie wydaje się właściwe, ponieważ DML z definicji służy do manipulowania danymi - ich tworzenia, usuwania i uaktualniania.

Na pograniczu obu języków znajduje się polecenie SELECT INTO, które dodatkowo modyfikuje (przepisuje, tworzy) dane.

20.5 Przykładowe zapytania

Przykłady użycia wyżej wymienionych rodzajów zapytań:

SELECT *
FROM pracownicy
WHERE pensja > 2000
ORDER BY staz DESC;

Zwraca tabelę (listę) utworzoną ze wszystkich kolumn (*) tabeli „pracownicy" (FROM pracownicy) zawierającą pracowników, których pensja jest większa niż 2000 (WHERE pensja > 2000) i sortuje wynik malejąco według parametru staz (ORDER BY staz DESC).

INSERT INTO pracownicy
(imie, nazwisko, pensja, staz)
VALUES
('Jan', 'Kowalski', 5500, 1);

Dodaje do tabeli „pracownicy" (INTO pracownicy) wiersz (rekord) zawierający dane pojedynczego pracownika.

UPDATE pracownicy
SET pensja = pensja * 1.1
WHERE staz > 2;

Podwyższa o 10% pensję (SET pensja = pensja * 1.1) pracownikom, których staż jest większy niż 2 (np. lata).

DELETE FROM pracownicy
WHERE imie = 'Jan' AND nazwisko = 'Kowalski';

Usuwa z tabeli „pracownicy" wszystkie wiersze (rekordy) dotyczące pracownika o imieniu „Jan" i nazwisku „Kowalski" (czyli takie, w których pole "imię" ma wartość Jan, a pole "nazwisko" – Kowalski).

CREATE TABLE pracownicy
(
imie VARCHAR(255),
nazwisko VARCHAR(255),
pensja FLOAT,
staz INT
);

Tworzy tabelę „pracownicy" zawierającą pola tekstowe zmiennej długości (varchar) o nazwach „imie" (imię) i „nazwisko", o maksymalnej długości 255 znaków, zapisaną za pomocą liczby rzeczywistej (float od ang. floating point) pensję oraz zapisany za pomocą liczby całkowitej (int od ang. integer) staż.

DROP TABLE pracownicy;

Usuwa z bazy tabelę „pracownicy".

ALTER TABLE pracownicy
ADD dzial VARCHAR(255);

Dodaje do struktury tabeli „pracownicy" kolumnę „dzial" (dział), jako pole tekstowe o długości maks. 255 bajtów.

21 Instrukcje definiujące dane w SQL

Dzięki DDL (Data Definition Language) można operować na strukturach, w których dane są przechowywane – czyli np. dodawać, zmieniać i kasować tabele lub bazy. Najważniejsze polecenia tej grupy to:

- CREATE (np. CREATE TABLE, CREATE DATABASE, ...) – utworzenie struktury (bazy, tabeli, indeksu itp.),
- DROP (np. DROP TABLE, DROP DATABASE, ...) – usunięcie struktury,
- ALTER (np. ALTER TABLE ADD COLUMN ...) – zmiana struktury (dodanie kolumny do tabeli, zmiana typu danych w kolumnie tabeli).

22 Zapytania wybierające SQL

Patrz wyżej DQL.

23 Relacyjne bazy danych[8]

Model relacyjny – model organizacji danych bazujący na matematycznej teorii mnogości, w szczególności na pojęciu relacji. Na modelu relacyjnym oparta jest relacyjna baza danych (ang. Relational Database) – baza danych, w której dane są przedstawione w postaci relacyjnej.

W najprostszym ujęciu w modelu relacyjnym dane grupowane są w relacje, które reprezentowane są przez tablice. Relacje są pewnym zbiorem rekordów o identycznej strukturze wewnętrznie powiązanych za pomocą związków zachodzących pomiędzy danymi. Relacje zgrupowane są w tzw. schematy bazy danych. Relacją może być tabela zawierająca dane teleadresowe pracowników, zaś schemat może zawierać wszystkie dane dotyczące firmy. Takie podejście w porównaniu do innych modeli danych ułatwia wprowadzanie zmian, zmniejsza możliwość pomyłek, ale dzieje się to kosztem wydajności.

System zarządzania relacyjną bazą danych (ang. Relational Database Management System, RDBMS) – to zestaw programów służących do korzystania z bazy danych opartej na modelu relacyjnym. Większość wewnętrznych języków RDBMS jest w pewnym stopniu zgodna ze standardem języka zapytań SQL. Język ten doczekał się już dwóch standardów – SQL92 i SQL99, jednak różnice pomiędzy teoretycznie SQL-owymi systemami są zbyt duże, żeby możliwe było przeniesienie nawet względnie prostej aplikacji z jednego systemu na drugi.[9]

24 Diagram związków encji

Wikipedia :: Diagram związków encji

Diagram związków encji lub Diagram ERD (od ang. Entity-Relationship Diagram) – rodzaj graficznego przedstawienia związków pomiędzy encjami używany w projektowaniu systemów informacyjnych do przedstawienia konceptualnych modeli danych używanych w systemie.

Systemy CASE, które wspierają tworzenia tych diagramów, mogą na ich podstawie automatycznie tworzyć bazy danych odpowiadające relacjom na diagramie.

Diagram pokazuje logiczne związki pomiędzy różnymi encjami, związki te mają dwie cechy:

Opcjonalność – która mówi o tym, czy każda encja musi, czy też może wystąpić równocześnie z inną. Np. TOWAR musi zostać zakupiony przez co najmniej jednego KLIENTA, ale KLIENT może być nabywcą TOWARU. W reprezentacji graficznej linia przerywana oznacza opcjonalność związku, natomiast ciągła wymóg związku.

Krotność – określającą ile encji wchodzi w skład związku:

1:1 ("jeden do jeden") – encji odpowiada dokładnie jedna encja,

1:N ("jeden do wielu") – encji odpowiada jedna lub więcej encji,

M:N ("wiele do wielu") – jednej lub więcej encjom odpowiada jedna lub więcej encji.

W przypadku związków M:N często stosuje się normalizację diagramu, która polega na dodaniu encji pośredniczącej i zastąpienie związku M:N dwoma związkami 1:N z nową encją.

25 Normalizacja baz danych[10]

Postać normalna – postać relacji w bazie danych, w której nie występuje redundancja (nadmiarowość), czyli powtarzanie się tych samych informacji. Doprowadzeniu relacji do postaci normalnej służy normalizacja bazy danych.

[8] http://pl.wikipedia.org/wiki/Relacyjny_model_danych
[9] http://pl.wikipedia.org/wiki/System_zarządzania_relacyjną_bazą_danych
[10] http://pl.wikipedia.org/wiki/Postać_normalna_(bazy_danych)

Pierwsza postać normalna (1NF)

Relacja jest w pierwszej postaci normalnej, jeśli:

- opisuje jeden obiekt,
- wartości atrybutów są elementarne (atomowe, niepodzielne) - każda kolumna jest wartością skalarną (atomową), a nie macierzą lub listą czy też czymkolwiek, co posiada własną strukturę,
- nie zawiera kolekcji (powtarzających się grup informacji),
- posiada klucz główny,
- kolejność wierszy może być dowolna (znaczenie danych nie zależy od kolejności wierszy).

Właściwości, które muszą zaistnieć w 1 formie :

1. Jest zdefiniowany klucz relacji.
2. Wszystkie atrybuty niekluczowe są w zależności funkcyjnej od klucza.

Druga postać normalna (2NF)

Relacja jest w drugiej postaci normalnej wtedy i tylko wtedy, gdy jest w I postaci normalnej i każda kolumna zależy funkcyjnie od całego klucza głównego (a nie np. od części klucza).

