Zawartość

Aplikacje Internetowe rozproszone

Koncepcja i obszary zastosowań architektury z cienkim i grubym klientem

Terminy **cienki klient** (thin client) oraz **gruby klient** (fat client) odnoszą się do mocy i jakości przetwarzania po stronie klienta w architekturze klient-serwer.

Model cienkiego klienta: klient posiada niezbyt wielką moc przetwarzania, ograniczoną do prezentacji danych na ekranie. Przykładem jest klient w postaci przeglądarki WWW. Fest najczęstszym rozwiązaniem w sytuacji, kiedy system scentralizowany jest zamieniany na architekturę klient-serwer. Wadą jest duże obciążenie serwera i linii komunikacyjnych.

Model grubego klienta: klient posiada znacznie bogatsze możliwości przetwarzania, w szczególności może zajmować się nie tylko warstwą prezentacji, lecz także warstwą przetwarzania aplikacyjnego (logiki biznesu). Używa większej mocy komputera klienta do przetwarzania zarówno prezentacji jak i logiki biznesu. Serwer zajmuje się tylko obsługą transakcji bazy danych. Popularnym przykładem grubego klienta jest bankomat. Zarządzanie w modelu grubego klienta jest bardziej złożone.

Dwuwarstwowa architektura K/S z cienkim klientem

- Systemy spadkowe (legacy), gdzie oddzielenie przetwarzania i zarządzania danymi jest niepraktyczne.
- Aplikacje zorientowane na obliczenia, np. kompilatory, gdzie nie występuje lub jest bardzo mała interakcja z bazą danych.
- Aplikacje zorientowane na dane (przeglądanie i zadawanie pytań) gdzie nie występuje lub jest bardzo małe przetwarzanie.

Dwuwarstwowa architektura K/S z grubym klientem

- Aplikacje w których przetwarzanie jest zapewnione przez wyspecjalizowane oprogramowanie klienta, np. MS Excel.
- Aplikacje ze złożonym przetwarzaniem (np. wizualizacją danych, przetwarzaniem multimediów).
- Aplikacje ze stabilną funkcjonalnością dla użytkownika, użyte w środowisku z dobrze określonym zarządzaniem.

Trzywarstwowa lub wielowarstwowa architektura K/S

- Aplikacje o dużej skali z setkami lub tysiącami klientów.
- Aplikacje gdzie zarówno dane jak i aplikacje są ulotne (zmienne).
- Aplikacje integrujące dane z wielu rozproszonych źródeł.

Usługa DNS - cechy i zastosowania

DNS (ang. Domain Name System) to system serwerów, protokół komunikacyjny oraz usługa zapewniająca zamianę adresów znanych użytkownikom Internetu na adresy zrozumiałe dla urządzeń tworzących sieć komputerową. Adresy DNS składają się z domen internetowych rozdzielonych kropkami. DNS to złożony system komputerowy oraz prawny. Zapewnia z jednej strony rejestrację nazw domen internetowych i ich powiązanie z numerami IP. Z drugiej strony realizuje bieżącą obsługę komputerów odnajdujących adresy IP odpowiadające poszczególnym nazwom.

Wewnątrz każdej domeny można tworzyć tzw. subdomeny – stąd mówimy, że system domen jest 'hierarchiczny'. Przykładowo wewnątrz domeny .pl utworzono wiele domen: regionalnych ('opole.pl', 'warmia.pl'), funkcjonalnych ('com.pl', 'gov.pl') należących do firm, organizacji lub osób prywatnych ('wp.pl', 'zus.pl').

System DNS posiada następujące cechy:

- Nie ma jednej centralnej bazy danych adresów IP i nazw. Najważniejszych jest 13 głównych serwerów rozrzuconych na różnych kontynentach.
- Serwery DNS przechowują dane tylko wybranych domen.

- Każda domena posiada jeden główny dla niej serwer DNS (tzw. master), na którym to wprowadza się konfigurację tej domeny, wszystkie inne serwery obsługujące tę domenę są typu slave i dane dotyczące tej domeny pobierają automatycznie z jej serwera głównego po każdej zmianie zawartości domeny.
- Serwery DNS mogą przechowywać przez pewien czas odpowiedzi z innych serwerów (ang. caching), a więc proces zamiany nazw na adresy IP jest często krótszy niż w podanym przykładzie.
- Na dany adres IP może wskazywać wiele różnych nazw. Na przykład na adres IP 207.142.131.245 mogą wskazywać nazwy pl.wikipedia.org oraz de.wikipedia.org
- Przy zmianie adresu IP komputera pełniącego funkcję serwera WWW, nie ma konieczności zmiany adresu internetowego strony, a jedynie poprawy wpisu w serwerze DNS obsługującym domenę.
- Protokół DNS posługuje się do komunikacji serwer-klient głównie protokołem UDP, serwer pracuje na porcie numer 53, przesyłanie domeny pomiędzy serwerami master i slave odbywa się protokołem TCP na porcie 53.

Struktura serwera aplikacji

Zestaw oprogramowania (platforma) wspierająca programistę przy tworzeniu aplikacji. Umożliwia oddzielenie logiki biznesowej od usług dostarczanych przez producenta platformy (bezpieczeństwo, zarządzanie transakcjami, skalowalność, czy też dostęp do baz danych). Do serwerów aplikacji należą m.in.: JBoss, BEA WebLogic, IBM WebSphere oraz platforma .NET Microsoftu.

Serwer aplikacji Microsoftu oferuje informatykom i deweloperom aplikacji następujące korzyści:

- Podstawowe środowisko wykonawcze obsługujące efektywne wdrażanie wydajnych aplikacji biznesowych i zarządzanie nimi.
- Środowisko programistyczne .NET Framework z uproszczonym modelem programowania oraz wydajnym modelem wykonawczym dla aplikacji serwerowych.
- Przyjazny dla użytkownika kreator instalacji, który zapewnia wybór różnych usług ról i funkcji niezbędnych do uruchamiania aplikacji w organizacji.
- Narzędzia instalacyjne, które automatycznie instalują funkcje wymagane przez poszczególne usługi ról.

Koncepcja hipertekstu i leksji oraz rola CSS w tworzeniu stron HTML

Hipertekst organizacja danych w postaci niezależnych leksji połączonych hiperłączami. Hipertekst cechuje nielinearność i niestrukturalność układu leksji. Oznacza to, że nie ma z góry zdefiniowanej kolejności czytania leksji, a nawigacja między nimi zależy wyłącznie od użytkownika. Najbardziej znanym systemem hipertekstowym jest sieć WWW.

Leksja (fragment) to podstawowa jednostka hipertekstu. Względnie spójna i niepodzielna jednostka tekstu i obrazu, nazywana często hipertekstowym węzłem lub stroną. W sieci WWW, wiele stron jest leksjami, lecz niektóre są tak długie, że przekraczają ramy niepodzielności i spójności. Leksja różni się od klasycznie pojmowanego fragmentu. Jej cechą musi być spójność i bardziej niż względna autonomiczność. Leksje są czytane w różnych i zmieniających się w trakcie lektury kontekstach. Leksje muszą być na tyle spójnymi całostkami tekstu, by wraz z zerwaniem toku narracji nie zerwany został tok lektury. Zawartością leksji może być nie tylko tekst, ale również grafika i film.

CSS – Cascading Style Sheets; umożliwienie separacji stylów względem struktury i treści dokumentu; bardziej precyzyjna kontrola nad układami, fontami, kolorami, tłem, efektami typograficznymi; możliwość modyfikowania większego zbioru stron poprzez edytowanie pojedynczego dokumentu; zapewnienie zgodności pomiędzy przeglądarkami, zapewnienie sprawnego ładowania stron. **Rola CSS:** Zastępuje tradycyjne metody projektowania WWW; Krótszy czas tworzenia strony; Większe

możliwości kreowania wyglądu strony; Łatwe do tworzenia; Projektowanie wyglądu strony oddzielone od projektowania treści; Podział formy dokumentu HTML na strukturę i zdobnictwo.

Definicja i obszary zastosowania języka XML

XML jest metajęzykiem służącym do opisu znaczenia danych zawartych w dokumencie, a nie sposobu ich prezentacji, jak to ma miejsce w HTML. Jest językiem „ramowym" do wymiany informacji między aplikacjami, czytelny dla maszyny i dla człowieka. XML pozwala zdefiniować znaczenia frazy (cena, wysokość podatku, tytuł książki). Standard przewiduje też możliwość sprawdzania poprawności danych zawartych w dokumencie (służy do tego mechanizm DTD - Document Type Definition). Podstawową możliwość definiowania języków adjustacji opartych na składni XML, ukierunkowanych praktycznie dowolnie, zaletą XML jest jego rozszerzalność. **Zastosowania XML:**

- Wymiana dokumentów między aplikacjami
- Standardy rozszerzające przeglądarki
- Mathematical Markup Language MML – przeglądarka Amaya - wzory matemayczne
- XHTML – bardziej sformalizowany HTML
- Scalable Vector Graphics SVG- grafika wektorowa
- SOAP – protokół wymiany obiektów
- Web Services Protocol Stack, XML-RPC, WSDL, UDDI
- Bazy danych – repozytoria XML

Zadania DTD i XML Schema dla aplikacji współpracujących w standardzie XML

Zasadniczym zadaniem obu tych technologii jest stworzenie mechanizmu, który pozwala na kontrolę poprawności składniowej dokumentu XML – więc zestawu użytych znaczników, ich atrybutów oraz struktury tych znaczników i atrybutów.

Powody, dla których warto jest posiadać taki mechanizm:

- Skuteczniejsza kontrola błędów przy tworzeniu dokumentu oraz możliwość zaimplementowania podpowiedzi w edytorach XML.
- Uproszczenie przetwarzania dokumentu, dla którego uprzednio sprawdzono poprawność strukturalną.
- Możliwość uzgodnienia formatów dla elektronicznej wymiany dokumentów pomiędzy instytucjami.
- Definiowanie sposobów konwersji pomiędzy dokumentami sporządzonymi w różnych formatach.
- Podstawa dla określania sposobów wizualizacji poszczególnych rodzajów elementów.

Języki zapytań do repozytoriów XML, ze szczególnym uwzględnieniem XQUERY

Lorel - silne wsparcie dla wyrażeń ścieżkowych, pozwalające dobrze radzić sobie z sytuacją słabej znajomości struktury przetwarzanego dokumentu.

Język **XML-QL** - bazuje na klauzulach where i construct. W klauzuli where można przy ich pomocy nałożyć dodatkowe ograniczenia na to, kiedy wzorzec zostanie uznany za pasujący. W znajdującej się za nią klauzuli construct, zawierającej fragment XMLa, będący rezultatem zapytania, wystąpienia tych zmiennych zostaną zastąpione przez ich zawartość.

XML-GL – operuje na diagramach. Zapytanie składa się z pionowej kreski, po której lewej stronie jest narysowany wzorzec a po prawej - wynik zapytania. Wiązanie zmiennych i dodawanie ograniczeń przy ich pomocy jest również wyrażane graficznie po lewej stronie.

XQL - Pozwala tylko na wybranie i filtrowanie elementów i tekstu.

SQL/XML - Nowe standardowe rozszerzenie SQL, pozwalające na generowanie zagnieżdżonych wyników w XML. Każda krotka wynikowa jest odwzorowywania na XMLowy element *row.* ***xmlelement*** *tworzy elementy XM ,* ***xmlattributes*** *tworzy atrybuty*

XQuery - służący do przeszukiwania dokumentów XML. XQuery obficie korzysta z wyrażeń ścieżkowych (XPath). Pozwala definiować własne funkcje i wykonywać operacje agregacyjne. Może działać na repozytorium XML wykorzystując nazwane kolekcje dokumentów. Zapytania opierają się głównie na pętli FLWOR (for-let-where-order by-return), podobnej do konstrukcji SQL: select – from – where - order by. W ramach niej następuje iteracja po elementach podanych przez rozszerzone wyrażenie XPath, wypełnienie zmiennych, filtrowanie po warunku, sortowanie a następnie utworzenie wynikowej konstrukcji XML.

JavaScript - sposoby przekształcania zawartości dokumentów XML w interfejsie DOM

W JS model DOM służy do odwoływania się do elementów okna, dokumentów itd. Stosowanie modelu DOM do struktur XML odbywa się w następujący sposób

- Tworzymy plik HTML zawierający JavaScript, który
- Ładuje dokument XML z danymi,
- I korzystając z modelu DOM modyfikuje prezentację dokumentu HTML

XSTL oparty na xml jezyk o charakterze deklaratywnym oparty na regułach przekształceń drzew xml.Scenariusze przetwarzania:

- zmiana struktury dokumentu XML;
- utworzenie arkusza XSLT do przetwarzania innych dokumentów;
- obudowanie danych z pliku XML informacjami o sposobie prezentacji.
- Przetwarzania dokumentów w oparciu o wzorce służą programy określane jako procesory XSLT.

Budowa i rodzaje pająków internetowych

Robot internetowy – jest programem zbierającym informacje o strukturze i stronach umieszczanych w indeksie wyszukiwarek i służą m.in. do: dodawania do wyszukiwarek; sprawdzania kodu strony; zbierania informacji o stronie; monitorowania "co nowego"; tworzenia mirrorów stron.

Kontrolowanie robotów. Roboty nie potrafią indeksować nie podlinkowanych plików, więc ominą te które leżą sobie "luzem" w katalogach na serwerze. Webmaster może kontrolować, które katalogi/pliki są odwiedzane przez roboty edytując plik robots.txt, bądź też implementując odpowiednie Meta Tagi.

Podążanie za linkami. Ty wybierasz stronę startową, a roboty indeksujące zażądają jej od serwera i pobiorą, tak jak zwykła przeglądarka. "Indekser" zachowa każde słowo ze strony i podąży za każdym linkiem, indeksując i zagłębiając się coraz dalej w strukturę witryny.

Odświeżanie indeksów. Aby update'ować indeks, robot utworzy zapytanie do serwera WWW o status każdej połączonej linkiem strony. Pobierze nagłówek HTTP używając zapytania "HEAD". Serwer może odpowiedzieć na zapytanie odsyłając nagłówek strony z wewnętrznego cache`u, bez otwierania i czytania całego pliku. Wtedy indexer porównuje daty z nagłówka i tą, którą sam pobrał dawniej z serwera.

Metody rankingu dokumentów w wyszukiwarkach internetowych; budowa wyszukiwarki

Metody rankingu dokumentów w wyszukiwarkach internetowych: Aby przybliżyć się do oczekiwań użytkownika, opracowano szereg strategii szeregowania stron, zwanych też rankingiem (rank) czy też

oceną lub wagą (score) dokumentu. Ranking stron/dokumentów decyduje zwykle o kolejności wyświetlania ich opisów w odpowiedzi systemu na kwerendę użytkownika. Ranking stron może zostać przygotowany dynamicznie (np. przez obliczenie "odległości" czy "podobieństwa" między zapytaniem użytkownika a dokumentem w trybie on-line) albo też statycznie (przez ocenę wartości strony w porównaniu z całą populacją stron WWW w trybie off-line). Zwykle stosuje się statyczną ocenę rankingu (z uwagi na koszt obliczeń). Stosuje się tu szereg podejść:

- Opinie redaktorów - W niektórych wyszukiwarkach istnieje możliwość zaznaczenia przez internautę, czy jest usatysfakcjonowany ogólnie dokumentami, które znalazła wyszukiwarka, i wtedy modyfikuje się ranking znalezionych dokumentów.
- Popularność - Niektóre wyszukiwarki kontrolują, do których stron znalezionych przez wyszukiwarkę rzeczywiście wchodzą internauci.
- Zawartość informacyjna - Stronę można określić jako interesującą, jeżeli zawiera dużo tekstu. Układ strony może stronę deprecjonować, jeśli sugeruje on "reklamówkowość" jej zawartości. Można badać "kwiecistość" tekstu, tj. czy słowa powtarzają się niezbyt często. Można też badać liczbę linków wychodzących ze strony (dobra strona startowa).Może też być istotne występowanie lub nie określonych słów kluczowych na określonych pozycjach. Zawartość informacyjna jest na obecnym etapie niemierzalna obiektywnie w wypadku, gdy treścią strony jest grafika.
- Lokalizacja - Jako składową miary ważności strony stosuje się czasami głębokość na ścieżce w jej URLu. Zakłada się, że strona umieszczona w korzeniu jest ważniejsza niż np. w podkatalogu na głębokości np. 5. W zależności od zainteresowań wyższą rangę można nadawać dokumentowi pochodzącemu z określonej domeny (rządowej, edukacyjnej, komercyjnej itp.). Dokumenty umieszczane na serwerach darmowych mogą być uważane za mniej ważne niż umieszczane na serwerach płatnych.
- Finansowanie - Inną metodą rankingu stron jest stopień ich popierania przez sponsorów wyszukiwarki i/lub reklamodawców.
- PageRank - Ranking PageRank zakłada, że osoby umieszczające linki do określonych stron WWW na swojej stronie, zapoznali się z treścią tych dokumentów i uznali je za godne uwagi, (czyli zrecenzowali go). Tak więc waga dokumentu winna zależeć od tego, ile dokumentów nań wskazuje. Ale dokument dokumentowi nierówny. Jeżeli na dany dokument wskazuje wiele innych dokumentów, to tenże dokument jest swego rodzaju "autorytetem". Tak więc wskazanie na inny dokument przez "wysoki autorytet" winno mieć większą wagę niż wskazanie przez autorytet mały. Algorytm PageRank dąży do stworzenia "bilansu" między stopniem autorytetu "wpływającym" na stronę i z niej "emanującym" na inne strony WWW.

Budowa wyszukiwarki:

- Pająk – program który automatycznie porusza się po stronach internetowych.
- Indekser – mechanizm tworzący indeks czyli specjalizowaną bazę danych, zawierającą skompilowaną wersje dokumentów ściągniętych przez pająka.
- Lista inwersyjna – Zawiera listę dla każdego słowa kluczowego listę dokumentów które ją zawierają. Każda lista jest posortowana wg numeru dokumentu. Podczas wyszukiwania nie są szukane wszystkie dokumenty zawierające dana frazę, ale jedynie ich listy inwersyjne.
- Podsystem wyszukujący – dialog z użytkownikiem - Wyszukiwanie to operacja polegająca na identyfikacji podzbioru zbioru dokumentów, zawierających pożądana treść lub mających pożądane cechy.

Architektura Systemów Komputerowych

Omów stałopozycyjną reprezentację liczb dodatnich i ujemnych w systemie komputerowym

Reprezentacja stałopozycyjna charakteryzuje się stałym położeniem kropki dziesiętnej.

- Na część całkowitą liczby oraz na część ułamkową przeznaczona jest stała, z góry określona liczba bitów.
- Jeśli na część ułamkową przeznaczone jest 0 bitów to reprezentacja stałopozycyjna służy do przechowywania liczb całkowitych.
- Jeśli liczba, którą chcemy przedstawić w tej reprezentacji, mieści się na ustalonej liczbie bitów, to może być reprezentowana dokładnie.
- Jeśli wynik działań wykonywanych na liczbach stałopozycyjnych nie mieści się na ustalonej liczbie bitów, powstaje nadmiar w obliczeniach (wynik jest błędnie interpretowany przez komputer).
- Deklaracja odpowiedniego typu zmiennych w programie określa dopuszczalny zakres danych.
- Mając do dyspozycji n bitów możemy w reprezentacji stałopozycyjnej przedstawić liczby całkowite z zakresu: $-2^{n-1} .. 2^{n-1}-1$

Do reprezentacji liczb całkowitych stosowane są kody stałopozycyjne:

- zapis znak-moduł - Zapis znak-moduł tworzy się przez dodanie przed MSB dodatkowego bitu znaku do zapisu NKB: 0 - liczba dodatnia; 1 - liczba ujemna; „0” ma podwójną reprezentację: 1000...000 lub 0000...000
- zapis U1 - W zapisie U1 (uzupełnień do 1) MSB jest także bitem znaku : 0 - liczba dodatnia; 1 - liczba ujemna; ale w zależności od jego wartości dalsze bity mają różne znaczenie. Dla „0” (liczba dodatnia) dalsze bity reprezentują liczbę w NKB. Dla „1” (liczba ujemna) dalsze bity reprezentują moduł liczby ujemnej, w taki sposób, że zanegowane ich wartości odpowiadają modułowi tej liczby w NKB. „0” ma podwójną reprezentację: 1111...111 lub 0000...000
- zapis U2 - Zapis U2 (uzupełnień do 2) jest podobny do U1 ale dla liczb ujemnych. Moduł liczby ujemnej powstaje tak, że do zanegowanych pozycji słowa jest arytmetycznie dodawana jedynka i dopiero tak utworzone słowo odpowiada w NKB modułowi tej liczby. „0” ma pojedynczą reprezentację: 0000...000
- zapis polaryzowany (BIAS) - Zapis BIAS (polaryzowany) przedstawia liczby w taki sposób, że „0” jest reprezentowane przez n-bitowe słowo 1000..000 czyli przez liczbę 2n-1 kodu NKB. Wszystkie inne liczby A są przedstawione na n pozycjach jako binarne wartości liczby 2 n-1+A

Liczba	ZM	U1	U2	BIAS	BCD
-127	11111111	10000000	1 0000001	00000001	1000100100111
-126	11111110	1 0000001	10000010	00000010	1000100100110
...					
-2	1000001 0	11111101	11111110	11111110	1000000000010
-1	10000001	11111110	11111111	11111111	1000000000001
0	10000000	11111111	00000000	10000000	0000000000000
0	00000000	00000000	00000000	10000000	0000000000000
1	00000001	00000001	00000001	10000001	0000000000001
2	0000001 0	00000010	00000010	10000010	0000000000010
3	00000011	00000011	00000011	10000011	0000000000011
...					
126	01111110	01111110	01111110	11111110	0000100100110
127	011111111	011111111	011111111	11111111	0000100100111

Omów ogólną budowę systemu komputerowego oraz przedstaw rolę jego istotnych elementów funkcjonalnych

Wariant struktury systemu komputerowego z jednym kompletem magistrali: danych, adresowej i sterowania zaproponował w latach 40-tych amerykański matematyk John von Neuman. Na jego cześć struktura nazywana jest **strukturą von Neumana**. Później dla systemów komputerowych specjalnego przeznaczenia w Uniwersytecie Harvarda zaproponowano strukturę z dwoma kompletami magistrali: danych, adresowej i sterowania, jeden do przesyłania poleceń programu, drugi do przesyłania operandów operacji. Struktura otrzymała nazwę **struktury Harvardzkiej**.

Wejścia systemu komputerowego. - dostarczają informacji do systemu komputerowego pochodzących ze świata zewnętrznego.

Wyjścia systemu komputerowego. - Urządzenia wyjściowe w systemie komputerowym służą do przekazania informacji wynikających z działania systemu komputerowego do świata zewnętrznego.

Jednostka centralna (CPU). - Bez tego elementu nie obejdzie się żaden system komputerowy, a więc również i mikrokontroler. Procesor centralny składa się z trzech podstawowych części: jednostki arytmetyczno-logicznej ALU (ang. - Arithmetic and Logic Unit), jednostki sterującej oraz rejestrów. Zadaniem CPU jest precyzyjne wykonywanie rozkazów programu dostarczonego przez programistę.

Zegar czyli generator taktujący. - Wszystkie systemy komputerowe używają generatora sygnału zegarowego po to aby wymusić na jednostce centralnej wykonanie kolejnego kroku operacji, można powiedzieć że sygnał zegarowy synchronizuje pracę systemu.

Pamięć. - Pamięć w systemie komputerowym służy do przechowywania programu, danych, wyniku działania kolejnych operacji

Magistrala (szyna)- Poszczególne bloki komputera, sterowane przez procesor, sprzęgnięte są wspólnym systemem komunikacyjnym, który ma zazwyczaj strukturę magistralową. W strukturze tej każdy z bloków ma podobny dostęp do magistrali informacyjnej umożliwiającej jednoznaczny przepływ informacji między blokami. W systemie występują trzy zasadnicze magistrale (szyny):

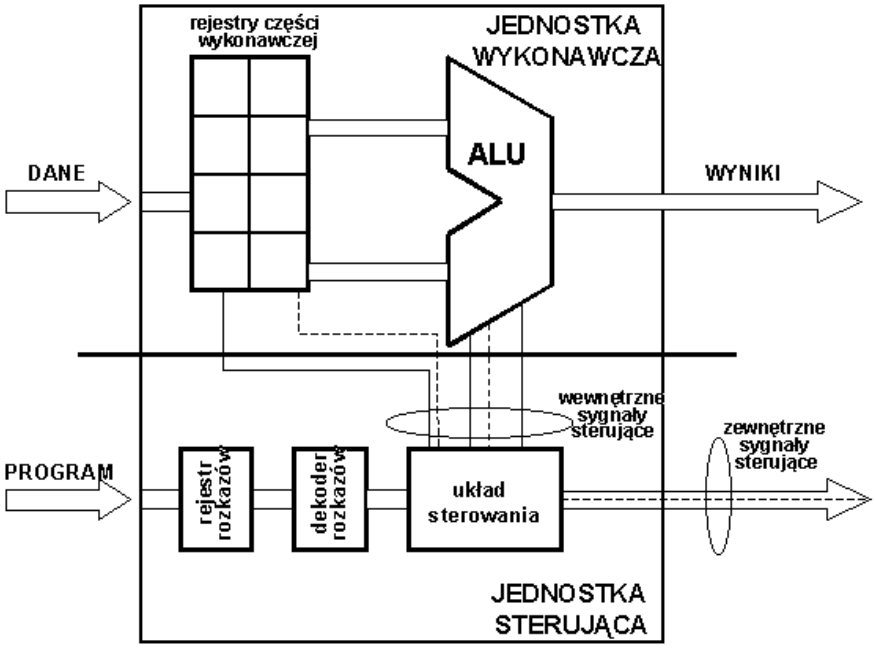
- szyna adresowa stanowi wiązkę jednokierunkowych linii, których liczba wyznacza możliwości adresowania np. 20 linii to umożliwia zaadresowanie 220 komórek
- szyna danych stanowi dwukierunkową wiązkę linii, których liczba określa długość słowa procesora (np. 8, 16, 32, ... + bit detekcji i korekcji błędów)
- szyna sterowania, którą stanowi dwukierunkowa wiązka linii (zwykle kilkanaście - kilkadziesiąt linii)

Program - To jest chyba najważniejsza rzecz w każdym systemie komputerowym. Składnikami programu są rozkazy, które wynikają z listy rozkazów dla danego procesora czy mikrokontrolera. Napisany program jest przechowywany w pamięci systemu komputerowego.

Omów ogólną budowę procesora oraz przedstaw rolę jego istotnych elementów funkcjonalnych

W architekturze procesora można wyróżnić dwa podstawowe elementy funkcjonalne:

- **jednostkę wykonawczą**, której zadaniem jest wykonywanie operacji arytmetycznych i logicznych w zależności od wewnętrznych sygnałów sterujących (dane zapisywane są do rejestrów, jednostka arytmetyczno logiczna wykonuje operacje, wysterowaną przez jednostkę sterowania, po czym zapisuje wynik do rejestrów)
- **jednostkę sterującą**, której zadaniem jest dekodowanie z programu rozkazu i na jego podstawie generowanie wewnętrznych i zewnętrznych sygnałów sterujących.



Scharakteryzuj główne dostępne programowo rejestry procesora

Osiem **rejestrów powszechnego zastosowania** (każdy o długości 16 bitów) są używane do najczęściej stosowanych rozkazów, jako miejsce, skąd pobieramy dane, miejsce przeznaczenia, wskaźniki do pamięci i wreszcie jako liczniki. Każdy z tych ośmiu rejestrów może być załadowany zarówno z pamięci, jak też z nich można do pamięci załadować, można ich używać do operacji arytmetycznych i logicznych. Rejestr AX - znany jako akumulator. Jest to najbardziej efektywny rejestr używany w operacjach arytmetycznych, logicznych, przesyłania danych. Rejestr BX - domyślnie rejestr BX, wraz z rejestrom segmentowym DS jest używany jako wskaźnik pamięci. Rejestr CX - używa się go głównie jaka licznika odliczającego powtarzające się fragmenty programów bądź pojedynczych rozkazów. Rejestr DX - jego głównym przeznaczeniem jest użycie go jako wskaźnika adresów w rozkazach wejścia/wyjścia.

Rejestry wskaźnikowe i indeksowe. Rejestr SI - podobnie jak rejestr BX, może być użyty juko wskaźnik pamięci. Rejestr DI - jest bardzo podobny w użyciu do rejestru SI. Może być użyty jako wskaźnik pamięci; ma specjalne własności, gdy zastanie zastosowany w rozkazach związanych z łańcuchami znakowymi. Rejestr BP - podobnie jak BX, SI, DI może być użyty jako wskaźnik pamięci. Rejestr BP, służąc za wskaźnik pamięci, odnosi się do rejestru SS, rejestru segmentu stosu. Rejestr SP - znany jest jako wskaźnik stosu i należy do ostatnich rejestrów powszechnego stosowania. Rejestr SP daje położenie bieżącego wierzchołka stosu i jest analogiczny do IP.

Rejestry segmentowe tak jak rejestry powszechnego stosowania odgrywają swoje specyficzne role, tak też rejestry segmentowe przeznaczone są do wyspecyfikowanych zadań.

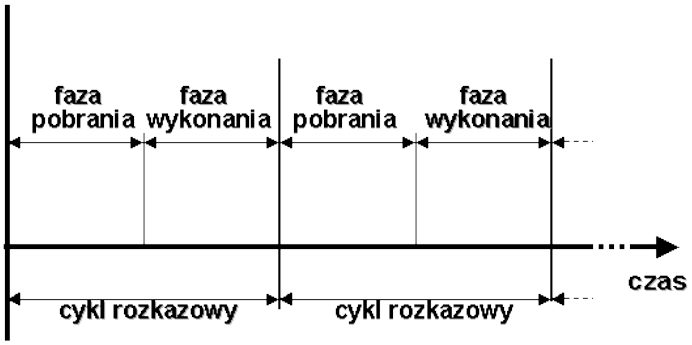
Rejestr IP - nazywany jest **wskaźnikiem rozkazów** i zawiera zawsze offset pamięci, w którym zawarty jest następny rozkaz do wykonania. Bazowy segmentu kodu zawarty jest w rejestrze CS. Pełny adres logiczny wykonywanego rozkazu wskazywany jest więc przez parę rejestrów CS:IP.

Rejestr znaczników (rejestr flagowy) - FLAGS jest czternastym rejestrem procesora 8086. Rejestr ten jest zbiorem poszczególnych bitów kontrolnych, zwanych znacznikami (flagami), które wskazują wystąpienie określonego stanu. Każdy znacznik jest bitem w rejestrze, który wskazuje czy wystąpił określony stan. Znaczniki mogą być wykorzystywane zarówno przez procesor, jak też i programistę na dwa sposoby: ustawienie znacznika dla zapamiętania określonego stanu po wykonaniu rozkazu i

testowanie znacznika celem umożliwienia decyzji o sposobie dalszego postępowania (przetwarzania danych)

Omów cykl rozkazowy procesora

Realizując program, system mikroprocesorowy wykonuje pewne powtarzające się czynności, polegające na cyklicznym pobieraniu kodów rozkazów z pamięci i wczytywaniu ich do układu sterowania mikroprocesora, a następnie realizacji rozkazu, którego kod został pobrany.



Przebieg jednego cyklu rozkazowego można opisać za pomocą następującego algorytmu:

1. Zawartość miejsca pamięci wewnętrznej wskazywanego przez licznik rozkazów LR zostaje przesłana do układów sterujących procesora.
2. W układach sterujących następuje rozdzielenie otrzymanej informacji na dwa pola: pole operacji i pole argumentów. Pole operacji zawiera adres rozkazu, który należy wykonać. Pole argumentów zawiera adresy, pod którymi są przechowywane dane oraz adres przeznaczenia wyniku.
3. Na podstawie wyznaczonych adresów następuje przesłanie z pamięci wewnętrznej argumentów do odpowiednich rejestrów, a na podstawie adresu rozkazu arytmometr wykonuje odpowiednie działanie (operację arytmetyczną lub logiczną) na zawartościach rejestru.
4. Wynik przetwarzania (wynik wykonanej operacji) jest wysyłany do pamięci wewnętrznej pod adres przeznaczenia wyniku.
5. Następuje zmiana wartości licznika rozkazów LR tak, aby wskazywał on kolejny rozkaz dla procesora.

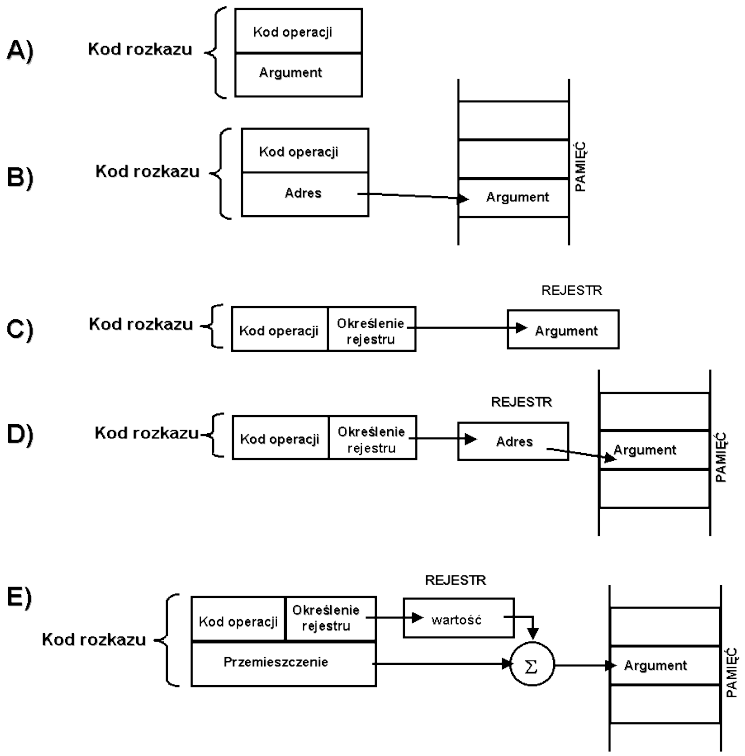
Omówić znane ci tryby adresowania w procesorze

Rozkazy, jak każdy inny rodzaj informacji w systemie mikroprocesorowym, są przechowywane w postaci kodów binarnych. **Kod rozkazu** musi zawierać określenie rodzaju wykonywanej operacji, czyli tak zwany kod operacji. **Kod operacji** musi być określony w początkowej części (pierwszym bajcie lub bajtach) kodu rozkazu w celu określenia, w jaki sposób ma przebiegać dalsza realizacja rozkazu przez mikroprocesor.

Trybem adresowania nazywa się sposób określenia miejsca przechowywania argumentów rozkazu. Można rozróżnić następujące tryby adresowania:

- adresowanie natychmiastowe, w którym argument rozkazu zawarty jest w kodzie rozkazu;
- adresowanie bezpośrednie, w którym kod rozkazu zawiera adres komórki pamięci przechowującej argument rozkazu;
- adresowanie rejestrowe, w którym w kodzie rozkazu określony jest rejestr zawierający argument rozkazu;

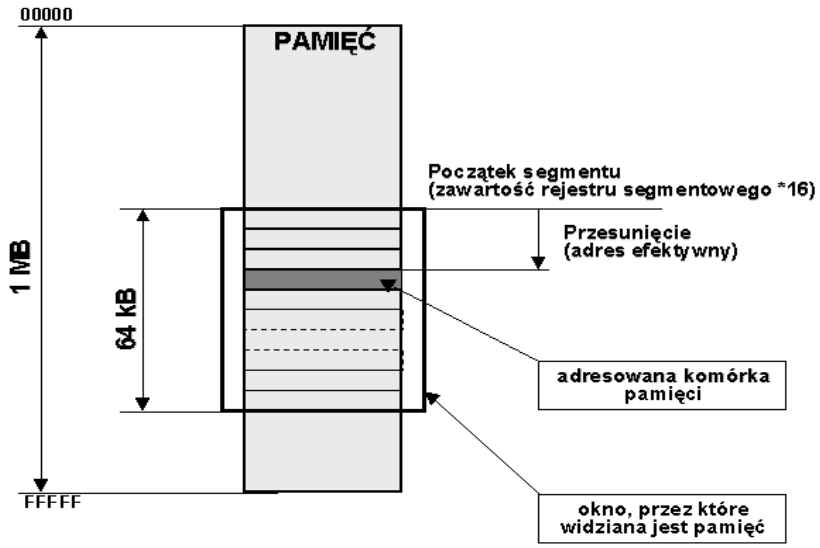
- adresowanie pośrednie (rejestrowe pośrednie), w którym kod rozkazu zawiera określenie rejestru, bądź rejestrów zawierających adres komórki pamięci z argumentem rozkazu;
- adresowanie indeksowe z przemieszczeniem, w którym adres argumentu przechowywanego w pamięci obliczany jest jako suma zawartości rejestru i wartości (tzw. przemieszczenia) określonych w kodzie rozkazu.



a) adresowanie natychmiastowe, b) adresowanie bezpośrednie, c) adresowanie rejestrowe, d) adresowanie pośrednie, e) adresowanie indeksowe z przemieszczeniem

Na czym polega praca procesora w trybie rzeczywistym? Omów sposób tworzenia adresu fizycznego w tym trybie

Tryb rzeczywisty – tryb pracy mikroprocesorów z rodziny procesorów x86, w którym procesor pracuje tak jak procesor Intel 8086. Tryb rzeczywisty nie zapewnia ochrony pamięci przed użyciem przez inny proces oraz obsługi wielozadaniowości. W trybie tym pracowały programy w systemie operacyjnym DOS. W trybie rzeczywistym dostępna jest 1-megabajtowa przestrzeń adresowa. Adres logiczny (programowy) składa się z dwóch liczb 16-bitowych: segmentu (numeru segmentu) oraz przemieszczenia względem początku segmentu (ang. offset). Ponieważ segment i przemieszczenie mają 16 bitów, dlatego w trybie rzeczywistym można maksymalnie zaadresować 1088 kB pamięci, ale procesory Intel 8086, 8088, 80188 oraz 80186 mają 20-bitową szynę adresową i z tego powodu mogą zaadresować tylko 1024 kB pamięci – przy próbie sięgnięcia do adresu powyżej 1024 kB w rzeczywistości odwołają się do adresu o 1 MB niższego.



Na czym polega praca procesora w trybie chronionym (wirtualnym)? Omów rolę tablicy deskryptorów

Tryb chroniony umożliwia adresowanie pamięci w większym zakresie niż 1 MiB (tryb rzeczywisty). Jeżeli szyna adresowa procesora ma 32 bity wówczas istnieje możliwość fizycznego zaadresowania pamięci nie tylko 1MB jak to ma miejsce w trybie rzeczywistym ale aż do 2^{32} = 4GB. W trybie wirtualnym chronionym pracy procesora adres logiczny składa się, podobnie jak w trybie rzeczywistym z 16 bitowej części segmentowej nazywanej selektorem segmentu oraz 32 bitowego przemieszczenia.

Zbiór wszystkich adresów logicznych tworzy logiczną przestrzeń adresową. Natomiast dwuwymiarowa budowa adresu logicznego prowadzi do podziału pamięci na segmenty zwanego segmentacją przestrzeni logicznej. Selektor segmentu zawiera następujące pola:

- numer deskryptora segmentu (13 bitów) będący indeksem do tablicy deskryptorów,
- kod TI typu tablicy deskryptorów: g1obala/lokalna (1 bit),
- poziom ochrony zadania żądającego dostępu do pamięci (2 bity) ,

Numer deskryptora wskazuje na odpowiednią pozycję w tablicy deskryptorów. Tablica deskryptorów składa się z = 8192 pól zwanych deskryptorami segmentów po 2^3 = 8 bajtów na każdy deskryptor. Zajmuje ona zatem jeden segment o objętości 2^{16} = 64kB. Deskryptory segmentów wchodzące w skład tej tablicy stanowią pełny opis segmentu pamięci w trybie wirtualnym. Każdy deskryptor segmentu zawiera następujące pola: adres bazowy segmentu (32 bity), wielkość segmentu (20 bitów), atrybuty, czy segment obecny fizycznie w pamięci (1 bit), pole dostępne dla programu (1 bit), poziom ochrony (2 bity), ziarnistość segmentu (segment o długości 2^{20} = 1MB lub 2^{32} = 4GB) (1 bit), długość słowa (1 hit), czy segment pamięci (1 bit), typ segmentu (kodu/danych/odczyt/zapis) (3 bity), czy segment użyty (1 bit).

Opisz ideę i elementy systemu pamięci cache

System pamięci kieszeniowej cache składa się z trzech zasadniczych elementów:

- banku danych pamięci cache (pamięci danych) przeznaczonego do przechowywania zapisywanej i odczytywanej informacji,
- katalogu pamięci cache (tzw. TAG-RAM) służącego do szybkiego sprawdzania, czy poszukiwana informacja znajduje się w pamięci danych,

- sterownika pamięci cache, którego zadaniem jest zapewnienie współpracy pamięci kieszeniowej z systemem, a przede wszystkim zapewnienie zgodności zawartości pamięci cache z pamięcią główną.

W celu zapewnienia możliwości szybkiego sprawdzenia, czy komórka pamięci, na której ma być wykonana operacja, jest odwzorowana w pamięci cache, bank danych i katalog, tworzą tzw. jednoblokową pamięć asocjacyjną.

Pamięć cache stanowi jeden blok, który jest dzielony na zestawy. Pamięć główna dzielona jest na strony o rozmiarze równym rozmiarowi bloku pamięci cache. Strony są z kolei również dzielone na zestawy, przy czym ilość zestawów w stronie jest identyczna jak w bloku pamięci cache. Każdy zestaw w pamięci cache ma swoją pozycję w katalogu TAG-RAM. Zawiera ona adres strony z którego pochodzi dany zestaw. Każdy zestaw jest wpisywany na miejsce w pamięci cache do zestawu o numerze zgodnym z numerem zestawu na stronie. Umożliwia to szybkie sprawdzenie obecności zestawu poprzez sprawdzenie adresu bloku w określonej pozycji katalogu.

Lista i format rozkazu. Omów przykładowe rozkazy realizowane przez procesor

Rozkazem (instrukcją maszynową) nazywamy najprostszą operację, której wykonania programista może zażądać od procesora. Rozkazy (jak inne informacje) są przechowywane w systemie mikroprocesorowym w postaci słów binarnych

Formatem rozkazu nazywamy sposób rozmieszczenia informacji w kodzie rozkazu Kod rozkazu: musi zawierać określenie rodzaju wykonywanej operacji (kod operacji) - w pierwszym bajcie (bajtach) kodu rozkazu. Może zawierać operandy i/lub adresy operandów wykonywanych operacji

Listą rozkazów nazywamy zestaw wszystkich instrukcji maszynowych (rozkazów), jakie potrafi wykonać dany procesor

- rozkazy przesłań
- rozkazy arytmetyczne i logiczne
- rozkazy sterujące (skoki, wywołania podprogramów, pętle, itp.)
- inne (np. sterowanie pracą koprocesora, rozkazy testujące, operacje w trybie chronionym)

Przykładowe rozkazy:

- Rozkazy przesłań: MOV - move - przesłanie słowa między dwoma rejestrami; STR - store - przesłanie słowa ze wskazanego w rozkazie rejestru do komórki pamięci; LD - load - przesłanie słowa z pamięci do wskazanego rejestru; XCH - exchange - wymiana zawartości rejestrów
- Rozkazy arytmetyczno-logiczne: ADD - addition – dodawanie; SUB - subtraction – odejmowanie; MUL - multiplication – mnożenie; DIV - division – dzielenie; NEG - negate – negacja; DEC - decrement - zmniejszanie wartości o jeden; INC - increment - awiększanie wartości o jeden; AND - conjuction – iloczyn; OR - disjunction – suma; EXOR - exclusive-OR - suma mod 2; NOT - complement - negacja
- Rozkazy sterujące pracą programu: JMP - jump - skok bezwarunkowy do wyspecyfikowanego miejsca w pamięci; CALL - jump to subroutine - wywołanie podprogramu; RET - return - skok powrotny z podprogramu; SKIP - skip - przeskok przez jeden rozkaz
- Rozkazy sterujące pracą procesora : HALT - stop - zatrzymanie wykonania programu; NOP - no operation - rozkaz pusty.

Bazy danych

Relacyjne bazy danych – teoria, istota, właściwości, model danych

RBD bazuje na pojęciu **relacji** (podzbiór iloczynu kartezjańskiego zbiorów). RBD dostarcza tylko jednego sposobu reprezentacji danych: Jest nim **dwuwymiarowa tabela** nazywana relacją. **Relacyjna baza danych - baza danych złożona z dwóch lub więcej tabel powiązanych ze sobą za pomocą relacji.** Celem tworzenia relacyjnych baz danych jest istotne zmniejszenie ich rozmiarów w stosunku do prostych baz danych. Istotą relacyjnych baz danych jest rozbicie pojedynczej bazy danych na kilka logicznie powiązanych mniejszych.