Trzecia postać normalna (3NF)

Mamy z nią do czynienia wtedy i tylko wtedy, gdy tabela jest w 2NF oraz gdy wszystkie pola niebędące polami klucza głównego są od niego zależne bezpośrednio.

Postać normalna Boyce'a-Codda (BCNF lub 3.5NF)

W tej postaci zależności funkcyjne muszą mieć następującą postać: jeżeli $X \rightarrow A$ i atrybut A nie jest zawarty w X, to X jest kluczem lub zawiera klucz.

Czwarta postać normalna (4NF)

Relacja jest w czwartej postaci normalnej, jeżeli zbiór atrybutów X określa wartościowo Y, to zachodzi jeden z następujących warunków (trywialne zależności wielowartościowe)

- Y jest puste lub zawiera się w X,
- suma zbiorów X i Y jest pełnym zbiorem atrybutów,
- X zawiera klucz.

Ponadto 4PN zachodzi wówczas gdy:

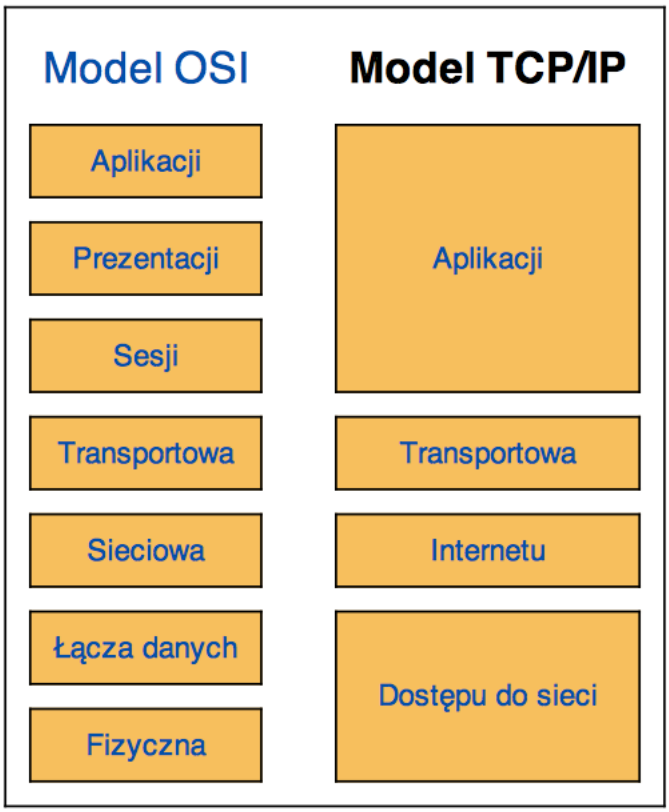
- spełnione są warunki PNB-C
- istnieją nietrywialne zależności gdzie Y wynika z X, X i Y są rozdzielne oraz X jest kluczem.

Czwarta i piąta postać normalna są w zasadzie używane wyłącznie przy okazji rozważań teoretycznych.

26 Narzędzia typu CASE

Wikipedia :: CASE Wikipedia :: Narzędzia UML

27 Scharakteryzować wielowarstwowy model sieci komputerowej



Model TCP/IP[11]

Teoretyczny model warstwowej struktury protokołów komunikacyjnych. Model TCP/IP został stworzony w latach 70. XX wieku w DARPA, aby pomóc w tworzeniu odpornych na atak sieci komputerowych. Potem stał się podstawą struktury Internetu.

Podstawowym założeniem modelu TCP/IP jest podział całego zagadnienia komunikacji sieciowej na szereg współpracujących ze sobą warstw (ang. layers). Każda z nich może być tworzona przez programistów zupełnie niezależnie, jeżeli narzucimy pewne protokoły według których wymieniają się one informacjami. Założenia modelu TCP/IP są pod względem organizacji warstw zbliżone do modelu OSI. Jednak liczba warstw jest mniejsza i bardziej odzwierciedla prawdziwą strukturę Internetu. Model TCP/IP składa się z czterech warstw.

Warstwa aplikacji

Warstwa procesowa czy warstwa aplikacji (ang. process layer) to najwyższy poziom, w którym pracują użyteczne dla człowieka aplikacje takie jak np. serwer WWW czy przeglądarka internetowa. Obejmuje ona zestaw gotowych protokołów, które aplikacje wykorzystują do przesyłania różnego typu informacji w sieci. Wykorzystywane protokoły to m.in.: HTTP, Telnet, FTP, TFTP, SNMP, DNS, SMTP, X Windows.

Warstwa transportowa

Warstwa transportowa (ang. host-to-host layer) gwarantuje pewność przesyłania danych oraz kieruje właściwe informacje do odpowiednich aplikacji. Opiera się to na wykorzystaniu portów określonych dla każdego połączenia. W jednym komputerze może istnieć wiele aplikacji wymieniających dane z tym samym komputerem w sieci i nie nastąpi wymieszanie się przesyłanych przez nie danych. To właśnie ta warstwa nawiązuje i zrywa połączenia między komputerami oraz zapewnia pewność transmisji.

[11] http://pl.wikipedia.org/wiki/Model_TCP/IP

Warstwa Internetu

Warstwa Internetu lub warstwa protokołu internetowego (ang. internet protocol layer) to sedno działania Internetu. W tej warstwie przetwarzane są datagramy posiadające adresy IP. Ustalana jest odpowiednia droga do docelowego komputera w sieci. Niektóre urządzenia sieciowe posiadają tę warstwę jako najwyższą. Są to routery, które zajmują się kierowaniem ruchu w Internecie, bo znają topologię sieci. Proces odnajdywania przez routery właściwej drogi określa się jako trasowanie.

Warstwa dostępu do sieci

Warstwa dostępu do sieci lub warstwa fizyczna (ang. network access layer) jest najniższą warstwą i to ona zajmuje się przekazywaniem danych przez fizyczne połączenia między urządzeniami sieciowymi. Najczęściej są to karty sieciowe lub modemy. Dodatkowo warstwa ta jest czasami wyposażona w protokoły do dynamicznego określania adresów IP.

Model OSI[12]

OSI (ang. Open System Interconnection) lub Model OSI (pełna nazwa ISO OSI RM, ang. ISO OSI Reference Model – model odniesienia łączenia systemów otwartych) – standard zdefiniowany przez ISO oraz ITU-T opisujący strukturę komunikacji sieciowej.

Model ISO OSI RM jest traktowany jako model odniesienia (wzorzec) dla większości rodzin protokołów komunikacyjnych. Podstawowym założeniem modelu jest podział systemów sieciowych na 7 warstw (ang. layers) współpracujących ze sobą w ściśle określony sposób.

Warstwa 7: aplikacji

Warstwa aplikacji jest warstwą najwyższą, zajmuje się specyfikacją interfejsu, który wykorzystują aplikacje do przesyłania danych do sieci (poprzez kolejne warstwy modelu ISO/OSI). W przypadku sieci komputerowych aplikacje są zwykle procesami uruchomionymi na odległych hostach. Interfejs udostępniający programistom usługi dostarczane przez warstwę aplikacji opiera się na obiektach nazywanych gniazdami (ang. socket).

Warstwa 6: prezentacji

Podczas ruchu w dół zadaniem warstwy prezentacji jest przetworzenie danych od aplikacji do postaci kanonicznej (ang. canonical representation) zgodnej ze specyfikacją OSI-RM, dzięki czemu niższe warstwy zawsze otrzymują dane w tym samym formacie. Kiedy informacje płyną w górę, warstwa prezentacji tłumaczy format otrzymywanych danych na zgodny z wewnętrzną reprezentacją systemu docelowego. Wynika to ze zróżnicowania systemów komputerowych, które mogą w różny sposób interpretować te same dane. Dla przykładu bity w bajcie danych w niektórych procesorach są interpretowane w odwrotnej kolejności niż w innych. Warstwa ta odpowiada za kodowanie i konwersję danych oraz za kompresję / dekompresję; szyfrowanie / deszyfrowanie. Warstwa prezentacji obsługuje np. MPEG, JPG, GIF itp.