RBD jest **zbiorem relacji**. Relacja to skończony zbiór krotek (krotka to jeden wiersz tabeli). Każdy wiersz tabeli musi być różny, natomiast każda kolumna musi być tego samego typu. Porządek kolumn i wierszy nie jest istotny. Kolumny są atrybutami relacji natomiast wiersze to krotki. **Atrybuty** mogą posiadać następujące właściwości: kluczowy, niekluczowy, obowiązkowy, nieobowiązkowy, puste(wartość null), wyliczane, jedno i wielowartościowe, segmentowe. Pojęcie **klucza relacji** określa atrybut tabeli, od którego wszystkie pozostałe atrybuty relacji są funkcyjnie zależne.

Klasyczne modele danych a model obiektowy – analiza porównawcza

Klasyczne modele danych reprezentują schematy za pomocą struktur rekordów i powiązań między nimi. Semantyczne modele danych dostarczają bogatszy zbiór narzędzi do budowy schematów i umożliwiają twórcom BD określenie w schemacie bazy danych znaczenia lub semantyki dotyczących dziedziny aplikacji. OMD dostarcza środków do realizacji tożsamości obiektów – rozróżnienie dwóch obiektów o takich samych cechach.

Model klasyczny

- **Hierarchiczne** – ma strukturę odwróconego drzewa, jedna z tabel pełni role korzenia a pozostałe mają postać gałęzi biorących swój udział w korzeniu. HMBD używa dwóch struktur danych: typy rekordów i związków nadrzędny-podrzędny. Relacje w HMBD są reprezentowane w kategoriach *ojciec- syn*, tzn. tabela typu *ojciec* może być powiązana z wieloma tabelami typu *syn*, lecz pojedynczy *syn* może mieć tylko jednego *ojca*. Aby dotrzeć do konkretnych danych, zaczynamy od korzenia i przedzieramy się przez całe drzewo do interesującej nas danej.
- **Sieciowe** – rozszerzenie modelu hierarchicznego. Ich struktura wygląda tak jak w HMBD z tą różnicą, że kilka drzew może dzielić ze sobą gałęzie, a każde drzewo stanowi część ogólnej struktury BD (wiele korzeni, wspólne gałęzie). Relacje są definiowane przez tzw. typy kolekcji. Kolekcja – niejawna konstrukcja formalna łącząca dwie tabele przez przypisanie jednej z nich roli właściciela, a drugiej – roli członka. Umożliwiają wprowadzenie relacji *jeden do wielu.*
- **Relacyjne** - dane przechowuje się w *domenach,* czyli tabelach. Każda tabela składa się z *krotek,* czyli rekordów, oraz *atrybutów,* czyli pól. Fizyczna kolejność pól i rekordów w tabeli jest bez znaczenia. Relacje w RMBD można podzielić na trzy grupy: *jeden do jednego, jeden do wielu* oraz *wiele do wielu.* Każde dwie tabele, między którymi istnieje relacja, są ze sobą jawnie powiązane, ponieważ dzielone przez nie pola zawierają odpowiadające sobie wartości.

Model obiektowy. Obiektowa baza danych składa się z obiektów i klas obiektów, powiązanych pewną liczbą mechanizmów abstrakcji:

- obiekt jest pakietem danych i procedur
- dane są trzymane w atrybutach obiektu
- procedury są definiowane za pomocą za pomocą metod obiektu
- metody są uaktywniane przez komunikaty przekazywane między obiektami

Rozproszone bazy danych – architektury wielowarstwowe, klient –serwer

Architektura wielowarstwowa (trójwarstwowa)

Podział zadań: serwer danych, serwer aplikacyjny, serwer http i klient. W tym modelu wszystkie funkcje aplikacji zostały rozdzielone, a więc powstały specjalizowane części odpowiedzialne za poszczególne operacje:

- interfejs użytkownika – komunikacja użytkownika z systemem
- reguły biznesowe – przetwarzanie danych
- baza danych – składowanie informacji

Architektura klient – serwer:

- serwer plików – zapytanie SQL wyrażone przez klienta jest wysłane do serwera jako zapotrzebowanie na odpowiedni plik wymagany do wykonania danego zapytania.
- serwer SQL – zapytanie SQL jest wysyłane do serwera, który je wykonuje i zwraca szukane dane.

Normalizacja baz danych – teoria, zasady, analiza użytkowa

Normalizacja – jest procesem identyfikowania logicznych związków między elementami danych oraz projektowania BD w taki sposób, aby reprezentowała te związki. Logiczne związki między elementami danych nazywamy związkami zależności lub determinowania. Zależność między elementami danych wskazuje kierunek w związku.

Pierwsza Postać Normalna – (*first Normal Form*), pierwsza postać normalna – Relacja jest w 1NF gdy każdy jej atrybut niekluczowy jest funkcyjnie zależny od klucza głównego. Oznacza konieczność utworzenia dwóch tabel dla funkcyjnie zależnych atrybutów oraz dla funkcyjnie niezależnych atrybutów.

Druga Postać Normalna – (*second Normal Form*), druga postać normalna – Relacja jest w 2NF gdy jest w 1NF i każdy atrybut niekluczowy (nie należący do żadnego klucza) jest w pełni funkcyjnie zależny od klucza głównego. W przeprowadzaniu relacji z 1NF do 2NF należy usunąć niepełne zależności funkcyjne, tj. atrybuty zależne od części klucza – rozdzielamy determinujące i zależne elementy danych na osobne tabele.

Trzecia Postać Normalna – Relacja jest w 3NF gdy jest w 2NF i każdy jej niekluczowy atrybut jest bezpośrednio zależny od klucza głównego. Aby przejść z 2NF do 3NF trzeba usunąć tzw. zależności przechodnie między danymi.

Interfejs Systemu zarządzania Bazą Danych – język SQL podstawowe funkcje

Zbiór programów umożliwiających tworzenie i reprezentację BD, jej definiowanie, konstruowanie i przetwarzanie. **Funkcje:**

- definiowanie, specyfikowanie typów danych przechowywanych w BD
- konstruowanie, zapamiętywanie danych na nośniku kontrolowanym przez SZBD
- przetwarzanie i generowanie zapytań

Język SQL to strukturalny język zapytań. Jego podstawowe funkcje to:

- definiowanie danych
- tworzenie zapytań – wyszukiwania danych w bazie danych;
- operowanie danymi – wstawianie, modyfikowanie i usuwanie danych z bazy danych;
- sterowanie danymi – sterowanie transakcjami, zatwierdzanie i wycofywanie.
- przenośność aplikacji

Składnia języka SQL – podstawowe instrukcje

- SELECT – mówi SZBD, które kolumny tabeli należy wyszukać z podanych tabel. SELECT nazwa_atrybutu, funkcja sumaryczna, wyrażenie, *, stała – wartość.
- FROM – określa tabele z których pobierane są atrybuty. FROM nazwa_tabeli.
- WHERE – definiuje warunki, według których chcemy wyświetlić krotki, filtruje wiersze
- ORDER BY – porządkuje zestaw krotek według wskazanego pola, zawsze na końcu zapytania
- GROUP BY – pogrupowanie tabel według określonych kryteriów. Grupuje wiersze. Związane z nią są operacje sumaryczne, grupować możemy tylko te atrybuty, które mają niewielką, skończoną liczbę wartości.
- HAVING – towarzyszy funkcji GROUP BY, nie występuje sama, określa grupę, która spełnia określony warunek.
- SUM, AVG, MIN, MAX, COUNT – operacje sumaryczne
- INSERT, DELETE, UPDATE – wstawianie/usuwanie/aktualizacja danych do/z tabeli

Przekształcenia zapytań w złączenia, warunki i ograniczenia

Baza danych zawiera informacje rozproszone po kilku tabelach, powiązanych ze sobą kluczami. Problem złączenia zewnętrznego pojawia się, gdy chcemy złączyć dwie tabele, z których jedna nie ma wiersza pasującego do pewnego wiersza w drugiej. Należy pamiętać o podaniu nazw tabel w klauzuli FROM i predykatów złączeniowych, aby powiązać wiersze, które mają równą wartość w kolumnie lub kolumnach.

```
SELECT D.NUMDZ, D.NAZWDZ, COUNT(*)
FROM DZIAŁ D, PRAZ P
WHERE D. NUMDZ = P.NUMDZ – predykat złączeniowy
```

Aby poradzić sobie z problemem złączenia zewnętrznego można użyć operacji **UNION**. Operacja ta umożliwia połączenie dwóch lub więcej instrukcji SELECT z jednoczesnym sumowaniem ich wyników. Wiersze wynikowe każdej instrukcji SELECT zostają obliczone i ustawione jeden pod drugim, po czym następuje sortowanie w celu wyeliminowania powtarzających się wyników. Operacja **UNION ALL** jest inną postacią UNION, nie dopuszcza do sortowania wierszy – pozostają więc powtarzające się wyniki.

Klauzule SELECT w kolejnych zapytaniach muszą mieć tę samą liczbę wyrażeń odpowiednio zgodnych typów danych.

Zarządzanie transakcjami

Aby zapewnić integralność bazową, muszą istnieć mechanizmy do obsługi awarii w bazach. Musi istnieć również mechanizm umożliwiający współbieżną pracę na bazie danych. Jądro Systemu Zarządzania Bazą Danych – nadzoruje jak wykonywane są transakcje.

W Systemie Bazy Danych z wieloma użytkownikami **transakcjami** nazywamy procedury, które wprowadzają zmiany do BD lub, które wyszukują dane.

Właściwości transakcji:

- niepodzielność – albo transakcja zostanie cała wykonana albo wcale
- spójność – wszystkie transakcje muszą być wykonane tak, aby po jej wykonaniu baza pozostała spójna
- izolacja – dla transakcji tworzy się barierę aby inne transakcje nie wpływały na aktualną, izoluje od innych
- trwałość – jeżeli transakcja kończy się, zmiany w bazie powinny być utrwalone

Sytuacja konfliktowa wystąpi gdy dwie transakcje T1 i T2 są zainteresowane dostępem do tego samego obiektu – cztery możliwe przypadki:

- odczyt – odczyt – dwie transakcje żądają odczytu wartości

- zapis – zapis – utrata aktualizacji
- zapis – odczyt – odczyt niewłaściwy
- odczyt – zapis – odczyt niepowtarzalny

Mechanizmy porządkujące kolejność transakcji, które wagują je. Metody:
- metody porządkowania wg etykiet
- metody walidacji
- metody blokowania

Tabele słownikowe – znaczenie w Systemach Zarządzania Bazami Danych

Słownik danych lub katalog systemowy jest sercem SZBD. W czasie konfigurowania SZBD, administrator BD tworzy grupy użytkowników, określa dla nich hasła, przydziela uprawnienia do danych i operacji dla każdej grupy użytkowników. Wszystkie te informacje zostają zapisane do zbioru tabel systemowych.

Informacje przechowywane w słowniku danych:
- opis relacji w BD – nazwy relacji, nazwy kolumn, typy danych kolumn
- deklaracje kluczy głównych i obcych
- opis perspektyw
- deklaracje grup użytkowników i uprawnień
- informacje nt. indeksów, rozmiarów plików, struktur plików i klastrów

Fizyczny projekt bazy danych

Zbiór danych tworzących BD musi być fizycznie przechowywany na komputerowym nośniku informacji. Większość BD jest za duża, aby zmieścić się w pamięci głównej, dlatego muszą być przechowywane w pamięci pomocniczej.

Fizyczne przechowywanie danych:
- dyski (urządzenia pamięci o swobodnym dostępie);
- każda ścieżka przechowuje informację mierzoną w KB i jest często podzielona na sprzętowo określone sektory;
- ścieżka podzielona jest na pewną liczbę bloków o równych rozmiarach;
- podział na bloki jest realizowany metodami programowymi podczas formatowania dysku system operacyjny;
- między blokami są pozostawione wolne miejsca zwane przerwami międzyblokowymi;
- dane są przesyłane między pamięcią główną a dyskiem porcjami będącymi wielokrotnością bloków;
- adres fizyczny bloku składa się z kombinacji numeru powierzchni, numeru ścieżki na powierzchni i numeru bloku na ścieżce;
- adres jest przekazywany do dyskowego urządzenia we – wy (polecenie odczytu – kopiowanie bloku do bufora tzn. do zarezerwowanego obszaru pamięci głównej, o pojemności jednego bloku; polecenie zapisu – kopiowanie zawartości bufora na odpowiedni blok na dysku).

Rozmieszczenie rekordów:
- dane są zazwyczaj fizycznie przechowywane w postaci rekordów (zbiór powiązanych ze sobą logicznie wartości danych składających się z pewnej liczby bajtów i odpowiadających poszczególnym polom);
- każdy rekord ma określony typ rekordów;
- plik jest sekwencją takich rekordów (wszystkie rekordy w pliku są zazwyczaj tego samego typu);
- pliki i rekordy odwzorowywane są na bloki;
- plik zazwyczaj składa się z wielu bloków*;*

- każdy plik ma zwykle nagłówek (dyskowe adresy bloków i typ rekordów);
- organizacja plików oraz operacje uzyskiwania dostępu i we – wy są ze sobą istotnie powiązane;
- wyszukanie rekordu na dysku wymaga przekopiowania do buforów pamięci głównej jednego lub więcej bloków, a następnie wyszukania wymaganego rekordu w buforze, korzystając z informacji zapisanej w nagłówku pliku
- celem dobrego fizycznego projektu BD jest zminimalizowanie operacji we – wy.

Model pamięci zewnętrznej (PZ):

- PZ jest logicznie podzielona na segmenty;
- każdy segment stanowi wirtualny obszar złożony z wielu stron (bloków – minimalna jednostka przydziału PZ i transmisji między pamięcią główną a pomocniczą);
- każda strona (blok) ma swój unikalny adres;
- każdy plik zapisuje się w jednym segmencie (segment może zawierać wiele plików);
- plik jest zbiorem rekordów (rekord składa się z pól);
- każdy rekord ma swój adres bezwzględny lub adres, który jest złożeniem adresu strony na której się znajduje i przesunięcia względem jej początku;
- relacja jest pamiętana w jednym pliku;
- dostęp do danych dokonuje się za pomocą ścieżek dostępu, np. indeksu (plik zawierający rekordy wskazujące na rekordy pliku).
- BD i katalog są zazwyczaj przechowywane na urządzeniach dyskowych
- dostęp do urządzeń dyskowych jest kontrolowany przez system operacyjny gospodarza;
- menedżer danych kontroluje wszystkie operacje dostępu do informacji w BD (używa w tym celu funkcji zarządzania danymi, dostarczonymi przez system operacyjny gospodarza w jego systemie zarządzania plikami lub używa swoich własnych programów);
- kompilator języka definicji danych „tłumaczy” instrukcje wyrażone w DDL i uaktualnia katalog systemowy;

Strategia projektowania aplikacji bazodanowej z interfejsem webowym

Systemy scentralizowane:

- Jako duże komputerowe systemy, np. mainframe. W systemach tych zintegrowano aplikacje bazodanowe z systemem zarządzania bazą danych. Obsługują one wielu użytkowników za pośrednictwem terminali
- Jako małe systemy (1 PC nie podłączony do sieci), których przeznaczeniem jest współpraca z jednym użytkownikiem.

Systemy wielowarstwowe:

- Dwuwarstwowe: klient – serwer
- Trójwarstwowe: klient – serwer aplikacji – serwer bazy danych (SZBD)
- Wielowarstwowe: przeglądarka internetowa – serwer WWW – serwer aplikacji – serwer bazy danych.

Od strony technicznej istnieją 3 typy aplikacji bazodanowych, które różnią się dopuszczalną ilością użytkowników i rozmiarem bazy danych.

- Lokalna aplikacja z wbudowanym systemem BD – tania i prosta w budowie, łatwa w instalacji, niskie wymagania sprzętowe, nie wymaga oddzielnego serwera; ograniczeniem jest nieduża liczba jednoczesnych użytkowników, kiepska skalowalność i utrudnienia w aktualizacjach oprogramowania.
- Architektura zawierająca serwer BD, np. MS SQL Server czy PostgreSQL, działający na oddzielnej maszynie w sieci lokalnej; użytkownicy łączą się ze swoich komputerów z serwerem BD przy pomocy aplikacji klienckich; wlicza się to także aplikacje webowe; może

obsłużyć dużą liczbę użytkowników, jednoczesny dostęp, rozwiązywanie konfliktów danych, możliwość wycofywania transakcji.

- Najbardziej zaawansowana architektura z serwerem aplikacji, w którym zaimplementowana jest logika biznesowa aplikacji (reguły zarządzania danymi i przeprowadzane operacje); serwer aplikacji łączy się z bazodanowym i udostępnia zaawansowane operacje na danych, wymuszając ścisłe reguły i wymagania co do tej operacji; aplikacje użytkowników łączą się z serwerem aplikacji i pozwalają na wykonywanie operacji na danych przy pomocy wygodnego interfejsu; wysoki poziom spójności danych, skalowalność, możliwość szybkiej zmiany logiki biznesowej.

Powszechnie stosowana jest architektura trójwarstwowa, wprowadzający logiczny i funkcjonalny podział aplikacji na 3 warstwy:

- Podstawowa – system zarządzania bazą danych; przechowuje dane używane przez system, obsługuje bazę z danymi;
- Pośrednia – serwer WWW; odpowiedzialny za odbieranie żądań od klienta, łączenie się z serwerem bazy danych, generowanie stron WWW na podstawie danych pobranych z BD;
- Końcowa – klient; poprzez przeglądarkę WWW; protokół TCP/IP przekazuje dane między aplikacjami przez Internet i ma mały wpływ na projektowanie aplikacji WWW bazodanowych.

Grafika i komunikacja człowieka z komputerem

Porównaj grafikę rastrową z grafiką wektorową

Grafika rastrowa - prezentacja obrazu za pomocą pionowo-poziomej siatki odpowiednio kolorowanych pikseli na monitorze komputera, drukarce lub innym urządzeniu wyjściowym. (np. pliki JPG, GIF, PNG).

Grafika wektorowa (obiektowa) –obraz opisany jest za pomocą figur geometrycznych (w przypadku grafiki dwuwymiarowej) lub brył geometrycznych (w przypadku grafiki trójwymiarowej), umiejscowionych w matematycznie zdefiniowanym układzie współrzędnych, odpowiednio dwu- lub trójwymiarowym. (pliki SVG.)

Podstawowa różnica to skalowalność. Zwiększając obraz SVG nie tracimy jakości bo obiekty opisane są jako punkty w przestrzeni, natomiast w grafice rastrowej tracimy na jakości bo obraz ma swój rozmiar.

Wymień i scharakteryzuj przekształcenie występujące w grafice komputerowej

Przekształcenia obrazów.

- Klasa 1 w klasę 2 – taki proces nazywamy segmentacją i identyfikuje on obszary, gdzie kolor i jasność są w przybliżeniu jednakowe.
- Klasa 2 w klasę 3 – możliwe przekształcenia to znajdowanie konturu (obszar odwzorowywany jest w krzywą zamkniętą) i ścienianie (tworzony jest zbiór zwany szkieletem obrazu).
- Klasa 3 w klasę 4 – proces ten nazywany jest segmentacją krzywych. Ma on na celu znalezienie punktów krytycznych na konturze, np. punktami krytycznymi dla wielokątów są ich wierzchołki.
- Klasa 4 w klasę 3 – obejmuję interpolację, w której krzywa gładka przechodzi przez punkty i aproksymację, gdy krzywa gładka przechodzi w pobliżu punktów.
- Klasa 3 w klasę 2 – obejmuje zagadnienia wypełnienia konturów (nazywane jest to czasem cieniowania). Jeżeli na wejściu jest szkielet obiektu to obraz musi być zrekonstruowany poprzez rozszerzanie.

- Klasa 2 w klasę 1 – Przejście pomiędzy tymi klasami polega na poprawieniu estetyki obrazu, osiąga się to poprzez zastosowanie filtrów dolnoprzepustowych lub z zastosowaniem filtrów zakłócających drżenie kolorów.

Przedstawione wyżej przekształcenia obrazów obejmują rozpoznawanie obrazów dla przejścia z klasy niższej do wyższej oraz grafikę komputerową dla przekształcenia obrazu z klasy wyższej do niższej.

Innymi przekształceniami mogą być przekształcenia wewnątrz klasowe – przykładem może być polepszanie jakości obrazu. Ważną klasę zagadnień stanowią przekształcenia między obrazami dwuwymiarowymi, a trójwymiarowym. Obejmują one rzutowanie – przejście od obrazu trójwymiarowego do dwuwymiarowego i reprojekcja – rekonstrukcja obrazu trójwymiarowego na podstawie rzutów. Inną klasą zastosowań są przekształcenia geometryczne obiektów. Rozważa się także przy nich problemy usuwania zasłoniętych linii lub zasłoniętych powierzchni.

Rzutowanie jest uważane za podstawowe przekształcenie w grafice komputerowej, gdyż komputerowa wizualizacja obiektu wymaga by był on odwzorowany na płaską kartkę papieru. Rzuty można podzielić na dwie klasy: równoległe i perspektywiczne.

Omów wybrane metody rysowania podstawowych prymitywów w grafice komputerowej

TCanvas jest obiektem należącym to innego obiektu. Używa się go na takich obiektach jak: Form, Bitmap, Image, PaintBox, Printer. Niezależnie od obiektów kreślenie grafiki zawsze odbywa się tak samo. Obiekty graficzne są kreślone na płótnie pracując w układzie współrzędnych kartezjańskich.

Klasa TCanvas zawiera funkcje wyświetlające prymitywy (linie, łuki okręgów, elipsy, wielokąty, tekst). Do rysowania używamy funkcji:

- Arc – funkcja rysująca łuk elipsy wpisanej w prostokąt
- Ellipse – funkcja rysująca elipsę wpisaną w prostokąt o danych wierzchołkach.
- FloodFill – funkcja wypełniająca ograniczony obszar bieżącą barwą
- LineTo – funkcja rysująca linię od aktywnej pozycji pióra (PenPos) do podanej pozycji
- MoveTo – przenoszenie pióra do podanej pozycji
- Pie – rysowanie wycinka elipsy wypełnionego bieżącym stylem i barwą
- Cord – rysowanie odcinka elipsy
- Polygon – rysowania wielokąta o podanych wierzchołkach
- Polyine – rysowanie łamanej o podanych wierzchołkach
- Rectangle – rysowanie prostokąta
- FillRect – rysowanie wypełnionego prostokąta bieżącym stylem i barwą wypełnienia
- RoundRect – rysowanie prostokąta o zaokrąglonych wierzchołkach
- Refresh – metoda odświeżająca i przywracająca stan domyślny (wszystkie piksele otrzymują określony kolor we właściwościach Color formularza)
- Brush - Ustalanie koloru i stylu wypełniania
- TextOut - Wypisanie ciagu znaków na ekranie od podanego miejsca
- Pixels - Wyświetlanie punktu w podanym kolorze
- Pen - Sprawdzenie pozycji pióra

Zasadniczo myślę, że nie o to chodzi, a o algorytmy rysowania tych prymitywów, ale to i tak nikt tego nie zapamięta (http://goo.gl/4XhmU).

Omów wykorzystanie algorytmów okienkowania

Oknem nazywamy obszar w rzeczywistej przestrzeni urządzenia bądź programu.

W zastosowaniach elementy rysunku definiowane są w rzeczywistym układzie współrzędnych. Podczas przenoszenia na urządzenie graficzne następuje przejście od układu rzeczywistego do układu

urządzenia graficznego. Można wtedy określać, jaki fragment rysunku ma być zobrazowany, oraz jak mają być usytuowane i jaka ma być ich wielkość. Wykonujemy wtedy dwie rodzaje operacji: okienkowanie, czyli odwzorowanie obszaru określonego we współrzędnych rzeczywistych na obszar zdefiniowany we współrzędnych danego urządzenia graficznego oraz obcinanie, czyli wyznaczenie elementów rysunku, leżących wewnątrz okna.

Operacja okienkowania jest więc sposobem transformacji obrazu z układu współrzędnych rzeczywistych na układ współrzędny urządzenia (np. tradycyjny układ współrzędnych w aplikacjach z oknami – punkt 0,0 znajduje się w lewym górnym rogu a obszar cechuje się podobnymi właściwościami jak pierwsza ćwiartka układu współrzędnych).

Omów wykorzystanie algorytmów obcinania

Elementy rysunku są definiowane w rzeczywistym układzie współrzędnych, jednaka podczas przenoszenia na urządzenie graficzne następuje przejście od układu rzeczywistego do układu urządzenia graficznego. Można wtedy określać jaki fragment rysunku ma być zobrazowany, oraz jak mają być usytuowane i jaka ma być ich wielkość. Wykonywane są wtedy dwie operacje: okienkowanie i obcinanie.

Obcinanie – wyznaczanie elementów rysunku, leżących wewnątrz okna. Wykorzystywane są wtedy algorytmy wyznaczania czy dany punkt leży wewnątrz dowolnego wielokąta oraz algorytmy obcinania odcinka. W przypadku odcinka stosuje się jeden z dwóch algorytmów: Cohena-Shutelanda, który polega na tym, że koniec odcinka o kodzie niezerowym jest zastępowany punktem przecięcia badanego odcinka z prostą zawierającą jeden z boków odcinka, co powoduje odrzucenie fragmentu odcinka lezącego poza oknem. Następnie pozostała część jest obcinana prostymi zawartymi w pozostałych bokach. Innym algorytmem jest algorytm Lianga-Barsky'ego, w którym zostało wykorzystane również równanie prostej przechodzącej przez dwa punkty, ale parametryczne – umożliwiające obniżenie kosztów.

Co to są współrzędne jednorodne. Omów ich wykorzystanie w grafice komputerowej

Współrzędne jednorodne – sposób reprezentacji punktów n-wymiarowych za pomocą n+1 współrzędnych. Punkt w przestrzeni dwuwymiarowej opisuje para liczb (x,y), a we współrzędnych jednorodnych trójka (x,y,W).

Wykorzystanie w grafice komputerowej:

- Przekształcenia macierzowe – przekształcenia typu: skalowanie, obrót, pochylenie i translacja można zapisać przy pomocy macierzy; przy pomocy współrzędnych jednorodnych można zapisać wszystkie przekształcenia na jednej macierzy, bez rozpisywania ich pojedynczo.
- Obcinanie – ogólnie obcinanie ogranicza się do tworzenia ostrosłupa widzenia, w którego obrębie znajdują się obiekty widoczne w danym rzucie; taka technika jest skompilowana obliczeniowo; można ją zastąpić poprzez przekształcenie ostrosłupa w sześcian, a żeby zapewnić temu poprawność wyników, to obcinanie należy wykonać przy pomocy współrzędnych jednorodnych. Jest to powszechnie stosowane podejście w rozwiązaniach sprzętowych.

Omów stosowane rodzaje rzutów w grafice komputerowej

Rzutowanie jest podstawowym przekształceniem w grafice komputerowej, gdyż komputerowa wizualizacja obiektu wymaga by był on odwzorowany na płaską kartkę papieru. Rzuty można podzielić na dwie klasy: rzuty równoległe i rzuty perspektywiczne

Własności rzutów równoległych:

- Zachowuje równoległość prostych
- Zachowuje stosunek długości odcinków równoległych

- Zachowuje związki miarowe figur płaskiej równoległej do płaszczyzny rzutowania
- W rzucie równoległym wszystkie proste rzutowania mają kierunek równoległy, gdy jest on prostopadły do płaszczyzny rzutowania, to jest to rzut równoległy ortogonalny
- Stosuje się go w rysunku technicznym

Własności rzutów perspektywicznych:

- Pozwala na bardziej realistyczną wizualizację obiektów trójwymiarowych
- Daje wrażenie głębi
- W rzucie środkowym (szczególny przypadek rzutu perspektywicznego) zmienione zostają relacje długości, na przykład rzuty odcinków leżących bliżej rzutni są dłuższe niż rzuty odcinków tej samej długości ale bardziej oddalonych od płaszczyzny rzutowania
- W rzucie perspektywicznym wszystkie proste (promienie rzutowania) mają punkt wspólny, nazywany jest on środkiem rzutowania. Odległość tego punktu decyduje o deformacji rysunku.

Rzutowanie w układzie obserwatora:

- Rzut równoległy ortogonalny – kierunek rzutu jest prostopadły do płaszczyzny rzutowania. Punkt P=(x,y,z) będzie miał współrzędne P`=(x,y,0).
- Rzut równoległy nieortogonalny – kierunek rzutu tworzy z rzutnią kąt alfa.
- Rzut środkowy - własnością takiego rzutu jest tzw. Punkt zbieżności prostych równoległych

Scharakteryzuj operacje na wielokątach występujące w grafice komputerowej.

Położenie punktu względem wielokąta. Jest to zadanie geometrii obliczeniowej. Istnieje wiele algorytmów; jednym z nich jest algorytm parzystości.

Wyznaczanie powłoki wypukłej wielokąta. Definicja zbioru wypukłego: zbiór Z nazywamy wypukłym, jeżeli zawiera wszystkie odcinki, których końcami są dowolne punkty ze zbioru Z. Definicja powłoki wypukłej. Powłoką wypukła nazywamy najmniejszy zbiór wypukły, do którego należą dane punkty. Algorytm Grahama wyznaczania powłoki wypukłej n punktów.

Triangulacja wielokątów. Triangulacja jest podziałem wielokąta na sumę trójkątów. Ułatwia ona wiele zadań, do których należą np. wypełnianie obszarów, określanie zasłaniania i oświetlania obiektów trójwymiarowych, wyznaczanie linii i ich przecięcia. Ważne jest by liczba trójkątów była jak najmniejsza.

Scharakteryzuj reprezentacje obiektów w grafice 2D i 3D

Obiekt graficzny – wszystko co można narysować przy pomocy komputera. Może nim być krzywa, figura, powierzchnia, bryła.

Figury i obszary płaskie: Brzeg obszaru może być definiowany kawałkami za pomocą krzywych, z kolei wnętrze za pomocą drzew czwórkowych lub za pomocą przecięcia, dodawania, odejmowania kilku ustalonych elementarnych klocków. Drzewo czwórkowe to niepusty zbiór etykietowanych węzłów takich, że istnieje wyróżniony węzeł (korzeń), a pozostałe są podzielone na n>=0 poddrzew z korzeniami. Jeśli chodzi o opis wnętrza to rozważany obszar jest opisany kwadratem, a jemu z kolei odpowiada korzeń drzewa czwórkowego; potomkowie będą związani z podziałem drzewa na mniejsze kwadraty;

Bryły: Często stosowanym opisem brył jest stosowanie drucianego szkieletu. Każda ściana jest określona zbiorem swoich krawędzi, a te z kolei są zdefiniowane przez wierzchołki. Prowadzi to do utworzenia tablicy wierzchołków i tablicy krawędzi. Kolejną strukturą jest lista ścian, której elementami są skończone ciągi numerów krawędzi stanowiących boki wielokątów ścian. Kolejna tablica zawiera listę krawędzi kolejnych ścian. Bryła może też być opisana poprzez drzew

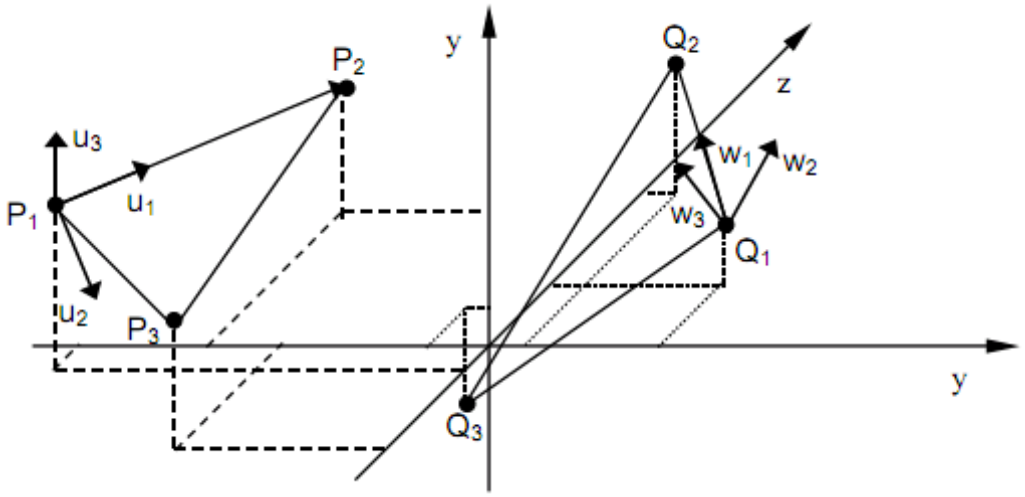
ósemkowych, i wpisać bryłę w sześcian, któremu odpowiada korzeń drzewa ósemkowego. Ten z kolei dzieli się na osiem mniejszych. Gdy cały sześcian jest wewnątrz, to węzłowi jest przypisany kolor czarny, biały gdy cały jest na zewnątrz. Gdy jest szary to sześcian jest dzielony na mniejsze i proces powtarza się do uzyskania jednolitego koloru lub do uzyskania minimalnego rozmiaru. Bryły można budować także w wyniku składania klocków, którymi mogą być dowolne półprzestrzenie punktów.

Wyjaśnij ideę przekształcenia 3 – punktowego

Zastosowanie szczególnie przydatne przy budowaniu scen trójwymiarowych, gdy jeden obiekt chcemy „dokleić" do drugiego.

Dane są dwie trójki punktów P_1, P_2, P_3 i Q_1, Q_2, Q_3. Szukamy takiej izometrii (odwzorowanie, które nie zmienia odległości między punktami), która:

- odwzorowuje punkt P1 w Q1
- kierunek $P = P_2 – P_1$ przekształca w kierunek $Q = Q_2 – Q_1$
- transformuje płaszczyznę przechodzącą przez P_1, P_2, P_3 na płaszczyznę wyznaczoną punktami Q_1, Q_2, Q_3.



Szukane przekształcenie jest złożeniem znanego już obrotu R i pewnej translacji T, którą pozostaje określić. Przypomnijmy, że obrazem punktu P_1 ma być Q_1. Punkt P_1 zostanie wcześniej poddany obrotowi R, a zatem T ma być przesunięciem o wektor $w = Q_1 - P_1R$.

Inżynieria Oprogramowania

Wymień najważniejsze modele cykle życia oprogramowania

Ocenia się, że w ostatnich trzydziestu latach efektywność sprzętu komputerowego wzrosła ok. 100 razy, podczas gdy efektywność procesu tworzenia oprogramowania tylko ok. 10 razy. Inżynieria oprogramowania, która doprowadziła do powstania modeli cyklu życia oprogramowania (procesy tworzenia oprogramowania) takich jak:

- Model kaskadowy;
- Realizacja kierowana dokumentami;
- Metodyka HIPO;
- Model ewolucyjny;
- Model przyrostowy;
- Model spiralny;
- Prototypowanie;

- Model formalnych przekształceń;
- Model piłki baseballowej;

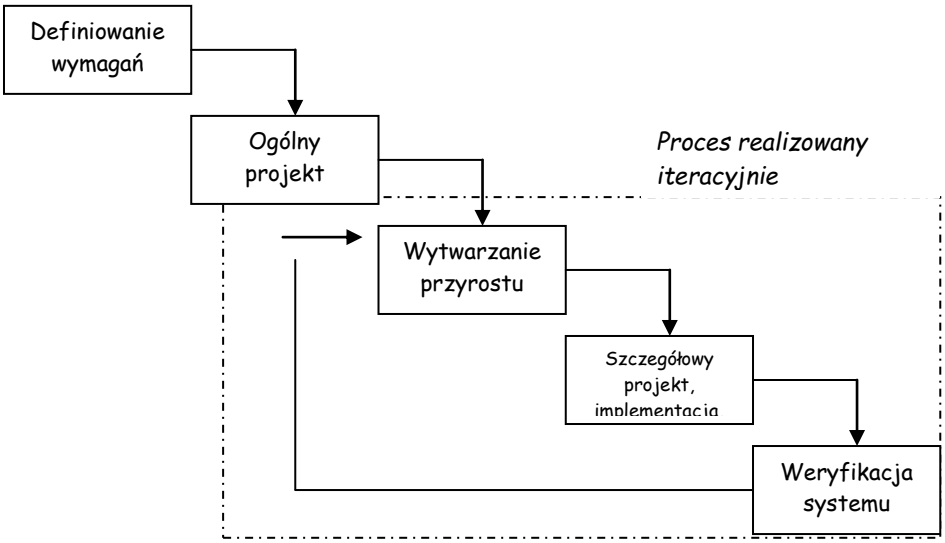
Scharakteryzuj model kaskadowy cyklu życia oprogramowania

Model kaskadowy (ang. *waterfall model*) – zwany również modelem liniowym lub wodospadowym, jest klasycznym modelem procesu tworzenia oprogramowania. Składa się on z kilku faz następujących kolejno po sobie z dobrze zdefiniowanymi wejściami i wyjściami.

- Analiza wymagań – podczas tej fazy ustalane są w toku narad z przyszłymi użytkownikami systemu: cele, wymagania i ograniczenia systemu. Wynikiem tej fazy jest specyfikacja wymagań;
- Projektowanie – wynikiem przebiegu tej fazy jest szczegółowy projekt systemu zgodny z wcześniej sporządzoną specyfikacją. Faza ta obejmuje identyfikację i opis zasadniczych abstrakcji oprogramowania i związków między nimi;
- Implementacja i testowanie – w tej fazie opracowany wcześniej projekt jest implementowany w konkretnym środowisku programistycznym, a następnie podlega sprawdzeniu czy każdy moduł spełnia swoją specyfikację;
- Integracja i testowanie systemu – podczas tej fazy poszczególne moduły i funkcje są integrowane w kompletny system, a następuje jako całość jest poddany testowaniu;
- Pielęgnacja – zazwyczaj jest to najdłuższa faza modelu kaskadowego. W tej fazie usuwane są błędy, które nie zostały wykryte we wcześniejszych fazach cyklu oprogramowania.

Scharakteryzuj model realizacji przyrostowej

Model przyrostowy jest modelem wytwarzania oprogramowania, zakładającym, że nie wszystkie elementy systemu zostaną wykonane w pierwszym okresie jego realizacji i zakłada dalszą rozbudowę wykonanego i przekazanego użytkownikowi Systemu Informatycznego.

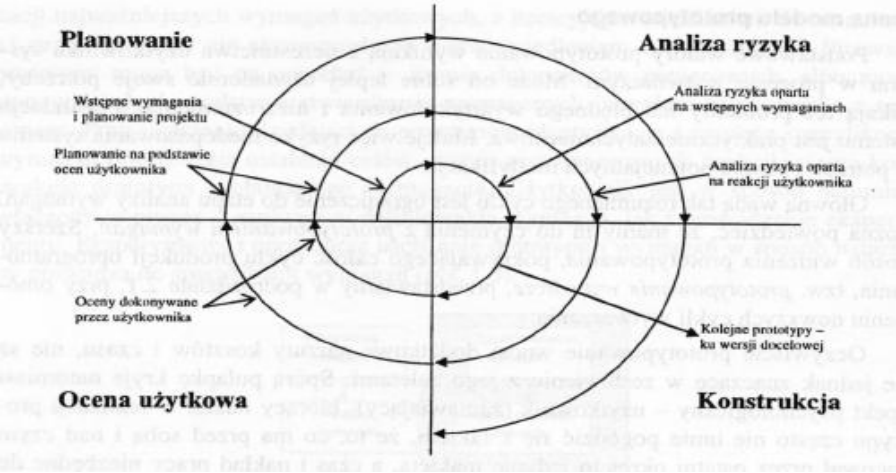


Zalety modelu przyrostowego:

- Klienci nie muszą czekać na dostarczenie całego systemu zanim zaczną czerpać korzyści wynikające z jego użytkowania.
- Klienci w oparciu o dostarczone prototypy mają możliwość zdobywania doświadczeń, które posłużą do bardziej precyzyjnego formułowania nowych wymagań.
- Zmniejsza się ryzyko porażki.
- Klient w pierwszej kolejności otrzymuje usługi zaspokajające jego priorytetowe potrzeby.

Scharakteryzuj model spiralny

Model spiralny polega na cyklicznym powtarzaniu czynności: planowania, analizy ryzyka, konstruowania i oceny, przy czym każde z powtórzeń bazując na wynikach wcześniejszej czynności jest wykonywane przy innych warunkach początkowych.



- **Planowanie** - Na tym etapie definiuje się cele i rozpoznaje zagrożenia dla procesu wytwarzania systemu. Tworzy się szczegółowe plany realizacji procesu.
- **Analiza ryzyka** - Prowadzi się szczegółową analizę każdego z rozpoznanych zagrożeń i podejmuje kroki zmierzające do ich likwidacji lub przynajmniej zmniejszenia ich wpływu na realizację przedsięwzięcia.
- **Konstrukcja** - Wybiera się model procesu wytwarzania systemu. Jeśli najpoważniejsze zagrożenie jest związane z interfejsem użytkownika, to odpowiednim modelem może być prototypowanie. Jeśli najbardziej zagrożone jest bezpieczeństwo, to tworzenie oparte na przekształceniach formalnych może być najwłaściwsze. Zastosowanie modelu kaskadowego może być najwłaściwsze, gdy głównym rozpoznanym ryzykiem jest integracja systemów.
- **Ocena** - Analizuje się uzyskane rezultaty i podejmuje decyzję o ewentualnym rozpoczynaniu nowej pętli spirali.

Scharakteryzuj fazę strategiczną procesu wytwarzania oprogramowania

Faza strategiczna (ang. Strategy phase) jest wykonywana zanim zostanie podjęta ostateczna decyzja o realizacji dalszych etapów przedsięwzięcia (np. negocjacje z klientem). Zasadniczym celem fazy strategicznej jest **dokonanie analizy potrzeby budowy systemu informatycznego**.

Najważniejsze czynności wykonywane w fazie strategicznej:

- Dokonanie serii rozmów (wywiadów) z przedstawicielami klienta;
- Analiza potrzeb informacyjnych przyszłego użytkownika;
- Określenie zakresu i kontekstu przedsięwzięcia;
- Prezentacja klientowi kilku możliwych rozwiązań;
- Wykonanie studium osiągalności;

Podstawowe rezultaty fazy strategicznej

- Określenie zakresu i kontekstu przedsięwzięcia
- Wykonanie studium osiągalności
- Opracowanie harmonogramu przedsięwzięć (diagram Gantta)
- Ocena rozwiązań
- Wybór rozwiązania
- Wstępna specyfikacja wymagań dla wybranego rozwiązania - opis ogólny wymagań i ograniczeń charakteryzujących projektowany system.

Kluczowe czynniki sukcesu tej fazy

- Szybkość pracy – szczególnie ważne dla firm wykonujących oprogramowanie na zamówienie. Faza ta wymaga zaangażowania niewielu kompetentnych osób, które potrafią wykonać pracę w krótkim czasie.
- Zaangażowanie kluczowych osób ze strony klienta
- Uchwycenie ogólnej całości systemu. Częstym błędem jest koncentracja na pewnych fragmentach systemu.

Scharakteryzuj fazę określania (formułowania) wymagań procesu wytwarzania oprogramowania

Celem tej fazy jest **dokładne określenie wymagań klienta wobec tworzonego systemu**. W fazie tej dokonywana jest więc zamiana celów klienta na konkretne wymagania zapewniające osiągnięcie tych celów. Proces określenia wymagań należy więc rozumieć nie jako proste zbieranie wymagań klienta, lecz jako proces, w którym klient wspólnie z przedstawicielami producenta (analitykami) konstruuje zbiór wymagań wobec systemu zgodny z postawionymi celami.

Uwarunkowania związane z określaniem wymagań

- Różni klienci stosują różną terminologię mówiąc o tych samych problemach.
- Klienci spodziewają się poprawy stanu aktualnego organizacji, często nie przewidując jakie zmiany w organizacji spowoduje nowy system
- Zleceniodawcy i użytkownicy to często inne osoby. Zleceniodawca nie potrafi zawsze właściwie przewidzieć potrzeby rzeczywistych użytkowników.

Kluczowe czynniki sukcesu tej fazy:

- Zaangażowanie właściwych osób ze strony klienta
- Pełne rozpoznanie wymagań, wykrycie przypadków oraz dziedzin szczególnych i nietypowych. Błąd popełniany w tej fazie polega na koncentrowaniu się na sytuacjach typowych.
- Sprawdzenie kompletności i spójności wymagań.

Scharakteryzuj fazę analizy procesu wytwarzania oprogramowania

Po określeniu wymagań są one poddawane analizie, której produktem jest **ogólny model logiczny systemu**, który umożliwia odpowiedź na pytanie: jak system będzie działać bez określania szczegółów implementacyjnych.

Cel fazy analizy. Zasadniczym celem fazy analizy jest określenie sposobu działania systemu i zasad jego współpracy z innymi systemami. Na tym etapie dokonuje się analizy obszaru działania przyszłego systemu. Przeprowadza się szczegółowe rozpoznanie procesów biznesowych i organizacyjnych w obrębie podmiotów, które będą przyszłymi użytkownikami systemu, lecz abstrahując od szczegółów implementacji. Wynikiem taj fazy jest logiczny model systemu, opisujący sposób realizacji przez system postawionych wymagań.

Podstawowe rezultaty fazy analizy:

- Poprawiony dokument opisujący wymagania;
- Słownik danych zawierający specyfikację modelu;
- Dokumenty opisujące stworzony model;
- Harmonogram fazy projektowania;
- Wstępne przypisanie zespołów do zadań**;**

Scharakteryzuj fazę projektowania procesu wytwarzania oprogramowania

Produktem fazy analizy jest ogólny model logiczny systemu. Tworzenie oprogramowania na bazie tego modelu nie jest jednak zalecane. Należy dokonać podziału ogólnego zbioru wymagań na

mniejsze, precyzyjniej sformułowane wymagania. Realizowane jest to poprzez dekompozycję modelu logicznego systemu powstałego w fazie analizy na mniejsze komponenty. W procesie dekompozycji należy uwzględnić specyfikę środowiska implementacji. Z powyższego stwierdzenia wynika, że w początkowej fazie projektowania należy dokonać wyboru środowiska implementacji. Projekt ma udzielić odpowiedzi na pytanie jak zaimplementować system.

Produkty fazy projektowania:

- W przypadku technik strukturalnych:
  - diagramy związków encji
  - diagramy przepływów danych
  - diagramy przejść stanów (ewentualnie wybrane funkcje w zależności od wielkości projektu)
  - diagramy przypadków użycia
  - zestawienia zawierające: definicje encji, definicje atrybutów, definicje funkcji (procesów), definicje plików danych (zbiorników danych), definicje danych złożonych i elementarnych, definicje modułów
- W przypadku technik obiektowych: diagramy klas, diagramy interakcji obiektów, diagramy stanów, inne diagramy (np. modułów, konfiguracji)
- Zestawienia zawierające: definicje klas, definicje atrybutów, definicje danych, definicje metod, zasoby interfejsu użytkownika (np. menu, okien dialogowych), projekt bazy danych, projekt fizycznej struktury systemu, harmonogram fazy implementacyjnej.

Scharakteryzuj fazę implementacji procesu wytwarzania oprogramowania

Zasadniczym celem fazy implementacji jest opracowanie kodu modułów tworzących oprogramowanie systemu informatycznego. Faza implementacji uległa w ostatnich latach znaczącej automatyzacji wynikającej ze stosowania:

- Języków wysokiego poziomu
- Gotowych elementów
- Narzędzi szybkiego wytwarzania aplikacji - RAD (*Rapid Application Development*)
- Generatorów kodu – to składowe narzędzi CASE, które na podstawie opisu projektu automatycznie tworzą kod programu (lub jego szkielet).

Podstawowym problemem tej fazy są błędy popełniane przez programistów. Pełne ich uniknięcie nie jest możliwe. Jednak pewne podejścia pozwalają w znacznym stopniu ich minimalizacje. Podejścia takie to: unikanie niebezpiecznych technik; unikanie instrukcja goto; przemyślanie korzystanie z wskaźników; stosowanie obliczeń równoległych; stosowanie zasady ograniczonego dostępu; stosowanie kompilatorów sprawdzających zgodność typów;

Kluczowe **czynniki sukcesu** tej fazy to:

- Wysoka jakość i wystarczająca szczegółowość projektu;
- Dobra znajomość środowiska implementacji
- Zachowanie przyjętych standardów;
- Unikanie błędów;

Scharakteryzuj fazę pielęgnacji procesu wytwarzania oprogramowania

Zwykle jest to najdłuższa faza cyklu życia. W tej fazie system jest użytkowany. **Pielęgnacja polega na usuwaniu błędów**, których nie wykryto we wcześniejszych fazach cyklu życia, poprawianiu implementacji modułów i wzbogacaniu systemu o nowe usługi w miarę formułowania nowych wymagań przez użytkownika.

Modelowanie grafiki 3D

Scharakteryzuj rodzaje krzywych wykorzystywane w modelowaniu 2D i 3D

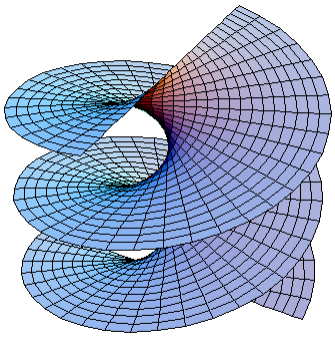
Krzywą zwartą nazywa się continuum o wymiarze 1. Innymi słowy jest to zbiór, w którym każdy jego punkt ma dowolnie małe otoczenia o zero wymiarowym brzegu.

Krzywe Béziera są krzywymi parametrycznymi, tzn. każda współrzędna punktu krzywej jest pewną funkcją liczby rzeczywistej będącej wspomnianym parametrem; aby określić krzywą na płaszczyźnie potrzebne są dwie funkcje, aby określić krzywą w przestrzeni – trzy, itd. Ze względu na rodzaj tych funkcji mówi się o krzywych wielomianowych oraz krzywych wymiernych.

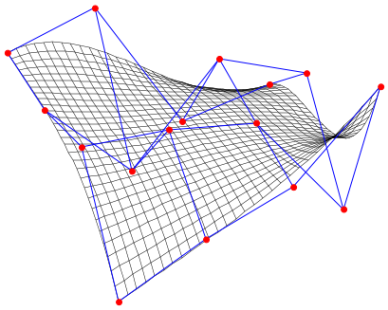
Krzywe B-sklejane, podobnie jak inne krzywe parametryczne używane w grafice komputerowej, są wyznaczane przez ciąg punktów kontrolnych $p_{0,\ldots\ldots,}p_{m-n+1}$. Krzywa taka jest reprezentowana przez $m - 2n$ krzywych wielomianowych stopnia n (mówi się wówczas, że krzywa B-sklejana jest n-tego stopnia), które łączone są z określoną ciągłością parametryczna, zazwyczaj C_n.

Scharakteryzuj rodzaje powierzchni wykorzystywane w modelowaniu 2D i 3D

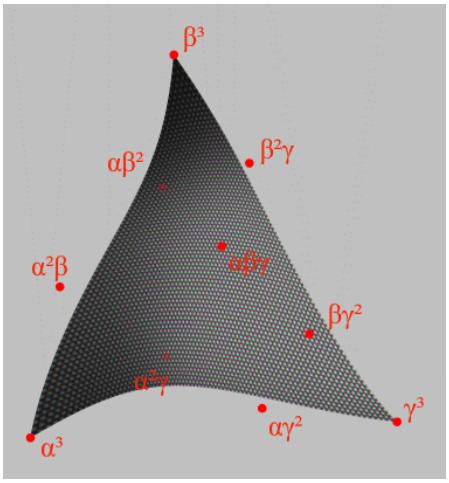
Helikoida to powierzchnia, którą tworzy prosta obracająca się wokół innej prostej ze stałą prędkością kątową i jednocześnie przesuwająca się równolegle do tej prostej ze stałą prędkością liniową.



Powierzchnie Beziera mogą być zdefiniowane w obszarze prostokątnym lub trójkątnym. W pierwszym przypadku określa się płat powierzchni iloczynem tensorowym.



Płaty trójkątne Porcje powierzchni Beziera mogą być określane również jako nie tylko w prostokątach ale również w trójkątach. Za ich pomocą można budować bardziej skomplikowane obszary niż za pomocą prostokątów.



Powierzchnie B-sklejane leżą w powłoce wypukłej swoich punktów. Pierwsza własność jest istotna dla modelowania, gdyż mówi, że zmiana parametrów funkcji Nim wpływa jedynie lokalnie na zmianę krzywej, czyli punkty de Boora odkształcają ją jedynie lokalnie.

Scharakteryzuj wykorzystanie krzywych i powierzchni sklejanych w modelowaniu

Trójwymiarowe skomplikowane modele, zbudowane są zarówno z krzywych jak i powierzchni sklejanych. W modelowaniu powierzchni mają zastosowanie następujące krzywe (linie proste, profile łukowe, krzywe parametryczne (Beziera, B-sklejana). W modelowaniu powierzchniowym wykorzystujemy następujące rodzaje powierzchni:

- powierzchnie płaskie, powierzchnie swobodne, w tym parametryczne oraz kwadryki (powierzchnie drugiego stopnia),
- prymitywy (gotowe powierzchnie wygenerowane przez program.

Większość metod tworzenia powierzchni opiera się na istniejących już w modelu krzywych. W przypadku ich braku, nie ma możliwości konstruowania niektórych rodzin powierzchni np. powierzchni powłokowych i powstałych poprzez wyciągnięcie, ponieważ stanowią one „warunek brzegowy” dla tych powierzchni.

Scharakteryzuj modele kolorów wykorzystywane w grafice komputerowej

RGB (Red, Green, Blue)– jeden z modeli przestrzeni barw, opisywanej współrzędnymi RGB. Z połączenia barw RGB w dowolnych kombinacjach ilościowych można otrzymać szeroki zakres barw pochodnych. Model RGB jest modelem teoretycznym a jego odwzorowanie zależy od urządzenia. Najczęściej stosowany jest 24-bitowy zapis kolorów (po 8 bitów na każdą z barw składowych), w którym każda z barw jest zapisana przy pomocy składowych, które przyjmują wartość z zakresu 0-255. W modelu RGB wartość 0 wszystkich składowych daje kolor czarny, natomiast 255 - kolor biały.

CMYK (Cyjan (taki niebieskawy), Magenta (taki purpurowy), Yellow, końcowa litera K może oznaczać albo literę ostatnią słowa Black albo skrót *key colour)* – zestaw czterech podstawowych kolorów farb drukarskich stosowanych powszechnie w druku kolorowym w poligrafii i metodach pokrewnych. Farby w ww. kolorach nie są określone jednoznacznie, toteż odcienie ich kolorów różnią się u różnych producentów, szczególnie w różnych regionach świata. Barwy wynikowe w metodzie CMYK otrzymuje się poprzez łączenie barw podstawowych w proporcjach (dla każdej z nich) od 0% do 100%.

Model **HSV** (ang. Hue Saturation Value) nawiązuje do sposobu, w jakim widzi ludzki narząd wzroku, gdzie wszystkie barwy postrzegane są jako światło pochodzące z oświetlenia. Według tego modelu wszelkie barwy wywodzą się ze światła białego, gdzie część widma zostaje wchłonięta a część odbita

od oświetlanych przedmiotów. Model jest rozpatrywany jako stożek, którego podstawą jest koło barw.

Symbole w nazwie modelu to pierwsze litery nazw angielskich dla składowych opisu barwy:

- H –barwa światła (ang. Hue) wyrażona kątem na kole barw przyjmująca wartości od 0° do 360°.
- S – nasycenie koloru (ang. Saturation) jako promień podstawy
- V – (ang. Value) równoważna nazwie B – moc światła białego (ang.Brightness) jako wysokość stożka.

Przyporządkowanie częstotliwości fal świetlnych na kole barw w modelu HSV jest takie samo jak w modelach HSL, tzn. centrum barwy czerwonej odpowiada kąt 0° lub 360°. Centrum barwy zielonej odpowiada kąt 120°. Centrum barwy niebieskiej odpowiada kąt 240°. Pozostałe barwy pośrednie dla składowej Hue są odpowiednio rozłożone pomiędzy kolorami czerwonym, zielonym i niebieskim.

Omów wykorzystanie tektur w grafice komputerowej

Teksturowanie - technika stosowana w grafice trójwymiarowej, której celem jest przedstawienie szczegółów powierzchni obiektów przestrzennych za pomocą obrazów bitmapowych (**tekstur**) lub funkcji matematycznych (**tekstur proceduralnych**). **Mapowanie tekstury** określa w jaki sposób powiązać piksele (nazywane w tym kontekście tekselami) lub wartości funkcji z powierzchnią obiektu. Tekstury niosą informacje o barwie powierzchni, jak również innych parametrach generowanego obrazu, związanych np. z modelem oświetlenia: barwa światła odbitego, rozproszonego, stopień przezroczystości, współczynnik załamania światła itp.

Tekstury bitmapowe to na ogół zdjęcia powierzchni rzeczywistych przedmiotów (ścian, tkanin, kory drzew, desek, itp.); **tekstury proceduralne** to parametryzowane wzory generowane programowo, np. szachownica, marmur, drewno, granit, chmury. Tekstury mogą być jedno, dwu i trójwymiarowe.

Teksturowanie jest alternatywą dla modelowania geometrycznego, bowiem przedstawienie wszystkich detali za pomocą geometrii jest trudne, niepraktyczne i w większości przypadków niemożliwe. Specjalnym przypadkiem teksturowania jest bumpmapping, gdzie tekstura wpływa na sposób obliczania natężenia światła, symulując niewielkie nierówności powierzchni.

Omów zasady występujące w obliczeniach kolorów

W grafice komputerowej należy pozyskać obraz, który będzie postrzegany przez receptory mózgu jako obraz naturalny. Do tego są potrzebne modele barw, modele i algorytmy wykonywania obliczeń barw, urządzenia do reprodukcji obrazów barwnych.

Modele kolorystyczne:

- Addytywne (addytywnie miesza się barwy), np. RGB
- Substraktywne (zasada substraktywnego mieszania barw), np. CMY
- Percepcyjne (wykorzystuje się zasady percepcji barw)

Obliczenia w przestrzeniach barw:

- Schemat 1 – polega na bezpośrednim wyznaczaniu wartości parametrów barwy piksela za pomocą równań wynikających z przyjętego modelu oświetlania danej sceny, istotne są wtedy obliczenia wynikające z dodawania wektorów reprezentujących barwne światła oraz z mnożenia wektorów przez skalar;
- Schemat 2 – polega na wyznaczaniu barwy piksela na podstawie barwy innych pikseli. Ten schemat stosuje się wtedy gdy wykonujemy obliczenia związane z eliminacją zakłóceń związanych przy wypełnianiu wnętrza wielokąta o znanych barwach wierzchołka.

Rodzaj obliczeń zależy także od przyjętego modelu barw. Dla modeli liniowych obliczenia są prostsze niż w przypadku modeli nieliniowych. Przy wykonywaniu obliczeń należy je przeprowadzać

niezależnie dla każdego parametru określającego barwę, np. dla modelu RGB należy wykonać obliczenia dla każdej z barw.

Interpolacja liniowa umożliwia określenie ciągu barw leżących na prostej łączącej dwie barwy początkowe.

Inny rodzaj obliczeń występuje przy wyznaczaniu barwy piksela pokrywanego w całości lub częściowo przez dwa obiekty o różnych barwach C1, C2. Takie sytuacje występują wtedy gdy składamy dwa obrazy w jeden, wygładzamy krawędzie wspólne dla dwóch obszarów o różnych barwach. Szukana barwa jest wtedy zależna nie tylko od barw ale i sposobu pokrycia. Z reguły korzysta się wtedy z koncepcji parametru alfa. Jest on związany z każdym obszarem wpływającym na barwę danego piksela. Jeżeli dany obszar o barwie C pokrywa piksel w części, to jego udział wynosi αC, gdzie parametr alfa przyjmuje wartości od 0 do 1. W przypadku gdy piksel jest pokryty przez dwa obszary, to wyznacza się dwie wielkości, np. αA, αB.

Scharakteryzuj modele oświetlenia wykorzystywane w modelowaniu 3D

Przez **model oświetlenia** rozumiemy sposób wyznaczenia obserwowanej barwy obiektu przy znanych właściwościach optycznych obiektu oraz znanych warunkach oświetlenia.

Istotne znaczenie w modelu oświetlenia odgrywa dobór źródeł światła uwzględnianych przy wyznaczaniu obserwowanej barwy obiektu. Model oświetlenia składa się z pierwotnego modelu oświetlenia (bierze pod uwagę jedynie oświetlenie bezpośrednie, tzn. pochodzące ze źródeł emitujących światło) oraz wtórnego modelu oświetlenia (bierze pod uwagę obiekty załamujące lub odbijające światło). Często model oświetlenia uwzględniający pierwotny i wtórny model oświetlenia nazywany jest globalnym modelem oświetlenia. Wynika stąd, że na warunki oświetlenia ma wpływ nie tylko rozmieszczenie źródeł emitujących światło, ale także rozmieszczenie i własności optyczne innych obiektów sceny.

Modele empiryczne Światło otoczenia – bezkierunkowe światło padające na scenę nie oświetloną żadnym kierunkowym źródłem oświetlenia, a obiekty znajdujące się wewnątrz są oświetlone ze wszystkich kierunków jednakowo.

- Model Lamberta – podstawowy model oświetlenia, który wchodzi w skład wielu innych, opiera się na odbiciu opisanym przez prawo cosinusów.
- Model Phonga opiera się na dwóch założeniach: maksimum odbicia zwierciadlanego występuje dla α równego zero i szybko spada ze wzrostem kata α.
- Model Blinna-Phonga jest alternatywa dla modelu Phonga, jest istotny nie ze względu na lepszą symulację odbicia światła od powierzchni (używa tej samej empirycznej funkcji),ale dlatego, że unika obliczania wektora odbicia zwierciadlanego Θs, dzięki czemu jest on nieco szybszy

Modele bazujące na fizyce.

- Model mikrościanek- jest fizycznym modelem powierzchni odbijającej. Zakłada się w nim, że powierzchnia jest izotropowym zbiorem płaskich mikroskopijnych ścianek, z których każda idealnie odbija światło. Geometria, rozkład mikrościanek oraz kierunek padania światła określają natężenie i kierunek odbicia zwierciadlanego.
- Model Cooka-Torrance'a. Funkcja odbicia została tu podzielona na dwie części. Pierwsza jest odpowiedzialna za odbicie rozproszone światła, a druga za odbicie zwierciadlane.
- Model HTSG – najbardziej kompletny i eksperymentalnie weryfikowalny model oświetlenia, bierze pod uwagę niemal wszystkie zjawiska towarzyszące odbiciu światła, wliczając w to zjawiska falowe takie jak: polaryzacja, dyfrakcja i interferencja. Model HTSG bazuje na optyce fali i opisuje odbicie rozproszone, rozproszone kierunkowe i zwierciadlane. Odbite światło zależy od długości fali, kąta padania, dwóch parametrów opisujących szorstkość powierzchni i indeksu refrakcji.

Modele hybrydowe.

- Model Straussa jest modelem przybliżonym, który został zaprojektowany tak, aby dać animatorom i twórcom trójwymiarowych obrazów intuicyjny zbiór parametrów, za pomocą których można kontrolować wygląd powierzchni.
- Model Warda jest prosty, atrakcyjny w implementacji, fizycznie poprawny, a zarazem odpowiedni dla większości rodzajów powierzchni, opiera się na funkcji Gaussa.
- Model Ashikhmina-Shirleya spełnia zasady wzajemności i zachowania energii, pozwala na modelowanie odbicia od powierzchni anizotropowych, postać funkcji odbicia jest określona przez intuicyjne parametry, zawiera współczynnik Fresnela, dzięki czemu odbijalność zwierciadlana powierzchni wzrasta wraz ze wzrostem kąta padania, posiada zmienny współczynnik odbicia rozproszonego, co daje zmniejszenie odbijalności rozproszonej wraz ze wzrostem kąta padania.
- Model Lafortune'a opiera się na użyciu klasy prostych funkcji z nieliniowymi parametrami do reprezentacji odbicia. Tak zdefiniowane funkcje spełniają własności fizyczne określone przez zasady wzajemności i zachowania energii.

Omów metodę „ray-tracingu"

Ray-Tracing jest techniką modelowania scen w grafice trójwymiarowej poprzez symulację drogi promieni świetlnych emitowanych od źródła światła lub jasnych (oświetlonych) obiektów do innych obiektów w scenie. Ray-Tracing w dosłownym tłumaczeniu oznacza *„śledzenie promieni"*. Technikę Ray-Tracingu można inaczej określić jako „programowe obliczanie rozkładu cieni i refleksów światła z różnych źródeł". Metoda ta została zaimplementowana w środowisku Pov-Ray.

Porównaj metody Gourandta i Phonga

Cieniowanie Gourauda polega na przypisywaniu punktom cieniowanego wielokąta jasności obliczonej poprzez interpolację wartości odpowiednich dla każdego wierzchołka.

Cieniowanie Phonga to technika cieniowania wielokątów, w której interpolowany jest wektor normalny do powierzchni. Dla każdego przetwarzanego piksela jest wyznaczany wektor normalny, a następnie stosuje się jakiś model oświetlenia w celu określenia koloru piksela.

Porównując cieniowanie Gouraud i Phonga można wskazać właściwości każdego z nich.

- Obie metody zapewniają ciągłą zmianę barwy eliminując skokowe zmiany cieniowania płaskiego.
- Cieniowanie Gouraud nie daje możliwości powstania lokalnego ekstremum (np. rozbłysku światła) w ramach jednego elementu płaskiego. Powoduje to uśrednienie jasności na powierzchni obiektu. Cieniowanie Phonga nie ma tej wady.
- Interpolacja wektora oddaje poprawnie lokalne ekstrema również w ramach pojedynczego elementu płaskiego.
- Wadą cieniowania Phonga jest fakt, że jest ono ponad dwukrotnie droższe obliczeniowo od cieniowania Gouraud.
- Obie metody są wrażliwe na orientację cieniowanego obiektu. Jeśli obiekt ma różne jasności w wierzchołkach i zostanie obrócony, to interpolacja da różne efekty w zależności od położenia obiektu.

Wykorzystanie apliakcji „Pov-Ray" do modelowania scen

Zastosowanie Pov-Ray:

- Używa metody Ray-tracingu
- Łatwy język opisu sceny
- Dużo przykładowych scen
- Zawiera wiele plików z predefiniowanymi krzywiznami, kolorami, teksturami

- Końcowe rysunki mogą być bardzo dobrej jakości
- Wiele różnych typów światła (liniowe, punktowe, powierzchniowe), perspektyw, tekstur
- Wiele różnych efektów atmosferycznych
- Różne modele rzeczywistych obiektów, jak ogień, chmury
- Możliwość generowania rysunku w różnych formatach
- Podstawowe obiekty: kule, prostopadłościany, stożki, powierzchnie
- Zaawansowane obiekty, tekst, krzywe
- Dodawanie animacji

Systemy wbudowane

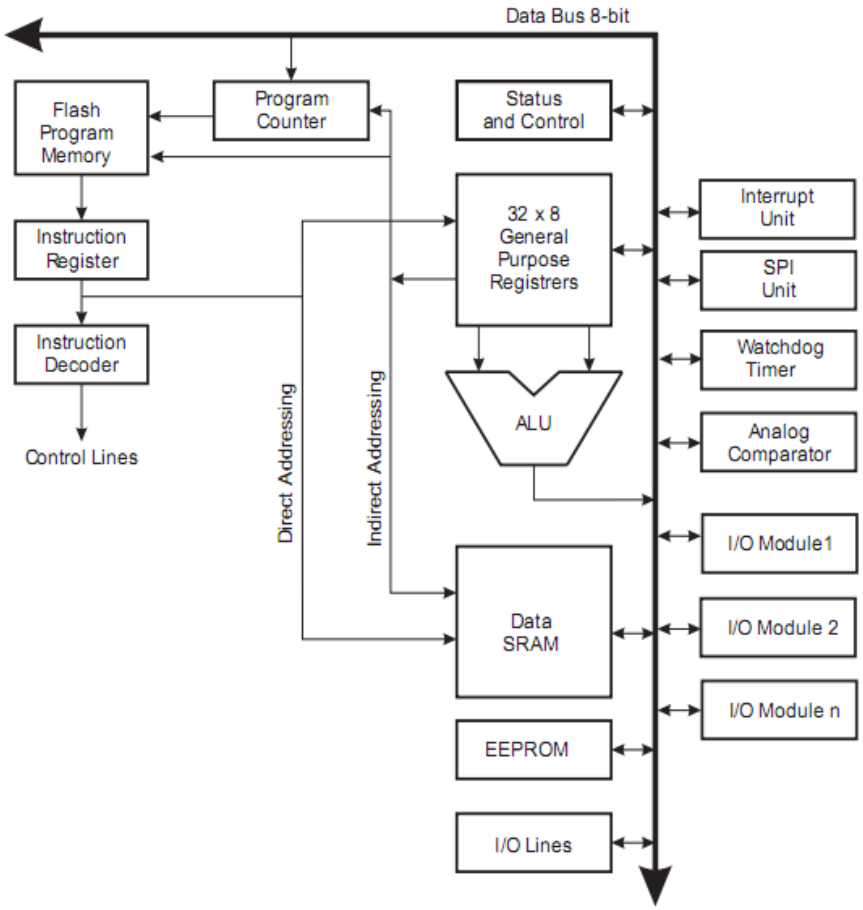
Zdefiniuj pojęcie "System wbudowany". Opisz uogólniony funkcjonalny schemat systemu

System wbudowany (ang. Embedded system) - system komputerowy (w przypadku tych pytań chodzi głównie o mikrokontrolery) specjalnego przeznaczenia, który staje się integralną częścią obsługiwanego przez niego sprzętu (jest weń wbudowany – stąd pochodzi nazwa)

Centralnym elementem systemu wbudowanego jest mikrokontroler do którego podłączony jest szereg peryferii. Peryferia mają na celu zapewnienie komunikacji z zewnętrznymi komputerami (np. konwerter napięć do standardu RS232, kontroler USB, kontroler sieci Ethernet), możliwości odczytu pomiarów z czujników (np. temperatury, obrotów), prezentacji danych (np. diody, wyświetlacz LCD), wysterowania elementów wykonawczych (np. silników, przekaźników wysokiego napięcia).

Należy zaznaczyć że większość systemów wbudowanych staje się coraz bardziej zintegrowana i kompaktowa. Dzisiejsze mikrokontrolery posiadają zintegrowaną pamięć programową (np. pamięć Flash) oraz programowalną dla mikrokontrolera nieulotną pamięć EEPROM, pamięć SRAM. Poza tym wspomagają obsługę szeregu interfejsów (np. USART oraz w bardziej zaawansowanych mikrokontrolerach USB, a nawet interfejs Ethernet – mikrokontrolery firmy Atmel z rodziny ARM), magistral SPI lub I2C.

Poniżej schemat blokowy mikrokontrolera. Po prawej stronie szyny danych mamy większość modułów które pośredniczą z wspomnianymi peryferiami systemu wbudowanego.



Scharakteryzuj interfejs RS-485 oraz funkcje modułu USART (Universal Synchronous Asynchronous Receiver Transmitter)

Standard RS485 składa się z różnicowego nadajnika, dwuprzewodowego toru transmisyjnego i różnicowego odbiornika. Dla standardu RS485 nie ma konieczności prowadzenia przewodu powrotnego (ze względu na różnicowy przesył). Standard RS485 umożliwia podłączenie wielu nadajników i odbiorników (maksymalnie do 32). Ograniczenie wynika z ograniczeń energetycznych nadajnika. Najczęściej stosowaną topologią dla takich standardów jest topologia magistrali.

Zasięg tego standardu to około 1200m. Prędkości transmisji jakie można uzyskać to 35Mbit/s (do 10m), i 100Kbit (do 1200m). RS485 jest najczęściej stosowanym interfejsem przewodowym w sieciach przemysłowych - z jednego prostego powodu, przesył różnicowy zapobiega wpływowi zakłóceń zewnętrznych (np. sprzętu indukcyjnego jak silniki) na transmisję danych. Na bazie tego interfejsu opracowano wiele protokołów komunikacyjnych.

Wiele aplikacji komunikuje się szeregowo jedynie w jednym kierunku jednocześnie , gdzie pojedyncze urządzenie nadrzędne/sterujące (master) wysyła polecenia do jednego z wielu urządzeń podrzędnych (slave) pracujących w sieci. Typowo jedno urządzenie (węzeł) jest adresowane przez komputer główny (host) i odpowiedź zostaje otrzymana od tego urządzenia.

W przypadku RS485 rządzenie nadrzędne (master) nadaje prośbę o rozpoczęcie komunikacji do węzła podrzędnego (slave) poprzez zaadresowanie tej jednostki

Moduł USART, ze względu na swoją uniwersalność, pozwala podłączyć i komunikować się mikrokontrolerowi z różnymi urządzeniami peryferyjnymi (np. komputer, moduł Bluetooth, moduł GPS, przyrządy pomiarowy). Obsługa modułu USART jest najczęściej wspomagana przez

mikrokontroler zatem sprowadza się do odczytu danych z rejestru i sprawdzania bitów (bitów rejestru) potwierdzających odbiór oraz przesył danych z buforu.

Scharakteryzuj interfejs I2C (Inter-Integrated Circuit).

I2C - zeregowa, dwukierunkowa magistrala służąca do przesyłania danych w urządzeniach elektronicznych. Została opracowana przez firmę Philips na początku lat 80. Znana również pod akronimem IIC, którego angielskie rozwinięcie Inter-Intergrated Circuit oznacza "pośredniczący pomiędzy układami scalonymi". Standard I²C określa dwie najniższe warstwy modelu odniesienia OSI: warstwę fizyczną i warstwę łącza danych.

OSI (ang. Open System Interconnection) lub Model OSI – standard zdefiniowany przez ISO (Międzynarodowa Organizacja Normalizacyjna, ang. International Organization for Standardization) oraz ITU-T (Sektor Normalizacji Telekomunikacji ITU, ang.: International Telecommunication Union - Telecommunication Standardization Sector) opisujący strukturę komunikacji sieciowej.

Warstwa fizyczna I²C do transmisji wykorzystuje dwie dwukierunkowe linie: SDA (linia danych, ang. Serial Data Line) i SCL (linia zegara, ang. Serial Clock Line). Obydwie linie są na stałe podciągnięte do źródła zasilania poprzez rezystory podciągające (ang. pull-up). I²C używa logiki dodatniej, a więc stan niski na magistrali odpowiada "0" logicznemu, natomiast stan wysoki "1" logicznej.

Warstwa łącza danych I²C jest magistralą zorientowaną bajtowo, a więc bity grupowane są po 8. Po przesłaniu 8 bitów w jednym kierunku, przesyłany jest dodatkowy bit potwierdzenia odebrania danych ACK (lub NACK w przypadku braku potwierdzenia) w kierunku przeciwnym.

Pierwszym bajtem jest zawsze nadawany przez urządzenie master adres urządzenia slave, który oprócz 7 bitów właściwego adresu zawiera bit kierunku transmisji na najmłodszej pozycji. Wartość "0" tego bitu oznacza transmisję od mastera do slave'a (zapis), podczas gdy wartość "1" kierunek przeciwny (odczyt). Po pierwszym bajcie przesyłane zostają dane.

Opracowany na początku lat 80. standard zakładał 7-bitową przestrzeń adresową, czyli możliwość zaadresowania do 128 urządzeń. W praktyce część adresów jest zarezerwowana, pozostawiając do dyspozycji 112 wartości. Jednym z zarezerwowanych adresów jest tzw. General call (adres 0), który powoduje wysłanie danych do wszystkich urządzeń podłączonych do magistrali.

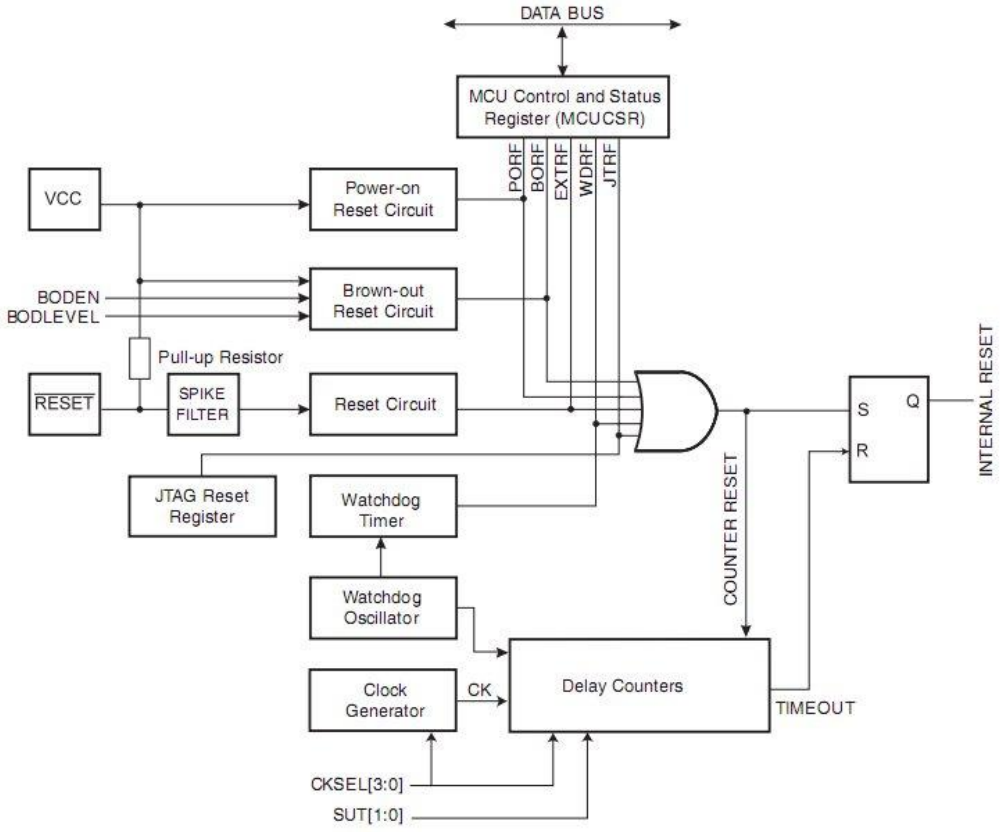
Scharakteryzuj funkcje modułów zerowania mikrokontrolerów

Należy pamiętać, że proces zerowanie nie jest natychmiastowy, a do jego

zakończenia stan wyjść może być nieokreślony. Po rozpoczęciu zerowania następuje ustawienie domyślnych wartości dla rejestrów mikrokontrolera.

Zwykle trwa on dwa cykle maszynowe i jest wykonywany w odpowiedzi na wykrycie stanu aktywnego (=1) na wejściu zerującym RESET, oznaczanym zwykle RST. Stan wejścia RST testowany jest zwykle jeden raz w ciągu każdego cyklu maszynowego. Założenie, że stan aktywny musi trwać minimum dwa cykle maszynowe jest słuszne jedynie przy spełnieniu warunku, że razem z włączeniem zasilania startuje generator taktujący pracę mikrokontrolera. W praktyce bezpieczny czas trwania impulsu zerującego powinien wynosić co najmniej 10-20 ms, a jego przekroczenie poza czas dwóch cykli maszynowych nie pociąga za sobą żadnych konsekwencji, poza powtórzeniem procesu zerowania. Powrót wejścia RESET do stanu niskiego rozpoczyna normalną pracę systemu.

Figure 15. Reset Logic



Scharakteryzuj funkcję modułu ICSP (In-Circuit Serial Programming)

ICSP - (In Circuit Serial Programming) jest metodą bezpośredniego programowania, najczęściej mikrokontrolerów. ICSP jest ulepszoną techniką ISP zaimplementowaną przez Microchip (*ww1.microchip.com/downloads/en/devicedoc/30277d.pdf*

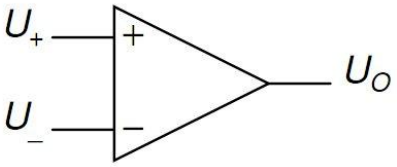
ISP (In System Programming) – rozwiązanie konstrukcyjne stosowane w mikrokontrolerach i układach PLD umożliwiające zaprogramowanie układu bez demontażu z urządzenia w którym pracuje.

Główną zaletą wynikającą ze stosowania ISP jest możliwość połączenia programowania i testowania w jednej fazie produkcyjnej, co sprzyja szybszemu projektowaniu urządzenia i ułatwia modyfikacje prototypów. Technologia ta pozwala także producentowi obejść się bez zakupu już zaprogramowanych mikrokontrolerów czy układów PLD (gdyż proces programowania może odbyć się na linii produkcyjnej urządzenia). Ponadto ISP ułatwia serwisowanie oraz aktualizację oprogramowania.

Układy scalone wyposażone w ISP mają wewnętrzne obwody, generujące napięcia, niezbędne do zaprogramowania wbudowanej pamięci, z typowego napięcia zasilającego, a także interfejs szeregowy, umożliwiający komunikację z programatorem. Do komunikacji większość układów wykorzystuje protokół JTAG, choć są w tym celu wykorzystywane także inne protokoły, np. SPI.

Scharakteryzuj funkcję komparatora analogowego (ang. Analog Comparator)

Komparator służy do porównywania wartości dwóch napięć elektrycznych. Wyjście UO jest w stanie wysokim, gdy U+ > U− a w stanie niskim, gdy U+ < U−.



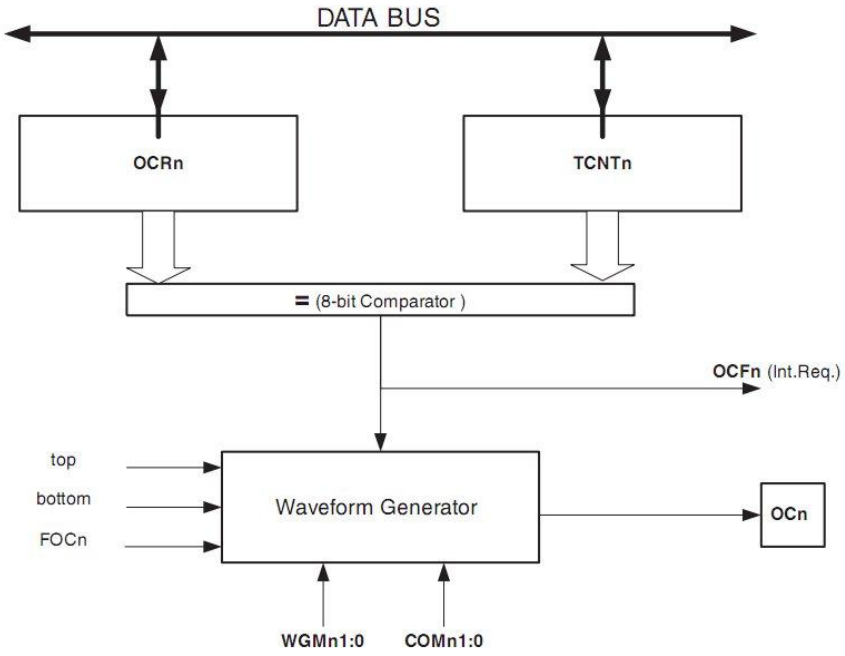
Komparatory są używane do przetworników analogowo-cyfrowych. W mikrokontrolerach najczęściej jest spotykany wbudowany układ przetwornika AC pozwalający na porównywanie napięć względem przyłożonego napięcia odniesienia – wynik zapisywany jest w rejestrach.

Scharakteryzuj funkcję modułu modulacji szerokością impulsu (ang. Pulse Width Modulation)

PWM (ang. Pulse-width modulation) – metoda regulacji sygnału prądowego lub napięciowego, polegająca na zmianie szerokości impulsu o stałej amplitudzie, używana we wzmacniaczach, zasilaczach impulsowych oraz układach sterujących pracą silników elektrycznych. Układ PWM zasila urządzenie bezpośrednio lub przez filtr dolnoprzepustowy, który wygładza przebieg napięcia lub prądu.

Mikrokontrolery najczęściej posiadają moduł PWM który pozwala na generowanie sygnału sparametryzowanego przez szereg rejestrów. Najczęściej jest dostępne kilka rodzajów modulacji – w zależności od producenta mikrokontrolera.

Figure 29. Output Compare Unit, Block Diagram



Scharakteryzuj funkcję bloku sterowania portami mikrokontrolera

Do bloku sterowania portami mikrokontrolera używane są moduły wejścia-wyjścia (I/O Modules). Modułów może być kilka – mogą też być wyposażone w dodatkowe funkcje jak interfejs USART, przetwornik AC, interfejs SPI.

Każdy moduł posiada przypisane mu rejestry odpowiedzialne za ustawienie kierunku wyprowadzenia (wejście/wyjście) oraz odczytu i zapisu stanu wyprowadzenia. W zależności od dodatkowych funkcji moduły posiadają dodatkowe rejestry (np. obsługa zewnętrznych przerwań, moduł modulacji sygnału – PWM – posiada kilka rejestrów parametryzujących modulację). Wyprowadzenia modułów zwykle

wyposażone są w rezystory podciągające zapewniające prąd rzędu 20mA każdy przy ustawieniu wyjścia w stan wysoki lecz nie posiadają zabezpieczeń przeciw przeciążeniowych.

Scharakteryzuj elementy języka asemblera MPASM (Microchip's Universal Assembler)

Liczby - W asemblerze MPASM mogą być przedstawione liczby tylko całkowite. Liczby mogą być zapisane w systemie liczbowej dziesiętnej, dwójkowej, ósemkowej lub szesnastkowej. Szesnastkowy system jest systemem domyślnym. Dyrektywa RADIX może zmienić system domyślny.

Literały - Literały tekstowe mogą mieć rozmiar do 255 znaków. Literał musi być zapisany w cudzysłowie. Do wprowadzania literału tekstowego służy dyrektywa DW (lub dw), przykładowo: DW „Tekst testowy” W przypadku braku końcowego znaku cudzysłowu, literał kończy się z końcem wierszu. Aby przedłużyć literał w następnym wierszu należy powtórzyć dyrektywę DW. Rozmiar literału w kodzie zawsze parzysty. Jeśli ilość znaków jest nieparzysta, to asembler dodaje bajt z wartością 0. Za pomocą dyrektywy "define" można wprowadzić nazwę literału, na przykład: #define tekstkom "Komunikat testowy"

Zmienne - Nazwa (identyfikator) zmiennej, którą zadeklarował programista, nie może być identyczna ze słowem kluczowym. Liczba znaków w identyfikatorze nie może być większa niż 32, a pierwszy znak musi być literą lub znakiem "_".

Wyrażenia - W tekście w języku asemblera MPASM można używać wyrażenia. Wyrażenia jako konstrukcje językowe zawierają liczby, literały, identyfikatory zmiennych i separatory (nawiasy okrągłe). Elementy wyrażenia są związane operatorami arytmetycznymi i logicznymi. Operatory asemblera MPASM i kolejność ich priorytetów są zapożyczone z języka C.

Scharakteryzuj instrukcji asemblera MPASM (Microchip's Universal Assembler)

Instrukcje asemblera MPASM są zależne od serii mikrokontrolerów które je implementują. Można wśród nich wyróżnić kilka charakterystycznych grup instrukcji.

- instrukcje bajtowe, np.
  cpfseq f,a - Porównanie f i WREG oraz ominięcie następnej instrukcji, jeśli równo
  incf f,d,a - Zwiększenie o jeden zawartości rejestru f oraz zapisywanie wyniku
- Instrukcje bitowe, np.
  btfss f,b,a Testowanie bitu b w f oraz ominięcie następnej instrukcji, jeśli b równy 1 btg f,b,a Negacja bitu b w f
- Instrukcje ze stałą, np.
  andlw k Mnożenie logiczne stałej k na WREG
  sublw k Odejmowanie stałej k od WREG
- Przesyłanie tablic, np.
  tblrd* Odczyt tablicy
  tblwt*+ Zapisywanie tablicy i zwiększenie adresu
- Instrukcje sterowania, np.
  bc n Skok jeśli C=1
  retfie s Powrót z przerwania
  call n,s Wywołanie podprogramu dalekie
  bra n Skok bezwarunkowy

Platformy Programowania

Wymienić i przedstawić najważniejsze składniki Platformy J2EE

- **Java Servlet** - komponenty pozwalające na tworzenie aplikacji www tzw. serwletów.
- **JavaServer Pages (JSP)** - język i technologia do tworzenia dynamicznych stron WWW.
- **JavaServer Pages Standard Tag Library (JSTL)** - biblioteka tagów w języku JSP do tworzenia aplikacji WWW.
- **Common Annotations for the Java Platform** - tzw. adnotacje wprowadzone dopiero w JEEv5.
- **Enterprise JavaBeans** - dodatkowe komponenty, które mogą być wykorzystywane przez wszystkie aplikacje.
- **Java API for XML-Based Web Services (JAX-WS)** - komponenty do tworzenia tzw. Web Serwisów opartych na XML.
- **Java Architecture for XML Binding (JAXB)** - komponenty do operowania na dokumentach XML.
- **SOAP with Attachments API for Java (SAAJ)** - komponenty do realizacji komunikacji XML w technologii SOAP.
- **Streaming API for XML** - komponenty do serializacji dokumentów XML.
- **JavaServer Faces (JSF)** - komponenty do tworzenia warstwy interfejsu użytkownika aplikacji WWW.
- **JavaMail API** - komponent implementujący obsługę e-mail.
- **Java Message Service API (JMS)** - komponenty implementujące komunikacje asynchroniczną.
- **Java Persistence API (JPA)** - komponenty do realizacji połączenia z bazami danych wykorzystujące tzw. persystencje oraz mapowanie obiektów baz danych.
- **Java Security** - komponenty implementujące mechanizmy zapewnienia bezpieczeństwa.
- **Java Transaction API (JTA)** - komponent do tworzenia aplikacji realizujących tzw. transakcje biznesowe.