Warstwa 5: sesji

Warstwa sesji otrzymuje od różnych aplikacji dane, które muszą zostać odpowiednio zsynchronizowane. Synchronizacja występuje między warstwami sesji systemu nadawcy i odbiorcy. Warstwa sesji "wie", która aplikacja łączy się z którą, dzięki czemu może zapewnić właściwy kierunek przepływu danych – nadzoruje połączenie. Wznawia je po przerwaniu.

Warstwa 4: transportowa

Warstwa transportowa segmentuje dane oraz składa je w tzw. strumień. Warstwa ta zapewnia całościowe połączenie między stacjami: źródłową oraz docelową, które obejmuje całą drogę transmisji. Następuje tutaj podział danych na części, które są kolejno numerowane i wysyłane do docelowej stacji. Na poziomie tej warstwy do transmisji danych wykorzystuje się dwa protokoły TCP (ang. Transmission Control Protocol) oraz UDP (ang. User Datagram Protocol).

[12] http://pl.wikipedia.org/wiki/Model_OSI

Warstwa 3: sieciowa

Warstwa sieciowa jako jedyna dysponuje wiedzą dotyczącą fizycznej topologii sieci. Rozpoznaje, jakie drogi łączą poszczególne komputery (trasowanie) i decyduje, ile informacji należy przesłać jednym z połączeń, a ile innym. Jeżeli danych do przesłania jest zbyt wiele, to warstwa sieciowa po prostu je ignoruje. Nie musi zapewniać pewności transmisji, więc w razie błędu pomija niepoprawne pakiety danych. Standardowa paczka danych czasami oznaczana jest jako NPDU (ang. Network Protocol Data Unit). Nie znajdują się w nim żadne użyteczne dla użytkowników aplikacje. Jedyne jego zadanie, to zapewnienie sprawnej łączności między bardzo odległymi punktami sieci. Routery są podstawą budowy rozległych sieci informatycznych takich jak Internet, bo potrafią odnaleźć najlepszą drogę do przekazania informacji. Warstwa sieciowa podczas ruchu w dół umieszcza dane wewnątrz pakietów zrozumiałych dla warstw niższych (kapsułkowanie). Jednocześnie warstwa sieci używa czterech procesów (adresowanie, enkapsulacja, routing, dekapsulacja). Protokoły warstwy sieci to: (IPv4, IPv6, ICMP, NOVELL IPX, APPLE TALK, CLNS/DECN et).

Warstwa 2: łącza danych

Warstwa łącza danych jest czasami nazywana warstwą liniową lub kanałową. Ma ona nadzorować jakość przekazywanych informacji. Nadzór ten dotyczy wyłącznie warstwy niższej. Warstwa łącza danych ma możliwość zmiany parametrów pracy warstwy fizycznej, tak aby obniżyć liczbę pojawiających się podczas przekazu błędów. Zajmuje się pakowaniem danych w ramki i wysyłaniem do warstwy fizycznej. Rozpoznaje błędy związane z niedotarciem pakietu oraz uszkodzeniem ramek i zajmuje się ich naprawą. Podczas ruchu w dół w warstwie łącza danych zachodzi enkapsulacja pakietów z warstwy sieciowej tak, aby uzyskać ramki zgodne ze standardem. Czasami są one oznaczane jako LPDU (ang. data Link Protocol Data Unit).

28 Przedstawić klasyfikację sieci komputerowych

Lan networks

Podział sieci ze względu na obejmowany obszar:

- lokalna sieć (LAN - Local Area Network) - łączy stacje robocze na niewielkim obszarze(pomieszczenie, piętro, budynek), charakteryzuje się niskimi kosztami eksploatacji oraz dołączania kolejnych komputerów oraz prostym oprogramowaniem.
- sieć kampusowa (Campus Network) - obejmuje zasięgiem kilka budynków na terenie uczelni, firmy lub osiedla.
- miejska sieć (MAN - Metropolitan Area Network)- integruje sieci LAN na obszarze całego miasta , oparta jest na laczach o duzej przepustowości np. na światłowodach.
- rozległa sieć (WAN - Wide Area network) - łączy wiele sieci typu MAN i WAN na dużych terenach geograficznych np. region, państwo, kontynent. Wykorzystuje publiczną sieć telekomunikacyjną ale w wielu przypadkach również kanały satelitarne lub radiowe.
- sieć radiowa (Radio Network) sieć do której łączenia nie korzysta się z przewodów, do każdej stacji dołączany jest nadajnik - odbiornik, zasięg takich sieć jest być uwarunkowany mocą nadajniko-odbiorników
- sieć satelitarna (Satellite Network) sieć, w której sygnały przekazywane są ze stacji naziemnej do satelity i z satelity do innej stacji naziemnej, sygnał jest wzmacniany w Satelicie.

29 Omówić podstawowe narzędzia administratora sieci komputerowej

ping[13]

ping — nazwa programu używanego w sieciach komputerowych TCP/IP (takich jak Internet) służącego do diagnozowania połączeń sieciowych. Pozwala na sprawdzenie czy istnieje połączenie pomiędzy hostami testującym i testowanym. Umożliwia on zmierzenie liczby zgubionych pakietów oraz opóźnień w ich transmisji, zwanych lagami.

[13] http://pl.wikipedia.org/wiki/Ping

traceroute[14]

traceroute - program służący do badania trasy pakietów w sieci IP. Program traceroute jest szeroko dostępny we wszystkich uniksowych systemach operacyjnych; jest także dostępny program mtr, który łączy funkcjonalność traceroute z narzędziem ping. Istnieje również program tracert o podobnej funkcjonalności, zawarty w systemach z rodziny Microsoft Windows.

route[15]

route – program narzędziowy w systemach uniksowych oraz Windows, który wyświetla i umożliwia zmiany tablicy trasowania pakietów sieciowych.

ip[16]

ip - polecenie do konfiguracji interfejsów sieciowych, tablic tras czy tuneli w systemach operacyjnych Linux. W pełni wspiera protokoły IPv4 i IPv6, działa też z gniazdami BSD. Polecenie wchodzi w skład pakietu Iproute2 - narzędzi do zarządzania siecią i kontroli ruchu sieciowego (QoS) w systemach Linux. Iproute2 zastępuje starsze narzędzia ifconfig czy route jeszcze zachowane w dystrybucjach dla kompatybilności.

iptables[17]

iptables to program sterujący filtrem pakietów (głównie używanym jako zapora sieciowa bądź NAT) opracowany dla systemu operacyjnego Linux.

Program może być używany jako filtr pakietów, bądź tzw. stanowa zapora dla systemów Linux z jądrem począwszy od serii 2.4.x, kontrolujący połączenia wchodzące i wychodzące do sieci komputerowej lub stacji roboczej.

30 Scharakteryzować politykę bezpieczeństwa w systemach komputerowych

Środki do zabezpieczania systemów informacyjnych i sieci:

- Techniczne: kopie zapasowe, odtwarzanie po awariach, konfiguracja systemu operacyjnego i baz danych, aktualizacja oprogramowania, itp.,
- Kryptograficzne: poufność, integralność, niemożność zaprzeczenia.
- Kryptograficzne i systemowe: autoryzacja, audyt, uwierzytelnianie.
- Kryptograficzne, protokolarne, usługowe: śluzy ogniowe, tunelowanie, zabezpieczenia sieci TCP/IP: IP i IPSEC, SSL/TSL, https, ssh, itp.
- Uwierzytelnianie i identyfikacja w sieciach: karty inteligentne, KERBEROS, DCE, SESAME, i inne.

31 Wyjaśnij terminy: identyfikacja, uwierzytelnienie, autoryzacja, spójność danych, ślad kontroli

Identyfikacja[18]

Podmiot deklaruje swoją tożsamość.

[14] http://pl.wikipedia.org/wiki/Traceroute
[15] http://pl.wikipedia.org/wiki/Route
[16] http://pl.wikipedia.org/wiki/Ip_(Linux)
[17] http://pl.wikipedia.org/wiki/Iptables
[18] http://pl.wikipedia.org/wiki/Uwierzytelnianie

Uwierzytelnianie

Strona ufająca stosuje odpowiednią technikę uwierzytelniania (authentication mechanism) w celu weryfikacji zadeklarowanej wcześniej tożsamości.

Autoryzacja[19]

Potwierdzenie, czy dany podmiot jest uprawniony do uzyskania dostępu do żądanego zasobu. Na tym etapie autentyczność podmiotu jest już potwierdzona, nie musi on jednak być uprawnionym do uzyskania dostępu w żądanym zakresie.

Celem autoryzacji jest kontrola dostępu (access control), która potwierdza, czy dany podmiot jest uprawniony do korzystania z żądanego zasobu. Dla określenia uprawnień danego podmiotu konieczne jest najpierw stwierdzenie jego tożsamości, dlatego w typowym zastosowaniu autoryzacja następuje dopiero po potwierdzeniu tożsamości podmiotu za pomocą identyfikacji i uwierzytelnienia.

Integralność danych[20]

Funkcja bezpieczeństwa polegająca na tym, że dane nie zostały zmienione, dodane lub usunięte w nieautoryzowany sposób.

W technice informatycznej i telekomunikacyjnej ochrona integralności zapobiega przypadkowemu zniekształceniu danych podczas odczytu, zapisu, transmisji lub magazynowania. Wykorzystuje się tutaj sumy kontrolne i kody korekcyjne takie jak CRC.

W bezpieczeństwie teleinformatycznym ochrona integralności zapobiega celowej modyfikacji danych dokonanej z użyciem zaawansowanych technik, mających na celu ukrycie faktu dokonania zmiany. Wykorzystuje się tutaj techniki kryptograficzne takie jak kody MAC odporne na celowe manipulacje. W odniesieniu do relacyjnych baz danych integralność definiowana jest jako połączenie trzech koncepcji: dokładność (ang. accuracy), prawdziwość (ang. correctness), oraz aktualność (ang. validity).

Ślad kontroli

System przechowuje historię operacji dokonywanych przez użytkowników.

32 [TODO] Scharakteryzować komponenty zintegrowanych programów bezpieczeństwa w systemach komputerowych

WTF?

33 Podstawowe modele kontroli dostępu do systemów operacyjnych i baz danych.

DAC określa podstawowe metody kontroli dostępu do obiektów w systemie plików. Administratorem uprawnień jest sysadmin lub właściciel obiektu (pliku). Rozbudowana wersją DAC jest model ACL – list kontroli dostępu.

MAC ochrania i zabezpiecza procesy, dane i urządzenia systemowe przed szkodliwym nadużyciem/wykorzystaniem za pomocą tzw. etykiet. Właściciel obiektu po jego utworzeniu może nie mieć prawa do późniejszego np. odczytu!

RBAC ogranicza dostęp do zasobów na podstawie roli, jaką użytkownik pełni w systemie. Przystosowany do organizacji, w której działa. Ukierunkowany na czynności użytkownika. Może wykorzystywać DAC, MAC i inne modele.

[19] http://pl.wikipedia.org/wiki/Autoryzacja_(informatyka)
[20] http://pl.wikipedia.org/wiki/Integralność_danych

przejmij - przekaż

- Rozszerzenie modelu macierzowego (DAC) – ujmuje kwestie delegacji i przejmowania uprawnień.
- Informacja o uprawnieniach jest pamiętana w postaci grafu, który opisuje wszystkie powiązania dot. przepływu informacji.
- Podmiot ma (w tym modelu!) uprawnienia do obiektu albo innego podmiotu.
  - Przekaż - podmiot x przekazuje uprawnienie h, jakie ma do z (obiektu lub podmiotu), obiektowi lub podmiotowi y pod warunkiem, że x ma uprawnienie g do przekazywania uprawnień.
  - Przejmij - podmiot x przekazuje uprawnienie h (pozbywając się uprawnienia h do obiektu z), jakie ma do z (obiektu lub podmiotu), obiektowi lub podmiotowi y pod warunkiem, że y ma uprawnienie t do przejęcia uprawnień.

Wooda Skierowany na realizację polityki bezpieczeństwa w środowisku baz danych.

- poziomy
  - Zewnętrzny,
  - Konceptualny,
  - Wewnętrzny.
- podmioty to:
  - Użytkownicy,
  - administratorzy uprawnień.
- obiekty to
  - tablice bazy danych
  - kolumny tablic
  - tablice wirtualne (widoki)
  - ew. funkcje.
- Dla poziomu zewnętrznego i konceptualnego są definiowane reguły dostępu, które muszą być niesprzeczne.

Jajodhi-Sadhu Kontrola dostępu MAC dla baz danych.

- Do modelu relacyjnego są wprowadzone atrybuty klasyfikacyjne zw. etykietami bezpieczeństwa: jawny $<$ poufny $<$ tajny $<$ ściśle tajny
- Schemat relacyjny wielopoziomowy: Każdy atrybut oprócz wartości ma nadany atrybut klasyfikacyjny, a ponadto cała krotka ma również taki atrybut.
- Polityka bezpieczeństwa to kontrola dostępu do realizacji operacji typu dołącz, aktualizuj, usuń na relacjach wielopoziomowych.

34 Porównanie kryptografii symetrycznej i asymetrycznej

Cecha	Symetryczne
Klucz	ten sam klucz do szyfrowania i deszyfrowania
Długość klucza	typowo: 112-128 bitów
Szybkość szyfrowania	Dziesiątki Gb/s, a kilkaset razy wolniej dla realizacji programowych
Podstawy teoretyczne	szyfry podstawieniowe i przestawieniowe, przekształcenia jednokierunkowe
Główne zastosowania	zabezpieczanie łączy komunikacyjnych, transmisja danych i uwierzytelnianie

podpisy cyfrowe, wymi

35 Scharakteryzować podpis elektroniczny[21]

Matematyczny sposób potwierdzania autentyczności cyfrowego dokumentu. Istnieje wiele schematów podpisów cyfrowych, obecnie jednak najpopularniejszym jest schemat podpisu dokumentów cyfrowych w systemach kryptograficznych z kluczem publicznym i jednokierunkową funkcją skrótu - w systemie tym do oryginalnej wiadomości dołączany jest skrót dokumentu, zaszyfrowany prywatnym kluczem nadawcy. Potwierdzenie autentyczności wiadomości jest możliwe po odszyfrowaniu skrótu kluczem publicznym nadawcy i porównaniu go z wytworzonym skrótem odebranego dokumentu.

36 Charakterystyka protokołów uwierzytelniania

- w jakim celu stosujemy:
  - chroni dokument przed podrobieniem,
  - TTP może ocenić i zadecydować o ważności dokumentu (metody symetryczne).
- 2 podstawowe sposoby:
  - symetryczne - w obrębie grupy osób, które sobie ufają,
  - asymetryczne (za pomocą podpisów cyfrowych).
- Uwierzytelnianie symetryczne - technologia: wiarygodność źródła (pochodzenia) informacji jest zastępowana przez obliczenie skrótu informacji i kryptograficzną ochronę tego skrótu - możliwości:
  - utajnienie algorytmu skrótu albo
  - utajnienie wartości skrótu albo
  - autentyczność wartości skrótu (nie jest to klasyczny podpis).