Omówić cykl życia serwletów i dynamicznie tworzonych stron JSP/ASP

Klasa Javy działająca po stronie serwera WWW. **Cykl życia serwletu**:

- Klasa serwletu jest ładowana do pamięci tylko raz, przy starcie kontenera webowego lub przy pierwszym wczytaniu strony.
- Tworzona jest instancja klasy - wywoływana jest metoda init() inicjalizująca obiekt. Inicjalizacja obiektu dokonywana jest tylko raz - przed obsłużeniem pierwszego żądania.
- Po zainicjalizowaniu, instancja klasy serwletu rezyduje w pamięci, oczekując na przydział żądań. W chwili gdy serwer otrzymuje żądanie od klienta, tworzy obiekty (instancje klas Javy) reprezentujące żądanie klienta i odpowiedź serwera, następnie uruchamia nowy wątek, który wywołuje metodę service() przekazującą obiekty żądania i odpowiedzi do odpowiedniej metody zaimplementowanej przez programistę.
- Po zakończeniu pracy serwletu wywoływana jest metoda destroy() zwalniająca zasoby alokowane przez uruchomiony serwet.

Strony JSP są wykonywane po stronie serwera. Interpretacja i wykonywanie kodu jest kontrolowane przez specjalny proces zarządzający.

- serwer odbiera żądanie pobrania strony JSP,
- proces zarządzający sprawdza, czy kod strony nie był odnawiany. Jeśli były w nim zmiany, strona jest tłumaczona do postaci źródłowej, tzn. na podstawie kodu JSP jest tworzony kod serwletu,
- kod serwletu jest kompilowany, zapisywany do pliku o rozszerzeniu class i uruchamiany,
- proces zarządzający tworzy instancję klasy serwletu, inicjuje ją poprzez wywołanie metody init i tworzy obiekty request i response,

- proces zarządzający wywołuje metodę jsp_service aby obsłużyć żądanie użytkownika, jeśli serwer lub administrator stwierdzi, że strona JSP nie jest już potrzebna,
- proces zarządzający wywołuje metodę destroy, która niszczy instancję klasy serwletu.

Przedstawić najważniejsze rozwiązania technologiczne zastosowane w platformie .Net w porównaniu z modelem WIN32

Przenośność kodu pomiędzy różnymi platformami za pomocą kodu pośredniego **CIL** (Common Intermediate Language), który nie jest kodem maszynowym przeznaczonym do konkretnego procesora, lecz specjalnym kodem pośrednim. **CLR** (Common Language Runtime) odpowiadającego za przekształcenie CIL na kod maszynowy i uruchomienie aplikacji .NET.

Zwiększa bezpieczeństwo aplikacji przez nadzorowanie kodu. Zanim zostanie przetłumaczony na kod maszynowy można sprawdzić czy nie wykonuje niedozwolonych operacji lub czy kod został dopuszczony do wykonania za pomocą zasad systemowych.

CLI jest nazywany **kodem zarządzanym**. Aplikacje .NET z reguły nie zwalniają same pamięci, lecz zajmuje się tym garbage collector, który śledzi obiekty w pamięci i zwalnia już niepotrzebne zasoby. Ułatwia to pracę programistom i zapobiega powstawaniu błędów, ale pozwala też tworzyć bardziej stabilne i niezawodne aplikacje.

Niezależność języka programowania, głównym językiem zaprojektowanym specjalnie pod kątem platformy .NET, jest C#, ale równie dobrze aplikacje można pisać w językach Visual Basic .NET, Delphi .NET, C++ .NET, J# .NET oraz w co najmniej dwudziestu innych. Co więcej, poszczególne elementy aplikacji można napisać w różnych językach, a wszystkie zostaną skompilowane do postaci tego samego kodu pośredniego CIL. Taka współpraca jest możliwa między innymi dlatego, że wszystkie języki zgodne z platformą .NET stosują ujednolicony system typów **CTS** (Common Type System), który gwarantuje, że np. podstawowy typ całkowity będzie zawsze reprezentowany za pomocą 32 bitów.

CIL jest kodem obiektowym. Umożliwia to kompilacje poszczególnych metod, zamiast całego programu. A dzięki przechowywaniu funkcji w pamięci, ich ponowne wywołania nie powodują powtórnej kompilacji.

Obiektowa biblioteka **FCL** (Framework Class Library) jest bezpiecznym interfejsem do funkcji Windows API. Aplikacje .NET mogą z nich korzystać za pośrednictwem uporządkowanej hierarchicznej struktury klas oraz za pomocą przestrzeni nazw, które grupują obiekty powiązane funkcjonalnie.

Przedstawić podstawowe konstrukcje języka C# wskazując na elementy odróżniające go od języków Java i C++

C# a C++

- Brak wskaźników w C# (tzn. są ale niejawne)
- foreach
- C# dziedziczenie tylko po jednej klasie bazowej
- Ręczne zwalnianie zasobów w c++ i Garbage Collector w C#

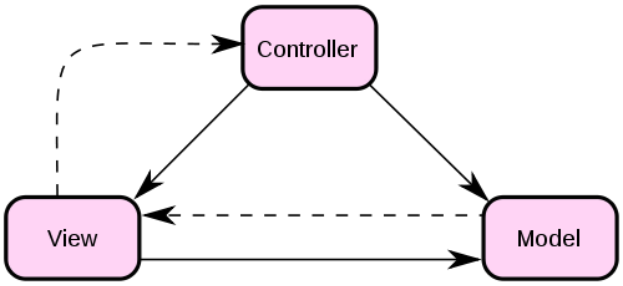
JAVA a C#

- przekazywanie parametrów do metod w C# (out, ref) (w Javie tylko typy proste są przekazywane przez wartość, reszta jest przekazywana przez referencję).
- W C# dostępne są struktury
- właściwości dla pól zawartych w class (niby jak gettery ale trochę sprytniejsze)

```
public int Age {
 get { return age - 10; }
 set { age = value; }}
```

Omówić paradygmat tworzenia trójwarstwowych aplikacji webowych MVC

MVC (Model – View – Controller) – wzorzec projektowy do organizowania struktury aplikacji posiadających graficzny interfejs użytkownika. **Model** – reprezentacja problemu bądź logiki aplikacji. Model jest aktywną reprezentacją pewnego fragmentu komputerowej reprezentacji problemu. Model pasywny nie zmienia swojego stanu bez interakcji użytkownika. Model aktywny może zmieniać swój stan samoczynnie. **View –** opisuje, jak wyświetlić pewną część modelu w ramach interfejsu użytkownika. Może składać się z podwidoków odpowiedzialnych za mniejsze części interfejsu. **Controller –** Zadaniem kontrolera jest odbiór, przetworzenie oraz analiza danych wejściowych od użytkownika. W typowej aplikacji źródłami danych wejściowych będą klawiatura i mysz. Po przetworzeniu odebranych danych kontroler może wykonać następujące czynności: zmienić stan modelu, odświeżyć widok, przełączyć sterowanie na inny kontroler. Każdy kontroler posiada bezpośrednie wskazania na określone modele i widoki, z którymi współpracuje, a jednocześnie w aplikacji może istnieć wiele kontrolerów. W danym momencie tylko jeden z nich steruje aplikacją.



Linie przerywane reprezentują połączenia niebezpośrednie (np. z wykorzystaniem wzorca obserwator). Linie nieprzerywane są to połączenia bezpośrednie

Zalety: Brak zależności modelu od widoków. Łatwiejsza rozbudowa. **Wady:** Złożoność - implementacje MVC wprowadzają dodatkową warstwę abstrakcji oraz dodatkowe sposoby interakcji. Kosztowne zmiany modelu. Trudne testowanie widoków.

Omówić najpopularniejsze sposoby dostępu do danych JDBC EJB, oraz DAO wskazując na różnice w ich zastosowaniu

JDBC (ang. Java DataBase Connectivit) – Interface umożliwiający niezależnym od platformy porozumiewanie się z bazami danych za pomocą języka SQL. Główna klasa wykorzystywana w celu obsługi dostępu do danych jest klasa jdbcTemplate. Zarządza ona komunikacja z baza danych oraz obsługuje wyjątki. Klasa JdbcTemplate udostępnia kilka metod, za pomocą których można wykonywać proste wyrażenia języka SQL. Najważniejsza z nich jest metoda execute().

EJB umożliwia oddzielenie implementacji trwałości danych od logiki aplikacji. Przy dostępie do danych operujemy jedynie na obiektach zamiast na tabelach i polach bazy danych. Komponenty EJB działają po stronie serwera, który jest nazywany kontenerem EJB. Kontener zapewnia automatycznie trwałość obiektów oraz tworzenie i zamykanie sesji. Do formułowania zapytań do baz danych w EJB jest używany obiektowy język zapytań EJB QL. Zapytania stanowią zawsze argumenty różnych wariantów metody find(), która jest wywoływana w komponencie EJB. Podczas implementowania komponentów EJB używamy mechanizmu adnotacji JPA.

DAO jest to komponent dostarczający jednolity interfejs do komunikacji między aplikacją, a źródłem danych. Stosowany do oddzielenia dostępu do danych od logiki biznesowej i warstwy prezentacji.

Przedstawić sposób użycia mechanizmu adnotacji w aplikacjach tworzonych na platformie J2EE

Inną technologią umożliwiającą zrealizowanie dostępu do baz danych są persystencje ze standardu EJB3 (ang. Enterprise Java Beans). EJB jest częścią platformy J2EE. Podobnie jak Hibernate, umożliwia ona oddzielenie implementacji trwałości danych od logiki aplikacji. Przy dostępie do danych operujemy jedynie na obiektach zamiast na tabelach i polach bazy danych.

Podczas implementowania komponentów EJB używamy mechanizmu adnotacji. Adnotacje są dodatkowymi elementami kodu, które umożliwiają zapamiętywanie danych i dodawanie dodatkowych informacji dla kompilatora. Mogą spełniać one także podobną rolę jak fragmenty konfiguracji w plikach XML. W przypadku implementowania obiektów trwałych (presystencji) służą one do wiązania tabel, kolumn i innych elementów relacyjnej bazy danych z obiektami Java. Adnotacje wykorzystywane są do następujących celów:

- definiowania klas, które mogą być usługami webowymi (Web Services);
- definiowania komponentów EJB;
- opisu sposobu odwzorowania klas Java na tabele w bazie danych ;
- wstrzykiwania referencji do wykorzystywanych komponentów oraz różnych zasobów;
- konfiguracji ustawień bezpieczeństwa aplikacji.

Kod źródłowy sesyjnego komponentu EJB 3.0 składa się z dwóch elementów:

- Klasy komponentu, która implementuje interfejs biznesowy komponentu i metody cyklu życia.
- Interfejsów biznesowych, które deklarują metody udostępniane przez komponent klientom zdalnym i lokalnym. Są one tworzone w formie zwykłych interfejsów java, w których są wykorzystywane **adnotacje**.

Najczęściej używanymi adnotacjami w encjach są:

- @Entity – definiuje klasę jako encję;
- @Table – odwzorowuje encję na tabelę bazy danych;
- @Id – definiuje klucz dla encji; każda encja powinna mieć klucz główny.
- @GeneratedValue – nakazuje wygenerowanie wartości danej kolumny;
- @Column – odwzorowuje obiekt na kolumnę bazy danych;
- @ManyToOne, @ManyToMany, @OneToMany itd. odwzorowuje relacje pomiędzy tabelami bazy danych na relacje pomiędzy obiektami;
- @JoinColumn - definiuje powiązanie kolumn z tabel, które zostały oznaczone jedną z adnotacji określających relacje.

Przedstawić wybraną technologie realizującą paradygmat MVC

Struts z wykładu: Technologia pozwalającą na implementowanie aplikacji zgodnych z wzorcem MVC jest Struts. Aplikacja Struts składa się z widoków, które są implementowane jako strony JSP, oraz modeli. Modele te są programowane jako serwlety. Ich działanie polega na wywoływaniu odpowiednich metod, w zależności od odebranych zadań użytkownika. Metody te są nazywane akcjami.

Oprócz implementacji widoków i modeli, aplikacja zawiera także plik konfiguracyjny w formacie XML. W tym pliku zapisuje się m.in. mapowania ścieżek odczytywanych z adresów żądań na odpowiednie akcje.

Działanie serwletów Struts polega na wywoływaniu metod, które są nazywane akcjami. Dlatego serwlety Struts muszą być klasami dziedziczącymi po klasie Action. Każda klasa serwletu Struts powinna posiadać przynajmniej jedna metodę execute. Klasami potrzebnymi do implementacji aplikacji Struts są także klasy: ActionForward, ActionMapping, oraz ActionForm.

Do tworzenia widoków, czyli stron JSP wykorzystuje się dodatkowo dwie biblioteki tagów, które są dostarczane przez tę technologię. Zawierają one tagi ułatwiające tworzenie elementów HTML takich jak formularze, przyciski, pola edycyjne i inne kontrolki.

Opisane inaczej: (by Piotrek) Podstawowym pojęciem w struts jest akcja. Akcją jest to wszystko, co powoduje wysłanie requestu do serwera - np. kliknięcie linka, zapisanie formularza, pobranie zasobu itp. Framework nie różnicuje akcji będących obsługą formularza czy obsługi normalnych linków - obie

sytuacje to standardowe akcje. Z punktu widzenia implementacyjnego akcja jest metodą, która spełnia warunki:

- jest publiczną metodą instancyjną w odpowiedniej klasie
- zwraca String
- nie przyjmuje parametrów
- jest oznaczona adnotacją @Action

Nazwa metody nie musi być zgodna z nazwą akcji. Właściwie nazwa metody nie jest w żaden sposób wykorzystywana przez framework - może być zupełnie dowolna.

Rezultaty - Struts definiuje 3 podstawowe typy rezultatów: SUCCESS, INPUT, ERROR. Nazwy wydają się tłumaczyć zastosowanie. W adnotacji @Action znajduje się parametr results, który jest tablicą adnotacji @Result.

Linki - Zakładając, że aplikacja jest wdrożona pod adresem: http://localhost:8080/app/ Wtedy przejście do adresu: http://localhost:8080/app/actionName.action spowoduje wykonanie metody oznaczonej tą akcją). Wynik wykonania zależy od tego, co zwróci metoda i jak to się będzie miało do konfiguracji @Result.

Klasy - Właściwie jedynymi wymaganiami dla klasy są: - publiczna klasa, nie abstrakcyjna i oznaczona odpowiednimi adnotacjami. Jedna klasa akcji może zawierać dowolną ilość akcji.

Walidatory - Kolejna z rzeczy, która stanowi o sile tego frameworka. Tutaj, nałożenie walidacji na parametr to kwestia dwóch adnotacji. Co dostajemy w zamian? Po pierwsze oczywiście pewność, że metoda nie wykona się bez spełnienia warunków. Po drugie - kompleksową dekorację widoku. Każde pole, które jest oznaczone taką walidacją zostaje wyróżnione poprzez pokolorowanie labela oraz wyświetlenie nad polem informacji pobranej z zasobów i18n.

Wymienić i krótko scharakteryzować najbardziej znane technologie do tworzenia szablonów w warstwie interfejsu użytkownika w modelu MVC

Struts – otwarty framework dla aplikacji tworzonych w języku programowania Java. Podstawowymi zastosowaniami Struts są prezentacja danych oraz kontrolowanie danych (widok i kontroler w modelu MVC). Struts pozwala w łatwy sposób mapować adresy stron www aplikacji na metody klas obsługujących żądania. Sercem Struts jest serwet wywołujący odpowiednie metody w zależności od jego konfiguracji zapisanej w pliku XML.

Spring jest szkieletem tworzenia aplikacji (ang. application framework) w języku Java dla platformy Java EE/J2EE (jakkolwiek istnieje też wersja dla środowiska .NET). Spring powstał jako alternatywa dla programowania aplikacji z użyciem Enterprise JavaBeans. Spring może być rozważany jako zbiór pomniejszych szablonów lub "szablon zbudowany z szablonów". Większość z tych szablonów została zaprojektowana aby pracować niezależnie, jakkolwiek użycie ich razem zapewnia większą funkcjonalność. Szablony te można podzielić ze względu na podstawowe komponenty: Kontener IoC, Szablon programowania aspektowego, Szablon dostępu do danych, Szablon obsługi transakcji, Szablon Model – Widok – Kontroler, Szablon zdalnego dostępu, Szablon autoryzacji i uwierzytelniania, Szablon zdalnego zarządzania JMX, Szablon komunikatów JMS, Szablon obsługi testowania.

Swing jest to podstawowa biblioteka graficzna Javy. Powstał jako następstwo AWT zapewnia wygląd I zachowanie okien zależnie od platformy na której program jest uruchamiany.

Ruby on Rails (często nazywany RoR lub po prostu Rails) – framework open source do szybkiego tworzenia aplikacji webowych. RoR został napisany w języku Ruby. Na framework Rails składają się cztery główne elementy:

- ActiveRecord – mechanizm ORM (Object-Relational mapping) dla Ruby, odpowiada za tworzenie modeli w architekturze MVC

- ActionPack – biblioteka zawierająca klasy ActionController i ActionView, które odpowiadają za tworzenie odpowiednio kontrolerów i widoków
- ActiveSupport – zbiór użytecznych dodatków do standardowej biblioteki Ruby, zawiera m.in. rozszerzenia klas String czy Time
- ActionMailer – biblioteka służąca do wysyłania wiadomości email (najczęściej przez aplikację Rails)

Omówić mechanizmy odwrócenia kontroli i wstrzykiwania zależności oraz ich zastosowanie w aplikacjach tworzonych przy użyciu Spring Framework

Kontener IoC (główny silnik w Spring Framework) działa zgodnie z mechanizmem odwrócenia kontroli. W standardowym podejściu aplikacja w swoim kodzie sama musi utworzyć instancje obiektów potrzebnych do jej działania. **Odwrócenie kontroli** polega na tym, że obiekty są dostarczane dla aplikacji z zewnątrz. Framework wywołuje metody w klasach aplikacji oraz dostarcza jej potrzebnych obiektów. Kod aplikacji jest wywoływany przez sam framework, który koordynuje przepływ pracy, jednak aby to było możliwe, kod aplikacji musi być dostosowany, tzn. udostępniać swoje zależności (metody, pola). Ponadto w konfiguracji frameworka muszą być wskazania do udostępnianych zależności aplikacji. Jest to rozwiązanie bardzo wygodne.

Wstrzykiwanie zależności (Mechanizm wstrzykiwania zależności jest jedną z form odwrócenia kontroli) opiera się na standardowych konstrukcja językowych Javy, a nie na interfejsach konkretnych środowisk. Klasy aplikacji ujawniają swoje zależności za pośrednictwem metod (getterów i setterów) lub konstruktorów, które są wywoływane przez kontener IoC. Dzięki temu kontener może przekazać wartości do obiektów aplikacji. Wartości te są określane w konfiguracji, która zwykle jest tworzona w formie XML lub plików właściwości.

Sposoby wstrzykiwania zależności:

- Wstrzykiwanie oparte na metodach ustawiających (ang. setter injection) za pomocą setterów.
- Wstrzykiwanie oparte na konstruktorach (ang. constructor injection). Zależności są udostępniane jako argumenty dodatkowych konstruktorów klasy.
- Wstrzykiwanie za pomocą metod (method injection) – rzadziej stosowana forma. Obiekt definiuje chronioną metodę abstrakcyjną, natomiast kontener implementuje tę metodę w czasie działania aplikacji.

Użycie mechanizmu wstrzykiwania zależności polega na zaimplementowaniu klasy, która udostępnia swoje zależności a następnie umieszczeniu w konfiguracji aplikacji definicji odpowiednich właściwości. Umożliwią one frameworkowi wstrzyknięcie obiektów do klasy implementowanej aplikacji.

Podstawy bezpieczeństwa systemów komputerowych

Ogólne zasady ochrony informacji w systemach komputerowych

- Kompleksowość przedsięwzięć i działań w zakresie zapewniania bezpieczeństwa SI;
- Tworzenie specjalnych instytucji bezpieczeństwa odpowiedzialnych za zbieranie i rejestrowanie informacji o działalności wywiadowczej i sabotażowej osób i instytucji oraz informowanie o przyczynach i skutkach zagrożeń oraz doradzanie przełożonym odpowiednich metod i środków przeciwdziałania;
- Organizowanie właściwej, ścisłej współpracy pomiędzy różnymi instytucjami w celu uzgadniania jakie informacje i zasoby muszą być ochraniane oraz wspólnych standardów bezpieczeństwa i ochrony;
- Odpowiedni dobór środków bezpieczeństwa, które powinny:

    - określać i obejmować wszystkie **osoby** mające dostęp do wiadomości niejawnych, **urządzenia jako nośniki informacji** oraz **pomieszczenia**, gdzie znajdują się takie informacje;
    - być tak skonstruowane, aby wykrywać osoby, których zatrudnienie mogłoby naruszyć bezpieczeństwo informacji niejawnych i ważnych urządzeń, a także określać sposoby zwalniania osób naruszających zasady bezpieczeństwa;
    - uniemożliwiać dostęp osób nieuprawnionych do informacji niejawnych;
    - zapewniać, że informacje niejawne będą przekazywane na podstawie **zasady ograniczonego dostępu**, która jest zasadą fundamentalną we wszystkich aspektach bezpieczeństwa.
- Określanie poziomu niejawności dokumentu zgodnie z odpowiadającym mu poziomem wartości informacji oraz przyjmowanie takiego systemu klauzul tajności, który stanowić powinien instrument wykonania tej zasady;
- Planowanie środków na zapewnienie bezpieczeństwa i na ochronę w taki sposób, aby największe były dla najważniejszych urządzeń i podzespołów systemu informatycznego oraz dla najważniejszych obiektów;
- Takie organizowanie bezpieczeństwa teleinformacyjnego, aby obejmowało ono wszystkie szczeble organizacyjne systemu informatycznego.

Zakres i poziom ochrony prawnej systemów informatycznych w Polsce i na świecie

Model bezpieczeństwa systemu informatycznego stanowi podstawę do projektowania systemu zabezpieczeń. Za zapewnienie bezpieczeństwa teleinformatycznego odpowiada kierownik (szef, dyrektor) firmy, instytucji, jednostki organizacyjnej lub korporacji. System ochrony systemu informacyjnego, jest odpowiednim powiązaniem wymaganych środków ochrony z dodatkowymi rozwiązaniami uwzględniającymi specyfikę firmy (organizacji, korporacji), a tym samym - jej systemu informacyjnego wraz z urządzeniami technicznymi i ludźmi obsługującymi te urządzenia oraz ludźmi spełniającymi wyznaczone im funkcje ochronne, a także stosownymi dokumentami legislacyjnymi

Na system ochrony systemu informacyjnego składają się zarówno regulacje prawne (obowiązujące ustawy oraz zarządzenia w jednostce organizacyjnej) i organizacyjne, jak i organizacja systemu ochrony.

Organizacja systemu ochrony informacji uwzględnia następujące główne elementy składowe:

- zabezpieczenia wewnętrzne – (przy projektowaniu systemu): koncepcja, replikacja danych, synchronizację pracy systemu, model usług plikowych, danych dzielonych i wykonywanych na nich transakcji, sterowanie współbieżnością.
- zabezpieczenia zewnętrzne – zabezpieczenia odnoszące się do otoczenia.
- szyfrowanie informacji - uniemożliwienie, a co najmniej utrudnienie dostępu oraz poprawnej interpretacji danych przez osoby niepowołane.

System powinien spełniać zasady: kompleksowości, elastyczności, weryfikowalności, wiarygodności, dopasowania oraz minimalnych kosztów.

Normalizacja w dziedzinie bezpieczeństwa systemów informatycznych

Wobec istnienia zróżnicowanego sprzętu i oprogramowania pochodzących od różnych producentów muszą istnieć międzynarodowe normy pozwalające na ocenę systemu informatycznego w przestrzeni zwanej bezpieczeństwem tego systemu. Aby móc określić poziomy (klasy) bezpieczeństwa w sposób zunifikowany, zaczęto opracowywać ujednolicone kryteria bezpieczeństwa systemów informatycznych. Przyjęte standardy powinny być dla producentów i użytkowników swoistą "biblią informatyczną" - szczególnie przy tworzeniu systemów, w których poufność, integralność informacji i niezawodność mają znaczenie specjalne.

Należało chronić informacje zastrzeżone w systemach o zdalnym dostępie i z podziałem zasobów. W tym celu powstała tzw. Pomarańczowa księga, która definiuje cztery poziomy bezpieczeństwa – od D do A, od systemu najmniej do najlepiej zabezpieczonego.

Specyfika systemów informatycznych a zagrożenia bezpieczeństwa

Specyfika systemów informatycznych sprawia, że w aspekcie bezpieczeństwa systemu i znajdujących się w nim informacji pewne właściwości systemu komputerowego czynią go bardzo podatnym na zagrożenia.

Duża gęstość upakowania informacji. Wykorzystywane w systemach komputerowych podstawowe nośniki informacji, takie jak: dyski twarde, CD-ROM-y/DVD-ROM-y, dyskietki, taśmy magnetyczne mają stosunkowo niewielkie rozmiary, a jednocześnie zawierają duże, albo nawet bardzo duże ilości danych, które bardzo łatwo mogą ulec nawet przypadkowemu zniszczeniu.

Specyfika zapisu danych elektronicznych. Dane zapisane na komputerowych nośnikach informacji trudno poddawać bezpośredniej kontroli na zawartość, a także śledzić zachodzące w nich zmiany. Dokonanie jakiejkolwiek modyfikacji bazy danych przekazywanej w postaci niezabezpieczonej jest bardzo proste, lecz stwierdzenie tego faktu sprawia sporo trudności.

Istnienie promieniowania ujawniającego. Elektroniczny sprzęt informatyczny emituje promieniowanie elektromagnetyczne, które – przy zastosowaniu odpowiedniego sprzętu radioelektronicznego, komputerowego i oprogramowania – może posłużyć do odbioru i odtworzenia przetwarzanej informacji przez nieupoważnione osoby.

Łatwość tworzenia kopii danych. Względna niezależność elementów oprogramowania systemów komputerowych powoduje, że niezabezpieczone odpowiednio dane informatyczne mogą być bardzo łatwo skopiowane, bez pozostawiania jakiegokolwiek śladu tego zabiegu.

Śladowa trwałość zapisu danych na nośnikach magnetycznych. Dane zapisane na nośniku magnetycznym nawet po ich systemowym skasowaniu pozostawiają trwały ślad możliwy do odczytania (histereza magnetyczna).

Funkcjonowanie systemów komputerowych w układach sieciowych. Sieci komputerowe, to zwielokrotnienie możliwości pojedynczych komputerów. Pojedyncze komputery są instalowane tylko w sytuacjach, gdy są niewielkie potrzeby użytkownika albo gdy wymaga tego bezpieczeństwo informacji przetwarzanych w takim komputerze.

Niebezpieczeństwa:

- udostępnianie zasobów pojedynczych komputerów innym użytkownikom sieci komputerowej,
- wspólne użytkowanie określonych zasobów informacyjnych,
- niedoskonałość systemów operacyjnych,
- niedoskonałość opracowania i wdrożenia procedur bezpieczeństwa,
- lekceważenie zasad bezpieczeństwa,
- zwiększenie poziomu anonimowości – poczucie bezkarności.

Istota systemów kryptograficznych symetrycznych

Algorytm symetryczny DES - jest algorytmem szyfrującym symetrycznym, tzn. ten sam klucz służy do szyfrowania, jak i deszyfracji. DES jest szyfrem przedstawieniowo – podstawieniowym o 64-bitowej długości bloków danych, klucz szyfrujący ma długość 56 bitów, uzupełnianych 8 bitami parzystości.

Zalety: Opis algorytmu podano do wiadomości publicznej jako dowód, że wartość metody szyfrowania nie polega na utajnieniu algorytmu, ale jego konstrukcji odpornej na analizę kryptograficzną. Zasady generowania podkluczy są jawne i również zostały opublikowane. DES może być implementowany zarówno hardware'owo, jak i software'owo.

Wady: Długość klucza wobec postępu technologicznego może okazać się niewystarczająca, wiąże się z tym coraz większe ryzyko złamania szyfru metodą przeszukiwania wszystkich kluczy. Dotychczas nie zostały opublikowane prace dające metodzie DES solidne podstawy matematyczne, więc ciągle pojawiają się wątpliwości, czy jest ona efektywna. Na jednym z etapów jego implementacji był modyfikowany przez organ bezpieczeństwa narodowego USA, co rodzi podejrzenia, że agencja ta zna "ukryte drzwi" algorytmu i może to wykorzystać do złamania szyfrogramów DES-a. Problemem w stosowaniu szyfrowania symetrycznego, jest generowanie, przechowywanie i dystrybucja kluczy. Koszty i inne problemy z tym związane są na tyle wysokie, że metody szyfrowania symetrycznego stosuje się w ograniczonym zakresie tam, gdzie tajemnica i ryzyko jej ujawnienia uzasadniają ponoszone koszty.

Istota systemów kryptograficznych asymetrycznych

Wśród wielu rozwiązań w tym zakresie najszersze zastosowanie znalazł szyfr RSA. Od momentu publicznego przedstawienia RSA, były na nim dokonywane liczne próby w kierunku złamania szyfrów generowanych tą metodą. Osiągnięto na tym polu bardzo ograniczone rezultaty, a ich konsekwencją było przede wszystkim zwiększenie długości kluczy używanych przez RSA. Jest to bowiem bardzo istotna cecha algorytmu, że – w przeciwieństwie do DES-a – w RSA możliwa jest dowolna długość klucza, ustalona przez użytkownika.

W systemach asymetrycznych (z kluczem publicznym), każdy użytkownik tego systemu, jeśli potrzebuje użyć szyfrowania, generuje parę ściśle związanych ze sobą kluczy: klucz publiczny (jawny) oraz klucz prywatny (tajny). klucz publiczny zostaje udostępniony wszystkim użytkownikom, którzy są zainteresowani przekazywaniem zaszyfrowanych wiadomości do użytkownika, który ten klucz wygenerował i udostępnił. Klucz prywatny (tajny) pozostaje dostępny tylko dla użytkownika, który go wygenerował w parze z kluczem publicznym. Jeśli użytkownik, który otrzymał klucz publiczny (nadawca), zechce nadać zaszyfrowaną wiadomość do użytkownika, który mu ten klucz przekazał (odbiorcy), to szyfruje wiadomość przy użyciu otrzymanego klucza i przekazuje szyfrogram odbiorcy.

W procedurach autentyfikacji i niezaprzeczalności nadawca szyfruje wiadomość kluczem tajnym, a odbiorca odbiera go przy użyciu klucza publicznego. W ten sposób odbiorca uzyskuje potwierdzenie, że nadawca jest tym jedynym, który posiada klucz tajny - a więc właściwym, autentycznym nadawcą, a ponadto nadawca nie może zaprzeczyć, że to on nadał określoną wiadomość, którą odbiorca odebrał przekazanym mu przez niego kluczem publicznym.

Podpis elektroniczny i cyfrowy, zakres zastosowań

Zaawansowany (bezpieczny) podpis elektroniczny – czyli podpis, który za pomocą odpowiednich środków technicznych (kryptograficznych) jest jednoznacznie i w sposób trudny do sfałszowania związany z dokumentem oraz autorem. Kategoria ta odnosi się do większości systemów tradycyjnie nazywanych podpisem elektronicznym i wykorzystujących różne algorytmy kryptograficzne dla zapewnienia bezpieczeństwa.

Kwalifikowany podpis elektroniczny – czyli taki podpis zaawansowany, który został złożony przy pomocy certyfikatu kwalifikowanego oraz przy użyciu bezpiecznego urządzenia do składania podpisu (SSCD).

Podpis cyfrowy - 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.

Zakres zastosowań:

- przetargi i aukcje elektroniczne.
- kontakty z urzędami administracji publicznej.
- e-zdrowie, e-daklaracje, e-faktura, e-KRS, e-GIDO, podpis dla ZUS.

Istota uwierzytelniania, zakres zastosowań

Uwierzytelnianie (ang. authentication, niepoprawnie autentykacja) – proces polegający na zweryfikowaniu zadeklarowanej tożsamości osoby, urządzenia lub usługi biorącej udział w wymianie danych. Uwierzytelnienie następuje po identyfikacji, czyli zadeklarowaniu swojej tożsamości przez użytkownika (np. przez podanie loginu). Zadeklarowana, ale jeszcze niezweryfikowana, tożsamość jest potwierdzana w procesie uwierzytelnienia (np. przez podanie hasła).

Ostatnio wprowadzane jest zabezpieczenie biometryczne (np. odcisk palca, potocznie: "kim jesteś"), jednak z kryptograficznego punktu widzenia jest to odmiana zabezpieczenia "co masz".

Zastosowanie:

- e-banki,
- wszelkiego rodzaju e-transakcje,
- kontakt z ZUSem i innymi organami administracji publicznej
- wszelkiego rodzaje portale internetowe na których logujemy się podając hasło

Polityka bezpieczeństwa firmy (instytucji, korporacji)

Polityka Bezpieczeństwa Firmy (PBF) jest zintegrowanym zbiorem ogólnych zasad i dyrektyw wewnętrznych w zakresie bezpieczeństwa. Odzwierciedla uregulowania obejmujące Centralę i oddziały Firmy. PBF ma charakter przymusowy, czyli żaden pracownik nie może działać inaczej bez specjalnej zgody kierownika jednostki organizacyjnej Firmy

Podstawowe założenia modelu zabezpieczeń: Bezpieczeństwo systemów informacyjnych musi być zapewnione przede wszystkim w trzech obszarach obejmujących:

- Tajność (dostęp do informacji musi być ograniczony tylko do kręgu użytkowników autoryzowanych);
- Integralność (informacja musi być zachowana w swej oryginalnej postaci, za wyjątkiem przypadków, gdy jest ona aktualizowana lub usuwana legalnie przez osoby do tego upoważnione);
- Dostępność (informacja musi być dostępna dla osób upoważnionych na ich żądanie, w każdej chwili).

Audyt bezpieczeństwa – istota, cel, metody

Audyt bezpieczeństwa to okresowe badanie i monitorowanie stanu zabezpieczeń wszystkich elementów systemu informatycznego oraz eliminowanie wykrytych luk i nieprawidłowości. Ideałem byłoby, gdyby audyt wykonywany był regularnie, np. co kilka miesięcy, a zapis o nim był zawarty w dokumentach polityki bezpieczeństwa firmy, do której należy system informatyczny. Planując audyt należy przede wszystkim ustalić co jaki czas ma być realizowany oraz dobór osób które mają go wykonać. Mogą to być osoby z instytucji macierzystej lub można to zlecić innej firmie.

Proces audytu można podzielić na: zewnętrzny (wykrycie ataków z sieci zewnętrzny w tym Internetu) i wewnętrzny (ataki przez pracowników). Niektóre testy audytu mogą zakłócić pracę systemu, np. restart serwera, rekonfiguracja oprogramowania. Dlatego dobrze jest zaplanować czas przeprowadzania audytu zwłaszcza jeśli koliduje z pracą systemu.

Metody:

- Testy penetracyjne mają na celu określenie realnej odporności systemu informatycznego na ataki hackerów: skanowanie przestrzeni adresowej sieci lokalnej, skan portów serwerów, symulują włamań, ocena odporności systemu na ataki typy DoS.

- Testy kontrolne – nieprawidłowości w konfiguracji systemu. Obejmują etapy: kontrola wersji systemu operacyjnego, weryfikacja poprawności konfiguracji, weryfikacja śladów włamań, weryfikacja poziomu bezpieczeństwa.
- Analiza systemowa zabezpieczeń – podejmuje problem oceny poziomu zabezpieczenia systemu informatycznego jako całości.

Podstawy przetwarzania rozproszonego

Podać definicję systemu rozproszonego

System rozproszony jest zbiorem niezależnych komputerów, które z punktu widzenia użytkowników systemu sprawiają wrażenie pojedynczego komputera. jest zbiorem komputerów, które nie mają wspólnej pamięci ani zegara. jest zbiorem niezależnych komputerów połączonych siecią komunikacyjną, która umożliwia wymianę komunikatów. **Cechy:**

- Dzielenie zasobów – wielu użytkowników systemu może korzystać z danego zasobu (np. drukarek, plików, usług, itp.).
- Otwartość – podatność na rozszerzenia, możliwość rozbudowy systemu zarówno pod względem sprzętowym, jak i oprogramowania.
- Współbieżność – zdolność do przetwarzania wielu zadań jednocześnie.
- Skalowalność – cecha systemu umożliwiająca zachowanie podobnej wydajności systemu przy zwiększaniu skali systemu (np. liczby procesów, komputerów, itp.).
- Tolerowanie awarii– właściwość systemu umożliwiająca działania systemu mimo pojawiania się błędów i (lub) uszkodzeń (np. przez utrzymywanie nadmiarowego sprzętu).
- Przezroczystość – właściwość systemu powodująca postrzeganie systemu przez użytkownika jako całości, a nie poszczególnych składowych.
- Wady systemów rozproszonych
- Oprogramowanie (zdecydowanie bardziej złożone; wymaga opracowania wspólnych standardów)
- Sieci (może ulec awarii lub zostać przeciążona)
- Bezpieczeństwo (komputer podłączony do sieci jest mniej bezpieczny)

Scharakteryzować najważniejsze cechy systemów rozproszonych: przeźroczystość, otwartość, skalowalność.

Otwartość (ang. openness) – podatność na rozszerzenia, możliwość rozbudowy systemu zarówno pod względem sprzętowym, jak i oprogramowania.

Skalowalność (ang. scalability) – cecha systemu umożliwiająca zachowanie podobnej wydajności systemu przy zwiększaniu skali systemu (np. liczby procesów, komputerów, itp.).

Przezroczystość (ang. transparency) – właściwość systemu powodująca postrzeganie systemu przez użytkownika jako całości, a nie poszczególnych składowych.

Na czym polega istota przetwarzania rozproszonego

Istotą przetwarzania rozproszonego jest podział przetwarzania pomiędzy wieloma autonomicznymi systemami – zwanymi systemami rozproszonymi. Przetwarzane jednostki są zależnymi od siebie operacjami obliczeniowymi.

Przetwarzanie rozproszone wymaga nie tylko synchronizacji procesów (tak, jak w przypadku przetwarzania współbieżnego) ale wymagają jeszcze opracowania metod komunikacji pomiędzy odległymi od siebie systemami komputerowymi oraz opracowania protokołu wymiany danych pomiędzy tymi systemami. Obliczenia rozproszone stanowią więc większe wyzwanie niż obliczenia współbieżne wykonywane na jednym komputerze.

Zaletą architektury rozproszonej jest naturalnie zwiększenie wydajności obliczeń bez względu na charakter tych obliczeń – co nie zawsze jest osiągalne przy przetwarzaniu równoległym czy współbieżnym.

Scharakteryzować model przetwarzania rozproszonego z pamięcią wspólną

Maszyny z pamięcią wspólną inaczej globalną, dzieloną, współdzieloną. W maszynach tego typu wszystkie procesy mają tę samą przestrzeń adresową. Dane osiągalne są zgodnie z regułami zasięgu obowiązującymi w zastosowywanym do wytworzenia programu języku programowania.

Model PRAM (Parallel Random Access Machine) zapewnia możliwość jednoczesnego dostępu każdego spośród N procesorów o architekturze RAM do pamięci współdzielonej. Komunikacja między procesami i ich synchronizacja odbywa się z użyciem wspólnych zmiennych. Modele PRAM dzielimy:

- EREW PRAM – nie zezwala na żaden równoczesny zapis oraz odczyt informacji w danej komórce pamięci
- CREW PRAM – możliwa jest jedynie równoczesna operacja odczytu przez wiele procesów, przy ograniczeniu możliwości jednoczesnego zapisu do pojedynczego procesora
- CRCW PRAM – zezwala na jednoczesność zarówno odczytów jak i zapisów

Scharakteryzować model przetwarzania rozproszonego z pamięcią rozproszoną

Model przetwarzania rozproszonego z pamięcią rozproszoną (nie wiadomo czy to dobre) - procesy komunikują się ze sobą przez pewien układ komunikacyjny a nie wspólną pamięć.

Model klient-serwer, jest podstawowym modelem przetwarzania danych w systemach rozproszonych. Klient jest stroną żądającą dostępu do danej usługi lub zasobu, zaś serwer jest stroną, która świadczy usługę lub udostępnia zasoby. Oznacza to, że klient i serwer są procesami porozumiewającymi się za pomocą wymiany komunikatów. Klient i serwer mogą pracować na tym samym komputerze, używając mechanizmów komunikacji lokalnej, lub na różnych komputerach, wykorzystując komunikację przez sieć. Klient i serwer nie odgrywają symetrycznych ról. Proces serwera jest uruchamiany, a następnie wprowadzany w stan oczekiwania na żądania pochodzące od ewentualnych klientów. Procesy klientów są zwykle uruchamiane interakcyjnie i wysyłają żądania do serwera. Serwer przetwarza otrzymane żądania i ewentualnie prowadzi dialog z klientem. Następnie powraca do oczekiwania na kolejne żądania.

Wyróżnia się dwa typy serwerów:

- serwery iteracyjne - proces serwera sam zajmuje się obsługą klienta
- serwery współbieżne - aby obsłużyć klienta, proces serwera uruchamia nowy proces

Co to jest sekcja krytyczna?

Sekcja krytyczna - w programowaniu współbieżnym fragment kodu programu, w którym korzysta się z zasobu dzielonego, a co za tym idzie w danej chwili może być wykorzystywany przez co najwyżej jeden wątek. System operacyjny dba o synchronizację, jeśli więcej wątków żąda wykonania kodu sekcji krytycznej, dopuszczany jest tylko jeden wątek, pozostałe zaś są wstrzymywane. Dąży się do tego, aby kod sekcji krytycznej był krótki - by wykonywał się szybko. Sekcja krytyczna może być użyta ażeby zagwarantować, że wspólny zasób, na przykład drukarka, jest używana tylko przez jeden proces w określonym czasie. Sposób implementacji sekcji krytycznych jest zależny od systemu operacyjnego.

Sekcje krytyczne realizuje się np. z wykorzystaniem blokowania przerwań, muteksów lub semaforów. Brak wzajemnego wykluczania się wykonywania sekcji krytycznych może spowodować błędy wykonania, np. dwukrotne zapisanie danej albo niepoprawna modyfikacja zasobu.

Scharakteryzować semafory

Semafory to wysoko poziomowe mechanizmy synchronizacji procesów współbieżnych, wprowadzone przez Dijkstrę. Semafor jest chronioną zmienną lub abstrakcyjnym typem danych, który stanowi klasyczną metodę kontroli dostępu przez wiele procesów do wspólnego zasobu w środowisku programowania równoległego.

Semafor binarny Dijkstry –może przyjmować tylko jedną z dwóch możliwych wartości: 1 lub 0. Operacja opuszczenia(P) polega na zmniejszeniu jego wartości o jeden jeśli jest to możliwe. Tak więc jeśli semafor ma wartość 1, to wówczas przyjmuje wartość 0. Jeśli natomiast jest już "wyzerowany", to uważany jest jako "niedostępny" lub "pusty" a zadanie wykonujące taką operację jest blokowane.

Semafor ogólny Dijkstry – to mechanizm, w którym wykorzystuje się jedynie operacje inkrementacji i dekrementacji pewnej całkowitoliczbowej nieujemnej zmiennej globalnej s. Po nadaniu wartości semaforowi s, mogą być na nim wykonywane jedynie funkcje P(opuszczania-czekania) i V(podnoszenia-sygnalizowania).

Scharakteryzować monitory

Monitory to struktury programowe, które są używane w programach współbieżnych do synchronizowania współpracujących procesów oraz do ochrony danych współdzielonych.

Monitor definiuje dwa typy operacji:

- operacje monitorowe – operacje, które mogą być wykonywane przez jeden wątek jednocześnie, są wykonywane na danych współdzielonych, często modyfikując te dane – więc współbieżny dostęp do tych danych mógłby być niebezpieczny,
- inne operacje – najczęściej operacje tylko do odczytu, które nie wymagają synchronizacji.