36.1 uwierzytelnianie symetryczne

- Podstawowy protokół symetryczny:
  - A wysyła do B: M, EK(H(M)), gdzie: K – klucz współdzielony przez A i B, H(M) – skrót wiadomości M,
  - B oblicza z M skrót H(M) i porównuje z DK(EK(H(M)));
    jeśli oba skróty są zgodne, to wiadomość jest autentyczna (wiarygodna).
- Obliczanie skrótu wiadomości - podstawowe techniki:
  - MDC - Message Digest Cipher - funkcje skrótu bezkluczowe - muszą być dobrej jakości;
  - MAC - Message Authentication Code - skrót z tajnym kluczem K może być wytworzony przez kogoś, kto zna K.
- nie wolno stosować tego samego klucza do MAC i do szyfrowania danych!!!
- jako funkcji skrótu można użyć:
  - dowolnego dobrego szyfru blokowego w trybie np. CBC z wektorem inicjalnym IV=0, bo obliczamy skrót!
  - inne popularne algorytmy MAC to:
    - CBC-MAC (DES w trybie CBC - MAC to 32 pierwsze bity wyniku)
    - HMAC: h((K+p2), h((K+p1), x), gdzie pi to ustalone łańcuchy, h to np. funkcja RIPEMD-160
- W przypadku MDC skrót jest szyfrowany za pomocą klucza tajnego (współdzielonego przez grupę użytkowników lub przez użytkownika i TTP) dla algorytmu symetrycznego. Dla MAC może jeszcze wystąpić dodatkowe szyfrowanie, ale nie musi.
- Problemy - ograniczone zastosowania, duża liczba kluczy

[21] http://pl.wikipedia.org/wiki/Podpis_cyfrowy

36.2 uwierzytelnianie asymetryczne

- Podstawowa technika - podpisy cyfrowe. Zasadniczo nie podpisuje się całych wiadomości, Dlaczego?
  - czas realizacji podpisu,
  - możliwość ataków, gdy podpisywane wiadomości są krótkie (RSA).
- Realizacja podpisu Alice:

  S A = E Kp (H(M)); do wysłania dodatkowo można go zaszyfrować razem z wiadomością (Kp to klucz prywatny Alice); ale na ogół wysyłamy: M, SA
- Pytanie: jak weryfikujemy podpis? (co musi zrobić Bob, aby sprawdzić poprawność podpisu?) Czego używamy? Jaka jest ewentualna rola TTP?
- Problemy:
  - zarządzanie kluczami publicznymi,
  - słabości niektórych algorytmów.
- Stosowane algorytmy: RSA (1978), ElGamal(1985), DSA(1994)
- Przyszłość: algorytmy oparte na logarytmie dyskretnym w obrębie grupy krzywych eliptycznych w ciele Galois GF(p),
- Zaleta: krótsze klucze - ok. 200 bitów w stos. do 1024 (RSA)

37 Omówić cechy systemów rozproszonych[22]

To zbiór niezależnych urządzeń technicznych połączonych w jedną, spójną logicznie całość. Zwykle łączonymi urządzeniami są komputery, rzadziej – systemy automatyki. Połączenie najczęściej realizowane jest przez sieć komputerową, jednak można wykorzystać również inne – prostsze – magistrale komunikacyjne. Urządzenia są wyposażone w oprogramowanie umożliwiające współdzielenie zasobów systemowych.

1. Dzielenie zasobów (ang. resource sharing) – wielu użytkowników systemu może korzystać z danego zasobu (np. drukarek, plików, usług, itp.).
2. Otwartość (ang. openness) – podatność na rozszerzenia, możliwość rozbudowy systemu zarówno pod względem sprzętowym, jak i oprogramowania.
3. Współbieżność (ang. concurrency) – zdolność do przetwarzania wielu zadań jednocześnie.
4. Skalowalność (ang. scalability) – cecha systemu umożliwiająca zachowanie podobnej wydajności systemu przy zwiększaniu skali systemu (np. liczby procesów, komputerów, itp.).
5. Tolerowanie awarii (ang. fault tolerance) – właściwość systemu umożliwiająca działania systemu mimo pojawiania się błędów i (lub) uszkodzeń (np. przez utrzymywanie nadmiarowego sprzętu).
6. Przezroczystość (ang. transparency) – właściwość systemu powodująca postrzeganie systemu przez użytkownika jako całości, a nie poszczególnych składowych.

38 Scharakteryzować przykładowe architektury systemów rozproszonych

Klient-serwer: rozproszony system ma wyróżniony węzeł zwany serwerem, oraz szereg podłączonych do niego węzłów zwanych klientami.

- Związek nie jest symetryczny: serwer wykonuje usługi zlecane przez klientów, nie może im odmówić i nie może im zlecić wykonanie usług.

[22] http://pl.wikipedia.org/wiki/System_rozproszony

Klient-multi-serwer: podobnie jak dla architektury klient-serwer, ale istnieje wiele serwerów, np. WWW.

Koleżeńska (peer-to-peer, P2P): wiele węzłów świadczy sobie wzajemne usługi poprzez bezpośrednie połączenie; nie ma wyraźnego podziału na usługodawców i usługobiorców.

- Np. Gnutella, NXOR, Napster, Kazaa; JXTA jako narzędzie do P2P.
- Komercyjny buzz dookoła P2P.

Architektura oparta na oprogramowaniu pośredniczącym (middleware): nie występuje podział na klientów i serwery, np. CORBA i WebServices.

Architektury gridowe: wirtualny komputer sumujący zasoby wielu komputerów w sposób przezroczysty dla użytkowników.

39 Porównać technologie CORBA i JAVA RMI

Common Object Request Broker Architecture

Technologia zapewniająca komunikację pomiędzy obiektami pracującymi w heterogenicznych (różnorodnych) systemach komputerowych. Obiekty pełniące dowolne funkcje mogą być zaimplementowane w różnych językach programowania, na dowolnej platformie sprzętowej, pod kontrolą różnych systemów operacyjnych.

Opis obiektów, a właściwie interfejsów do nich, znajduje się w pliku IDL (ang. Interface Definition Language), który jest kompilowany na kod zajmujący się przekazywaniem metod (w przypadku implementacji technologii CORBA w niektórych językach interpretowanych, plik IDL jest interpretowany w czasie wykonania).

Obiekty mają swoje adresy IOR (ang. Interoperable Object Reference). Są to kilkusetznakowe adresy kodujące wiele informacji o obiekcie, m.in. adres komputera, adres programu na komputerze, informacje o kolejności zapisu bajtów (czy jest to big endian, czy little endian), numer obiektu, typ obiektu, itd.

Adresy IOR mogą dotyczyć także niskopoziomowych protokołów transmisji danych – zwykle GIOP (ang. General Inter-ORB Protocol) lub IIOP (ang. Internet Inter-ORB Protocol).

Remote Method Invocation

To mechanizm umożliwiający zdalne wywołanie metod obiektów. Obiekty te mogą znajdować się w innych maszynach wirtualnych Javy, które mogą znajdować się na innych komputerach.

Obiekty zdalne rejestrowane są pod wybranymi nazwami w serwisie RMI Registry. Aplikacja kliencka ściąga z RMI Registry tzw. stub tego obiektu, który umożliwia komunikację z obiektem zdalnym przy użyciu wyeksportowanych metod w ten sam sposób, jakby chodziło o obiekt lokalny. Rola RMI Registry w tym miejscu kończy się - nie pośredniczy on w komunikacji pomiędzy aplikacją kliencką a obiektem zdalnym. Parametry metod będące obiektami przy wywołaniu zdalnym są serializowane.

40 Omówić usługi nazewnicze i role rejestrów w systemach rozproszonych

To je to:

41 Sklasyfikować i omówić języki programowania serwisów Internetowych

PHP

Java

C#

Ruby

Scala

Lisp

Perl

Python

ColdFusion

Visual Basic

JavaScript

C++

Clojure

???

42 Scharakteryzować język PHP

Wykonywany po stronie serwera.

Nie posiada kompilatora, ani maszyny wirtualnej, tylko intepreter.
Tylko interpreter może wykonać skrypt.
Nie jest typowany statycznie, ale można sprawdzać typy i rzutować.
GC po zakończeniu skryptu.
Zaczynał jako język strukturalny, pełną obiektowość osiągnął w wersji 5.
Od wersji 5.3 posiada wsparcie dla programowania funkcyjnego (wyrażenia lambda i funkcje anonimowe)

43 Wymienić i krótko omówić instrukcje sterujące w PHP

elseif/else if

Wybrany blok kodu jest wykonywany w zależności od wartości wyrażenia logicznego.

while

Wykonywanie bloku kodu jest powtarzane dopóki wartość wyrażenie logicznego jest prawdą.

do-while

Jak pętla *while* ale najpierw jest wykonywana jedna iteracja, a dopiero po jej zakończeniu jest sprawdzana wartość wyrażenie logicznego decydującego o zakończeniu iteracji.

for

Służy do budowania pętli bazujących na zmiennych sterujących o danej wartości startowej, wyrażeniu logicznego decydującego o zakończeniu iteracji oraz bloku zmieniającego wartość zmiennych sterujących.

foreach

Ułatwia iteracje po wartościach przechowywanych w tablicach.

break

Kończy wykonywanie instrukcji *for*, *foreach*, *while*, *do-while* lub *switch*. Przyjmuje parametr określający liczbę zagnieżdżonych instrukcji których wykonywanie kończy.

continue

Wykorzystywane jest do zakończenia przetwarzania obecnie wykonywanej pętli *while*, *do-while*, for lub *foreach* i przejścia do wykonania następnej iteracji.

switch

Podobne do *if*, użyteczne tam gdzie chcemy porównać wartość jednej zmiennej z wieloma innymi wartościami i uruchomić inny blok kodu w zależności od tego z którą wartością zmienna się równa.

declare

Umożliwia ustawianie dyrektyw wykonywalnych dla bloku kodu PHP. Podstawową dyrektywą jest dyrektywa *encoding* umożliwiająca zmianę kodowania znaków. Dyrektywa *tick* jest pomocna przy profilowaniu kodu.

return

Kończy wywołanie funkcji. Wartość podana jako argument do instrukcji *return* jest wartością zwracaną przez funkcję której wywołanie kończy.

require

To to samo co *include* ale w razie błędu powoduje zakończenie działania skryptu z błędem.

include

Załącza i wykonuje kod PHP znajdujący się we wskazanym pliku.

require_once

To samo co *require* ale nie wykonywane jest ponownie jeżeli już dany plik został załadowany.

include_once

To samo co *include* ale nie wykonywane jest ponownie jeżeli już dany plik został załadowany.

goto

Umożliwia przełączenie punktu wykonywania kodu do innego miejsca programu.

44 Scharakteryzować język JavaScript

Wykorzystywany przede wszystkim na stronach.
Język imperatywny z wieloma elementami jeżyka strukturalnego.
Wykonywany po stronie klienta (przeglądarki).
Oparty o zdarzenia, tzn. funkcjonuje tylko na zdarzeniach.
W pełni obiektowy - wszystko jest obiektem (nawet prymitywy).
Można dynamicznie definiować obietky oraz rozszerzać istniejące w oparciu o prototypy.
Nie jest typowany statycznie, ale można sprawdzać typy i rzutować, rzutowanie poprzez odpowiednie funkcje.
Zarządzanie pamięcią jest automatyczne.
Funkcje mogą przyjmować nieokreśloną ilość parametrów niezależnie od deklaracji funkcji.

45 Omówić kolejne kroki obsługi technologii AJAX z wykorzystaniem PHP

Obsługa XML w PHP? SimpleXML (parsuje do drzewa obiektowego), XMLReader/Writer.
X_REQUESTED_WITH => XMLHttpRequest
"HTTP_ACCEPT"=>"application/json, */*; q=0.01",

46 Wymienić i opisać funkcje i operatory stosowane w SQL/XML

Funkcje

- XMLElement – tworzy element XML z danych nie-XML
- XMLForest – tworzy las elementów XML z danych nie-XML
- XMLGen – tworzy element XML z wykorzystaniem szablonu
- XMLConcat – tworzy las elementów przez konkatenację elementów XML wywiedzionych z jednego wiersza relacji
- XMLAgg – tworzy las elementów z kolekcji elementów XML wywiedzionych z różnych wierszy relacji

Operatory

- existsNode(). This is used in the WHERE clause of a SQL statement to restrict the set of documents returned by a query. The existsNode() operator takes an XPath expression and applies it an XML document. The operator and returns true (1) or false (0) depending on whether or not the document contains a node which matches the XPath expression.
- extract(). This takes an XPath expression and returns the nodes that match the expression as an XML document or fragment. If only a single node matches the XPath expression, the result is a well-formed XML document. If multiple nodes match the XPath expression, the result is a document fragment.
- extractValue(). This takes an XPath expression and returns the corresponding leaf level node. The XPath expression passed to extractValue() should identify a single attribute, or an element which has precisely one text node child. The result is returned in the appropriate SQL data type.

- updateXML(). This allows partial updates to be made to an XML document, based on a set of XPath expressions. Each XPath expression identifies a target node in the document, and a new value for that node. The updateXML() operator allows multiple updates to be specified for a single XML document.
- XMLSequence(). This makes it possible to expose the members of a collection as a virtual table

47 Podać i omówić przykład schematu XML w DTD

```
<!ELEMENT people_list (person)*>
<!ELEMENT person (name, birthdate?, gender?, socialsecuritynumber?)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT birthdate (#PCDATA)>
<!ELEMENT gender (#PCDATA)>
<!ELEMENT socialsecuritynumber (#PCDATA)>

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE people_list SYSTEM "example.dtd">
<people_list>
  <person>
    <name>Fred Bloggs</name>
    <birthdate>2008-11-27</birthdate>
    <gender>Male</gender>
  </person>
</people_list>
```

48 Podać i omówić przykład schematu XML w XML Schema

```
<?xml version="1.0"?>
<note
xmlns="http://www.w3schools.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.w3schools.com note.xsd">
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>

<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">

<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="to" type="xs:string"/>
      <xs:element name="from" type="xs:string"/>
      <xs:element name="heading" type="xs:string"/>
      <xs:element name="body" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

49 Omówić możliwość przetwarzania dokumentów XML w XQuery[23]

To język zapytań (jednakże posiadający pewne cechy języka programowania) służący do przeszukiwania dokumentów XML.

- Zapytania typu FLWOR (FOR, LET, WHERE, ORDER BY, RETURN)
- Wyrażenia ścieżkowe
  - osie
    - child
    - descendant
    - parent
    - ancestor
    - following-sibling
    - preceding-sibling
    - following
    - preceding
    - attribute
    - self
    - descendant-or-self
    - ancestor-or-self
  - testy
    - node() - jakikolwiek węzeł
    - text() - tylko węzły tekstowe
    - comment() - tylko węzły komentarza
    - element() - tylko elementy
    - schema-element(person) - elementy których schemat bazuje na schemacie elementu person
    - element(person) - elementy o nazwie person
    - attribute() - atrybuty
    - attribute(price) - atrybuty o nazwie price
    - document-node() - węzły dokumentu
  - predykaty
    - np. descendant::toy[attribute::color = "red"]
  - filtry
    - np. $products[price gt 100]
  - wyrażenia logiczne
    - np. 1 eq 2 and 3 idiv 0 = 1