Monitory zawsze są powiązane z konkretnymi danymi współdzielonymi, które to dane chronią. Monitory są obok semaforów podstawowymi mechanizmami synchronizacji procesów/wątków w aplikacjach. W języku Ada95 są implementowane przez typy chronione.

Scharakteryzować kanały

Kanały są mechanizmami pozwalającymi na dwustronne przekazywania komunikatów miedzy procesami. W kanałach można wyodrębnić podkanały, z których każdy może pełnić rolę odrębnego bufora. W rezultacie kanał jest abstrakcją półki z przegródkami, w których można pozostawić informację różnym osobom.

W modelu przesyłania komunikatów rolę łącza spełnia kanał komunikacyjny. Wymiana informacji jest realizowana poprzez wysyłanie komunikatów – **nadaj(komunikat) i odbierz(komunikat)**. Kanały komunikacyjne **channels** mogą być tworzone dynamicznie.

Scharakteryzować spotkania

Spotkania porównuje się do spotkania dwóch dobrych znajomych w z góry znanym miejscu, podczas którego osoby mogą sobie przekazywać informacje. Spotkania te są symetryczne ze względu na znajomość, lecz są niesymetryczne ze względu na kierunek przepływu informacji (przekaz może być jednostronny).

Spotkania niesymetryczne ze względu na znajomości jak i przepływ informacji. Klient wie gdzie przyszedł, ale jego w tym spotkaniu ogranicza się jedynie do oczekiwania wy wykonanie usługi. Pracownik serwisu nie zna klienta, za to podczas spotkania jest on aktywny.

Spotkania asymetryczne umożliwiają zarówno obustronną wymianę informacji (klient przynosi zepsutą rzecz, a pracownik ją naprawia i naprawioną zwraca klientowi) jak i jednostronną wymianę informacji (naprawy nie można wykonać na poczekaniu i należy umówić się na kolejne spotkanie). Idea ta realizowane jest w mechanizmie zdalnego wywoływania procedur **RPC** (Remote Procedure

Call). W procesie żądającym wykonania usługi (klient) jest wywoływana procedura zaimplementowana w innym procesie, który zazwyczaj wykonywany jest na serwerze.

Problemy społeczne i zawodowe informatyki

Omówić problemy społeczne informatyki. Podać przykłady

- Rozpowszechnianie w Internecie nielegalnych treści, w tym m.in. pornografii dziecięcej i materiałów propagujących przemoc i okrucieństwo.
- Anonimowość Internetu sprzyja pedofilom i tworzy dogodne środowisko dla wymiany treści pornograficznych i identyfikacji dzieci, które można seksualnie wykorzystać.
- Cyberbullingu, czyli prześladowania w Internecie oraz grooming - uwodzenia nieletnich w sieci.
- Uzależnienia od komputera i Internetu.
- Smog informacyjny.
- Wszelkiego rodzaju cyberprzestępstwa, np. phishing, pharming, hacking, które prowadzą to kradzieży ważnych danych (loginu i hasła), a następnie są kradzione nasze realne pieniądze z kont bankowych.

Omówić problemy zawodowe informatyki. Podać przykłady

- Problemy zdrowotne związane ze zły przygotowaniem stanowiska pracy. Na problemy zdrowotne, a nawet kalectwo mogą narazić się wszyscy Ci, którzy w nieodpowiedni sposób korzystają przez okres kilku lat z nie ergonomicznych stanowisk pracy. Problem tkwi chociażby w złym ustawieniu monitora, wysokości blatu biurka, wysokości krzesła, odległości pomiędzy dolną krawędzią stołu a siedzeniem itp. Nie bez znaczenia w tym wypadku jest również ułożenie dłoni, które powinny spoczywać na biurku. Następstwem powyższych niedociągnięć mogą być tzw. "choroby zawodowe", np. bóle kręgosłupa, problemy ze wzrokiem.
- Konieczność ciągłego podnoszenia kwalifikacji. Z każdy rokiem wchodzą nowe technologie, firmy wymyślają nowe, bardziej doskonałe rozwiązania problemu, w związku z czym informatyk chcąc być cennym na rynku pracy musi się stale dokształcać.
- Problem wypalenia zawodowego.

Wyjaśnić pojecie pomarańczowej księgi

W roku 1983 Departament obrony USA oficjalnie ogłosił "Kryteria oceny zaufania systemów komputerowych" TCSEC (ang. Trusted Computer System Evaluation Criteria) znane także pod nazwą **Pomarańczowej księgi**. Definiuje ona cztery poziomy bezpieczeństwa - od D do A, od systemu najmniej do najlepiej zabezpieczonego. Poziomy podzielono na klasy bezpieczeństwa oznaczone liczbowo tak, że bezpieczeństwo klasy w ramach poziomu rośnie wraz ze zwiększaniem się numeru klasy. Każdy poziom i klasa są charakteryzowane czterema głównymi elementami:

- taktyka (polityka) bezpieczeństwa (ang. securitv policv),
- identyfikacja, kontrola i sprawdzanie podmiotu, czyli jego odpowiedzialność (ang. accountability),
- ubezpieczenie eksploatacyjne i okres trwałości ubezpieczenia (ang. assurance),
- testy sprawdzające system i opis idei bezpieczeństwa systemu (ang. documentation).

Klasy i zakresy bezpieczeństwa:

- **Poziom D** - posiada tylko klasę D. Jest ona przeznaczona dla systemów, które były oceniane, lecz nie spełniły wymagań klas wyższych. Systemy należące do tego poziomu mają minimalne zabezpieczenia. Do tej grupy należy system DOS

- **Poziom C** - w skład tego poziomu bezpieczeństwa wchodzą systemy, w których bezpieczeństwo opiera się na dyskrecjonalnej kontroli dostępu.
- **Poziom B** - w skład tego poziomu bezpieczeństwa wchodzą systemy, w których bezpieczeństwo opiera się na obowiązkowej kontroli dostępu.
- **Poziom A** - obejmuje systemy, które mają zweryfikowane zabezpieczenia. Wymagana jest obszerna dokumentacja, której celem jest wykazanie, że baza bezpieczeństwa odpowiada wszelkim wymaganiom bezpieczeństwa na etapach projektu, implementacji i rozwoju.
- Każdy poziom (klasa) spełnia warunki poziomu (klasy) poprzedniego oraz pewną liczbę warunków dodatkowych.

Czynniki wpływające na klasyfikację systemów wg „Pomarańczowej księgi":

- Dyskrecjonalna kontrola systemu
- Ponowne użycie obiektów
- Etykiety
- Integralność etykiet
- Eksport etykietowanej informacji
- Eksport do wielopoziomowych urządzeń
- Eksport do jednopoziomowych urządzeń
- Etykietowanie wyjść odczytywanych przez użytkowników
- Regulowana kontrola dostępu
- Etykiety zależne od użytkownika
- Etykietowanie urządzeń
- Identyfikacja i uwierzytelnianie
- Nasłuch
- Wiarygodna ścieżka
- Architektura systemu
- Integralność systemu
- Testowanie bezpieczeństwa
- Specyfikacja i sprawdzanie projektu
- Analiza utajnionego kanału
- Zarządzanie bazą zaufania
- Zarządzanie konfiguracją
- Wiarygodne odzyskiwanie informacji
- Wiarygodna dystrybucja
- Przewodnik po bezpieczeństwie
- Przewodnik po bazie zaufania
- Dokumentacja testów
- Dokumentacja projektu

Wyjaśnić pojęcie smogu informacyjnego

Jest to problemy z nadmiarem i niedoborem informacji w Internecie. Smog informacyjny jest to problem, z którym obecnie boryka się współczesny Internet.

Jest to nadmiarowe zasypywanie Internetu informacjami, które często są nieprawdziwe, niepotrzebne bądź byle jakie (nie niosą istotnych wiadomości). Nadmiar tych informacji w dużej mierze przysłania interesujące treści. Wynikają z tego problemy z szybkim i dokładnym wyszukiwaniem. Za powstawanie smogu można obwiniać firmy reklamowe, jak również człowieka który umieszcza nieprawdziwe bądź zwielokrotnione istniejące treści w masowych ilościach.

Omówić budowę kart elektronicznych. Podać przykłady

Karta elektroniczna, karta chipowa (ang. smart card) — uniwersalny nośnik danych w postaci karty wykonanej z plastiku z umieszczonym na niej (lub wewnątrz niej) jednym lub kilkoma układami scalonymi (chip), które pozwalają na ochronę procesu logowania użytkownika, kontrolę dostępu i zawartych na niej danych. Może być odczytywana za pomocą urządzeń automatycznych, np. przy zawieraniu i rozliczaniu transakcji finansowych oraz w kasach cyfrowych. Karty elektroniczne mają rozmiar i wygląd zbliżony do tradycyjnych kart kredytowych z paskiem magnetycznym. Często posiadają również taki pasek i mogą być odczytywane w urządzeniach nie obsługujących kart elektronicznych.

Karty elektroniczne charakteryzują się:

- wielokrotnością usług - jedna karta może mieć kilka zastosowań, może być jednocześnie np. kartą identyfikacyjną, bankomatową i płatniczą;
- skuteczniejszą ochroną danych - odczytanie danych z karty wymaga bardziej skomplikowanych urządzeń od czytników pasków magnetycznych; część danych może być zaszyfrowana i niemożliwa do odczytania bez klucza;
- większą elastycznością - umożliwia szybką modyfikację danych; karta elektroniczna może zawierać znacznie więcej danych niż karty z paskiem magnetycznym.

Karty elektroniczne znalazły wiele **zastosowań** w życiu codziennym. Najczęściej służą jako:

- karty kontroli dostępu do pomieszczeń i zasobów;
- identyfikatory w postaci legitymacji elektronicznych jak i certyfikatów elektronicznych;
- karty płatnicze.

Omówić budowę kart magnetycznych. Podać przykłady

Karta magnetyczna – karta, w której nośnikiem danych jest pasek magnetyczny. Ich podstawowym elementem jest występująca w postaci paska magnetycznego pamięć. Umożliwia ona zapis informacji, które mogą być następnie odczytywane przy pomocy czytników kart magnetycznych reagujących na zmiany pola magnetycznego.

Fizyczne właściwości karty jak wymiary, giętkość, lokalizacja paska magnetycznego i jego właściwości magnetyczne definiują normy ISO. Współczynnik koercji określa odporność paska magnetycznego na rozmagnesowanie. Im wyższy tym pasek bardziej odporny na rozmagnesowanie.

Karty magnetyczne są powszechnie stosowane jako karty płatnicze, telefoniczne, identyfikacyjne, do kontroli dostępu, rejestracji czasu pracy, systemach lojalnościowych itp. Mogą posiadać dodatkowe zabezpieczenia w postaci hologramów lub mikrodruku.

Obsługa systemów komputerowych. Stawiane wymagania w zakresie bezpiecznej pracy

Nie ma bo nie było komu napisać.

Współczesne bariery rozwoju informatyki. Omówić podstawowe bariery rozwoju komputerów kwantowych

Ogólnie mówiąc, komputerem kwantowym nazywamy układ kwantowy, którego ewolucja (zgodna z prawami mechaniki kwantowej) reprezentuje rozwiązanie zadania obliczeniowego. Jednym z podstawowych wniosków, wypływających z tej definicji, jest to, iż komputery kwantowe sterowane są bezpośrednio przez prawa przyrody – to przez nie dokonywane są wszelkie zmiany stanu układu. Fizycy od dawna mają kandydatów na kubity – cząstki elementarne, np. foton lub elektron.

Komputer kwantowy, mimo że wykorzystywałby inne właściwości fizyczne niż klasyczne komputery, nie umożliwiałby rozwiązywania nowej klasy problemów. Każdy problem rozwiązywalny przez

komputer kwantowy może zostać rozwiązany przez komputer klasyczny. Jednak dzięki specyficznym własnościom komputerów kwantowych pewne problemy można byłoby rozwiązać znacznie szybciej, co w praktyce znacznie poszerzyłoby zakres problemów do jakich mogą być użyte komputery

Podstawowym problemem, który napotykamy przy budowie komputera kwantowego, jest zapewnienie środków służących przechowywaniu informacji. Informacja kwantowa zapisana jest w postaci kubitów – kwantowych analogów klasycznych bitów. Kubity, w przeciwieństwie do bitów, mogą znajdować się również w stanach pośrednich pomiędzy 0 i 1, nazywanych superpozycjami.

Jednym z największych problemów, z którymi zmagają się naukowcy pracujący nad zbudowaniem komputera kwantowego, jest **zjawisko dekoherencji**. W znacznym uproszczeniu, można powiedzieć, iż zjawisko dekoherencji polega na wycieku informacji z układu kwantowego do otoczenia. Dekoherencja polega zazwyczaj na zmianie superpozycji stanów na pojedynczy stan ustalony, przez co tracone są dane. W celu budowy komputera kwantowego konieczne jest zapobieżenie zjawisku dekoherencji oraz minimalizacja szumów kwantowych pochodzących od otoczenia.

Przestępstwa komputerowe. Podać przykłady

"W szerokim rozumieniu, przestępczość ta obejmuje wszelkie zachowania przestępcze związane z funkcjonowaniem elektronicznego przetwarzania danych, polegające zarówno na naruszaniu uprawnień do programu komputerowego, jak i godzące bezpośrednio w przetwarzaną informację, jej nośnik i obieg w komputerze oraz cały system połączeń komputerowych, a także w sam komputer. (komputer jako narzędzie do popełnienia przestępstwa), jak i skierowane przeciwko takiemu systemowi".

Zjawisko to pojawiło się wraz z rozwojem komputeryzacji i od tej pory towarzyszy mu nieustannie. Choć "przestępstwo przeciwko danym" jest takim samym przestępstwem jak każde inne, jest ono trudne do wykrycia. Sprawca może być trudny do ustalenia, a ilość śladów z nim związana może być znikoma. Specyfika systemów komputerowych powoduje niejednokrotnie ich całkowite zatarcie.

Według ekspertów Rady Europy przestępstwa komputerowe dzielą się na grupy:

- oszustwo związane z wykorzystaniem komputera,
- fałszerstwo komputerowe,
- zniszczenie danych lub programów komputerowych,
- sabotaż komputerowy,
- "wejście" do systemu komputerowego przez osobę nieuprawnioną (patrz: cracking, haker),
- "podsłuch" komputerowy,
- bezprawne kopiowanie, rozpowszechnianie lub publikowanie programów komputerowych prawnie chronionych,
- bezprawne kopiowanie topografii półprzewodników,
- modyfikacja danych lub programów komputerowych,
- szpiegostwo komputerowe,
- używanie komputera bez zezwolenia,
- używanie prawnie chronionego programu komputerowego bez upoważnienia,
- metoda salami – to forma przestępstwa polegająca na kradzieży małych sum z różnych źródeł.

Etyka w informatyce i etyka informatyków

Etyka jest to nauka o moralności, w informatyce jak i w życiu należy kierować się pewnymi zasadami. Doskonale i skrótowo określa to Dekalog etyki komputerowej (etyki informatyka):

- Nie będziesz używał komputera, by szkodzić bliźnim
- Nie będziesz przeszkadzał bliźnim w pracy z komputerem
- Nie będziesz grzebał w plikach bliźniego.

- Nie będziesz używał komputera do kradzieży.
- Nie będziesz używał komputera do składania fałszywych świadectw.
- Nie będziesz używał lub kopiował programów, za które nie zapłaciłeś.
- Nie będziesz używał zasobów komputerowych bliźnich bez zezwolenia.
- Nie będziesz przywłaszczał sobie własności intelektualnej bliźnich.
- Będziesz rozważał społeczne konsekwencje programów, które napiszesz.
- Będziesz używał komputera z rozwagą i szacunkiem.

Programowanie deklaratywne

Predykaty wbudowane w Prologu. Wyjaśnić pojęcie predykatu równości. Podać przykłady predykatów pozwalających wyświetlać teksty oraz je pobierać od użytkownika. Obsługa plików

W Prolog-u istnieją **predykaty wbudowane**, ich nazwy są zastrzeżone i nie można ich przedefiniować. Np. random(X), X div Y, sqrt(X). Predykaty wbudowane mogą realizować zadania, których nie można zaprogramować w czystym Prologu, mogą też realizować zadanie, których nie można zaprogramować w czystym Prologu, lub realizować zadania, które ułatwiają pracę programiście.

Predykat równości. Przykładowo, operacja przyrównania jest predykatem postaci: *=(X,Y)*, gdzie X, Y są zmiennymi tej samej dziedziny. Dla wygody zapisu przyjęto w wielu dialektach PROLOGu notację beznawiasową dla predykatów wbudowanych, tak więc predykatu związanego z operacją przyrównania można używać pisząc: *X=Y*, kiedy Prolog natyka się na taki cel, stara się zrównać *X* i *Y* dopasowując je do siebie. Jeśli jest o możliwe, cel jest uzgodniony. W przeciwnym razie cel zawodzi.

Predykat = = powoduje znacznie dokładniejsze porównanie niż =. Jeśli nie zawodzi *X==Y,* to nie zawiedzie też *X=Y*, ale nie odwrotnie. Predykat = = znacznie dokładniej analizuje zmienne. Predykat = traktuje zmienną nieukonkretnioną jako równą dowolnej wartości, zaś dla predykatu = = zmienna nieukonkretniona jest równa jedynie zmiennej z nią związanej.

Predykatem przeciwnym do równości jest X\=Y

Przykłady predykatów pozwalających wyświetlać teksty oraz je pobierać od użytkownika.

- get_char(X) – odczytywanie znaków z klawiatury
- put_char(X) – wypisywanie znaków

Obsługa plików.

- open(X,Y,Z) – otwiera plik o nazwie *X*, jeśli *Y* ma wartość *read,* plik jest otwierany do czytania, w przeciwnym razie plik jest otwierany do pisania, *Z* jest ukonkretniane specjalnym termem będącym nazwą strumienia, którego należy używać podczas odwoływania się do pliku.
- close(X) – predykat używany do zamknięcia strumienia o nazwie *X.* Strumień staje się niedostępny.
- set_input(X) – ustawia bieżący strumień wejściowy na strumień wskazany przez *X*. *X* jest termem zwracanym jako trzeci argument *open* lub atomem *user_input,* nakazującym pobieranie danych z klawiatury.
- set_output(X) – ustawia bieżący strumień wyjściowy na strumień wskazany przez *X*. *X* jest termem zwracanym jako trzeci argument *open* lub atomem *user_output,* nakazującym wypisywanie danych na monitorze.
- current_input(X) – cel jest uzgodniony, jeśli nazwa bieżącego strumienia wejściowego pasuje do X, a zawodzi w przeciwnym razie.
- current_output(X) – cel jest uzgodniony, jeśli nazwa bieżącego strumienia wyjściowego pasuje do X, a zawodzi w przeciwnym razie.

Dlaczego w środowisku deklaratywnym mówi się o tym, iż zmienna zastępuje obiekt? Czy zmienna anonimowa w Prologu zastępuje obiekt?

Zmienne w prologu traktuje się jako zastępstwa obiektów, których nie potrafimy w danej chwili nazwać. Zmienna nie nazywa konkretnego obiektu, ale może zastępować obiekt, którego nazwy nie potrafimy podać. Zmienna ukonkretniona – odpowiada konkretnemu obiektowi, natomiast nieukonkretniona – nie wiadomo, jakiemu obiektowi może odpowiadać. Póki argument nie jest ukonkretniony to może być zastępowany dowolnym innym argumentem występującym w danym miejscu. W momencie ukonkretnienia prolog przeszukuje bazę wiedzy celem znalezienia faktu.

W przypadkach, gdy nie ma znaczenia nazwa zmiennej, bądź po prostu nie jesteśmy zainteresowani wiedzą pochodzącą z danego obiektu, który miała przedstawić zmienna, używa się zmiennej anonimowej. Oznaczona jest podkreśleniem. Używa się ich także często, aby nie wymyślać nazw dla zmiennych, ponieważ występują one tylko w jednym miejscu i nigdzie indziej. Podsumowując zmienna anonimowa jak najbardziej zastępuje obiekt.

Czy w języku Prolog fakty się definiuje czy deklaruje? Podać przykład zadawania pytań o fakty

W języku Prolog fakty się definiuje, deklaracyjną formułę mają zapytania i reguły[1].

W prologu definiujemy sobie bazę faktów:
maKota(ala);
maKota(kasia);
jest Ojcem(tomek, kasia);

Zapytania o fakty w tym przypadku wyglądają podobnie tak, jak definiowanie ich, tj. podajemy interpreterowi:
maKota(ala);
Rezultat: *Yes*,
maKota(magda);
Rezultat: *No*,
jestOjcem(michał, kasia);
Rezultat: *No*,
itp.

Warto zwrócić uwagę, na dwie rzeczy:

- jeśli zapytamy o fakt, który nie znajduje się w bazie faktów, zawsze otrzymamy No, np.: maPsa(kasia); Rezultat: *No*;
- Prolog nie zdaje sobie sprawy z tego, co właściwie te fakty oznaczą – więc nie wie, że fakt jestOjcem(tomek, kasia) oznacza „Ojcem Kasi jest Tomek”, utrzymywanie znaczenia jest zadaniem programisty

1 Mgliste. Na dobrą sprawę w dostępnych mi materiałach nie odnalazłem jednoznacznej odpowiedzi. Cała kwestia rozbija się o określenie znaczenia pojęć definicji i deklaracji.
Jeśli deklarację potraktujemy w kontekście samego paradygmatu programowania deklaratywnego, to jest to sposób rozwiązywania problemu przez wyspecyfikowanie go, bez podania sposobu na jego rozwiązanie – innymi słowy piszemy to, CO chcemy osiągnąć, a nie JAK chcemy osiągnąć.
Jeśli chodzi o definicję – podajemy, czym jest dany obiekt.
Więc w jaki sposób można określić fakt: (c.d. nast. strona)
maKota(ala);
Deklaracja? Nie wydaje się, ze względu na to, że na dobrą sprawę nie mamy żadnego celu, który chcemy osiągnąć.

Budowa reguł w Prologu. Podać przykład głowy. Czy reguła jest klauzulą?

Reguła wnioskowania składa się ze zbioru wyrażeń nazywanych warunkami i zbioru wyrażeń nazywanych konkluzjami.

Reguła w prologu składa się z głowy i ciała. Aby spełniona była przesłanka reguły, spełnione muszą być wszystkie jej podcele (najczęściej łączone przecinkiem ‘**,**’ co oznacza logiczne AND). Głowa oddzielona jest od ciała symbolem dwukropka i myślnika ‘**:-**‘

Dla zdania X lubi Y, jeśli Y gra Futbol

Lubi(X,Y) :- gra(Y,Futbol).

Lubi(X,Y) jest **przykładem głowy**.

Należy zauważyć że reguła kończy się kropką. Treść gra(Y,Futbol) zawiera koniunkcję celów które muszą być spełnione aby głowa była prawdziwa.

W Prologu stwierdzenie zapisuje się w postaci klauzul. Klauzule w postaci

$\mathbf{A_1 \wedge A_2 \wedge \ldots \wedge A_m \Rightarrow B_1}$ gdzie m>0, A - poprzednik klauzuli, B - następnik klauzuli

Analogiczny zapis to $\mathbf{B_1 :- A_1 , A_2 , \ldots , A_m}$ Nazywa się regułami.

Można wyróżnić jeszcze klauzule w innych postaciach jak na przykład fakty i cele. Z tego można stwierdzić, że **reguła jest klauzulą**.

Wyjaśnić pojęcie atomu i termu. Czy struktura jest atomem czy termem. Podać w Prologu przykład struktury w formie drzewa

Atom – są to stałe znakowe, występują 2 rodzaje atomów:

- składające się z liter i cyfr (zaczyna się małą literą) np. ania, x_y, 123,
- ciąg znaków specjalnych/symboli np. <->, ==>, …., ?-
- Ciąg znaków pomiędzy apostrofami albo cudzysłowami np.: 'Ania', "Polska". Wykorzystujemy ten sposób zapisu do nazywania stałych zgodnie z zasadami gramatycznymi tzn. zaczynając nazwę z dużej litery. Jeśli atom ujęty jest w pojedynczy cudzysłów, może zawierać dowolne znaki dozwolone podkreślenie.

Term – wyrażenie składające się ze zmiennych oraz symboli funkcyjnych o dowolnej argumentowości (w tym o argumentowości 0, czyli stałych), z pewnego ustalonego zbioru. W wielu dziedzinach matematyki używa się określenia term na oznaczenie napisów (wyrażeń) formalnych, które mogą być traktowane jako nazwy obiektów matematycznych. Termy języka to elementy najmniejszego zbioru.

Czy struktura jest atomem czy termem – struktura to pojedynczy obiekt składający się z zestawu innych obiektów, nazywanych składnikami struktury. Struktury są nazywane w standardzie języka Prolog „termami złożonymi”. Przykładek struktur mogą być karta katalogowa w bibliotece. Czyli **struktura JEST TERMEM.**

Podać w Prologu przykład struktury w formie drzewa. - Każdą strukturę da się przedstawić w postaci drzewa. Gdy struktura jest dość złożona ułatwia to zrozumienie jej budowy. np.

```
książka(autor(karl, popper), pan_tadeusz, wydanie(wsip, 1982))
```

Podać przykłady operatorów infiksowych, prefiksowych i postfiksowych w Prologu. Klasy priorytetów. Kiedy stosuje się łączność operatorów?

W Prologu niektóre funktory warto jest zapisywać jako operatory, gdyż użycie takiej składni ułatwia czytanie programu. Przykładowo operacje arytmetyczne zapisuje się zwykle za pomocą operatorów. Intuicyjnie zapisuje się x+y*z, jednak w przypadku chęci zapisania podanego wyrażenia tradycyjnie, jako strukturę, zapis wyglądałby następująco: +(x, *(y,z)) i byłby poprawnym termem w prologu.

Operatory infiksowe – zapisywane pomiędzy swoimi argumentami, np. +, -, *, /

Operatory prefiksowe – występujące przed argumentami, jak minus w wyrażeniu –x, oznaczający zmianę znaku argumentu. Operatory postfiksowe – występujące za swoimi argumentami, np. silnia x!.

Priorytet operatora mówi, które operacje mają być przeprowadzane najpierw. Każdy operator w prologu ma klasę priorytetu. Jest ona liczbą całkowitą związaną z operatorem, zaś wartość tej liczby zależy od używanej wersji prologu. Zawsze jednak operator z wyższym priorytetem ma klasę priorytetu bliższą jedynki. Jeśli klasy mieszczą się w zakresie od 1 do 255, to operatory pierwszej klasy są wykonywane jako pierwsze.

Łączność operatorów jest ważna gdy występuje kilka operatorów o takim samym priorytecie. Operator mogą być łączne lewostronnie i prawostronnie. Pierwsze z nich muszą mieć po lewej stronie operacje o priorytecie takim samym lub niższym, a po prawej – priorytecie niższym. Wszystkie operatory arytmetyczne (dodawanie, odejmowanie, mnożenie i dzielenie) są łączne lewostronnie. W przypadku wyrażeń sprawiających trudności w interpretacji, stosuje się nawiasy, które pozwalają uniknąć niejasności.

Wyjaśnić pojęcie celów i mechanizmu nawracania w Prologu. Rola predykatu „!”

Cel w prologu jest to stwierdzenie, które chcemy udowodnić (lub obalić, tzn. udowodnić jego fałszywość). Formalnie jest to klauzula bez głowy; składniowo wygląda tak samo jak fakt. Po wpisaniu celu interpreter odpowie **true** lub **false**. Jeśli cel jest stwierdzeniem złożonym to, każda z zawartych w nim struktur jest podcelem. Jeśli cel posiada zmienne Prolog znajdzie instancje, dla których cel jest prawdziwy.

Mechanizm nawracania można porównać do rekurencji. Interpreter buduje swojego rodzaju drzewo, aby rozwiązać cel (cel posiada podcele). Przechodzi po „potomkach” drzewa (kolejnych klauzulach) aby uzgodnić go (głowę klauzuli) z bieżącym celem. Po niedopasowaniu (uzgodnienie zawodzi) wycofywane są wprowadzone zmiany i następuje pobranie następnego „potomka” (klauzuli) w celu uzgodnienia.

Rolą predykatu „!” jest odcięcie. Prolog umożliwia sterowanie mechanizmem nawracania. Dokonuje się tego przez zastosowanie standardowego predykatu odcięcia, oznaczanego przez wykrzyknik (**!**). Predykat ten jest zawsze spełniony i jego działanie uboczne polega na tym, że blokuje wszystkie możliwe pozostałe do wykonania sprawdzenia spełnienia predykatów znajdujących się w danej klauzuli. Tym samym predykat „!” uniemożliwia nawracanie w obrębie klauzuli w której występuje.

Wyjaśnić mechanizm przeszukiwania rekurencyjnego na przykładzie listy nazw miesięcy. Na czym polega porównywanie rekurencyjne?

Przeszukiwanie rekurencyjne. Często jest konieczne przeszukanie struktur w Prologu w celu znalezienia pewnych informacji. Kiedy struktury mogą mieć inne struktury jako argumenty, konieczne jest szukanie rekurencyjne.

Załóżmy, że mamy listę nazw miesięcy:
[styczen.luty.marzec.kwiecien.maj.czerwiec.lipiec.sierpien.wrzesien.pazdziernik.listopad.grudzien]

Teraz przyjmijmy, ze chcemy dowiedzieć się czy dany miesiąc znajduje się na liście. W Prologu najpierw sprawdzamy czy interesujący nas miesiąc jest zapisany w głowie listy. Jeśli tak, mamy już odpowiedź. Jeśli nie, sprawdzamy czy miesiąc występuje w ogonie listy. Oznacza to, że za każdym razem sprawdzamy głowę ogona listy, aż dojdziemy do listy pustej - wtedy nasze poszukiwanie kończy się negatywnie.

Porównywanie rekurencyjne. Prolog zawiera predykaty do porównywania liczb całkowitych, ale porównywanie struktur jest już znacznie bardziej skomplikowane, bo porównywać trzeba poszczególne składniki tych struktur. Wobec tego, że owe składniki mogą być strukturami, porównywanie musi być wykonywane rekurencyjnie. Wyobraźmy sobie, że chcemy ustalić opłacalność eksploatacji różnych samochodów. W tym celu przemierzamy każdym z samochodów ustaloną trasę i mierzymy zużycie paliwa. Z każdym samochodem wiążemy listę liczb określających liczbę litrów paliwa zużytych na poszczególnych trasach. Aby porównać dwa samochody, musimy oczywiście zwracać uwagę na kolejność elementów listy, gdyż można porównywać tylko zużycie paliwa na tych samych trasach.

Wyjaśnić rolę akumulatora w Prologu. Podać przykład użycia akumulatora

Akumulator – argument zawierający wartość pośrednią, uzyskaną podczas analizy struktur.

Jego podstawą rolą, jaką pełni jest rola zapisywania tymczasowego danych. Przydatne do zapisywania wartości na liście bez powtórzeń. Pomaga unikać złączania struktur, co jest zasadniczo wolną operacją.

Funktory w Prologu. Przykłady użycia funktora *is* do budowy wyrażeń arytmetycznych.

Obiekty mogą być reprezentowane przez stałe lub termy złożone (struktury). Term złożony ma postać f(arg1,...,argn), gdzie f jest funktorem o arności n. Struktury w prologu zapisujemy podając funktor oraz jego składniki. Nazwa funktora odpowiada typom w innych językach programowania. Składniki umieszczamy w nawiasach okrągłych rozdzielając je przecinkami.

`posiada(maria,ksiazka(wichrowe_wzgorza,autor(emily,bronte))).`
książka to struktura zawierająca w sobie tytuł i kolejną strukturę autor która zaś składa się z imienia i nazwiska. Teraz możemy zadać pytanie: Czy Maria posiada jakąkolwiek książkę autora o nazwisku Bronte? I jaki jest jej tytuł? `?- posiada(maria, ksiazka(X,autor(_,bronte))).`

W Prologu nie można w sposób ,,bezpośredni'' wykonywać obliczeń arytmetycznych. Służy do tego predykat is. Przykłady: `?- X is 2 + 2.`

Systemy Baz Danych

Składnia języka PL/SQL – podstawowe instrukcje

Schemat budowy bloku:

```
[<Block header>]
[declare
     <Constants>
     <Variables>
     <Cursors>
     <User defined exceptions>]
begin
     <PL/SQL statements>
     [exception
          <Exception handling>]
end;
```

Schemat budowy procedury:

```
create [or replace] procedure <procedure name> [(<list of
parameters>)] is
     <declarations>
```

```
begin
      <sequence of statements>
      [exception
            <exception handling routines>]
end [<procedure name>];
```

Instrukcja warunkowa

```
IF warunek THEN instrukcje;
[ELSIF warunek THEN instrukcje; ]
[ELSIF ...]
[ELSE instrukcje; ]
END IF;
```

Pętla

```
LOOP
      instrukcje
      EXIT lub EXIT WHEN warunek;
END LOOP;
```

Pętla while

```
WHILE warunek LOOP
      instrukcje;
END LOOP;
```

Pętla for

```
FOR licznik IN [REVERSE] dolna_granica..gorna_granica LOOP
      instrukcje;
END LOOP;
```

Instrukcja GOTO:

```
...
GOTO gdzies; -- skocz do instrukcji za etykieta
...
<gdzies> -- etykieta
...
```

Instrukcja pusta: `NULL;`

Pętla for z kursorem

```
declare
cursor c1 is ...; //po is należy wstawić jakies zapytanie SQL np:
SELECT imie FROM pracownicy
rekord c1%ROWTYPE;
begin
for rekord in c1 loop
      ...
end loop;
end;
```

System zarządzania bazą danych – podstawowe elementy i funkcje jądra

System zarządzania bazą danych danych (SZBD – ang. database managment system, DBMS) jest zorganizowanym zbiorem narzędzi umożliwiającym dostęp i zarządzanie jedną lub więcej BD, jest powłoką która otacza bazę danych i za jej pomocą dokonuje się wszystkie operacje na BD. Wszystkie żądania użytkowników o dostęp do bazy danych (wyszukiwanie informacji, aktualizacja danych w plikach i tabelach itd.) są obsługiwane przez DBMS. Innymi słowy.

Wyróżniamy trzy warstwy SZBD(ang. DBMS):

1. Jądro – centralna część systemu która realizuje niżej wymienione funkcje oraz zarządza operacjami współdzielenia danych między wielu użytkowników zwłaszcza współbieżnością.
2. Interfejs między jądrem a zestawem narzędzi – standardowy język umożliwiający komunikację z podstawowymi funkcjami zarządzania danymi. Jest to zwykle wariant języka baz danych SQL;
3. zestaw narzędzi – oprogramowanie zarządzania zapytaniami i administrowania bazami danych, generatory aplikacji itd. (może być dostarczenie przez producenta – duży zestaw narzędzi albo wbudowanie jako część SZBD);

Każda z tych trzech warstw może być umieszczona na innym komputerze (oprogramowanie jądra może znajdować się na jednym komputerze, a różne programy wchodzące w skład zestawu narzędzi mogą być umieszczone na innych – architektura klient – serwer);

Podstawowe funkcje jądra:

- organizacja plików;
- mechanizmy dostępu;
- zarządzania transakcjami (kontrola współbieżności i spójności);
- zarządzanie słownikami;
- zarządzanie zapytania;
- sporządzanie kopii zapasowych (backup i odtwarzanie);
- przy organizacji plików w pamięci dyskowej należy uwzględniać minimalną liczbę rekordów, które muszą być odczytane (uaktualnianie) w celu wykonania podstawowych operacji na plikach:
- odszukanie określonego rekordu;
- wprowadzanie nowego rekordu;
- modyfikacja rekordu;
- usunięcie rekordu;
- organizacja plików;

Model fizyczny bazy danych – rodzaje plików

Model fizyczny bazy danych – opisuje rozmieszczenie danych w pamięci komputera (sposób organizacji danych w pamięci).

Organizacja plików - sposób w jaki układane są dane w fizycznych urządzeniach przechowywania danych powiązanym metodami dostępu do danych:

- **pliki nieuporządkowane** - rekordy są ustawianie w pliku w porządku ich wstawiania, wstawianie jest więc bardzo sprawne, wyszukiwane zaś wymaga liniowego przeszukiwania całego pliku, rekord po rekordzie;
- **pliki sekwencyjne** - rekordy są uporządkowane g. Wartości jednego lub wielu pól w praktyce identyczny to zazwyczaj klucza głównego.
    - Rekordy są umieszczane sekwencyjne kolejnych stronach;
    - wstawianie rekordów wymaga większej ilości przetwarzania;
    - żaden rekord nie może znajdować się na dwóch różnych stronach;
    - strony są powiązane listą za pomocą wskaźników lub ich adresy zapisuje się w tablicy przechowywanej niezależnie;
    - wyszukiwanie odpowiedniego rekordu wymaga sekwencyjnego przeglądania wszystkich rekordów;
    - usuwanie rekordu realizuje się za pomocą znacznika zajętości umieszczonego w nagłówku rekordy lub strony;
- **pliki haszowane** dostarczają bardzo szybkiego dostępu do rekordów na postawie określonego kryterium, muszą być zadeklarowane za pomocą tzw. klucza haszowania.

Wstawianie rekordów do pliku oznacza, że klucz rekordu K jest przekazywany do funkcji haszującej.

- **pliki indeksowe** – podstawowa idea polega na zastosowaniu dodatkowego pliku o dwóch polach i dodawanego do SBD.

Model fizyczny bazy danych – rodzaje, istota i znaczenie indeksów

Indeks – jest specjalnym plikiem, o postaci (wartości klucza k, adres strony s) stworzonym do wcześniej powstałego pliku W uporządkowanego wg rosnących lub malejących wartości klucza. Dana para wartości (k,s) występująca w indeksie oznacza, że pierwszy rekord pamiętany na stronie s pliku W ma wartość k.

Istnieje wiele metod przeszukiwania indeksu w celu odszukiwania adresu odpowiedniej strony. Najważniejsze to przeszukiwanie:

- sekwencyjne z pamięci zewnętrznego są pobierane kolejne strony indeksu, aż do momentu odnalezienia odpowiedniego adresu strony;
- binarny algorytm wyszukiwania - wykorzystywana jest fakt uporządkowania par (k,s) wg wartości klucza.

Główny problem polega na utrzymanie odpowiednio małego indeksu (tak aby mógł być przechowywany w pamięci głównej);

Rodzaje indeksów:

- Indeks gęsty – indeks w którym są pamiętane klucze do wszystkich rekordów pliku;
- Indeks rzadki - indeks w którym pamiętane tylko klucze do rekordów umieszczonych np. jako pierwsze na stronie pamięci zewnętrznej;
- Szczegółowym, a zarazem najczęściej stosowanym indeksem jest indeks o strukturze B-drzewa

Indeksy zwiększają efektywność zapytań, a zatem jego głównym zadaniem jest przyśpieszenie procesu wyszukiwania danych.

Zarządzanie transakcjami – właściwości, współbieżność, metody blokowania.

Właściwości transakcji:

- niepodzielność (atomowość, albo cała transakcja zostanie wykonana albo w ogóle nic – nie może być wykonana częściowe);
- spójność (wszystkie transakcje muszą zachować spójność i integralność BD).
- izolacja (wynikiem transakcji modyfikującej dzielone dane mogą być tymczasowo niespójne dane. Takie dane muszą być niedostępne dla innych transakcji dopóty, dopóki transakcja nie zakończy ich używać. Manager transakcji musi być więc dostarczyć iluzji, że dana transakcja działa w izolacji od innych transakcji);
- trwałość (gdy transakcja skończy się, nawet w wypadku awarii sprzętu lub oprogramowania, wówczas zmiany dokonania przez nią powinny zostać w pełni utrwalone).
- współbieżność. Ze względu na efektywność SZBD jest ważne, aby kilka transakcji było wykonywanych współbieżne operacje: odczyt – zapis w odpowiednich obszarach pamięci oper. wartości pobrany z BD; zapis – wartość z pamięci oper. jest przepisywana do b.d., w miejscu na aktualnej już wartości z bazy.

Sytuacja konfliktowa wystąpi wtedy, gdy dwie transakcje T1 i T2 są zainteresowane dostępem do tego samego obiektu A. Mamy wtedy do rozważenia 4 możliwości grup.

- odczyt-odczyt – współdzielenie danych – obie transakcje żądają odczyt wartości A. Każda uzyskana ta sama wartość. Nie ma hasła.

- Zapis-zapis – Utrata aktualizacji T1 i T2 dokonają zmian wartości pewnej danej przed zakończeniem transakcji T1, nie uwzględniając aktualizacji dokonanej przez T1.
- zapis-odczyt – odczyt niewłaściwy transakcje T2 odczytuje wartość zmienioną przez T1 po czym transakcja T1 zostaje anulowana;
- odczyt-zapis – odczyt niepowtarzalny transakcja T2 wykonuje kilkakrotnie odczyt wartości pewnego obiektu A. Jeśli jednak między kolejnymi odczytami wartości a inną transakcja dokona zmiany tej wartości, transakcja T2 uzyska dwie różne wartości A.

Podstawowym problemem związanym ze współbieżnym wykonywaniem transakcji jest określenie takiego porządku wykonania akcji poszczególnych transakcji, przy którym jest zachowana spójność BD. Poprawne uporządkowanie akcji współbieżnie wykonywanych transakcji rozwiązywane jest za pomocą metod, które można podzielić na trzy grupy:

- metody porządkowania wg etykiet;
- metody walidacji;
- metody blokowania.

Metody blokowania: W metodzie blokowania, z każdą daną x jest związana blokada; Zakładanie blokad danych, do których transakcja żąda dostępu, eliminuje dostęp do nich przez inne transakcje w czasie, gdy ograniczenia integralnościowe mogą być przejściowo naruszone. Zakłada się, że każda transakcja przed odczytem lub zapisem danej musi założyć odpowiednią blokadę na tej danej oraz, że wszystkie blokady założone przez transakcję zostaną zdjęte przed zakończeniem jej wykonania. Wyróżniamy dwa podstawowe typy blokad:

- blokadę współdzielona – (shared lock) – powinna być zakładana na dane przez operację nie powodującą jej uaktualnienia (odczyt);
- blokadę wyłączną – (exclusive lock) – powinna być zakładana na dane przez operację powodującą jej uaktualnienie (zapis);

Dwie blokady są zgodne, jeżeli mogą być jednocześnie założone na tę samą daną przez dwie różne transakcje.

Hierarchiczna metoda blokowania konieczność rozwiązywania tzw. problemu ziarnistości blokad – polega ona na, doborze rozmiaru blokowanej jednostki dla której transakcje żądające dostępu będą efektywnie. BD jest widziana jako hierarchiczne jednostek blokowania zwanych ziarnami. Hierarchiczną ziaren można przedstawić w postaci acyklicznego grafu skierowanego, którego wierzchołki odpowiadają ziarnom natomiast leki określają relację zawieranie się jednych ziaren w drugich;

Przykład:

- transakcja T1 blokuje jedną krotkę relacji R1 w trybie wyłącznym;
- transakcja T2 żąda również dostępu do tej krotki w trybie wyłącznym ze względu na niezgodność blokad wyłącznych żądanie transakcji T2 zostanie odrzucone;
- transakcja T2 mogłaby założyć blokadę całej relacji R1 blokując w ten sposób również jej krotki /naruszonego by to warunek niezgodności blokad wyłącznych.

W celu zapewnienia poprawności hierarchicznego blokowania danych należy, zatem zagwarantować, że po założeniu przez transakcję blokady podstawowej danego wierzchołka R w hierarchii ziaren żadna inna transakcja nie uzyska zgodnie z nią blokady żadnego wierzchołka będącego poprzednikiem R. Jednym z możliwych rozwiązań jest wprowadzenie blokad intencjonalnych – żądanie blokady intencjonalnej na danym.

Blokady intencjonalne (GREY)

- Intencjonalna blokada współdzielona –pozwala na współdzielony dostęp do danych należących do wierzchołka R i na zakładanie blokady typu IS lub typu S, wszystkich następników tego wierzchołka w hierarchii ziaren;

- Intencjonalna blokada wyłączna – pozwala za – równo na dostęp współdzielony, jak i wyłączny do danych należących do wierzchołka R, i na zakładanie blokady typu IS, X, IS, S i SIX wszystkich następników tego wierzchołka w hierarchii ziaren;
- Blokada mieszana - połączenie podstawowej blokady współdzielonej i intencjonalnej blokady wyłącznej – SIX. Pozwala na dostęp współdzielony do danych na – leżących do wierzchołka R, i do wszystkich jego na – następników oraz pozwala na zakładanie blokad wyłącznych na nich.

Niezawodność i odtwarzanie bazy danych – model funkcjonalno-logiczny

Niezawodność BD i odtwarzanie po awariach sprzętowych i programowych - szczególną grupę stanowią transakcje, których wykonywanie zostało przerwane awarią systemu komputerowego – powinny one zostać wycofane w sposób automatyczny przez SZBD i ewentualnie zainicjowane po usunięciu awarii i odtworzeniu spójnego stanu BD.

Mechanizmy odtwarzania spójnego stanu bd sprzed awarii muszą uwzględniać:

- różne typy pamięci w których przechowuje się dane: pamięć operacyjną; pamięć zewnętrzną o dostępie bezpośrednim; pamięć zewnętrzną archiwizacji.
- różne typy awarii (awarie po których można odtworzyć stan poprzedni, i awarie powodujące zmiany nieodwracalne).
- poszczególne poziomy zabezpieczeń awaryjnych:
    - poziom fizyczny (pełne strony danych przechowywane w pamięciach dyskowych);
    - poziom logiczny (stan BD i zmieniające je transakcje).
- metody przywracania zgodnego stanu BD po awarii:
    - obrazy przed i po transakcji;
    - kronikowanie wprowadzonych modyfikacji, tworzenie kopi przechowywanych w pamięciach dyskowych lub archiwizacja (backup) na taśmie magnetycznej;
    - aktualizacja odroczona (lazy update);
    - okresowe składowanie BD.

BD jest przechowywana na dysku w tzw. segmentach seg n. Każdy segment składa się ze stron, będących jednostkami przydziału pamięci dyskowej i transmisji informacji do/z jednostki centralnej.

Strony są numerowane od 1 do q. Zbiór stron przydzielonych do segmentu **seg i** jest opisany przez mapę segmentu **msg** i zawierające numery fizyczne stron. Na dysku przechowywana jest również mapa stron **ms** składająca się z q elementów/ jeden element na stronie/ i informacja o tym czy strona jest wolna (0) czy przydzielona (1) do jednego z segmentów.

Mapy segmentów **msg i** , podobnie jak inne dane, są pamiętane na stronach, które w miarę potrzeby są wprowadzane do pamięci operacyjnej.

W pamięci operacyjnej można wyróżnić dwa szczególne bufory

- bufor stron map **(bsm);**
- bufor stron danych **(bsd).**

W celu sprowadzenia do pamięci operacyjnej strony i segmentu **seg k** należy wykonać następujące operacje :

- sprowadzić do bufora **bsm** stronę mapy **mseg k** , który zawiera i – ty element;
- określić adres j i – tej strony segmentu **seg k** ;
- sprowadzić do bufora **bsd** stronę o numerze j.

Optymalizacja zapytań.

proces optymalizacji: Mechanizm tłumaczenia zapytania nazywamy optymalizatorem zapytania. Istnieją dwie klasy optymalizatorów:

- optymalizatory oparte na składni heurystyce – wybierają one plan wykonania na podstawie składni instrukcji SQL (np. postać i kolejność warunków w klauzuli where – oznacza to, że dla

tego samego zapytania wyrażonego na dwa nieznacznie odmienne sposoby możemy uzyskać zupełnie inny czas wykonania);
- optymalizatory oparte na składni statystyce podają na początku zapytanie wstępnej obróbce – wynikiem tego etapu jest konwersja zapytania na postać bazową lub kanoniczną (to oznacza, że dwa różne składniowo, ale semantycznie podobne zapytania, wprowadzone do tego optymalizatora zostaną przekształcone w tę samą postać bazową). optymalizator wybiera następnie plan wykonania opierając się na definicjach tabel, kolumn i indeksów w słowniku danych, biorąc również pod uwagę statystyki, np. rozmiary plików.

Architektura bazy danych ORACLE – wewnętrzna struktura i obszary pamięci, oraz procesy drugoplanowe

Wew. Obiekty BD Oracle:

- Tabele i typy danych, kolumny, więzy integralności oraz typy danych – możliwe jest def. Własnych typów danych oraz specjalnego typu danych REF, umożliwiającego powiązanie obiektów o różnych typach;
- Partycje i podpartycje – dzielenie tabeli na kilka wierszy;
- Schemat użytkownika – zbiór obiektów, których właścicielem jest dany użytkownik. Każdy użytkownik powinien mieć konto;
- Typ danych użytkownika – obiekt, który można „wklejać" w inny obiekt, perspektywy.
- indeksy, klastry, klastry haszowane. Indeks(gęsty, rzadki) – dwukolumnowa tablica (w sensie logicznym).
- Perspektywy – replika lokalnej BD;
- Sekwencja – obiekt, można z niego korzystać, przemieszczać go, jest to lista kolejnych niepowtarzalnych liczb części kodu, definicje sekwencji przechowywane są w słowniku danych
- Powiązania baz danych – BD Oracle może korzystać z tych danych, które są poza lokalną BD. W celu uzyskania ścieżki dostępu do obiektu w odległej BD należy stworzyć powiązanie między Bazami Danych;
- Segmenty, ekstanty(ciągłe obszary bloków) oraz bloki danych. Segment to fizyczny obiekt, składa się z obszarów. Obszary składają się z bloków;
- Segmenty wycofania – utrzymują spójność BD np. żeby można było powrócić po awarii, do stanu sprzed awarii;
- Migawki - zapytania korzystają z odległej BD, jest to perspektywa, która dokumentuje stan odległej BD.
- Procedura – nie zwraca wartości do wywołującego ją programu
- Funkcja – zwraca wartość do wywołującego ją programu
- Wyzwalacze – wyzwalacze instrukcji, wyzwalanie na poziomie wiersza
- Synonimy – całkowity identyfikator obiektu (nazwa komputera, nazwa instancji, nazwa właściciela obiektu, nazwa obiektu
- Tabele systemowe – przechowują dane o tabelach

Wewnętrzne obszary pamięci:

- bufor bazy danych – przechowuje bloki danych – jest on częścią SGA (tzw. Obszar Globalny Systemu);
- bufor dziennika powtórzeń – zapisywane są pliki z dziennika powtórzeń;
- dzielony obszar SQL (hierarchia zapytań SQL) – zawiera plan wykonania oraz zestaw przeanalizowanych instrukcji SQL:
- bufor słownika danych – znajdują się tu tabele słownikowe;
- bufor biblioteczny
- wielki obszar (usprawnieni procesu archiwizacji);
- obszar JAVA;

- zwielokrotnione pole buforów;
- obszar kontekstu (obszar SQL);
- globalny obszar programu

Instancja jest to grupa buforów (tzw. Obszar Globalny Systemu – SGA) wykorzystywana do obsługi bazy oraz szereg procesów drugoplanowych (programów rezydujących w pamięci operacyjnej komputera pracującego jako serwer Oracle), wykonujących wszystkie czynności związane z obsługą bazy i jej użytkowników. Zwykle dla jednej bazy podniesiona jest jedna instancja. Możliwe jest jednak otwarcie wielu instancji pracujących na danych jednej bazy przy użyciu opcji Serwera Równoległego.

Procesy drugoplanowe są wywoływane z poziomu administratora BD. Obsługują i wymuszają połączenia między fizycznymi i logicznymi strukturami bazy danych. Każdy z procesów drugoplanowych tworzy plik śladu, który jest zapisywany w czasie pracy instancji. Nazwa tego pliku zawiera nazwę procesu oraz identyfikator procesu z poziomu SO. Pliki śladu są tworzone, aby była możliwość określenia wystąpienia błędu (można sprawdzić co było przed błędem).

Konfiguracja sprzętowa i logiczny model (układ) bazy danych ORACLE

Przegląd architektury

- używanie obszarów pamięci ma na celu poprawę wydajności działania BD, usprawnienie jej zarządzania oraz umożliwienie korzystania z tych samych danych przez wielu użytkowników;
- odczytywanie i zapisywanie danych pomiędzy sga a plikami danych w bazie zachodzi za pomocą zestawu procesów drugoplanowych.
- serwer bazy danych nazywany także instancją składa się z zestawu obszarów pamięci oraz procesów drugoplanowych. Mają one dostęp do plików danych.
- Baza danych składa się z fizycznych plików, obszarów oraz procesów. Rozmieszczenie tych elementów różni się zależnie od wybranej architektury.
- Parametry serwera określane są podczas uruchamiania serwera.

Autonomiczne hosty

- Pojedyncza baza danych na autonomicznym serwerze. Najprostszą strukturą bazy danych jest konfiguracja złożona z pojedynczego serwera, pojedynczej bazy danych oraz z jednego dysku twardego .
- Pojedyncza baza danych na autonomicznym serwerze z kilkoma dyskami. Rozdzielenie plików poprawia wydajność bazy danych poprzez zmniejszenie rywalizacji operacji wej – wyj o pliki bazy danych.

Autonomiczne hosty z zestawem dysków

- kopia lustrzana plików sterujących – baza danych oracle tworzy ją automatycznie
- kopie lustrzane plików dziennika powtórzeń
- kopia lustrzana zarchiwizowanych plików dziennika

Autonomiczne hosty z opcją powielania dysków. Wiele systemów operacyjnych umożliwia wykonanie zsynchronizowanych kopii plików za pomocą procesu zwanego powielaniem dysku lub powielaniem wolumenu - znany jest on również jako mirroring. Pierwsza korzyść – zestaw powielonych dysków stanowi kopię zapasową w przypadku awarii dysku; Druga korzyść - poprawa wydajności pracy bazy danych. Jeśli system operacyjny nie obsługuje asynchronicznych zapisów, proces powielania dysków na poziomie systemu operacyjnego może powodować obniżenie wydajności systemu.
Autonomiczne hosty z wieloma bazami danych: możliwe jest utworzenie kilku baz danych na jednym komputerze. Każda BD posiada wtedy osobny zestaw plików i każda jest obsługiwana przez inny serwer danych.

Hosty sieciowe:

- połączone bazy danych

- serwery klastrowe – serwer równoległy
- aplikacje typu klient – serwer
- architektura trójwarstwowa
- bazy danych typu standby

Logiczny układ bazy danych: Standardowa struktura zwana optymalną elastyczną architekturą:

- punkt startowy – przestrzeń tabel system; w Oracle stanowi to odpowiednik katalogu głównego; powinny się tam znajdować tylko tabele słownika danych oraz systemowy segment wycofania;
- oddzielenie segmentów danych – przestrzeń tabel DATE;
- oddzielenie segmentów indeksowych – przestrzeń tabel INDEXES; zaleca się oddzielenie ich od segmentów słownika danych z przestrzeni tabel SYSTEM;
- oddzielenie segmentów dla narzędzi – przestrzeń tabel TOOLS;
- oddzielenie segmentów wycofania – przestrzeń tabel RBS;
- oddzielenie segmentów tymczasowych – przestrzeń tabel TEMP; nie można w niej przechowywać żadnych stałych obiektów, takich jak tabele czy indeksy;
- oddzielenie użytkowników – przestrzeń tabel USERS;

Każdy projekt struktury logicznej bazy danych powinien spełniać następujące wymagania:

- segmenty tego samego typu powinny być przechowywane razem;
- system powinien być zaprojektowany z uwzględnieniem specyfiki pracy bazy danych/rozmiar transakcji, liczba użytkowników, liczba transakcji itd./;
- dla obiektów o dużej ważności powinny istnieć oddzielne obszary;
- rywalizacje operacji wej – wyj o przstrzenie tabel powinny być zminimalizowane;
- słownik danych powinien być oddzielony od reszty obiektów bazy danych

Ewolucja technologii dostępu do baz danych

Historycznie **ODBC** (ang. Open DataBase Connectivity - otwarte łącze baz danych) był chyba pierwszą próbą ujednoliconego interfejsu dostępu do baz danych. **ODBC** jest interfejsem pozwalający programom łączyć się z systemami zarządzającymi bazami danych. Jest to API niezależne od języka programowania, systemu operacyjnego i bazy danych.

JDBC (ang. Java DataBase Connectivity - łącze do baz danych w języku Java) interfejs umożliwiający niezależnym od platformy aplikacjom napisanym w języku Java porozumiewać się z bazami danych za pomocą języka SQL. Interfejs ten jest odpowiednikiem zaprojektowanego przez Microsoft łącza ODBC. Istnieje także uniwersalny sterownik JDBC-ODBC umożliwiający obsługę każdej bazy, do której opracowano sterownik ODBC.

OLE DB, rozwiązanie opracowane przez Microsoft m.in. po to, by ujednolicić sposób obsługi relacyjnych i nierelacyjnych źródeł danych. Jest to obiekt COM, który funkcjonuje w podobny sposób jak ODBC, ale w odniesieniu do dowolnego źródła danych, a nie tylko baz danych SQL.

W **.Net** Microsoft zdecydował się wprowadzić zupełnie inny sposób dostępu do baz danych. Poprzednie interfejsy starały się ujednolicać cechy wielu baz danych, teraz interfejs bazy tylko ''wypełnia'' zestaw danych DataSet. Następnie program pracuje na ''rozłączonym'' zestawie danych oryginalnych, a na koniec, w razie potrzeby, .Net Provider wykonuje ciąg poleceń, aktualizując dane w bazie.

Systemy Operacyjne

Przedstaw dwa główne modele komunikacji międzyprocesowej w systemie komputerowym

Model przekazywania komunikatów. Przed rozpoczęciem komunikacji należy nawiązać połączenie (funkcja systemowe odebranie połączenia). Musi być znana nazwa odbiorcy: inny proces w tym samym procesorze lub proces w innym komputerze. Proces odbiorcy musi zazwyczaj udzielić zgody na nawiązanie komunikacji za pomocą funkcji akceptującej połączenie. Większość procesów realizujących połączenia to tzw. demony, które wywołują czekania na połączenie i są budzone gdy połączenie zostanie nawiązane. Źródło komunikacji (klient) i demon odbiorcy (server) wymieniają komunikaty za pomocą funkcji systemowych czytania komunikatu i pisania komunikatu. Wywołanie funkcji zamknięcie połączenia kończy komunikację. Przesyłanie komunikatów jest przydatne do bezpośredniej wymiany mniejszych ilości danych oraz w komunikacji międzykomputerowej.

Model pamięci dzielonej. Wspólne użytkowanie przez procesy pewnego obszaru pamięci. Procesy wymieniają informację przez pisanie i czytanie do wspólnie użytkowanych obszarów pamięci. Procesy określają postać danych i ich miejsce w pamięci wspólnej; nie podlega to kontroli SO. Procesy muszą dopilnować, aby nie zapisywać jednocześnie tego samego miejsca pamięci. Pamięć dzielona zapewnia maksymalna szybkość i wygodę komunikacji, w obrębie jednego komputera może ona przebiegać z szybkością działania pamięci operacyjnej. Problemy z zakresu ochrony i synchronizacji.

Przedstaw problem producenta i konsumenta

Problem producenta i konsumenta to klasyczny informatyczny problem synchronizacji. W problemie występują dwa rodzaje procesów: producent i konsument, którzy dzielą wspólny zasób - bufor dla produkowanych (konsumowanych) jednostek. Zadaniem producenta jest wytworzenie produktu, umieszczenie go w buforze i rozpoczęcie pracy od nowa. W tym samym czasie konsument ma pobrać produkt z bufora. Problemem jest taka synchronizacja procesów, żeby producent nie dodawał nowych jednostek gdy bufor jest pełny, a konsument nie pobierał gdy bufor jest pusty.

Rozwiązaniem dla producenta jest uśpienie procesu w momencie gdy bufor jest pełny. Pierwszy konsument, który pobierze element z bufora budzi proces producenta, który uzupełnia bufor. W analogiczny sposób usypiany jest konsument próbujący pobrać z pustego bufora. Pierwszy producent, po dodaniu nowego produktu umożliwi dalsze działanie konsumentowi. Rozwiązanie wykorzystuje komunikacje międzyprocesową z użyciem semaforów. Nieprawidłowe rozwiązanie może skutkować zakleszczeniem.

Przedstaw algorytmy bez wywłaszczenia i z wywłaszczeniem planowania czasu procesora. Jaki algorytm jest optymalny?

Algorytmy bez wywłaszczania:

- FCFS (First Come First Served) — pierwszy zgłoszony, pierwszy obsłużony. Algorytm ten dokonuje najsprawiedliwszego przydziału czasu (każdemu według potrzeb), jednak powoduje bardzo słabą interakcyjność systemu - pojedynczy długi proces całkowicie blokuje system na czas swojego wykonania, gdyż nie ma priorytetów zgodnie z którymi mógłby zostać wywłaszczony.
- LCFS (Last Come First Served) — ostatni zgłoszony, pierwszy obsłużony.
- SJF (SJN, SPF, SPN, Shortest Job/Process First/Next) — najpierw najkrótsze zadanie.

Algorytmy z wywłaszczeniem:

- Planowanie rotacyjne (Round Robin, RR) — każde z zadań otrzymuje kwant czasu; po spożytkowaniu swojego kwantu zostaje wywłaszczone i ustawione na końcu kolejki;
- SRT (Shortest Remaining Time) — najpierw zadanie, które ma najkrótszy czas do zakończenia.
- SJF (SJN, SPF, SPN, Shortest Job/Process First/Next) — najpierw najkrótsze zadanie.

Algortm SJF może być wywłaszczający lub niewywłaszczający. Konieczność wyboru powstaje wówczas, gdy w kolejce procesów gotowych, podczas, gdy procesor jest zajęty innym procesem, pojawia się proces o krótszej fazie procesora, niż to co jeszcze zostało do wykonania w procesie bieżącym. Algorytm SJF - wywłaszczający usunie wówczas, bieżący proces do kolejki procesów gotowych i przydzieli mu proces o krótszej fazie z kolejki. **Jest to algorytm optymalny** ze względu na najkrótszy średni czas oczekiwania. Problemem tego algorytmu jest głodzenie długich procesów - może się zdarzyć, że cały czas będą nadchodzić krótsze procesy, a wtedy proces dłuższy nigdy nie zostanie wykonany.

W jaki sposób przewidywany jest czas trwania następnych faz zapotrzebowania procesu na procesor

Można tylko oszacować długość następnej fazy procesora. Można to zrobić na podstawie znanych już poprzednich faz procesora, używając średniej wykładniczej pomiarów poprzednich faz.

Wyjaśnij pojecie wiązanie adresów. W jaki sposób i kiedy ono następuje?

Adres wytworzony przez procesor w wyniku wykonania rozkazu programu nosi nazwę adresu logicznego (ang. logical address). Zbiór wszystkich adresów logicznych generowanych przez program tworzy logiczną przestrzeń adresową procesu. Adres fizyczny wskazuje konkretną komórkę pamięci operacyjnej. Zbiór wszystkich adresów fizycznych tworzy fizyczną przestrzeń adresową.

Wiązanie adresów polega na odwzorowaniu adresów logicznych generowanych w programie na adresy fizyczne w pamięci operacyjnej. Może następować w dowolnych z poniższych etapów:

- w czasie kompilacji, czyli tworzenia programu,
- w czasie ładowania programu do pamięci,
- w czasie wykonania programu.

Wiązanie **podczas kompilacji** wymaga dokładnej znajomości początkowego adresu, pod którym program zostanie załadowany do pamięci. Tylko w takiej sytuacji program może posługiwać się bezpośrednio adresami bezwzględnymi w pamięci. W czasie wykonywania programu adresy logiczne i fizyczne są już takie same. Wiązanie podczas kompilacji można zastosować jedynie w prostych systemach przeznaczonych dla jednego użytkownika (np. MS-DOS).

Jeśli początkowy adres w pamięci nie jest znany w czasie tworzenia programu, to wiązanie adresów musi być opóźnione do momentu **ładowania programu do pamięci**. Po załadowaniu program nie może być już przemieszczany w pamięci podczas wykonywania, ponieważ adresy logiczne i fizyczne są już takie same. Zmiana adresu początkowego pociąga za sobą konieczność ponownego załadowania całego programu, począwszy od nowego adresu w pamięci.

Największe możliwości swobodnego przemieszczania programu w pamięci daje wiązanie adresów **w czasie wykonywania programu**. Adresy logiczne muszą być jednak tłumaczone na bieżąco na adresy fizyczne, co wymaga zastosowania specjalnego sprzętu. Zajmuje się tym jednostka zarządzania pamięcią (ang. memory management unit - MMU). Złożoność tej jednostki warunkuje wybór strategii zarządzania pamięcią.

Czym jest fragmentacja i jak sie z nią walczy?

Fragmentacja – niekorzystne zjawisk, zachodzące w systemie plików, polegające na pojawianiu się nieciągłości obszarów zapisanych i niezapisanych na dysku twardym komputera. Ich przyczyną jest niedoskonała budowa systemu plików, usuwanie i dodawanie nowych plików oraz zmiany, polegające na dopisywaniu i skracaniu plików już istniejących. Fragmentacją określa się także zapisywanie poszczególnych plików na oddalonych od siebie sektorach, co powoduje duże opóźnienia w czasie ich odczytu lub zapisu. Najpoważniejszym skutkiem fragmentacji systemu plików jest spadek wydajności operacji na plikach. Duża fragmentacja zmusza do częstej relokacji głowicy dysku twardego, co jest

czasochłonne. Fragmentacja zmniejsza także prawdopodobieństwo odzyskania danych z dysku w wypadku awarii systemu plików.

Defragmentacja – operacja układająca pliki na dysku komputerowym tak, aby system miał do nich szybszy dostęp. Najczęściej jest to poukładanie bloków jednego pliku po kolei, kiedy są one rozrzucone po całej partycji.

Niektóre systemy plików takie jak HPFS, NTFS, ext4 nie muszą być często defragmentowane, gdyż są zaprojektowane tak, aby minimalizować fragmentację danych. Dzięki defragmentacji zwiększa się wydajność systemu plików.

Przestaw schemat zarządzania pamięcią nazywany segmentacja

Segmentacja - schemat zarządzania pamięcią operacyjną oparty na wyobrażeniu użytkownika o tym czym jest program/pamięć. Program, z punktu widzenia użytkownika to zbiór segmentów, będących logicznymi jednostkami, takimi jak program główny, podprogram, funkcja, itd. Każdy segment ma nazwę (numer) oraz długość; adres logiczny odwołania do segmentu tworzy więc parę: (numer-segmentu (s), odległość (o)). Należy zdefiniować implementację odwzorowującą 2-wymiarowe adresy określone przez użytkownika na 1-wymiarowe adresy fizyczne. Takie odwzorowanie daje tablica segmentów; pozycja tablicy segmentów składa się z bazy segmentu i granicy

Ochrona i wspólne użytkowanie. Zaletą segmentacji jest powiązanie ochrony pamięci z segmentami (segmenty rozkazów, segmenty danych). Sprzęt odwzorowujący będzie sprawdzał bity ochrony związane z każdą pozycją tablicy segmentów, kontrolujące read/write/execute status segmentu.

Przedstaw algorytm zastępowania stron w pamięci wirtualnej. Jaki algorytm jest optymalny?

Istnieje wiele opracowanych propozycji algorytmów zastępowania stron przy użyciu pamięci wirtualnej. Sam proces zastępowania stron jest potrzebny ze względu na to, że zazwyczaj wielkość pamięci operacyjnej (a właściwie ilość dostępnych ramek) jest mniejsza od wielkości pamięci wirtualnej na dysku. Jeśli wiele procesów używa pamięci przy wsparciu pamięci wirtualnej, to musi istnieć mechanizm, który ładuje część danych z dysku do pamięci operacyjnej na miejsce innych danych – więc zastępowanie.

Jednym z najprostszych algorytmów jest algorytm oparty o FIFO – zastępowana jest ta strona w pamięci, która jest najstarsza – tj. została umieszczona jako pierwsza w pamięci. Algorytm ten nie działa zbyt dobrze, gdyż może się okazać, że pomimo faktu, że strona jest najstarsza, jest ciągle silnie używana, co spowoduje, że niemal natychmiast po jej zastąpieniu pojawi się żądanie jej załadowania, co nie jest rozwiązaniem optymalnym.

Inny algorytm jest określany jako LRU – Last Recently Used. Algorytm ten zastępuje stronę, która jest najrzadziej używana. Określenie ostatniego użycia może być wykonywane za pomocą dwóch środków (zegara systemowego oraz stosu). Pełna implementacja LRU jest dość kosztowna, więc stosuje się czasem pewne modyfikacje.

Algorytm optymalny zastępuje te strony, które najdłużej będą nieużywane. Ze względu na to, że nie ma pewnej metody prognozowania użycia stron w pamięci, ten algorytm nie jest możliwy do zaimplementowania. Jest używany „po fakcie” jako punkt odniesienia do oceny innych algorytmów (np. żeby stwierdzić, że użyty algorytm jest ileś procent słabszy od teoretycznego algorytmu optymalnego).

Przedstaw zjawisko szamotania w systemach operacyjnych. Jakie są jego przyczyny i w jaki sposób zapobiegać jego powstawaniu?

Gdy wiele procesów rywalizuje o ramki w pamięci, algorytmy zastępowania stron można zakwalifikować do dwóch kategorii:

- Zastępowania globalnego – procesy mogą wybierać brakujące im ramki ze zbioru wszystkich ramek, nawet gdy ramka jest w danej chwili przydzielona do innego procesu; w ten sposób proces o wyższym priorytecie może zwiększać liczbę swoich ramek kosztem procesu o niższym priorytecie.
- Zastępowania lokalnego – liczba ramek przydzielonych procesowi nie zmienia się; proces wybiera do zastąpienia brakującą ramkę ze zbioru swoich ramek.

Proces, który ma w pamięci minimalną liczbę ramek i którego strony są aktywnie używane, spotyka się z problemem szamotania – szybko następuje brak strony i trzeba zastąpić jakąś stronę, która za chwilę okaże się potrzebna – proces szamoce się spędzając więcej czasu na stronicowaniu niż na wykonaniu. Powoduje to poważne zaburzenia wydajności. Przyczyna:

- System operacyjny nadzoruje wykorzystanie jednostki centralnej, w przypadku gdy jest zbyt niskie, to zwiększa się stopień wieloprogramowości wprowadzając nowy proces do systemu
- Strony zastępowane są wg globalnego algorytmu zastępowania stron bez brania pod uwagę do jakich procesów należą
- Planista przydziału procesora dostrzega spadek wykorzystania procesora, więc zwiększa stopień wieloprogramowości: Nowy proces aby wystartować zabiera ramki wykonywanym procesom, Gwałtowanie maleje przepustowość systemu

Sposoby zapobiegania:

- Zmniejszenie stopnia wieloprogramowości
- Zastosowanie lokalnego algorytmu zastępowania:
- Dostarczenie procesowi tyle ramek, ile potrzebuje

Przedstaw pojecie sekcji krytycznej. W jaki sposób rozwiązuje sie problemy związane z tym pojęciem?

Sekcja krytyczna - w programowaniu współbieżnym fragment kodu programu, w którym korzysta się z zasobu dzielonego, a co za tym idzie w danej chwili może być wykorzystywany przez co najwyżej jeden wątek. System operacyjny dba o synchronizację, jeśli więcej wątków żąda wykonania kodu sekcji krytycznej, dopuszczany jest tylko jeden wątek, pozostałe zaś są wstrzymywane. Dąży się do tego, aby kod sekcji krytycznej był krótki - by wykonywał się szybko. Sekcja krytyczna może być użyta ażeby zagwarantować, że wspólny zasób, na przykład drukarka, jest używana tylko przez jeden proces w określonym czasie. Sposób implementacji sekcji krytycznych jest zależny od systemu operacyjnego.

Sekcje krytyczne realizuje się np. z wykorzystaniem blokowania przerwań, muteksów lub semaforów. Brak wzajemnego wykluczania się wykonywania sekcji krytycznych może spowodować błędy wykonania, np. dwukrotne zapisanie danej albo niepoprawna modyfikacja zasobu.

Mutex (ang. Mutual Exclusion, wzajemne wykluczanie)

Semafor – jest chronioną zmienną lub abstrakcyjnym typem danych, który stanowi klasyczną metodę kontroli dostępu przez wiele procesów do wspólnego zasobu w środowisku programowania równoległego. Typowy semafor implementowany jest jako zmienna typu całkowitego. Semafory dzieli się na binarne i zliczające. Z drugiej strony semafor jest bezużyteczny w zapobieganiu zjawisku zakleszczenia, co ilustruje problem ucztujących filozofów.

Zintegrowane systemy informatyczne zarządzania

Opisać model zastosowań technologii informatycznej w organizacji

Można mówić o kilku etapach (poziomach) transformacji (jest to tzw. pięciofazowy model pozwalający świadomie stosować IT dla potrzeb biznesu):

- zastosowanie pojedynczych aplikacji IT. Obejmuje sferę działań dotyczącą wykorzystania technologii informacyjnej do wsparcia działalności firmy (organizacji) na poziomie operacyjnym.
- tworzenia systemów, których zasięg przekracza granice funkcji - wewnętrzna integracja. Spójne i usystematyzowane podejście do wykorzystania systemów informatycznych w całościowym zabezpieczeniu procesów biznesowych, a nie wydzielonych procesów funkcjonalnych. Etap ten to integracja w dwóch podstawowych obszarach: w obszarze technicznym i technologicznym (wspólna platforma programowo sprzętowa) oraz w obszarze organizacyjnym i procesowym.
- Reorganizacja procesów biznesowych. Wykorzystanie IT przynosi korzyści tylko wtedy, gdy towarzyszy mu fundamentalna zmiana organizacji.
- Reorganizacja sieci gospodarczej i zmiana filozofii działania. Etap ten charakteryzuje zmiana zasad (przebiegu) wymiany zasobów (dzięki zastosowaniu IT) pomiędzy uczestnikami wspólnej sieci gospodarczej.
- Zmiana zakresu działalności gospodarczej. Regułą jest dostarczanie na rynek rozwiązań o największej dla odbiorcy wartości. Będzie to możliwe jeśli organizacja zdoła ciągle udoskonalać własne, kluczowe kompetencje (w celu utrzymania przewagi konkurencyjnej), natomiast zlecać wykonanie pozostałych operacji tym zewnętrznym partnerom, którzy realizują to taniej i na wyższym poziomie.

Omów typologię i podstawowe standardy zintegrowanych systemów informatycznych zarządzania

System informatyczny zarządzania jest to część systemu informacyjnego realizowana przez techniczne środki informatyki, którego celem jest wspomaganie procesów zarządzania.

Podstawowe kryteria podziału **typologicznego SIZ** odnoszące się do podstawowych cechy oraz właściwości struktury i funkcjonowania przedsiębiorstwa:

- zakres merytoryczny SIZ (np. systemy cząstkowe, dziedzinowe, wielodziedzinowe, kompleksowe),
- zakres funkcjonalny (np. systemy ewidencji gospodarczej, analizy ekonomicznej, wspomagania procesów decyzyjnych),
- forma komunikacji użytkownika z systemem (np. systemy pośrednie i bezpośrednie),
- konstrukcja systemu (np. systemy jedno- i wielomodułowe),
- lokalizacja SIZ w strukturze organizacyjnej (np. systemy centrali, zarządu, dyrekcji, filii, zakładu),
- technologia przetwarzania danych (np. systemy z klasycznymi zbiorami danych, bazą danych, bazą wiedzy).

Oraz zakresu i formy wspomagania procesu zarządzania, w tym kryterium można wyodrębnić m.in. następujące subkryteria szczegółowe:

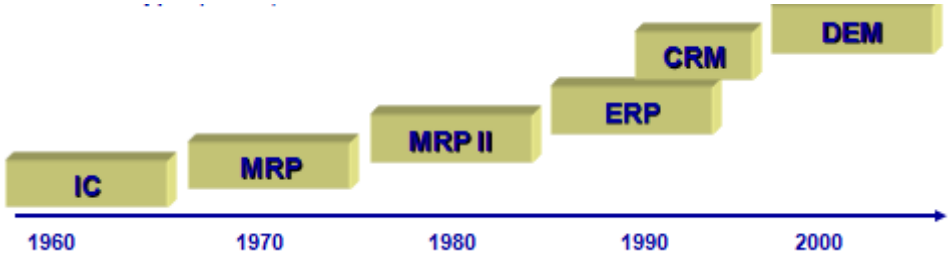
- złożoność organizacyjna obiektu gospodarczego (np. systemy dla przedsiębiorstw jedno- i wielozakładowych),
- obszar działalności (np. systemy regionalne, rządowe),
- różnorodność i intensywność interakcji z otoczeniem,
- rozproszenie terytorialne,
- złożoność informacyjna przedsiębiorstwa,
- przyjęty model zarządzania.

SIZ dzieli się na 5 generacji, w którym za kryterium przyjęto zawarty w nich zakres zasobów informacji oraz sposób i środki do ich wykorzystania:

1. Systemy transakcyjne – są zorientowane na ewidencję działalności gospodarczej organizacji, np. ewidencja zatrudnienia, płac, środków finansowych, itp.

2. Systemy informacyjne – są zorientowane na dostarczanie wyselekcjonowanych, zagregowanych i adresowanych informacji. Służą informowaniu wyższych szczebli zarządzania i noszą nazwę Systemów Informowania Kierownictwa,
3. Systemy wspomagania decyzji – umożliwiają w trybie konwersacyjnym identyfikację problemu decyzyjnego oraz tworzenie różnych wariantów rozwiązań prowadzących do wyboru optymalnej decyzji. Najczęściej są stosowane do podejmowania decyzji strategicznych, np. w dziedzinie planowania działalności gospodarczej, inwestycji.
4. Doradcze systemy decyzyjne (nazwane systemami ekspertowymi) – zawierają rozwiązania proceduralne, informatyczne i technologiczne prowadzące do znacznego ograniczenia lub eliminowania decydenta z procesu podejmowania decyzji
5. Zintegrowane Systemy Informatyczne Zarządzania (ZSI). Są to system zarządzania zorganizowany modułowo, obsługujące wszystkie sfery działalności przedsiębiorstwa, wszystkie zasoby danych, procedury zarządzania, sterowanie i regulacja procesów wytwórczych są przetwarzane przy wsparciu technologii informatycznej

Standardy Zintegrowanych Systemów zarządzania:



- IC (*Inventory Control*) - zarządzanie gospodarką magazynową
- MRP (*Material Requirments Planing*) - planowanie potrzeb materiałowych
- MRP II (*Manufacturing Resource Planing*) - planowanie zasobów produkcyjnych
- ERP (*Enterprice Resource Planing*) - (określana jako MRP III - *Money Resource Planing* lub MRP II Plus) planowanie zasobów przedsiębiorstwa wraz z procedurami finansowymi, w tym księgowość zarządcza, cash flow i rachunek kosztów działania
- CRM (*Customer Relationship Management*) - zarządzanie kontaktami z klientem
- DEM (*Dynamic Enterprice Modeler*) - dynamiczne modelowanie przedsiębiorstwa, umożliwiające bezpośrednie przejście od modelu firmy do gotowej konfiguracji aplikacji dla poszczególnych użytkowników

Omów podstawowe struktury i architektury wykorzystywane w zintegrowanych SI zarządzania

Struktury wykorzystywane w ZSIZ:

- Struktura funkcjonalna - Określa ona cele, funkcje i zadania systemu w układzie hierarchicznym. Wiąże ona SIZ z systemem zarządzania w układzie pionowym
- Struktura informacyjna, w tym:
  - zbiory danych wejściowych,
  - zbiory danych wyjściowych,
  - zbiory danych stałych (przechowywanych),
  - algorytmy przekształcania danych (technologia przetwarzania danych)
- Struktura techniczna - środki techniczne i oprogramowanie systemowe.
- Strukturę organizacyjno-przestrzenną - określa terytorialne rozmieszczenie składników systemu

Architektury ZSIZ:

1. Aplikacje jednostanowiskowe na komputerach niepodłączonych do sieci – najprostsze systemy, ograniczone wykorzystanie,
2. Aplikacje rozproszone w stylu klient-serwer:
    - Dwuwarstwowe (Two-tier): Aplikacja klienta łączy się z serwerem, logika i BD jest umieszczona na jednym i tym samym serwerze,
    - Trzy wartstwowe (Three-tier): Aplikacja klienta łączy się z serwerem, na którym znajduje się logika aplikacji, serwer aplikacji łączy się z serwerem BD,
    - N – warstwowe (Multi-Tier) – Wiele serwerów przetwarzających logikę i/lub wiele serwerów BD,
    - Internetowe – aplikacja całkowicie umieszczona na zdalnym serwerze, interfejs webowy;
    - Z powyższymi wiąże się jeszcze – thin client/fat client – odpowiednio: większość aplikacji po stronie serwera/po stronie klienta

Na czym polega i czym się przejawia złożoność realizacyjna SI zarządzania

Złożoność realizacyjna SIZ: Złożoność wynikająca z czynników realizacyjnych ZSI (organizacyjnych, technicznych, technologicznych i socjo – psychologicznych) narzuca konieczność uwzględnienia czynników zapewniających powodzenie realizacji ZSI (czynniki krytyczne sukcesu - sukcesu to takie obszary, w których wszystko musi być zrealizowane z najwyższą starannością. Są to więc uwarunkowania lub fakty, które mogą narazić projekt na niepowodzenie. Złożoność realizacyjna ZSI przejawia się w najczęściej popełnianych błędach:

- na poziomie definiowania przedsięwzięcia (brak jasno sprecyzowanych celów, niezgodność ze strategią gospodarczą i informatyzacji, brak poparcia kierownictwa, niedopasowanie technologii do rzeczywistych potrzeb firmy, zapominanie o barierach we wprowadzaniu zmian)
- na poziomie planowania przedsięwzięcia (stosowanie nieefektywnych metod i narzędzi, optymistyczne szacunki kosztów i terminów, brak analizy różnych scenariuszy)
- na poziomie organizowania i koordynowania prac (brak właściwych procedur organizacyjnych, brak organizacji zespołów dostosowanej do specyfiki projektu, niesprawne monitorowanie i kontroli prac, brak mechanizmów motywacyjnych)
- na poziomie realizacji (niekontrolowane zmiany w planie przedsięwzięć, niewłaściwa kolejność prac)
- na poziomie kontroli prac (wadliwy system komunikacji w zespołach, usuwanie skutków a nie przyczyn odstępstw od planu, brak działań korekcyjnych)
- na poziomie organizacji zespołów (zła organizacja pracy zespołowej, brak mechanizmów motywacyjnych, zły podział obowiązków i odpowiedzialności, niedocenianie roli kierownika przedsięwzięcia)

Ze złożoności wynika więc **zjawisko niepowodzenia realizacji SI**. Przyczyny niepowodzeń są na ogół grupowane w cztery kategorie:

- nieodpowiedniość - niezgodność z zakresem wymagań (jest to najczęstszy przypadek, eksponowany zwykle przez kierownictwo),
- niepowodzenie procesu tworzenia SI (albo wytworzony SI nie może zadziałać, albo nie może do końca powstać np. z powodu wyczerpania środków lub braku czasu),
- niepowodzenie współdziałania (chodzi o niezgodność wymagań i systemu lub złą jego wydajność),
- niepowodzenie oczekiwań (niezdolność SI do spełnienia oczekiwań najczęściej określonego grona osób u odbiorcy).

Omów znane ci scenariusze realizacji zintegrowanych systemów informatycznych zarządzania

1. Tworzenie ZSI od podstaw przez służby informatyczne przedsiębiorstwa

- klasyczne rozwiązanie oparte na sekwencyjnym realizowaniu poszczególnych etapów pracy nad systemem,
- mało realne - potrzeba kwalifikowanej kadry, długi okres prac,
- duże nakłady czasowo-finansowe - brak uzasadnienia ekonomicznego,
- wariant realizowany sporadycznie,
- ze względu na koszty próba rozpowszechniania powstałego produktu jako uniwersalnego (brak powodzenia).

2. Tworzenie ZSI od podstaw przez zewnętrzne firmy informatyczne
   - wymaga postępowania przetargowego,
   - całkowite uzależnienie od zewnętrznych wykonawców,
   - niebezpieczeństwo powstania produktu dalekiego od oczekiwań przedsiębiorstwa,
   - zwykle długi czas realizacji - odraczany efekt pełnego wdrożenia (konieczność testowania),
   - trudne do oszacowania rzeczywiste nakłady finansowo-czasowe,
   - wariant zalecany dla przedsiębiorstw o ustalonej specyfice organizacyjno-technicznej.
3. Wybór, zakup i wdrożenie wyrobu gotowego
   - zaspokojenie potrzeb przez typowe rozwiązanie dostępne na rynku,
   - procedura wyboru produktu może być zlecona wybranej firmie, powołanemu zespołowi mieszanemu lub zrealizowana samodzielnie,
   - wariant najczęściej stosowany.

Omów sposób realizacji procedury wyboru gotowego ZSI

Procedura wyboru gotowego ZSIZ powinna być realizowana wg poniższego schematu:

1. Ocena aktualnej technologii przetwarzania danych w informatyzowanym przedsiębiorstwie. Ocena (audyt) powinna obejmować:
   - inwentaryzację funkcjonujących systemów
   - analizę funkcjonujących systemów
2. Zdefiniowanie założeń przedsięwzięcia informatycznego. Obejmuje:
   1. wymagania funkcjonalno-techniczne na wysokim stopniu uogólnienia:
      - struktura funkcjonalna
      - struktura informacyjna
      - struktura techniczna
      - struktura przestrzenna
      - struktura konstrukcyjno-technologiczna
   2. Sposób przeprowadzenia postępowania przetargowego,
   3. Organizację wyboru ZSIZ
3. Opracowanie zapytania ofertowego, określenie postaci odpowiedzi ofertowej wraz z procedurą jej oceny. Zapytanie ofertowe zawiera:
   - funkcje realizowane przez ZSI
   - główne wejścia i wyjścia wraz ze specyfikacją obecnego zakresu operacji
   - organizację danych i ich zbiory wraz ze specyfikacją ich zawartości
   - wymagane sprzężenia z innymi systemami
   - wymagania w zakresie zabezpieczeń i kontroli.
4. Ocena odpowiedzi oferentów, ich klasyfikacja i utworzenie tzw. krótkiej listy,
5. Prezentacje i wizyty referencyjne
6. Negocjacje, wybór ZSI i podpisanie umów realizacyjnych
7. Wdrożenie ZSI

Omów krytyczne czynniki sukcesu realizacji zintegrowanych systemów informatycznych zarządzania

Zarządzanie organizacją powinno mieć charakter strategiczny. Należy pamiętać, iż bez ustalenia konkretnych celów biznesowych, jakie firma pragnie osiągnąć, wdrożenie nie będzie miało sensu.

- Systemy informatyczne mogą jedynie pomóc w gromadzeniu, przetwarzaniu i przekazywaniu informacji w organizacji. Nie są one też receptą na sukces, są tylko narzędziem wspierającym pracę organizacji.
- W organizacji musi panować atmosfera zaufania niezbędna do powstawania nowej jakości i dzielenia się wiedzą oraz doświadczeniem.
- Wymiana informacji zarządzania i sprawny proces decyzyjny jest niemożliwy bez wykorzystania nowoczesnych technologii informatycznej.
- Organizacja musi stworzyć własny system motywacyjny wspierający wykorzystanie nowych technologii w procesie zarządzania organizacją.
- Trwałe zmiany są niemożliwe bez silnego zaangażowania i wsparcia liderów organizacji, stąd konieczne jest zaangażowanie naczelnego kierownictwa jak również zapewnienie niezbędnych zasobów i środków.
- Istotne znaczenie ma wytworzenie przekonania o zdolności organizacji do przeprowadzenia zmian zarówno w zespołach pracowników jak i w każdym z pracowników indywidualnie.
- Wdrożenie informatycznego systemu zarządzania należy traktować jako zamierzenie kompleksowe – niekiedy jest procesem długotrwałym.
- Informatyzacja organizacji ma charakter twórczy, a tym samym może ewoluować w niespodziewanych kierunkach. Sytuacje takie należy przewidzieć, akceptować, a nawet wspierać.

Oczywiście podczas wdrażania ZSIZ obowiązywać muszą standardowe reguły planowania, analizy, projektowania i implementacji:

- planowanie strategiczne systemu, czyli stworzenie wizji i uzasadnienie potrzeby tworzenia systemu mającego skutecznie wspomagać strategiczne założenia firmy,
- analiza systemu, czyli ustalenie jego potrzeb w takim zakresie, aby zaspokoić wymagania użytkownika (w aspekcie funkcji, algorytmów, zakresu, ograniczeń projektowych) oraz stworzenie podstaw do realizacji prac projektowych poprzez opracowanie założeń modelowych,
- projektowanie systemu, czyli stworzenie projektu koncepcyjnego i technicznego wynikającego z analizy systemu poprzez wykorzystanie modelowego odwzorowania, struktury, procedur i wymagań,
- implementacja systemu, czyli budowa oprogramowania i infrastruktury technicznej zdolnej realizować założenia projektowe,
- wdrażanie systemu, polegające na instalacji sprzętu i oprogramowania, jego testowaniu oraz szkoleniu personelu w jego eksploatacji. .