```
for $d in doc("dzialy.xml")//nrdzialu
let $p := doc("pracownicy.xml")//pracownik[nrdzialu = $d]
where count($p) >= 10
order by avg($p/pensja) descending
return
  <ZbiorczoDzial>
     { $d,
        <zatrudnionych>{count($p)}</zatrudnionych>,
        <sredniapensja>{avg($p/salary)}</sredniapensja>
     }
  </ZbiorczoDzial>
```

[23] http://www.w3.org/TR/xquery/#id-path-expressions

50 Scharakteryzować obsługę baz danych w PHP

Istnieją implementacje dla większości popularnych baz danych i są to rozwiązania oparte o funkcje.

Wyjątkiem tutaj jest rozszerzenie MySQLi (imporoved mysql) które może być wykorzystywane zarówno obiektowo jak i poprzez funkcje. Połączenie jest reprezentowane przez obiekt.

Do tej całej watachy dochodzi jeszcze PDO - PHP Data Objects - obiektowa warstwa abstrakcji na większość popularnych baz danych.

Wspiera ODBC.

W PHP nie ma modelu O/R.

51 Scharakteryzować obsługę baz danych w Javie[24]

Java DataBase Connectivity (JDBC)

Interfejs programowania opracowany w 1996 r. przez Sun Microsystems, umożliwiający niezależnym od platformy aplikacjom napisanym w języku Java porozumiewać się z bazami danych za pomocą języka SQL. Interfejs ten jest odpowiednikiem standardu ODBC opracowanego przez SQL Access Group.

```
Connection conn = DriverManager.getConnection(
     "jdbc:somejdbcvendor:other data needed by some jdbc vendor",
     "myLogin",
     "myPassword" );
try {
     /* you use the connection here */
} finally {
    //It's important to close the connection when you are done with it
    try { conn.close(); } catch (Throwable ignore) { /* Propagate the original exception
instead of this one that you may want just logged */ }
}