Przygotowanie zmian w organizacji - budowanie świadomości: Niezmiernie istotnym elementem przedstawionych wyżej reguł jest budowanie socjologiczno-organizacyjnych podstaw takiego wdrożenia. Wprowadzenie informatycznego systemu zarządzania to ogromna zmiana w przedsiębiorstwie, której nie sprostają zarówno uzasadniona strategia, nowoczesna technologia czy sprawne procesy realizacyjne przedsięwzięcia bez odpowiedniego przygotowania firmy do takich zmian.

Wdrożenie informatycznego systemu zarządzania powinno oczywiście skutkować w przyszłości wyższą efektywnością organizacji stąd zmiany wynikające z takiego wdrożenia będą dotyczyły wszystkich obszarów organizacji (w tym również kombinacji wyspecyfikowanych obszarów), takich jak: struktury, realizowanych procesów biznesowych, technologii, ludzi.

Omów rolę i zadania integratora wdrożeniowego

integrator wdrożenia - firma odpowiedzialna za końcowy rezultat informatyzacji przedsiębiorstwa. Powinna ona wykonać następujące działania:

- Przygotowanie wybranego gotowego ZSI do wymagań i ograniczeń informatyzowanego przedsiębiorstwa:
    - identyfikacja i analiza potrzeb i wymagań przyszłych użytkowników,
    - weryfikacja rozwiązań w odniesieniu do potrzeb przedsiębiorstwa,
    - instalacja i próbna eksploatacja,
    - ostateczna weryfikacja całego rozwiązania
- Przygotowanie informatyzowanego przedsiębiorstwa do wymagań ZSI:
    - przygotowanie organizacyjne przedsiębiorstwa,
    - przygotowanie kadrowe przedsiębiorstwa,
    - przygotowanie infrastruktury technicznej rozwiązania.

Rola i zadania integratora: Roli integratora realizacji SIZ może się podjąć firma posiadająca:

- odpowiedni zespół specjalistów z różnych dziedzin objętych systemem
- rozwinięty system współpracy z dostawcami sprzętu, oprogramowania i usług poparty stałymi umowami i certyfikatami
- grupę konsultantów zewnętrznych, współpracujących z zespołem projektowo-wdrożeniowym
- odpowiednią bazę materiałowo-techniczną, szkoleniową, logistyczną i finansową

Wiedza i doświadczenie w realizacji SIZ muszą być odwzorowane w postaci optymalnego (przetestowanego) modelu integracji działań

Na wybranym przykładzie pokaż etapy wdrażania zintegrowanego systemu informatycznego zarządzania

Etap wstępny przygotowania wdrożenia.

- decyzja o przystąpieniu do prac mających na celu usprawnienie systemu zarządzania,
- wybór systemu informatycznego.

Etap właściwego przygotowania wdrożenia.

- szkolenie przyszłych użytkowników systemu,
- powołanie zespołu wdrożeniowego,
- opracowanie koncepcji planu wdrożenia,
- powołanie grup wdrożeniowych dla wyróżnionych dziedzin,
- precyzyjne określenie kierunku oraz zakresu dostrojenia systemu do wymagań użytkowników,
- opracowanie planu wdrożenia,
- instalacja sprzętu i oprogramowania,
- szkolenie informatyków użytkownika,
- procedury treningowe,
- przygotowanie projektów zmian organizacyjnych.

Etap realizacji wdrożenia i eksploatacji systemu.

- wdrożenie równolegle kilku w obszarach dziedzinowych zarządzania,
- testowanie podsystemu (dziedziny) w zakresie testowania całego systemu informatycznego, „zestrojenie" poszczególnych modułów oraz próbna eksploatacja
- ocena wdrożenia systemu, wyniki próbnej eksploatacji są podstawą do ostatecznej oceny zarówno systemu jak i procesu wdrożenia

Eksploatacja systemu, jago pielęgnacja i ciągłe doskonalenie.

Omów budowę, strukturę i przeznaczenie wybranego zintegrowanego systemu informatycznego zarządzania

System USOS WEB. Budowa, zawiera zakładki:

- dla studentów (spis ocen końcowych, wyniki sprawdzianów, decyzje dziekanatu, rankingi, płatności)
- dla pracowników (wystawianie ocen, spis studentów)
- katalog (wyszukiwanie studenta/pracownika, hierarchia jednostek organizacyjnych, informacje o przedmiotach, kierunek i programy prowadzonych studiów, informacje o akademikach, pomoc)

Struktura:

- dla studentów są dostępne wszystkie informacje dotyczące przebiegu studiów, płatności.
- dla pracowników jest dostępne wystawianie ocen studentom lub podejmowanie decyzji w odpowiedzi na złożone podania.

Przeznaczenie: System ma na celu ułatwienie komunikacji pomiędzy pracownikami uczelni a studentami.

Sztuczna inteligencja

Metody reprezentacji wiedzy. Omówić wektory wiedzy. Podać przykład

Metody reprezentacji wiedzy: Wiedza dotyczy m.in.: obiektów, faktów, zdarzeń, procedur, wiedzy o sobie (metawiedza). Baza wiedzy zawiera fakty i reguły, które są niezbędne do rozwiązania problemu w specyficznej dziedzinie. **Fakty** są zdaniami oznajmiającymi. Oprócz faktów baza wiedzy zawiera **reguły** o postaci: `IF warunek THEN wniosek AND/OR akcja`

Próby ogólnej definicji reprezentacji wiedzy: Przez reprezentacje wiedzy rozumie się sposób w jaki wiedza o świecie jest przedstawiana wraz z metodami przetwarzania, a zwłaszcza wnioskowania (inferencji).Główną siłą sprawczą wyznaczającą zakres i kierunek prac nad reprezentowaniem wiedzy jest to, do czego owa reprezentacja ma być stosowana oraz w - w pewnym stopniu - to w jaki sposób wiedza będzie pozyskiwana.

Najpopularniejsze metody reprezentacji wiedzy:

a) **Reprezentacja logiczna**
Logika formalna (rachunek zdań) Rachunek zdań, to wnioskowanie: z prawdziwych faktów wynikają nowe, prawdziwe fakty. Reguła wnioskowania nie zależy od wiedzy, z którą mamy do czynienia
Logika predykatów. Rachunek zdań jest na ogół zbyt ubogi do reprezentowania faktów, które chcemy zapisać w systemie. Nie można za jego pomocą wyrazić, że pewien obiekt ma pewną własność. Pozwala na to nieco bardziej rozbudowany system logiczny – rachunek predykatów. W rachunku predykatów stosuje się następującą notację:
   - - obiekt *a* ma własność *P* - *P(a)*
   - - każdy obiekt *x*, który ma własność *P*, ma też własność *Q* - $\forall x\,(P(x) \Rightarrow Q(x))$
   - - istnieje obiekt o własności *P*, który posiada również własność *Q* - $\exists x\,(P(x) \Rightarrow Q(x))$
   - - instance (x, y), x jest elementem zbioru obiektów y ($x \in y$)
   - - is-a (x, y), zbiór obiektów x stanowi część zbioru obiektów y ($x \subseteq y$)

   Własności: nie można w niej dowieść fałszywego twierdzenia i wszystkie prawdziwe twierdzenia mają dowód.
b) **Sieci semantyczne** Duże nadzieje wiązano z sieciami semantycznymi, w których symbole zapisywane są w węzłach grafów, a połączenia miedzy nimi (łuki grafu) reprezentują określone relacje. Kontekst, w jakim dopuszczalne jest użycie danego słowa, zapisany jest w

strukturze sieci semantycznej. Każdy węzeł jest symbolem, powiązania są relacjami, natomiast rozumowanie jest przeszukiwaniem grafu. Sieci semantyczne znalazły szerokie zastosowanie w analizie tekstów, gdyż łatwo korzystać z zakodowanej w nich wiedzy, wystarczy zbadać węzły połączone z symbolami występującymi w zdaniu. Proces tworzenia sieci semantycznych jest jednak trudny i uniemożliwia powstawanie spontanicznych asocjacji - wszystkie skojarzenia trzeba przewidzieć i w jawny sposób zaprojektować.

c) **Systemy produkcyjne** Modularna forma zapisu wiedzy. Wiedza zapisana za pomocą reguł produkcji: pary (warunek - działanie):`IF warunek1 AND warunek2 ... THEN akcja.` Należy się zastanowić: czy człowiek używa reguł logicznych? Jedynie w nielicznych przypadkach. Próby analizy tekstów w oparciu o reguły logiczne okazały się bardzo trudne, gdyż zbyt wiele takich reguł da się zastosować w konkretnym przypadku i w efekcie nie wiadomo, które zastosować. Jest to powszechny problem, związany z "eksplozją kombinatoryczną", czyli bardzo szybko rosnącą liczbą zagadnień lub reguł, które należy zastosować by rozwiązać pierwotne zadanie.
Systemy produkcyjne składają się z 3 części: bazy reguł produkcji, specjalnej struktury danych, nazywanej kontekstem, interpretera, kontrolującego aktywność systemu. **Zalety systemów produkcyjnych**: Modularność Jednolitość Naturalność **Wady**: trudno zapisać algorytm, reguły nie odwołują się do siebie

d) **Ramy (frames)** Reprezentacja wiedzy w postaci bardziej złożonych struktur, takich jak "ramy", daje większe możliwości. Ramy zawierają wiedzę ogólną o obiektach, gromadząc wewnątrz opis cech danego obiektu. Nieznane cechy mogą mieć wartości domyślne, najczęściej spotykane, mogą mieć też procedury związane z używaniem wiedzy zawartej w ramach. Ramy są złożonymi strukturami powstałymi w wyniku nagromadzenia się wcześniejszych doświadczeń. Posiadają tzw. haczyki (slots, hooks) na fakty lub procedury. Rozumowaniem jest zapełnianie tych haczyków Jednakże używanie ram nie rozwiązuje podstawowego problemu, jakim jest dobór odpowiedniej wiedzy kontekstowej do danej sytuacji, a więc wybór ram, które należy użyć. Reprezentacja wiedzy w postaci reguł lub ram działa bardzo dobrze jeśli mamy do czynienia z niewielką, ściśle zdefiniowaną domeną wiedzy, dla której znanych jest niewiele faktów.

Wektory wiedzy. Pewnym uogólnieniem reguł są wektory wiedzy. Sposób ten pozwana przedstawić reguły w postaci wektorowej. Baza reguł zapisana w tradycyjny sposób jest przedstawiana za pomocą symboli, np:

- * - warunek/wniosek nie występuje w regule
- T - warunek/wniosek jest prawdziwy
- N - warunek/wniosek jest fałszywy.

Poszczególne reguły w tradycyjnej bazie reguł powinny jednak zawierać jednakową liczbę warunków i wniosków. W rezultacie zakodowania przy pomocy symboli otrzymujemy zwarty i przejrzysty zapis reguł w postaci wektorów. Przykład bazy wiedzy zbudowanej z trzech reguł.

- R1 - IF jest ładna pogoda THEN pójdę na spacer
- R2 - IF jestem zmęczony THEN nie pójdę na spacer
- R3 - IF pada deszcz THEN wezmę parasol

Mechanizm wnioskowania i baza wiedzy w systemach ekspertowych. Podać przykład bazy wiedzy

Systemy ekspertowe są programami komputerowymi przeznaczonymi do rozwiązywania specjalistycznych problemów wymagających profesjonalnej ekspertyzy. Są one w zasadzie zorganizowane w ten sposób, że wiedza dotycząca danej dziedziny jest odseparowana od reszty systemu.

Baza wiedzy zawiera fakty i reguły (ogólnie: pewnych zasad określających sposób postępowania), które są niezbędne do rozwiązania problemu z określonej dziedziny; umożliwiające systemowi przeprowadzenie wnioskowania.

Wnioskowanie jest takim postępowaniem, w którym na podstawie stwierdzeń (zdań) uznanych za prawdziwe np. aksjomaty, twierdzenia, dochodzi się do uznania nowego stwierdzenia - konkluzji.

Maszyna wnioskująca – na podstawie zgromadzonej wiedzy wyszukuje rozwiązanie postawionego problemu – jest ona oddzielona od bazy wiedzy, dzięki czemu działa tak samo w systemach ekspertowych dla dowolnej dziedziny jak i w szkieletowych systemach ekspertowych; algorytm wyszukiwania zawiera szereg strategii przeszukiwań, heurystyk i metod wnioskowania – strategie wyznaczają kolejne kroki przeszukiwań, heurystyki pomagają zoptymalizować przestrzeń poszukiwań, a metody decydują w jaki sposób zachodzi proces myślenia (wnioskowane wstecz, w przód, czy inne);

Przykład bazy wiedzy: fakty: A, B, C, D, E. Reguły:

- R1: if A and B then F
- R2: if C and D then G
- R3: if F and G then H
- R4: if E and H then CEL

Mechanizmy wnioskowania. Omówić wnioskowanie progresywne. Na czym polega strategia świeżości?

Idea wnioskowania w przód: Na podstawie dostępnych reguł i faktów należy generować nowe fakty tak długo, aż wśród wygenerowanych faktów znajdzie się postawiony cel (hipoteza).

Podstawową cechą tego sposobu wnioskowania, która w pewnych sytuacjach może być jego wadą, jest **możliwość zwiększania się bazy faktów**. Postępowanie takie umożliwia, szczególnie w przypadku baz wiedzy o niewielkiej liczbie faktów, zwiększenie ich liczby, a co za tym idzie, przyspieszenie procesu sprawdzania postawionej hipotezy. Jednocześnie, z innego punktu widzenia, tworzenie nowych faktów w pewnych szczególnych sytuacjach może być zjawiskiem niepożądanym, gdyż zajmują one niepotrzebnie pamięć operacyjną komputera, co może doprowadzić do jej całkowitego zapełnienia.

Algorytm wnioskowania w przód można wyrazić w następujący sposób:
BW := Wiedza początkowa / Fakty
UNTIL osiągnięto cel lub nie można zastosować więcej reguł
DO

1. Określ zbiór C reguł w bazie wiedzy BW, dla których są spełnione przesłanki.
2. Ze zbioru C wybierz regułę R na podstawie strategii sterowania wnioskowaniem.

BW := Wynik uaktywnienia reguły R działający na BW plus BW.

Aby zastosować regułę ze zbioru możliwych korzysta się z metod sterowania wnioskowaniem, które ograniczają ilość reguł możliwych do uaktywnienia (spełniając funkcję filtru); oto najbardziej popularne z nich:

- strategia świeżości – polega na wyborze tej reguły, która najpóźniej została dołączona do zbioru reguł;
- strategia blokowania – eliminuje te reguły, które zostały już użyte w procesie wnioskowania;
- strategia specyficzności – opiera się na uwzględnianiu różnej liczby przesłanek w regułach – preferowane są te reguły, które mają większą liczbę przesłanek lub (w wypadku tej samej liczby przesłanek) te, które mają mniejszą liczbę zmiennych;
- strategia przypadkowości – używana jeśli w wyniku zastosowania poprzednich strategii wciąż istnieje więcej niż jedna reguła – wybieramy zatem regułę w sposób losowy;

Krzepkość jako centralne zagadnienie algorytmów genetycznych. Wyjaśnić równice pomiędzy funkcjami: celu, przystosowania i krzepkości

Idea **algorytmów genetycznych** zaczerpnięta jest z przyrody i teorii Darwina o przetwarzaniu osobników najlepiej dostosowanych. Służą do poszukiwań pewnych elementów.

- są poszukiwanymi algorytmami bazującymi na mechanizmach zaczerpniętych z naturalnej selekcji lub genetyki
- w odróżnieniu od algorytmów przypadkowych nie są prostym przeszukiwaniem - wykorzystują informacje przeszłe, aby wytworzyć nowe elementy w przestrzeni poszukiwań
- należą do klasy tzw. algorytmów adaptacyjnych - dostosowywanie do nowych warunków życia

Funkcja przystosowania jest miarą, w jakim stopniu dany chromosom rozwiązuje dany problem. Problem, jaki należy rozwiązać zazwyczaj jest zadany przez programistę.

Rola operatorów genetycznych. Omówić operator krzyżowania wielopunktowego z zanurzeniami

Krzyżowanie. Operatorom genetycznym, w tym również krzyżowaniu. podlegają tylko produkcje typu.

Mutacja. Operator mutacji przechodzi przez kolejne pozycje produkcji typu A —. BC i z prawdo podobieństwem p może zmienić symbol zdanej pozycji produkcji na losowo wybrany ze zbioru nieterminali N. W krańcowym zatem wypadku, efektem zadziałania operatora może być produkcja z losowo zmienionymi symbolami zarówno po lewej stronie, jak i prawej stronie produkcji.

Inwersja. W ogólnym przypadku operator inwersji permutuje kod chromosomu. W przypadku produkcji nieterminalnej modelu GCS permutacji podlegają dwa symbole prawej strony produkcji. Operator inwersji stosowany jest z prawdopodobieństwem P_i. Inwersja osobnika jest w tej chwili rzadziej stosowanym operatorem genetycznym, ze względu na możliwość stosowania n p krzyżowania równomiernego, które nie jest tak wrażliwe na permutację kodu, jak operatory krzyżowania jedno- i dwupunktowego. Ze względu jednak na epistatyczne właściwości populacji klasyfikatorów w modelu GCS operator inwersji wydaje się być pożyteczny. Warto bowiem zauważyć, że inwersja symboli po prawej stronie produkcji równoznaczna jest z zamianą kolejności poddrzew wyprowadzeń każdego z symboli, co z kolei nie powoduje zaburzenia związków epistatycznych pomiędzy klasyfikatorami.

Inwersja jest, oprócz wprowadzonych wcześniej płodności i ścisku, trzecim już mechanizmem, którego zadaniem jest utrzymanie wysokiego tempa zbieżności lub jej lepszym końcowym stopniem, pomimo istnienia wysokiej epistazy populacji produkcji modelu GCS.

Reguły uczenia sztucznych sieci neuronowych. Omówić regułę Hebba

Neuron posiada zdolność adaptacji, polegającą na tym, że wagi synaptyczne mogą się zmieniać. Wraz z nimi zmienia się więc również ranga informacji podawanej na poszczególne wejścia. To, czy na wyjściu neuronu pojawi się impuls, czy też nie, zależy jednak na ogół nie tyle od sygnału na określonym dendrycie, co raczej od całej konfiguracji sygnałów wchodzących do neuronu. I tak, jedne konfiguracje będą pobudzać neuron, inne zaś nie.

Neuron wyposażony w dodatkowe wejście sterujące może być uczony przez zewnętrznego nauczyciela. Uczenie to "po prostu" dobieranie wag połączeń między neuronami w taki sposób, aby po podaniu na wejście sieci jakichś wartości, na jej wyjściu uzyskać odpowiedni wynik. Oczywiście, nikt nie robi tego ręcznie, choć jest to możliwe. Stosuje się specjalne algorytmy.

Uczenie bez nadzoru stosujemy kiedy cel uczenia nie jest określony przez konkretne, prawidłowe przykłady. Jedyną informacją, która musi wystarczyć jest sama wewnętrzna struktura sygnałów

wejściowych (przykładów, przy pomocy których można wytrenować sieć). Na podstawie wewnętrznych zależności w podanych informacjach sieć neuronowa musi stworzyć własne kategorie i będzie rozpoznawać (klasyfikować) podane sygnały wejściowe (generując odpowiednie sygnały na wyjściu).

Zasada uczenia w tym schemacie polega na tym, że waga, i-tego wejścia m-tego neuronu wzrasta podczas prezentacji wektora wejściowego proporcjonalnie do iloczynu i-tej składowej sygnału wejściowego docierającego do rozważanej synapsy w t-tej „turze" i sygnału wyjściowego rozważanego neuronu w tej samej turze.

Można dostrzec, że wzmocnieniu ulegają te wagi, które są aktywne w sytuacji gdy ich neuron jest pobudzony. Jeśli pewien wzór pobudzeń X sygnalizowany jest przez pewne m-te wyjście sieci, to w miarę upływu czasu ta sygnalizacja staje się coraz bardziej wyraźna. Sieć uczy się w ten sposób rozróżniać nadchodzące do niej bodźce i grupować je w pewne kategorie (cluster), gdyż neuron wytrenowany do rozpoznawania pewnego sygnału X będzie zdolny do rozpoznawania także tych sygnałów, które są podobne do tego wzorcowego.

Reguła Hebba posiada istotną wadę, mianowicie prowadzi do procesu rozbieżnego (wagi są przez cały czas zwiększane). Możemy zapobiec rozbieżności prostej reguły Hebba ograniczając wzrost wektora wag.

Architektury sztucznych sieci neuronowych. Omówić sieci typu Perceptron

Rodzaj sieci neuronowych zależy od sposobu połączenia składowych neuronów, co określa tzw. architekturę sieci neuronowej. Do podstawowych typów sieci neuronowych należą:

- sieci jednokierunkowe jednowarstwowe
- sieci jednokierunkowe wielowarstwowe
- sieci rekurencyjne, zwane sieciami Hopfielda
- sieci komórkowe.

Perceptron jest najprostszym modelem sieci jednokierunkowej. Zbudowany jest z N neuronów wejściowych i M neuronów wyjściowych. Elementy zarówno wejściowe jak i wyjściowe przyjmują wartości binarne – {0,1}.Zadaniem perceptronu jest odwzorowanie zadanego zbioru wektorów wejściowych na zadany zbiór wektorów wyjściowych.

Rola funkcji aktywacji w sieciach neuronowych. Podać przykłady

Wartość **funkcji aktywacji** jest funkcją, na którą podawany jest sygnał wyjściowy neuronu, według niej jest obliczana wartość wyjścia neuronów sieci neuronowej Funkcja aktywacji zwana jest funkcja transferową, funkcją przejścia itp. Funkcja aktywacji przybiera jedną z trzech postaci:

- skoku jednostkowego (funkcja progowa)
- liniowa
- nieliniowa

Wybór funkcji aktywacji zależy od rodzaju problemu jaki stawiamy przed siecią do rozwiązania.

Przykład: Dla sieci wielowarstwowych najczęściej stosowane są funkcje nieliniowe, gdyż neurony o takich charakterystykach wykazują największe zdolności do nauki. Polega to na możliwości odwzorowania dowolnej zależności pomiędzy wejściem a wyjściem sieci ale w sposób płynny. Umożliwia to otrzymanie na wyjściu sieci informacji ciągłej a nie tylko postaci: TAK – NIE.

Systemy immunologiczne. Wyjaśnić pojęcie pamięci idiotopowej

Sztuczny system immunologiczny – rozproszony system przetwarzania informacji posiadający zdolność douczania się i adaptacji do zmiennego sztucznego otoczenia.

Algorytm działania SSI: Jeżeli organizm został zaatakowany przez konkretne patogenny, ich struktury zostają zapamiętane. W uproszczonym sensie układ odpornościowy jest binarnym klasyfikatorem, który klasyfikuje dowolną strukturę do jednej z dwóch klas „własny" lub „obcy". Zaklasyfikowanie struktury do klasy „obca" wywołuje ciąg reakcji prowadzących do usunięcia zagrożenia.

Pamięć idiotypowa - pamięć immunologiczna tworzy sieciową strukturę, która podlega modyfikacjom w miarę jak pojawiają się kolejne typy patogenów.

Środowisko MATLAB do projektowania systemów sztucznej inteligencji. Omówić Neural Network Toolbox

Neural Network Toolbox - zbiór funkcji do projektowania i symulacji sieci neuronowych.

NNT - jest rozwiąznaniem bibliotecznym zaimplementowanym w MATLAB służoącym do projektowania sieci neuronowych. Posiada metody do tworzenia sieci.

- wielowarstwowej
- jednowarstwowej złożonej z perceptronów
- jednowarstwowej złożonej z neuronów liniowych

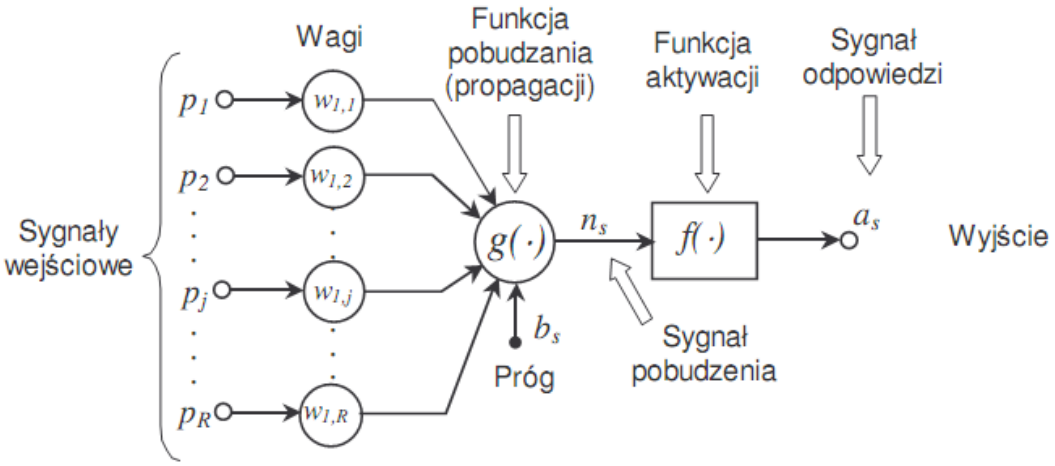
A także metody uczenia, symulacji i wykreślania położenia w/w sieci. Ogólnie rzecz ujmując jest to kompletna biblioteka służąca do tworzenia sieci neuronowych.

Środowisko wytworzone przez The MathWorks będące interaktywnym środowiskiem do wykonywania obliczeń naukowych oraz do tworzenia symulacji komputerowych.

Neural Network Toolbox rozszerzenie środowiska MATLAB. Dodaje funkcje do projektowania, implementacji wizualizacji oraz symulacji sieci neuronowych.

Sieć neuronowa jest charakteryzowana przez

- funkcje wg których neuron reaguje na docierające do niego pobudzenia, nazwane funkcjami pobudzenia (propagacji) i funkcji aktywacji
- strukturę połączeń pomiędzy neuronami – architektura siecie
- metodę określania wag tych połączeń, nazwaną algorytmem uczenia



gdzie:
R- ilość wejść,
S- ilość neuronów w warstwie,

Technologie Programistyczne Systemy Internetowe

Model warstwowy aplikacji – role warstw w aplikacjach z interfejsem graficznym

Role warstw

- Warstwa interfejsu:
  - Obiekty do przyjmowania danych od użytkownika, np. klikanie, wpisywanie tekstu;
  - Obiekty do prezentacji wyników obliczeń (grafika itp.)
  - Obiekty informujące (objaśnienia, itp.)
- Warstwa komunikacji interfejs – logika biznesowa
  - obiekty do przechwytywania działań użytkownika na obiektach interfejsu i przekazujące skojarzonym obiektom warstwy logiki biznesowej
  - reagują na działania na obiektach interfejsu i aktywują obiekty logiki biznesowej
- Warstwa logiki biznesowej
  - Obiekty aktywizowane przez obiekty warstwy komunikacji
  - Wykonują obliczenia
  - Wykorzystują wspólne dane z warstwy współdzielonych danych
  - Mogą modyfikować własności obiektów warstwy interfejsu dzięki referencji do nich pochodzących od warstwy komunikacji – jeśli zmiany mają dotyczyć obiektów interfejsu będących źródłem informacji o działaniu użytkownika. A także poprzez referencję do nich zapamiętane w warstwie współdzielonych danych podczas fazy 1 pracy aplikacji
- Warstwa współdzielonych danych: Obiekty przechowujące dane dla warstwy logiki biznesowej

Serwlety jako rozszerzenia serwera - charakterystyka i porównanie z CGI i innymi metodami dynamicznego HTML po stronie serwera

Rodzaje aplikacji wspierających dynamiczne strony po stronie serwera:

- Aplikacje jako procesy zewnętrzne w stosunku do serwera (programowanie CGI, FastCGI)
- Aplikacje jako moduły serwera (np.. ISAPI)
- Aplikacje działające wewnątrz kontenera („piaskownicy”) na serwerze (np.. Serwlety, PHP)

Porównanie Servletów z innymi technologiami:

Serwlety są bardziej efektywne, łatwiejsze w użyciu, bezpieczniejsze i tańsze od tradycyjnych programów CGI oraz technologii zbliżonych do CGI. Poza tym, mają większe możliwości oraz zapewniają lepszą przenaszalność.

- **Efektywność -** W przypadku CGI, dla każdego żądania HTTP tworzony jest nowy proces. Jeśli sam program CGI jest stosunkowo krótki, to przeważającą część wykonania programu może stanowić uruchomienie procesu. W przypadku serwletów, wirtualna maszyna Javy działa bez przerwy i obsługuje wszystkie nadsyłane żądania wykorzystując do tego niewielkie wątki Javy, a nie procesy systemu operacyjnego, które wykorzystują wiele zasobów systemowych. Co więcej, w przypadku korzystania z tradycyjnych programów CGI, jeśli jednocześnie zostanie nadesłanych N żądań skierowanych do tego samego programu, jego kod zostanie N razy załadowany do pamięci. W przypadku serwletów, w takiej sytuacji zostanie utworzonych N wątków, lecz wykorzystywana będzie wyłącznie jedna kopia klasy serwletu. I ostatnia sprawa. Gdy program CGI skończy obsługiwać żądanie, zostanie on zakończony Serwlety natomiast pozostają w pamięci nawet po zakończeniu obsługi żądania,
- **Wygoda -** Serwlety dysponują rozbudowanymi narzędziami służącymi do automatycznego przetwarzania i dekodowania danych przesyłanych z formularzy HTML, odczytywania i

określania nagłówków HTTP, obsługi cookies, śledzenia sesji oraz wieloma innymi narzędziami wysokiego poziomu.

- **Duże możliwości** - Serwlety udostępniają kilka możliwości, których implementacja w tradycyjnych programach CGI jest wyjątkowo trudna, lub wręcz niemożliwa. Serwlety mogą bezpośrednio wymieniać informacje z serwerami WWW. Kilka serwletów może także korzystać ze wspólnych danych, co znacznie ułatwia implementację wielokrotnego wykorzystywania połączeń z bazami danych oraz innych, podobnych rozwiązań optymalizujących wykorzystanie zasobów. Serwlety mogą także przechowywać informacje pomiędzy kolejnymi żądaniami.
- **Przenośność** - Serwlety są pisane w języku Java i wykorzystują standardowy interfejs programistyczny. W rezultacie, serwlety napisane z myślą o jednym serwerze będą poprawnie funkcjonować na innym.
- **Bezpieczeństwo** - Jedno z podstawowych zagrożeń istniejących w tradycyjnych programach CGI wynikało z faktu, iż programy te często były wykonywane przez powłoki systemowe ogólnego przeznaczenia. Z tego względu programiści tworzący programy CGI musieli zwracać szczególną uwagę na odnajdywanie i eliminację znaków traktowanych przez powłokę systemową w sposób specjalny, takich znaków jak odwrotne apostrofy (`) oraz średniki (;). Kolejnym źródłem problemów jest fakt, iż do tworzenia programów CGI używane są języki, które nie sprawdzają granic tablic i łańcuchów znaków.
- **Niewielkie koszty** - Dostępnych jest wiele darmowych lub bardzo tanich serwerów WWW doskonale nadających się do użytku „osobistego” lub do obsługi witryn o niewielkim natężeniu ruchu. Jednak za wyjątkiem serwera Apache, który jest dostępny bezpłatnie, większość innych serwerów WWW o komercyjnej jakości jest dosyć droga. Znacznie odróżnia to technologie wykorzystujące serwlety od alternatywnych rozwiązań CGI, w przypadku których konieczny jest zakup drogich bibliotek lub pakietów programistycznych.

Dostęp do baz danych z programów w Javie

Bazy danych w Javie mogą być obsługiwane na kilka różnych sposobów w zależności od tego jakiego typu są to bazy. W dokumentacji możemy odnaleźć dwa główne sposoby na łączenie się z bazami danych. Pierwszy z nich to JDO. Mechanizm ten jest rozwijany jako mechanizmy JPA oraz Hibernate. Ten mechanizm szerzej jest znany jako Mapowanie Obiektowo Relacyjne (ORM). Drugą metodą dostępu do bazy danych jest JDBC. Rozwiązanie to jest dedykowane przede wszystkim dla RDBMS ponieważ jest silnie powiązane z językiem SQL.

JDBC jest interfejsem, a zatem nie jest związane z konkretnym dostawcą RDBMS. Z jednej strony jest to bardzo dobre ponieważ migracja pomiędzy dostawcami jest przezroczysta dla kodu aplikacji. Z drugiej strony możliwość bezpośredniego użycia SQL powoduje, że kod może zostać zanieczyszczony rozwiązaniami charakterystycznymi dla konkretnych producentów RDBMS. JDBC zostało zaprojektowane jako interfejs niezależny od architektury bazy danych używanej podczas projektowania danego systemu. JDBC jest pomostem pomiędzy przestrzeniami bazy danych i aplikacji, napisanej w języku Java. Interfejs ten operuje na poziomie typowego dla baz danych języka SQL i pozwala w prosty sposób konstruować zapytania do bazy danych oraz wykorzystywać ich wyniki w kodzie Javy. Zatem, możemy przyjąć, że dzięki JDBC aplikacje bazodanowe napisane w Javie są niezależne od sprzętu oraz stosowanej bazy danych.

Ważnym elementem JDBC jest sterownik JDBC. Sterownikiem JDBC, łączącym naszą aplikację z konkretną bazą danych, nazywamy zestaw klas, które implementują interfejs JDBC. Zadaniem tej technologii jest ukrycie przed programistą wszelkich specyficznych własności danej bazy, pozostawiając widoczne jedynie te najbardziej ogólne, wspólne dla całego modelu. Dzięki temu programista może skupić się wyłącznie na swojej aplikacji nie wdając się w szczegóły związane z obsługą używanej przez niego bazy danych. Aby umożliwić niezależność od platformy, JDBC udostępnia menedżera sterowników, który dynamicznie zarządza obiektami sterowników. Obiekty te

będą wykorzystywać nasze zapytania do bazy danych. Menedżer rejestruje sterownik w momencie ładowania, a samo ładowanie można wymusić, używając na przykład konstrukcji Class.forName().

Omów technologię JSP porównaj z PHP

Technologia Java Server Pages (w skrócie JSP) pozwala na mieszanie zwykłego, statycznego kodu HTML z informacjami generowanymi dynamicznie przez serwlety. Wiele stron tworzonych przez programistów WWW to statyczne dokumenty, w których zawartość generowana dynamicznie ogranicza się do zaledwie kilku niewielkich miejsc. Strona JSP w procesie translacji jest zamieniana na serwlet. Każde wywołanie strony JSP z poziomu klienta wykonywane jest przez skompilowany serwlet. Jeśli użyta zostanie prekompilacja (kompilacja wstępna) to już podczas uruchamiania aplikacji wszystkie strony JSP zostaną przetłumaczone na serwlety.

Porównanie z PHP: PHP jest darmową technologią o ogólnie dostępnym kodzie źródłowym. PHP to język skryptowy przypominający nieco ASP i JSP, a pisane w nim programy umieszczane są bezpośrednio w kodzie dokumentów HTML. Zaletą JSP w porównaniu z PHP jest tworzenie dynamicznych części dokumentu przy użyciu języka Java, który już prawdopodobnie znasz i który dysponuje rozbudowanymi narzędziami programistycznymi do komunikacji sieciowej, wykorzystania baz danych, obsługi obiektów rozproszonych, itp. Natomiast wykorzystanie PHP wiąże się z koniecznością nauki nowego języka programowania.

Klasyfikacja metod kompresji

Kompresja dzieli się na bezstratną – w której z postaci skompresowanej można odzyskać identyczną postać pierwotną oraz stratną – w której takie odzyskanie jest niemożliwe, jednak główne właściwości, które nas interesują, zostają zachowane, np. jeśli kompresowany jest obrazek, nie występują w postaci odtworzonej widoczne różnice w stosunku do oryginału. Pomimo to może się już nie nadawać zbyt dobrze np. do dalszej przeróbki czy do wydruku, gdyż w tych zastosowaniach wymaga się zachowania innych właściwości.

Algorytmy kompresji dzieli się na algorytmy zastosowania ogólnego oraz algorytmy do danego typu danych. Z definicji nie istnieją algorytmy kompresji stratnej zastosowania ogólnego, ponieważ dla różnych typów danych konieczne jest zachowanie różnych właściwości. Na przykład kompresja dźwięku używa specjalnego modelu psychoakustycznego, który nie ma sensu w zastosowaniu do obrazu, poza bardzo ogólnymi przesłankami dotyczącymi sposobu postrzegania rzeczywistości przez człowieka.

Większość algorytmów bezstratnych to algorytmy zastosowania ogólnego oraz ich drobne przeróbki, dzięki którym lepiej działają z określonymi typami danych. Nawet drobne poprawki mogą znacząco polepszyć wyniki dla pewnych typów danych.

Najczęściej używane metody kompresji bezstratnej można podzielić na słownikowe i statystyczne, choć wiele metod lokuje się pośrodku.

Koncepty i zastosowanie technologii apletów (możliwości i ograniczenia)

Aplet Javy – aplet dostarczany w postaci kodu bajtowego Javy. Może zostać uruchomiony w przeglądarce internetowej wykorzystując wirtualną maszynę Javy albo w samodzielnej aplikacji AppletViewer służącej do testowania apletów Javy. Mogą być pisane zarówno w Javie jak i innych językach kompilowanych do kodu bajtowego - na przykład Jython.

Aplety Javy używane są w celu dostarczenia funkcjonalności (np. interakcji), które nie mogą być uzyskane przy stosowaniu HTML-a. Ponieważ kod bajtowy Javy jest wieloplatformowy, to aplety mogą być uruchamiane w wielu systemach operacyjnych, włączając w to Windows, Unix/Linux czy Mac OS.

Działanie apletu oparte jest na kilku głównych metodach: init(), start(), stop(), destroy(). Po załadowaniu strony przeglądarka wykonuje metodę init(), która zapewnia inicjalizację apletu. Następnie wykonywana jest metoda start() uruchamiająca wczytany aplet. Metoda ta jest wykonywana za każdym razem, gdy wczytana strona stanie się aktywna. Przy opuszczaniu strony wywoływana jest metoda stop(). Bezpośrednio przed zakończeniem pracy apletu wywoływana jest metoda destroy(), która zwalnia zasoby zajęte przez uruchomiony applet.

Technologia Enterprise Java Beans

Technologia Enterprise JavaBeans (EJB) jest naturalnym wyborem dla programisty Java, który planuje budowę modułowych, wielowarstowych aplikacji dla zastosowań intranetowych. Formalnie rzecz biorąc, specyfikacja Enterprise JavaBeans stanowi element standardu Java 2 Enterprise Edition (J2EE). Enterprise JavaBeans definiuje architekturę rozproszonych komponentów i klientów dla niezależnych od platformy sprzętowo-programowej aplikacji Java oraz obejmuje biblioteki programisty umożliwiające ich implementację. Idea EJB opiera się na tworzeniu komponentów (ziaren EJB), które mogą być osadzane na serwerze aplikacji (tzw. kontenerze EJB), który z kolei udostępnia je do wykonania lokalnie (dostęp z części aplikacji uruchomionej na tej samej wirtualnej maszynie) lub zdalnie poprzez protokół RMI over IIOP.

Zdalne ładowanie a zdalne wywołanie modułów - cele i problemy

Remote Procedure Call (RPC - zdalne wywołanie procedury) to protokół zdalnego wywoływania procedur, stworzony przez firmę Sun i swego czasu dość popularny na Uniksach, obsługiwany w bibliotekach języka Java. Współcześnie wypierany przez bardziej rozbudowane protokoły takie jak CORBA, XML-RPC, czy JSON-RPC.

CORBA (ang. Common Object Request Broker Architecture) to 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.

RMI (ang. Remote Method Invocation - zdalne wywołanie metod) to mechanizm umożliwiający zdalne wywołanie metod obiektów. Obiekty te mogą znajdować się w innych maszynach wirtualnych Javy, które mogą znajdować się na innych komputerach.

Problemy związane z RMI

- Klient i serwer muszą używać tej samej technologii.
- Technologie te zaprojektowano z myślą o zastosowaniach lokalnych (intranety) – stąd istnieją m.in. problemy z przekraczaniem zapór firewall.

Problemy związane z RPC

- Zróżnicowanie formatów danych na różnych maszynach
- Eksport i import powoduje narzuty
- Narzut utrzymywania logów
- Okresowe kontrole z klientem podczas dłuższych obliczeń

Podejścia i poziomy umiędzynarodawiania kodu

Dwa poziomy umiędzynarodowienia kodu:

- Umiędzynarodowienie (internacjonalizacja) - tworzenie aplikacji w taki sposób, aby nadawała się do jednoczesnego użycia dla różnych narodów bez konieczności zmiany kodu programu. Np. Dopasowanie języka, zapisu liczb, dat, kalendarza realnego czasu, komunikacji międzyludzkiej, stosunku do płci itd.. Nazwany także “i18n”
- Inkulturacja adaptacja oprogramowania poprzez dodanie specyficznych dla kraju czy języka lokalnych własności i modułów oraz dostarczenie tłumaczenia tekstów. Tzw. proces l10n

i18n koncentruje się głównie na technicznych aspektach umiędzynarodowienia - tj. dostarczanie aplikacji w wielu językach (chociażby poprzez zewnętrzne pliki zasobów i nie zaszywanie tekstów bezpośrednio w kodzie – dzięki temu tłumaczenia nie wymagają rekompilacji), prostą podmianę formatu daty, waluty itp.

l10n idzie odrobinę dalej – programista aplikacji zwraca wtedy uwagę na wszystkie kulturowe różnice, jak np. aspekty religijne, istnienie świąt, istnienie różnic w nazwiskach męskich/żeńskich, warunki dziedziczenia, monogamia/poligamia itp.

Porównanie technologii .NET i Java

Java	.NET
Kompilowane do kodu pośredniego (bytecode), obecność maszyny wirtualnej	Kompilowane do kodu pośredniego (CIL), obecność maszyny wirtualnej
Jeden oficjalnie wspierany język, istniejące projekty dostarczające języków opartych o JVM (jak Groovy czy Clojure)	Wsparcie oficjalnie dla wielu języków (C#, VB.NET, F# itp.), specyfikacja, która określa wymagania dla języka aby był zgodny z .NET i możliwy do wykorzystania na tą platformę
Środowisko uruchomieniowe na wszystkie platformy	Środowisko uruchomieniowe na platformę Windows (nieoficjalne porty na *nix, kompatybilne ze starszymi wersjami .NET)
Pełna otwartość, zarówno maszyny wirtualnej (co prawda JVM od Oracle jest faktycznie zamknięta, ale istnieje otwarta implementacja OpenJDK), używanych bibliotek i serwerów aplikacji	W całości zamknięta
Dystrybucja aplikacji: .jar, .war, .ear, .class	.exe, .dll
Dostęp do bazy danych: JDBC	ODBC, Ado.NET
Możliwości serwerowe: Servlety, JSP, JSF, EJB, WebServices	ASP.NET, ASP.NET MVC
Serwery aplikacji: Wiele, różnych producentów, jak: Tomcat, Jetty, Glassfish, Jboss, WebSphere etc.	Jeden dedykowany serwer – Microsoft IIS.
Mnóstwo zewnętrznych technologii i frameworków	Mniejsza niż w Javie liczba frameworków stworzonych przez firmy trzecie
Możliwość wołania kodu zewnętrznego	Możliwość wołania kodu zewnętrznego
Wiele darmowych, zaawansowancyh IDE (NetBeans, Eclipse, ...)	Darmowe IDE tylko w podstawowej wersji, niewielka aktywność firm trzecich na rynku IDE dla .NET

Technologie Sieciowe

Model programowanie sieciowego klient – serwer. Gniazda (sockety) TCP

Socket API w sposób jawny:

- kreowany, używany i zamykany przez aplikację,
- oparty na architekturze klient/serwer,
- dwa typy usługi transportowej przez socket API: niepewne datagramy UDP i pewne strumienie danych TCP.

Socket – jest interfejsem pomiędzy aplikacją a systemem operacyjnym; jest bramą, przez którą proces może wysyłać i odbierać wiadomości do/od innego procesu należącego do odległej (lub lokalnej) aplikacji.

TCP:

- Klient musi zacząć pierwszy – jest stroną inicjalizującą
- Serwer musi być uruchomiony najpierw żeby klient miał z czym się łączyć
- Serwer musi stworzyć tzw. socket „powitalny serwer" i czekać na klientów
- Klient łączy się z serwerem poprzez: stworzenie lokalnego klienckiego socketu TCP, musi podać adres IP serwera oraz nr portu, na którym serwer nasłuchuje.
- Kiedy klient tworzy socket TCP, ustanawia połączenie z serwerem TCP.
- Następnie serwer TCP tworzy nowy socket do komunikacji z tym klientem – pozwala to serwerowi na nawiązanie wielu połączeń z wieloma klientami.