Statement stmt = conn.createStatement();
try {
    ResultSet rs = stmt.executeQuery( "SELECT * FROM MyTable" );
    try {
        while ( rs.next() ) {
            int numColumns = rs.getMetaData().getColumnCount();
            for ( int i = 1 ; i <= numColumns ; i++ ) {
               // Column numbers start at 1.
               // Also there are many methods on the result set to return
               //  the column as a particular type. Refer to the Sun documentation
               //  for the list of valid conversions.
               System.out.println( "COLUMN " + i + " = " + rs.getObject(i) );
            }
        }
    } finally {
        try { rs.close(); } catch (Throwable ignore) { /* Propagate the original exception
instead of this one that you may want just logged */ }
    }
} finally {
    try { stmt.close(); } catch (Throwable ignore) { /* Propagate the original exception
instead of this one that you may want just logged */ }
}
```

[24] http://pl.wikipedia.org/wiki/Java_DataBase_Connectivity

Hibernate[25]

Framework do realizacji warstwy dostępu do danych (ang. persistance layer). Zapewnia on przede wszystkim translację danych pomiędzy relacyjną bazą danych, a światem obiektowym (ang. O/R mapping). Opiera się na wykorzystaniu opisu struktury danych za pomocą języka XML, dzięki czemu można "rzutować" obiekty, stosowane w obiektowych językach programowania, takich jak Java bezpośrednio na istniejące tabele bazy danych. Dodatkowo Hibernate zwiększa wydajność operacji na bazie danych dzięki buforowaniu i minimalizacji liczby przesyłanych zapytań.

52 Omówić metody zabezpieczania internetowych baz danych

Nie powinno się wystawiać bazy, a jedynie interfejs do niej (aplikacja).
Firewall
Niestandardowe nazwy użytkowników administracyjnych.
Trudne hasła.
Każda aplikacja powinna posiadać własnego użytkownika.
Maksymalne ograniczenie uprawnień dla danego użytkownika.
Używanie połączeń szyfrowanych (SSL).
Stosuje się różne metody szyfrowania baz danych.

53 Scharakteryzować modele danych multimedialnych w systemach internetowych

- Systemy komercyjne (Oracle, IBM DB2):
  - wykorzystanie rozszerzeń obiektowo-relacyjnych
  - rozwiązania firmowe dla metadanych sygnałowych
  - prawie brak wsparcia dla metadanych semantycznych
- Standardy:
  - SQL/MM – model obiektowo-relacyjny; brak wsparcia dla metadanych semantycznych
  - MPEG-7 – reprezentacja wszystkich typów metadanych w postaci XML; na razie słabe wsparcie przez systemy zarządzania bazą danych

54 Wymienić i omówić kolejne kroki wytwarzania witryn internetowych

1. Planowanie
   (a) specyfikacja wymagań, czyli co chce klient
2. Analiza
   (a) analiza wymagań i studium wykonalności, czyli co można faktycznie zrobić i jak
3. Projekt
   (a) projekt systemu, poszczególnych struktur, komponentów, modułów
4. Implementacja
   (a) wytwarzanie kodu
5. Testowanie

[25] http://pl.wikipedia.org/wiki/Hibernate

(a) testowanie poszczególnych elementów systemu

(b) testowanie systemu w całości

6. Wdrożenie i pielęgnacja

55 Omówić rolę administratora systemu informatycznego

Administrator[26]

Informatyk zajmujący się zarządzaniem systemem informatycznym i odpowiadający za jego sprawne działanie. Wyróżnia się administratorów:

- systemów operacyjnych
- baz danych
- serwerów
- sieci
- poszczególnych usług typu fora dyskusyjne, czaty itp., gdzie rola administratora sprowadza się przede wszystkim do moderowania.

Zadania

Do zadań administratora należy nadzorowanie pracy serwerów, dodawanie, ewentualna edycja danych, i kasowanie kont ich użytkowników, konfiguracja komputerów, instalowanie oprogramowania, dbanie o bezpieczeństwo systemu i opcjonalnie samych danych, nadzorowanie, wykrywanie i eliminowanie nieprawidłowości, asystowanie i współpraca z zewnętrznymi specjalistami przy pracach instalacyjnych, konfiguracyjnych i naprawczych, dbanie o porządek (dotyczy w szczególności forów internetowych) itp.

Ze względu na zakres obowiązków, specjalistyczna wiedza typowego administratora może wykraczać poza znajomość administracji powierzonego mu oprogramowania lub sieci, i dotyczyć pogranicza takich kategorii jak m.in.: elektronika, znajomość wielu różnych języków programowania, kryptografia i kryptoanaliza, etyka.

56 Charakterystyka infrastruktur klucza publicznego PKI

Zarządzanie kluczami (PKI - Public Key Infrastructure) to zestaw sprzętu, oprogramowania, ludzi i procedur potrzebnych do tworzenia, zarządzania, dystrybucji, wykorzystania, przechowywania i unieważniania cyfrowych certyfikatów.

Obejmuje:

- wytwarzanie kluczy,
- poświadczanie (certyfikaty),
- uzgadnianie (ustanawianie), wymiana,
- stosowanie (dobór) i użytkowanie,
- przechowywanie, w tym przekazywanie TTP i organom prawnym (escrow),
- niszczenie i unieważnianie.

PKI to infrastruktura pozwalająca na przypisanie klucza publicznego do podmiotu za pośrednictwem urzędów poświadczających klucze publiczne (CA).

- X.509

[26] http://pl.wikipedia.org/wiki/Administrator_(informatyka)

- hierarchiczny system urzędów certyfikujących (CA)
- tylko CA mogą podpisywać certyfikaty

- PGP
  - sieć zaufania (web of trust)
  - wszyscy mogą podpisywać certyfikaty

1. Dystrybucja kluczy
   (a) PGP - Klucze jawne PGP typowo rozprowadza się pocztą elektroniczną, za pomocą sieciowych biuletynów informacyjnych lub serwerów kluczy opartych na poczcie elektronicznej. W celu przeciwdziałania sabotażom klucze są podpisywane przez trzecie strony - innych użytkowników.
   (b) PEM - CA jest obdarzoną powszech-nym zaufaniem (TTP) bazą kluczy jawnych. Klucze jawne PEM mogą być rozprowadzane za pomocą dowolnych środków, zwykle pocztą elektroniczną lub za pomocą przeglądania katalogów zbudowanych w oparciu o standard X.500.
2. Unieważnianie kluczy
   (a) PGP - Dystrybucja kluczy jest doraźna i w znacznej części na drodze ustnej. W tej sytuacji niemożliwe jest zagwarantowanie unieważnienia certyfikatu, jeżeli jest on skompromitowany.
   (b) PEM - Listy unieważnień certyfikatów (CRL) są zachowane w katalogach typu X.500 lub w skrzynkach pocztowych utrzymywanych przez każdy urząd typu PCA

57 Protokół SSL/TSL i jego zastosowania[27]

TLS zapewnia poufność i integralność transmisji danych, a także uwierzytelnienie serwera, a niekiedy również klienta. Opiera się na szyfrowaniu asymetrycznym oraz certyfikatach X.509.

Według modelu OSI, TLS działa w warstwie prezentacji, dzięki czemu może zabezpieczać protokoły warstwy najwyższej – warstwy aplikacji, np.: telnet, HTTP, gopher, POP3, IMAP, NNTP, SIP.

58 Aktorzy i przebieg ataków typu DDoS

58.1 Ataki DoS/DDoS mają dwa typy ofiar

1. docelowa ofiara i
2. pośredniczące systemy, na których działają exploity I demony atakującego.
3. administrator siecie botnet

58.2 Fazy ataku DDoS:

1. Intruz wynajduje jeden lub więcej systemów w Internecie takich, że mogą dać się skompromitować i zainstalować exploity intruza. Najczęściej wykorzystuje do tego celu wykradzione dane legalnego użytkownika, wybierając przy tym systemy z szerokopasmowym dostępem, np. węzły akademickie.
2. Na skompromitowany system ładuje dowolną liczbę narzędzi hakerskich (np. skanery, wykrywacze typu SO, programy Dos/DDoS). Taki system staje się Masterem, który rozpoczyna poszukiwanie innych systemów możliwych do skompromitowania. Przeszukuje także systemy w sieci lokalnej pod katem uruchomionych usług, wynajdując te z podatnościami (masowe wnikanie). Automatyczne narzędzie do tej kompromitacji systemów nie są częścią narzędzi DDoS, ale pochodzą najczęściej od innych grup hakerów. Te skompromitowane systemy stają się początkowymi ofiarami DDoS – zostają na nich uruchomione demony DDoS przeprowadzające właściwy atak.

[27] http://pl.wikipedia.org/wiki/Transport_Layer_Security

3. Intruz dysponuje listą „posiadanych" systemów z demonami DDoS. Właściwa faza DDoS zaczyna się, gdy na systemie Master zostaje uruchomiony program, który komunikuje się z demonami, wydając polecenie rozpoczęcia ataku. Tę komunikację można tak ukryć, że staje się trudna do wykrycia, aczkolwiek istnieją pewne dowody na to, występuje. W większości przypadków administratorzy zaatakowanych systemów (demonów) nawet nie podejrzewają, że demony DDoS w nich działają. Nawet zostaną one zlokalizowane i usunięte, to nie można określić, gdzie jeszcze znajduje się takie oprogramowanie.

59 Bezpieczeństwo transakcji elektronicznych i protokół SET[28]

SET (Secure Electronic Transaction) to protokół bezpiecznych transakcji elektronicznych. Jest standardem umożliwiającym bezpieczne przeprowadzenie transakcji z wykluczeniem bezpośredniego kontaktu przez Internet, z użyciem kart kredytowych. Projekt ten został ogłoszony 1 lutego 1996 roku. Popierany przez dwie największe organizacje związane z kartami płatniczymi - Visa i MasterCard oraz wspierany przez firmę IBM. Protokół ten posiada cechy następujących technologii: SSL, STT, SEPP, SHTTP.

System SET umożliwia komunikującym się stroną wymianę informacji w bezpieczny sposób. SET wykorzystuje kryptograficzną metodę umożliwiającą zastąpienie numeru karty kredytowej certyfikatem. Dzięki temu sprzedawca nie otrzymuje numeru karty kredytowej.

Do przeprowadzenia transakcji wymagane są:

- Po stronie klienta - przegladarka internetowa, system operacyjny posiadajacy mechanizmy służące do obsługi certyfikatów (np. repozytorium)
- Po stronie sprzedawcy - serwer obsługujący protokół SET

Protokół SET nie uzyskał jednak adaptacji na szeroką skalę, powodami tego są:

- Wymagana jest instalacja specjalnego oprogramowania (e-portfela)
- Wysokie koszty i relatywna złożoność procesu wdrożenia przez sprzedawców w porównaniu z alternatywnymi metodami opartymi o protokół SSL
- Problematyczna dystrybucja certyfikatów klienta

60 Na czym polega bezpieczeństwo usługi znakowania czasem?

Bezpieczeństwo notariatu opiera się na podpisie złożonym przez zaufaną instytucję, stwierdzającym, że dany dokument rzeczywiście istniał o danym czasie i na braku możliwości zafałszowania informacji o dacie, zmiany treści, czy przeniesienia podpisu na inny dokument.

- Użytkownik wysyła skrót wiadomości do notariatu cyfrowego.
- Na serwerze notariatu gromadzone są skróty dokumentów przesłane w tej samej sekundzie od różnych klientów.
- Na ich podstawie budowany jest łączny skrót wielu dokumentów z tej samej sekundy, ten dodawany jest do sumy skrótów bazowych w tzw Super skrót bazowy (SHV super hash value) z poprzedniej sekundy dając nowy skrót dla danej sekundy.
- Takie rozwiązanie daje powiązanie skrótu danego dokumentu z wszystkimi innymi dokumentami z danej chwili i z poprzednimi poświadczeniami dokumentów.
- W każdej następnej chwili dołączane są kolejne skróty, dając w ten sposób umiejscowienie skrótu wśród innych tego rodzaju „odcisków palca".
- Zwrotnie użytkownik otrzymuje poświadczenie zarejestrowania skrótu i certyfikat dokumentu (Serwer nie może ich zmieniać) skróty są gromadzone w centralnym rejestrze i publikowane za pomocą różnych środków.

[28] http://pl.wikipedia.org/wiki/Secure_Electronic_Transaction

60.1 Elementy

1. dokument i jego skrót
2. serwer obliczający root hash (hash hashy dokumentów z danej sekundy) i SHV (Super Hash Value - root hash i SHV z poprzedniej sekundy)
3. certyfikat dokumentu - potwierdza zarejestrowanie skrótu dokumentu w systemie
4. publikacja SHV za pomocą różnych środków

60.2 Elementy

1. Powiązanie z innymi dokumentami
2. Umiejscowienie w czasie
3. Powiązany z całą historią skrótów
4. Masowa publikacja

60.3 Uzasadnienie

Zmiana dokumentu wymagała by przeliczenia wszystkich skrótów wystawionych do chwili obecnej oraz zmiany wszystkich opublikowanych wartości SHV po zgłoszeniu oryginalnego dokumentu.