Adresacja IP

Każdy host TCP/IP jest identyfikowany na podstawie logicznego adresu IP. Ten adres jest unikatowy dla każdego hosta komunikującego się za pomocą protokołu TCP/IP. Każdy 32-bitowy adres IP określa lokalizację systemu hosta w sieci – służy do identyfikacji urządzeń.

Tak jak adres zamieszkania składa się z dwóch części (nazwa ulicy i numer domu), tak i każdy adres IP jest podzielony na dwie części — identyfikator sieci i identyfikator hosta:

- Identyfikator sieci, znany także jako adres sieci, identyfikuje pojedynczy segment sieci należący do większej sieci złożonej (sieci składającej się z innych sieci) TCP/IP. Wszystkie systemy podłączone i mające dostęp do tej samej sieci mają w swych pełnych adresach IP wspólny identyfikator sieci. Identyfikator ten służy także do unikatowej identyfikacji każdej sieci w większej sieci złożonej.
- Identyfikator hosta, znany także jako adres hosta, identyfikuje węzeł TCP/IP (stację roboczą, serwer, router lub inne urządzenie TCP/IP) w każdej sieci. Identyfikator hosta każdego urządzenia służy do unikatowej identyfikacji każdego systemu w sieci, do której ten system należy.

Aby adresowanie IP było prostsze, adresy IP są wyrażane w zapisie kropkowo-cyfrowym. 32-bitowy adres IP składa się z czterech 8-bitowych oktetów. Oktety w systemie binarnym są konwertowane na oktety w systemie dziesiętnym (system z podstawą dziesiętną) i oddzielone kropkami.

Omówić DNS

Zasoby internetowe (hosty, routery, strony WWW) identyfikuje się na 2 sposoby:

1. poprzez adres IP – 32-bitowy, np. 148.81.7.73, adres używany do adresowania datagramów
2. nazwy – np. ii.ap.siedlce.pl, używane przez ludzi. Potrzebna jest więc translacja pomiędzy adresami AP a nazwami.

Domain Name System (DNS). To rozproszona baza danych implementowana jako hierarchia wielu serwerów nazw. Protokół warstwy aplikacji. Hosty, routery oraz serwery nazw komunikują się, aby rozwikłać nazwę (translacja adres IP/nazwa).

Dlaczego nie można zcentralizować DNS?

- Jeden punkt (serwer), który może „paść"
- Za duży narzut komunikacyjny
- Odległa scentralizowana baza danych
- Utrzymanie: dynamiczne zmiany, nowe nazwy.
- Żaden serwer DNS nie posiada wszystkich translacji nazwa – adres IP. Stosuje się lokalne serwery nazw – każdy ISP (Internet Sernice Provider), instytucja posiada własny lokalny (domyślny) serwer nazw. Zapytanie DNS od hosta wędruje wpierw do lokalnego serwera.

Autorytatywny serwer nazw – dla hosta: przechowuje jego adres IP i nazwę, może przeprowadzić translację nazwa/adres IP.

DNS: Root Name Servers – z nimi kontaktują się lokalne serwery nazw jeśli same nie potrafią rozwikłać nazwy.

Protokół HTTP, adresacja URL i HTML

HTTP: hypertext transfer protocol Web-owy protokół w warstwie aplikacji. Klient/serwer model:

- Klient: przeglądarka, która prosi, otrzymuje i wyświetla Web-owe obiekty;
- Serwer: Web serwer wysyła obiekty w odpowiedzi na prośby klienta

http (http1.0: RFC 1945 I http1.1: RFC 2068): usługa transportowa dostarczana przez TCP:

- Klient inicjuje połączenie TCP (kreuje socket) do serwera na porcie 80;
- Serwer akceptuje połączenie TCP od klienta;
- Przykazy zgodne z protokołem http są wymieniane przez przeglądarkę (http client) oraz www-serwer (http server);
- Połączenie TCP zamknięte.
- HTTP jest bezstanowe (http 1.0) – serwer nie trzyma informacji o byłych zapytaniach klientów (serwer parsuje zapytanie, odpowiada i zamyka połączenie TCP).
- Trwałe połączenia (http 1.1) – na tym samym połączeniu TCP, serwer parsuje zapytanie, odpowiada, itd.
- 2 RTT-y potrzebne do ściągnięcia każdego obiektu (Mniej RTT-ów i szybszy start)
- Klient wysyła zapytanie o wszystkie obiekty z referencjami tak szybko, jak tylko otrzyma bazową stronę HTML

Format przekazów http:

- request (ASCII) – zapytanie, komendy GET, HEAD (pobiera tylko informacje nagłówkowe bez samego dokumentu), POST.
- response – odpowiedź.

Metody http/1.0: GET, POST, HEAD.

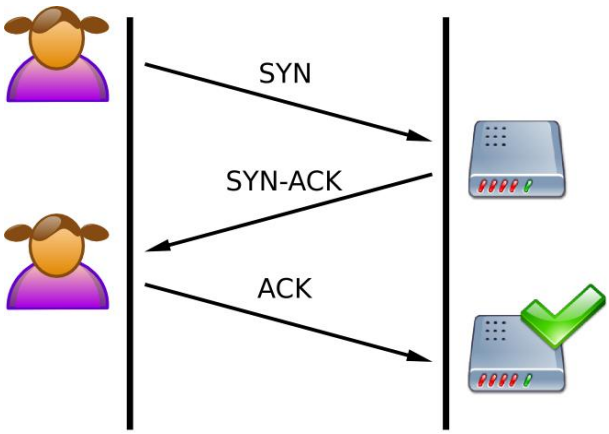
Metody http/1.1: GET, POST, HEAD, PUT (do przesyłania plików na serwer), DELETE (usuwa plik, wyspecyfikowany w polu URL).

URL – Universal Resource Location, adres strony www. Każdy adres URL posiada dwa składniki – nazwę hosta serwera przechowującego obiekt i ścieżkę identyfikacyjną jego lokalizację.

HTML – plik pełniący rolę obiektu wchodzącego w skład witryny WWW. Większość witryn jest złożona z podstawowego pliku HTML i kilku obiektów, do których są zdefiniowane odwołania. Plik odwołuje do obiektów poprzez adresy URL.

Protokół TCP i UDP, struktura segmentu TCP i UDP

TCP jest protokołem działającym w trybie klient-serwer. Serwer oczekuje na nawiązanie połączenia na określonym porcie. Klient inicjuje połączenie do serwera.



W przeciwieństwie do UDP, TCP gwarantuje wyższym warstwom komunikacyjnym dostarczenie wszystkich pakietów w całości, z zachowaniem kolejności i bez duplikatów. Zapewnia to wiarygodne połączenie kosztem większego narzutu w postaci nagłówka i większej liczby przesyłanych pakietów. Chociaż protokół

definiuje pakiet TCP, to z punktu widzenia wyższej warstwy oprogramowania, dane płynące połączeniem TCP należy traktować jako ciąg oktetów. Charakterystyczny dla TCP jest moment nawiązania połączenia, nazywany three-way handshake. W celu weryfikacji wysyłki i odbioru TCP wykorzystuje sumy kontrolne i numery sekwencyjne pakietów. Odbiorca potwierdza otrzymanie pakietów o określonych numerach sekwencyjnych ustawiając flagę ACK. Brakujące pakiety są retransmitowane. Prawidłowe zakończenie połączenia może być zainicjowane przez dowolną stronę. Polega ono na wysłaniu pakietu z ustawioną flagą FIN (finished). Pakiet taki wymaga potwierdzenia flagą ACK.

Opis nagłówka TCP

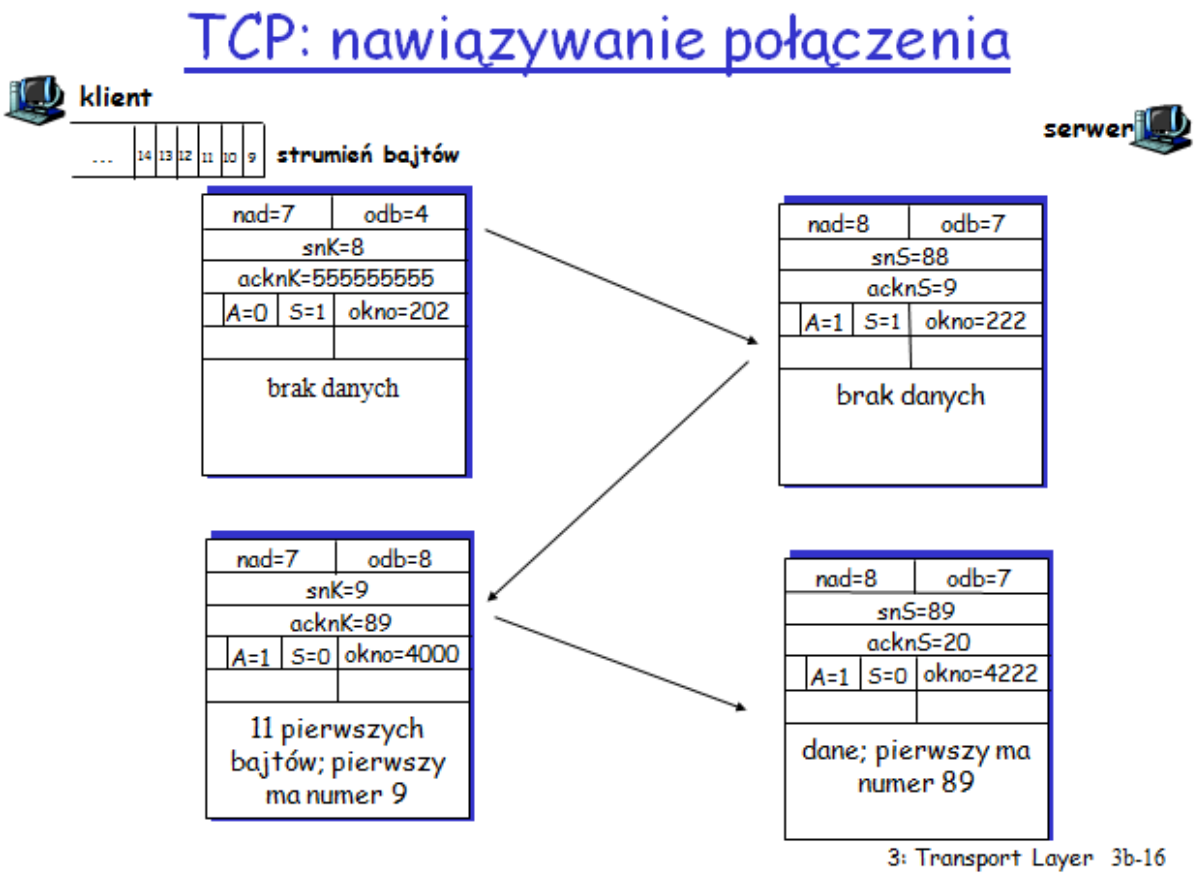
	Bity 0 – 3	4 – 7	8 – 15	16 – 31
0	Port nadawcy			Port odbiorcy
32	Numer sekwencyjny			
64	Numer potwierdzenia			
96	Długość nagłówka	Zarezerwowane	Flagi	Szerokość okna
128	Suma kontrolna		Wskaźnik priorytetu	
160	Opcje (opcjonalnie)			
160/192+	Dane			

UDP (ang. User Datagram Protocol – protokół pakietów użytkownika) – jeden z podstawowych protokołów internetowych. Umieszcza się go w warstwie czwartej (transportu) modelu OSI. Jest to protokół bezpołączeniowy, więc nie ma narzutu na nawiązywanie połączenia i śledzenie sesji. Nie ma też mechanizmów kontroli przepływu i retransmisji. Korzyścią płynącą z takiego uproszczenia budowy jest większa szybkość transmisji danych i brak dodatkowych zadań, którymi musi zajmować się host posługujący się tym protokołem. UDP udostępnia mechanizm identyfikacji różnych punktów końcowych (np. pracujących aplikacji, usług czy serwisów) na jednym hoście dzięki portom(porównaj: gniazdo). UDP zajmuje się dostarczaniem pojedynczych pakietów, udostępnionych przez IP, na którym się opiera. Kolejną cechą odróżniającą UDP od TCP jest możliwość transmisji do kilku adresów docelowych na raz (tzw. multicast).

Opis nagłówka UDP (dla IPv4)

	Bity 0 – 7	8 – 15	16 – 23	24 – 31
0	Adres źródłowy			
32	Adres docelowy			
64	Zera	Protokół	Długość UDP	
96	Port źródłowy		Port docelowy	
128	Długość		Suma kontrolna	
160	Dane			

Nawiązywanie połączenia TCP i kontrola zagęszczeń w TCP



Nadawca i odbiorca (klient i serwer) ustanawiają połączenia zanim zaczną wymieniać segmenty z danymi. Zmienne inicjalizujące połączenie TCP:

- numery sekwencyjne;
- bufory oraz info do kontroli

Klient inicjuje połączenie, serwer zgadza się na nie. **Potrójny uścisk**:

- Klient wysyła kontrolny segment tzw. TCP SYN do serwera; specyfikuje początkowy seq#;
- Serwer otrzymuje segment SYN i odpowiada kontrolnym segmentem SYNACK
    - Potwierdza poprzez ack# otrzymany segment SYN
    - Alokuje bufory i czas procesora;
    - Specyfikuje swój początkowy seq#
- Po odebraniu segmentu SYNACK klient również przydziela połączeniu bufory i zmienne.

Zagęszczenia występują wtedy, gdy zbyt dużo nadawców wysyła zbyt dużo danych zbyt szybko aby sieć mogła je obsłużyć. Objawiają się stratą segmentów (bufory router-ów przepełnione) oraz opóźnieniami (kolejki w buforach router-ów). Jest to jeden z 10 najważniejszych problemów w Internecie.

Podejścia do kontroli zagęszczeń. Kontrola zagęszczeń przez systemy końcowe:

- Sama sieć nie bierze udziału w kontroli. Zagęszczenie jest postrzegane przez systemy końcowe poprzez straty i opóźnienia. To podejście jest stosowane w TCP
- Sieć bierze udział w kontroli zagęszczeń. Routery informują systemy końcowe o zagęszczeniu poprzez pojedynczy bit informuje o zagęszczeniu. Wyznaczanie szybkości wysyłania pakietów.

Algorytm routingu Link State

Algorytmy te przedstawia się na grafach, gdzie wierzchołkami grafu są routery, a krawędzie fizycznymi łączami. Koszt połączenia – opóźnienia, koszt $, lub stopień zagęszczenia. „Dobra" ścieżka to zazwyczaj ścieżka o minimalnym koszcie.

Klasyfikacja:

- globalna informacja – routery znają kompletny graf oraz koszty połączeń – algorytmy **„link state"**
- rozproszona informacja – routery znają tylko sąsiadów, do których mają bezpośrednie połączenie oraz koszty tych połączeń – algorytmy **„distance vector".**

Algorytm Dijkstra (link-state). Graf oraz koszty połączeń są znane wszystkim routerom. Wylicza najkrótsze ścieżki od jednego węzła (zródło) do wszystkich pozostałych, wynikiem czego jest **tabela routingu** dla tego węzła. Po *k* iteracjach, znane są najkrótsze ścieżki do *k* węzłów.

Protokoły routingu Znane również jako Interior Gateway Protocols (IGPs). Najbardziej powszechne:

- RIP (Routing Information Protocol)
  - Oparty na algorytmie Distance Vector
  - Metryka odległości: ilość skoków
- OSPF (Open Shortest Path First)
  - Oparty na algorytmie Link State
  - Drogi są liczone za pomocą algorytmu Dijkstra
- IGRP (Interior Gateway Routing Protocol)
  - Następca RIP
  - Używa Distance Vector
  - Wiele różnorodnych metryk

Algorytm routingu Distance Vector

Algorytm Distance Vector. Algorytm iteracyjny – jest kontynuowany tak długo dopóki jakiś węzeł ma informację do przekazania.

Rozproszony – każdy węzeł komunikuje się tylko ze swoimi bezpośrednimi sąsiadami. Struktura danych w **tabeli odległości**:

- każdy węzeł oblicza swoją własną tabelę
- wiersz dla każdego routera-odbiorcy
- kolumna dla każdego z bezpośrednich sąsiadów

Minimalne wartości w tabeli odległości wyznaczają tabelę routingu.

Routing hierarchiczny i protokół BGP

Routing hierarchiczny jest to rozwiązanie problemów w dużych sieciach (Internet):

- Propagacja informacji o routingu
- Przechowywanie i obliczanie olbrzymich tablic routingu

Routing hierarchiczny oznacza że węzły tworzą logiczne grupy na kilku poziomach hierarchii.

Protokół BGP (Border Getway Protocol) wykonuje routing między domenowy w sieciach pracujących z TCP/IP. Wykonuje routing pomiędzy wieloma systemami autonomicznymi (domenami) i wymienia informacje o routingu i dostępności z innymi systemami posługującymi się protokołem BGP. Utrzymuje tablice routingu, przesyła uaktualnienia routingu i podejmuje decyzje o trasie kierowania ruchem. Protokół BGP wykonuje we współczesnych sieciach zadania związane z wyborem ścieżek dla ruchu międzydomenowego, oraz rozwiązuje problem skalowalności Internetu.

Uwzględni politykę przesyłania danych: np. żadna komunikacja z USA do Rosji nie może przechodzić przez Irak.

Modyfikuje algorytm distance-vector: każdy router pobiera od swoich sąsiadów nie tylko wektor odległości do różnych innych routerów, ale również ścieżki przypisane do tych odległości -- może od razu odrzucić ścieżki przechowujące przez niego samego (rozwiązanie problemu Count-to-infinity)

Adresacja w LAN, Ethernet oraz protokół ARP

Adresacja w LAN – Każda karta sieciowa posiada swój własny nie powtarzalny adres MAC umieszczony w pamięci ROM. Producent wykupuje pewną pulę adresów w IEEE.

Ethernet – technologia sieciowa wykorzystywana w budowie lokalnych sieci komputerowych.

ARP – protokół zapewniający węzłom mechanizm translacji adresów IP na adresy warstwy łącza danych (adres MAC). Każdy węzeł sieci przechowuje w swojej pamięci RAM tabelę protokołu ARP która zawiera mapowania adresów IP na MAC

IP	MAC	TTL
192.168.1.15	88-45-AD-DF-99-10	12:01:00
192.168.1.18	CC-40-EF-F1-11-11	12:08:00

W przypadku gdy węzeł nadawczy chce wysłać datagram do węzła posiadając jego adres IP (z zaznaczeniem że węzeł znajduje się w tej samej podsieci). Węzeł nadawczy odwołuje się do tabeli protokołu ARP i uzyskuje z niej odpowiedni adres MAC. W przypadku gdy dla danego adresu IP nie ma wpisu w tablicy, węzeł nadawczy utworzy pakiet ARP (składający się z pól IP i MAC dla węzła nadawczego i odbiorczego). Pakiet jest wysyłany pod adres rozgłaszania MAC w postaci FF-FF-FF-FF-FF-FF. Karta kapsułkuje pakiet w ramce warstwy łącza danych, a następnie umieszcza ją w podsieci. Ramka ta jest odbierana przez karty wszystkich węzłów w sieci. Każdy węzeł sprawdza czy jego adres IP jest zgodny z docelowym IP zawartym w pakiecie ARP. Jeżeli tak wtedy odsyła do nadawcy pakiet ARP odpowiedzi, na podstawie którego ten uaktualnia tablicę ARP.

Wstęp do programowania

Wymień i omów własności algorytmu

Własności algorytmu:

- adekwatność (również: poprawność) - czy algorytm realizuje obliczenia zgodnie z przyjętym celem realizacji obliczeń
- własność stopu (określane również jako skończoność) – czy zostały zdefiniowane kryteria zatrzymania wykonywania algorytmu w warunkach poprawnego i niepoprawnego przebiegu obliczeń
- jednoznaczność - algorytm jest zapisany w sposób na tyle precyzyjny, że jego wykonanie jest prawie automatycznym powtarzaniem kolejnych kroków
- powtarzalność - każde wykonanie algorytmu przebiega według takiego samego schematu działania i prowadzi do rej samej klasy rozwiązań
- złożoność obliczeniowa (również określane jako efektywność) - nakład czasu lub zasobów maszyny realizującej algorytm, niezbędny dla jego prawidłowego wykonania
- dyskretność – algorytm składa się ze skończonej liczby dyskretnych operacji

Wyjaśnij pojęcie translacji. Jakie znasz metody translacji

Tekst programu w postaci kodu źródłowego, zapisanego w jakimś języku programowania, musi zostać przetłumaczony na język maszynowy, zanim program będzie można uruchomić. Proces tłumaczenia programu na język maszynowy nazywamy **translacją**, a wykonywany jest za pomocą specjalnych

programów zwanych translatorami. Wadą tworzenia w ten sposób programów jest długi czas ich powstawania, a zaletą to, że można maksymalnie wykorzystać możliwości sprzętowe, dlatego najczęściej w taki sposób tworzone jest oprogramowanie systemowe.

Interpretacja - polega ona na wykonywaniu programu na podstawie jego tekstu. Tłumaczenie dokonywane jest wtedy krok po kroku a przetłumaczony fragment programu jest natychmiast wykonywany.

Asemblacja - proces tłumaczenia programu zapisany w języku asemblera (niskiego poziomu) na język maszynowy, jest wykonywany za pomocą specjalnych programów zwanych asemblerami.

Kompilacja - napisaniu programu w języku wysokiego poziomu za pomocą dowolnego edytora tekstu, tekst źródłowy poddawany jest kompilacji przez programy zwane kompilatorami. Następnie w fazie łączenia następuje połączenie procedur bibliotecznych z tekstem programu - realizują to linkery lub konsolidatory. Efektem końcowym jest program wynikowy(plik), który może być zapisany na dysku i później uruchamiany za pośrednictwem systemu operacyjnego.

Wymień i omów powody wprowadzania typów danych

Typ danych – w językach programowania opis rodzaju, struktury i zakresu wartości, jakie może przyjmować dany literał, zmienna, stała, argument, wynik funkcji lub wartość.

Języki programowania mają wiele typów standardowych, tj. takich które nie wymagają definiowania, oraz daje możliwości zdefiniowania dowolnego potrzebnego typu. Każdy typ, standardowy lub zdefiniowany musi mieć określony w sposób jednoznaczny dopuszczalny zbiór wartości (można to zamieścić w specyfikacji programu).

Powody wprowadzania typów.

- Typ danych pozwala osiągnąć logiczną jasność, wskazując czym jest dana zmienna (np. liczbą rzeczywistą) i jakie operacje można na niej wykonywać.
- Podczas wykonywania programu, z różnych powodów, bieżąca wartość zmiennej może być zapamiętywana w kilku komórkach. Typ danych pozwala translatorowi (kompilatorowi, interpretatorowi) na zarezerwowanie niezbędnej liczby komórek z przeznaczeniem na przechowywanie wartości zmiennych wartości.
- Każdy język programowania ma swój zestaw podstawowych instrukcji testujących wartości pewnych zmiennych (w wyniku otrzymujemy wartość logiczną), lub przekształcających wartości zmiennych. Wymaga to jasnej specyfikacji typu danych dla zmiennych uczestniczących w tych operacjach.

Scharakteryzuj typ tablicowy w języku C/C++

Tablica (zwana czasami zmienną tablicową) jest to struktura języka programowania, która zbudowana jest z elementów będących tego samego typu i zajmujących ciągły obszar pamięci. W C++ elementy tablicy mogą posiadać następujące typy: int, char, float, short, long, wskaźnikowy, mogą również być tablicami bądź klasami. Tablicę definiuje się za pomocą typu, identyfikatora oraz wymiaru.

int tab[100]; typem tab jest int[100], a więc 100 elementowa tablica liczb całkowitych typu int. Wartością wyrażenia sizeof(tab) będzie zatem rozmiar w bajtach całej tablicy.

Niestety, nie oznacza to, że tablice mają własność tablic znanych nam z Javy, tzn. że, tablica zna swój wymiar. W prawie każdej operacji wykonywanej na zmiennej tab zmienna ta jest niejawnie konwertowana do typu wskaźnikowego. Wartością tab jest wtedy adres pierwszego elementu tablicy, a więc elementu o indeksie zero. A zatem po deklaracji: int tab[20]; zmienna tab jest traktowana jako wskaźnik typu int* wskazujący na pierwszy element tablicy. W szczególności wynika z tego, że wyrażenie *tab jest nazwą pierwszego elementu tablicy.

Scharakteryzuj typ plikowy w języku C/C++

Typ ten jest reprezentowany w ramach biblioteki sdio w C++, która zawiera operacje na plikach, w tym również na konsoli. Typ FILE jest strukturą, która w każdym systemie może być zdefiniowana inaczej. Dostęp do struktury (wskaźnik) otrzymuje się poprzez funkcję fopen przy otwieraniu pliku, po czym należy go zamknąć poprzez funkcję fclose. Korzysta się z trzech podstawowych strumieniu plikowych, które są predefiniowane (jako stałe typu FILE*):

- stdin – standardowe wejście
- stdout – standardowe wyjście
- stderr – standardowe wyjście diagnostyczne

Podczas uruchamiania programów z konsoli, stdin to wejście z klawiatury, a stdout i stderr są kierowane na ekran.

Funkcje podstawowe:

- najprostszą funkcją wypisującą dane w danym strumieniu jest fputs o argumentach: fputs(napis, strumien)
- dla stout jest krótsza forma, puts, wymagająca tylko pierwszego argumentu
- można również wysłać tylko jeden znak przy pomocy funkcji fputc(napis, strumien);
- pojedyncze znaki są wczytywane funkcją fgetc;
- konsola ma zazwyczaj liniowy sposób wczytywania znaków, toteż nawet jeśli oczekuje się jednego znaku, należy po jego wprowadzeniu nacisnąć Enter.
- Wczytywany jest znak, a funkcja zwraca int
- Część typu int nie używana przez char pozostaje zawsze zerem, co umożliwia stwierdzenie końca pliku, co jest sygnalizowane przez zwrócenie jego wyniku specjalnej stałej EOF, nie odpowiadającej żadnemu znakowi.
- Znak po wczytaniu może zostać cofnięty;
- Funkcja fgets(tablica, ile, strumien) umożliwia wczytanie całej linii
- Funkcje fread umożliwia wczytanie porcji danych ze strumienia plikowego do wskazanej tablicy;
- Funkcja fwrite zapisuje do danego strumienia podaną porcję danych;

Można także skorzystać z biblioteki fstream.h; do odczytywania zawartości pliku należy utworzyć obiekt ofstream i powiązać go z plikiem (poprzez jego nazwę). Zapis odbywa się podobnie, należy użyć obiektu ifstream i również go powiązać z plikiem. Odczyt i zapis sformatowanych danych wykonywany jest między innymi przez operatory >> i <<.

Wymień i scharakteryzuj metody komunikowania się funkcji w C/C++

Funkcje w c++ są od siebie całkowicie odizolowane. Mogą się one wymieniać się informacjami, najgorszym sposobem są oczywiście **zmienne globalne**.

Zwracanie wartości przez funkcję – podczas deklaracji funkcji określa się typ jaki będzie ona zwracać. W ciele funkcji znajduje się słowo kluczowe return zwracające wartość.

Odczytywanie wartości zwracanej przez funkcję - Funkcja zwraca wartość, a ta wartość można odczytać.

Przesyłanie argumentów do funkcji przez wartość – podczas deklaracji funkcji określa się ilość i tym argumentów z jakimi funkcja będzie wywoływana. Do funkcji przekazywana jest kopia argumentów.

Przesyłanie argumentów przez referencje – pozwala na modyfikowanie zmiennych (nawet lokalnych) znajdujących się poza funkcją. Czyli możliwa jest komunikacja z funkcja w dwie strony. Referencja oznaczana jest przez &.

Argumenty domniemane – są to argumenty, które posiadają domyślna wartość, gdy argument taki nie jest podany, używana jest wartość domyślna.

Scharakteryzuj zmienne lokalne i globalne

Zmienna lokalna – o zasięgu obejmującym pewien blok, podprogram. W językach obsługujących rekurencję zazwyczaj są to zmienne automatyczne, natomiast w językach bez rekurencji mogą być statyczne.

Zmienna globalna – obejmuje swoim zasięgiem cały program, widziana w nim z wielu miejsc, istniejąca przez cały czas życia programu.

Nadużywanie zmiennych globalnych może prowadzić do problemów:

- W programach wielowątkowych zmienna może być modyfikowana przez dowolny wątek;
- Zaśmiecają przestrzenie nazw, co może doprowadzić do przepełnienia stosu
- Mogą kolidować ze zmiennymi lokalnymi

Zaleca się używanie zmiennych statycznych o ograniczonym polu widzenia, zabezpieczaniem dostępu blokadami.

Omów wykorzystanie parametrów domyślnych w C/C++

Jawne inicjowanie argumentów w deklaracji (nie w definicji) funkcji można traktować jako przykład przeciążenia funkcji; tematem tym zajmiemy się później bardziej szczegółowo. Weźmy następujący przykład: w prototypie funkcji, która symuluje ekran monitora, wprowadźmy inicjalne wartości domyślne dla szerokości, wysokości i tła ekranu `char* ekran(int x=80, int y=24, char bg = ' ');`

Wprowadzone wartości początkowe argumentów x, y oraz bg są domyślne w tym sensie, że jeżeli w wywołaniu funkcji nie podamy argumentu aktualnego, to na stosie funkcji zostanie „położona” wartość domyślna argumentu formalnego. Jeżeli teraz zadeklarujemy zmienną `char*` kursor, to wywołanie `kursor = ekran();` jest równoważne wywołaniu `kursor = ekran(80, 24, ' ');`

Deklaracja funkcji nie musi zawierać wartości domyślnych dla wszystkich argumentów, ale podawane wartości muszą się zaczynać od skrajnego prawego argumentu, np.

```
char* ekran( int x, int y, char bg = ' ' );
char* ekran( int x, int y = 24, bg = ' ' );
```

Wobec tego deklaracje są błędne:

```
char* ekran (int x = 80, int y, char bg = ' ');
char* ekran ( int x, int y = 24, char bg );
```

Omów przeciążanie funkcji w C/C++

Rodzina funkcji charakteryzująca się tą samą nazwą, ale różnymi: Typami parametrów; Liczbą parametrów; Typem zwracanej wartości. Przy czym dwie funkcje nie różnią się tylko typem zwracanej wartości. Celem stosowania przeciążeń funkcji jest optymalizacja programu z punktu widzenia czasu jego trwania.

Przykłady rodziny funkcji przeciążonej:

- `int pole(int x, int y) { return x*y };`
- `double pole(double x, double y) { return x*y };`
- `double pole(double x) { return x*x };`

Omów konsekwencje występowania nadmiaru w programowaniu

Przekroczenie zakresu – nadmiar. **Przekroczenie zakresu** nie jest sygnalizowane. Poniższy przykład pokazuje jaka wartość może być otrzymana. `short n = SHRT_MAX + 1`

Przy próbie reprezentacji zbyt dużej liczby powoduje wystąpienie wyjątku.

Zaawansowane systemy grafiki komputerowej

Omów zasady doboru barw stosowane w programach grafiki komputerowej

Pewne reguły wynikają raczej z psychologicznego punktu widzenia, a nie z estetycznych względów. Np. oko jest bardziej czułe na przestrzenne zmiany natężenia niż na zmiany barwy. Także linie, tekst, drobne szczegóły powinny odróżniać się od tła raczej jasnością (natężeniem światła) niż barwą.

Należy unikać barw podobnych do siebie, np. złą kombinację stanowi barwa niebieska z czarną, podobnie żółta z białą. Natomiast barwa żółta dobrze kontrastuje z barwą czarną, ponadto uwypuklenie barwy żółtej z czarnym tekstem jest lepsze, niż barwy białej z czarnym tekstem, także biały tekst na niebieskim tle daje lepszy kontrast niż biały na czarnym. Lepiej unikać kombinacji czerwonej barwy z zieloną. Także barwy nie powinny być wykorzystywane do odróżniania małych obiektów.

Omów wykorzystanie barw komplementarnych w grafice komputerowej

Mamy barwy addytywne, subtraktywne, jasność, odcień, nasycenie i barwy komplementarne, **Efekt barw komplementarnych**, to barwy, które w połączeniu tworzą monochromatyczną biel lub szarość. Zasada barw komplementarnych jest fundamentem w budowie materiałów światłoczułych. Jednocześnie, jest pierwszym sposobem subiektywnej oceny reprodukcji barw przez operatora w procesie negatyw pozytyw („czystość" reprodukcji szarości). **Barwy komplementarne dopełniają się zatem wzajemnie w ocenie obserwatora.** Tak więc, schematyczne koło barw (np. w modelu HSV) powstałe w wyniku optycznego nałożenia świateł, ukaże jako parę dopełniającą (komplementarną) barwy żółtą i niebieską; znajdą się one bowiem naprzeciw siebie.

Scharakteryzuj wykorzystanie „skali barw" w aplikacjach komputerowych

Podstawową skalą barwną jest skala RGB. Barwy są tutaj reprezentowane po zmieszaniu różnych natężeń czerwieni, zieleni, błękitu. Każdy z tych kolorów ma swój kanał. Wartości każdego z kanałów barwnych opisane są za pomocą 8 bitów danych na piksel (na kolor). Zatem każdy piksel obrazka RGB ma składową: 0-255 czerwieni, 0-255 zieleni, 0-255 niebieskiego. Gdy wszystkie składowe mają wartość zero, piksel jest czarny. Ponieważ kolory RGB łącząc się tworząc biel, nazywane są kolorami addywnymi. Oznacza to, że wszystkie odbite światła odbierane z powrotem przez oko. Ponieważ każdy z trzech kanałów barwnych jest 8-bitową skalą szarości, każdy piksel obrazka RGB może przyjąć jeden z 256 odcieni każdego koloru składowego. 256^3 = 16,7 mln barw. Tryb RGB jest stosowany jako standard w wyświetlaniu na monitorach. Stanowi on podstawę przy generowaniu obrazka na tryb CMYK (który z kolei ma zastosowanie w drukarkach offsetowych). Przestrzeń RGB jest dużo większa niż CMYK, a więc edycja w RGB daje większe możliwości. Wszystkie skanery pobierają obrazy kolorowe w modelu RGB. Niektóre naświetlarki slajdów i folii barwnych wymagają danych RGB (większość wymaga CMYK).

Omów elementy architektury systemów grafiki komputerowej

- Tor wizualizacji – relacje zachodzące między strukturami danych, WE/WY, procesorem i pamięcią obrazu;
- Podsystem przekształceń – przekształcenia dla obiektów pobranych z BD do postaci umożliwiających pasteryzację;
- Podsystem pasteryzacji – realizuje zadania eliminacji linii i powierzchni zasłoniętych, obliczanie wartości pikseli i zapisywanie w pamięci;
- Pamięć obrazu – interpretacja pamięci obrazu, np. 1B może obrazować 1 piksel, ale nie musi, piksele mogą być opisywane za pomocą wielu bajtów.

Omów schemat przetwarzania danych w systemach grafiki komputerowej

W systemie graficznym zadaniem toru wizualizacji jest utworzenie na ekranie obrazu obiektów opisanych w pewnym układzie współrzędnych. Niezależnie od architektury i parametrów toru wizualizacji, kolejność operacji jest następująca:

- I etap – wytworzenie danych i zorganizowanie ich w strukturę danych (dane te mogą być odczytywane z pamięci masowej);
- II etap – wybór z utworzonej struktury właściwej porcji danych i przekazanie ich do jednostek przetwarzających;
- III etap – współrzędne są przekształcane we współrzędne ekranowe z uwzględnieniem rzutowania perspektywicznego i obcinania, są także przeprowadzane obliczenia związane z przyjętym modelem oświetlenia;
- IV etap – odwzorowanie poszczególnych elementów obrazu (punktów, odcinków, wieloboków) w pamięci obrazu. Operacje te korzystają z parametrów wyliczonych w III etapie;
- V etap – wyświetlenie na ekranie pikseli na podstawie zawartości pamięci obrazu.

Pierwszy etap dotyczy niewielkich zbiorów danych, natomiast operacje realizowane podczas tworzenia opisu mogą mieć różnorodny charakter. Organizacja struktur danych, formaty danych, podział ekranu na okna, wybór danych w zależności od położenie obserwatora w istotny sposób wpływają na komplikację budowy opisu. Ostateczna postać obrazu musi być pod kontrolą programu użytkowego, dlatego też tworzeniem struktur danych zajmuje się program nadrzędny.

Wyświetlanie obrazu jest proste, wymaga jednak korzystania z dużych zbiorów danych i dostosowania układów o szybkości działania przekraczającej możliwości procesorów uniwersalnych. Z tego powodu we wszystkich systemach wyświetlanie obrazu jest realizowane przy użyciu specjalizowanych układów.

Pierwsze sterowniki nie zawierały żadnych układów obliczeniowych a ich funkcje ograniczały się do wyświetlania zawartości pamięci obrazu. Istnieją systemy, w których pewne operacje są realizowane przez układy pamięciowe – przechowywana w danej chwili zawartość pamięci graficznej stanowi pełny opis ramki (klatki) wyświetlanego obrazu.

Scharakteryzuj wykorzystanie współbieżności w systemach grafiki komputerowej

Najprostsze rozwiązanie polega na przeniesieniu niektórych zadań dotyczących tworzenia i wyświetlania obrazu procesora z procesora centralnego do specjalizowanej jednostki graficznej. Z uwagi na specyfikę realizowanego przetwarzania systemy graficzne są podatne na zrównoleglanie.

Współbieżność może dotyczyć przestrzeni obiektów (do obiektów stosujemy zrównoleglanie) lub przestrzeni obrazów (zrównoleglanie stosujemy np. do wierszy). W pierwszym przypadku jednocześnie są wykonywane operacje z różnymi obiektami będą ich fragmentami - np. Algorytm śledzenia promieni, gdzie każdej jednostce możemy przypisać oddzielny obszar analizowanej sceny.

Współbieżność w przestrzeni obrazu polega na jednoczesnym wykonywaniu operacji dotyczących różnych części obrazu. Typowym rozwiązaniem jest dzielenie obrazu na regularne obszary i przypisanie jednostki dla pewnego bloku obrazu. Można stosować te dwa podejścia jednocześnie.

Omów metody klasyfikacji architektur systemów grafiki komputerowej

Jedną z możliwości jest zastosowanie klasyfikacji systemów komputerowych dla SGK.

Inne możliwości wynikają np. z rozważania budowy systemu w aspekcie podziału algorytmu przetwarzania między jednostką centralną a specjalizowanymi układami graficznymi.

Spotyka się następujące oznaczenia:

- G – generowanie danych,
- T – przegladanie struktur danych,
- X – przekształcenia współrzędnych,
- S – wypełnianie pamięci,
- D – wyświetlanie.

Np.. GTXS-D oznacz że tylko wyświetlanie jest realizowane przez specjalizowane układy,

Inną możliwością jest przyjęcie za podstawą podsystemu rasteryzacji.

- Przy rasteryzacji w porządku obiektowym mamy: Porządek obiektowy oznacza, że dla każdego wielokąta są kolejno wyznaczane wartości wszystkich jego pikseli – może to prowadzić do wielokrotnego obliczania tego samego piskela. Można stosować np.. z-bufor, algorytm malarski lub algorytm dwójkowego podziału przestrzeni.
- Przy rasteryzacji w porządku pikselowym mamy: Porządek pikselowy jest typowy dla algorytmów skaningowych – dla każdego piksela są sprawdzane obiekty mogące mieć wpływ na jego postać.

Mimo podobieństw obydwie metody prowadzą do różnych rozwiązań konstrukcyjnych.

Scharakteryzuj rodzaje sprzętu wykorzystywanego w systemach grafiki komputerowej

Drukarka urządzenie współpracujące z komputerem, służące do przenoszenia danego tekstu, obrazu na różne nośniki druku (papier, folia, płótno itp).

CPU urządzenie cyfrowe sekwencyjne, które pobiera dane z pamięci, interpretuje je i wykonuje jako rozkazy. Wykonuje on ciąg prostych operacji (rozkazów) wybranych ze zbioru operacji podstawowych określonych zazwyczaj przez producenta procesora jako lista rozkazów procesora.

Skaner urządzenie służące do przebiegowego odczytywania: obrazu, kodu paskowego lub magnetycznego, fal radiowych itp. do formy elektronicznej (najczęściej cyfrowej). Skaner przeszukuje kolejne pasma informacji odczytując je lub rejestrując.

Procesor graficzny jest główną jednostką obliczeniową znajdującą się w nowych kartach. Głównym zadaniem GPU było wykonywanie obliczeń potrzebnych do uzyskania akcelerowanej grafiki 3D, co spowodowało częściowe odciążenie procesora CPU z konieczności wykonywania tego zadania. W tej sytuacji mógł on zająć się innymi obliczeniami, co skutkowało zwiększeniem wydajności komputera podczas renderowania grafiki. Nowoczesne procesory graficzne wyposażone są w szereg instrukcji, których nie posiada procesor komputera.

Monitor decyduje o jakości wyświetlanego obrazu.

Omów metody kompresji obrazów stosowane w grafice komputerowej

Kompresja obrazów cyfrowych pozwala na zmniejszenie rozmiaru danych opisujących obraz. Zależnie od tego, czy po dekompresji możliwe jest odtworzenie idealnego obrazu źródłowego, czy też dane zostaną częściowo zmienione dzielimy metody kompresji na stratne i bezstratne.

Nie istnieją algorytmy kompresji stratnej, które można stosować do dowolnego typu danych. Prostym przykładem kompresji stratnej jest np. zachowanie tylko co drugiego piksela, lub odrzucenie 2 najmniej istotnych bitów. Takie metody jednak nie dają zazwyczaj tak zadowalających rezultatów jak oparte na modelach psychozmysłowych.

Kompresję bezstratną umożliwiają np. pliki PNG, TIFF, JPEG 2000. Niektóre techniki zamieniają (zmniejszają, usuwają) pewne informacje, aby uzyskać mniejszy plik. Niestety nie są to bezstratne metody kompresji. Przykładami takich kompresji jest kompresja JPEG, a także stratne tryby kompresji PNG czy JPEG 2000.

Najbardziej powszechnym algorytmem kompresji obrazów jest JPEG. Wiele rozwiązań użytych w JPEG jest używanych także w innych algorytmach, więc warto je tutaj omówić:

- obniżenie rozdzielczości kanałów koloru. Tak radykalne cięcie danych nieznacznie wpływa na jakość, ponieważ rozdzielczość postrzegania kolorów przez ludzkie oko jest słaba.
- podzielenie każdego kanału obrazka na bloki 8x8. W przypadku kanałów kolorów, jest to 8x8 aktualnych danych, a więc zwykle 16x8. Zamiast wartości pikseli mamy teraz średnią wartość wewnątrz bloku oraz częstotliwości zmian wewnątrz bloku, obie wyrażone przez liczby zmiennoprzecinkowe. Zastąpienie średnich wartości bloków przez różnice wobec wartości poprzedniej. Poprawia to w pewnym stopniu współczynnik kompresji.
- kwantyzacja, czyli zastąpienie danych zmiennoprzecinkowych przez liczby całkowite. To właśnie tutaj występują straty danych.

Omów standardy wykorzystywane w grafice komputerowej

Potrzeba standaryzacji operacji i możliwości przenoszenia oprogramowania pojawiła się wraz z upowszechnieniem się sprzętu wykorzystywanego do grafiki. Pierwszy standard (**3D Core**) został opracowany przez ACM SIGGRAPH. Przez wiele lat, praktycznie do lat dziewięćdziesiątych powszechnie stosowanym standardem w rozwiązaniach przemysłowych był **GKS**. Wynikało to z prostej konstrukcji operacji 2D i typowego stosowania grafiki 2D we wspomaganiu prac inżynierskich. Bardziej rozbudowane możliwości zaoferował **PHIGS**, dając do dyspozycji hierarchiczne struktury prymitywów. Standard ten był rozszerzany o możliwości tworzenia grafiki realistycznej do postaci **PHIGS+** oraz **PHIGS PLUS**. Dodatkową zaletą tych systemów jest efektywne wykorzystanie wspomagania sprzętowego. Obecnie wydaje się, że najpowszechniejszym standardem jest **OpenGL** opracowany przez Silikon Graphics (**SGI**)– firmę przodująca w rozwiązaniach dla potrzeb grafiki realistycznej. Niezależnie popularność zyskał **Direct3D** zaproponowany przez firmę Microsoft dla potrzeb obsługi gier komputerowych. Jako **DirectX** jest najpowszechniej akceptowanym standardem w środowisku kart graficznych.

Formaty graficzne można podzielić na dwie grupy:

- Związane z grafiką rastrową (nieskalowalne), (bmp, gif, tiff, tga, JPG, png).
- Związane z grafiką wektorową (skalowalne), (wmp, eps, ps (Postscript), dxf, dwg (Autocad), svg).