# Modelowanie a metamodelowanie

**Model** oprogramowania jest to abstrakcja systemu lub jego części. W zależności od typu modelu może on przedstawiać prosty lub bardziej szczegółowy widok systemu. Utworzone modele zapewniają efektywną komunikację między projektantami, zwłaszcza, gdy modelowany system jest bardzo złożony i wymaga zaangażowania wielu osób (zespołów).

Modelowanie można określić jako tworzenie opisu obiektu/zjawiska rzeczywistego lub abstrakcyjnego, wykonywane w założonym celu. Jego efektem jest model, który powinien posiadać określone własności. Z tego względu nie ma modeli złych, są tylko takie, które nie realizują założonego celu. Model może natomiast być niepoprawny, tzn. nie spełniać definiujących go własności.

Dwie role modelu:

− Reprezentacja tego, co istnieje (np. modele dziedzinowe, biznesowe).

− Reprezentacja tego, co ma powstać (np. model projektowy).

**Meta-modelowanie** oznacza konstruowanie zbioru "koncepcji" (obiektów, terminów, itp.) w zakresie pewnej dziedziny. Uznając model za abstrakcję pewnego zjawiska ze świata rzeczywistego, to meta-model jest abstrakcją ukazującą właściwości owego modelu. Przykładem takiej zależności może być program komputerowy napisany w pewnym języku programowania oraz gramatyka owego języka programowania.

Meta-modelowanie stosuje się w paradygmacie obiektowym.

Model jest instancją metamodelu

Meta-model jest modelem, którego używa się do mówienia o rodzajach modeli jakie chcemy budować. Na przykład jeśli w przedsiębiorstwie zastosowano podejścia wykorzystujące diagramy związków encji i diagramy przepływu danych to metamodel definiowałby egzemplarze takich typów obiektowych jak Typ encji, Typ związku, Typ atrybutów, Przechowywanie danych, Przepływ danych. Te typy obiektowe determinowałyby wówczas sposób, w jaki by wyrażono następny poziom. Na przykład metamodel z typem obiektowym o nazwie Typ procesu pozwalałby tworzyć typy procesów takie jak Zmontuj część, czy Zapłać pracownikowi.

Ostatnimi czasy pojawił się trend tak zwanej Architektury Sterowanej Modelami (ang.

Model Driven Architecture, MDA), która jest zbiorem standardów dostarczanych przez Object Management Group, mających ujednolicić budowę modeli oprogramowania oraz ich transformację do wykonywalnego kodu. Kluczowym elementem MDA jest zunifikowany język modelowania (ang. Unified Modeling Language, UML), który jest od dawna wykorzystywany do modelowania struktury i zachowań systemów oprogramowania.

# Własności i zakres zastosowań języków UML i LOTOS

**Własności UML (Unified Modeling Language):**

* Język półformalny (składnia bezkontekstowa zdefiniowana na podzbiorze UML, składna kontekstowa zdefiniowana za pomocą OCL, zdefiniowania nieformalnie semantyka na języku naturalnym)
* Graficzny język modelowania (specyfikowania, projektowania, dokumentowania)
* Powszechnie akceptowany standard, wykorzystywany do odniesień dla pojęć obiektowości
* Reprezentacja modelu w postaci zbioru diagramów, pełniących określone role
* Opisuje zachowania systemu za pomocą diagramów
* Reprezentuje zazwyczaj zbiory bytów, a nie pojedyncze byty
* Koncentruje się na danych i operacjach ich przetwarzania

**Zakres zastosowań UML:**

* Budowa systemów informatycznych
* Modelowanie zachowania – interakcji zachodzącej pomiędzy elementami systemu oraz pomiędzy elementami systemu i jego otoczeniem (opis przebiegu przypadków użycia, przypadków testowych, procesu realizacji przypadków użycia, przebiegu procesów systemowych)
* Modelowanie dziedziny problemu – w przypadku stosowania go do analizy oraz do modelowania rzeczywistości, która ma dopiero powstać - tworzy się w nim głównie modele systemów informatycznych
* Głównie używany wraz z jego reprezentacją graficzną – jego elementom przypisane są symbole, które wiązane są ze sobą na diagramach. Uniwersalnym formatem zapisu języka UML jest XMI - język służący do zapisywania modeli UML za pomocą

Głównym przeznaczeniem UML jest budowa systemów informatycznych. Służy do modelowania dziedziny problemu - w przypadku stosowania go do analizy oraz do modelowania rzeczywistości, która ma dopiero powstać - tworzy się w nim głównie modele systemów informatycznych.

**Własności LOTOS (Language Of Temporal Ordering Specification):**

* Algebraiczny język procesowy
* Algebraiczna specyfikacja typów danych
* Algebraiczna specyfikacja zachowania
* Język interaktywny opisujący interakcje systemu z otoczeniem za pomocą komunikacji poprzez bramki (porty)
* Analityka nie interesuje wnętrze systemu (czarna skrzynka), a jedynie zmiany jego stanów
* Umożliwia modelowanie procesów w systemach zagnieżdżonych, gdzie jeden system zawiera w sobie kilka pomniejszych procesów
* Umożliwia modelowanie procesów współbieżnych
* Przy modelowanie nie jest ważne wnętrze systemu (czarna skrzynka), a jedynie zmiany jego stanów
* Trze wersje języka: bazowa (basic), pełna (full), rozszerzona. Różnica wynika z wprowadzenia abstrakcyjnych typów danych w wersji pełnej. Tzn. w wersji bazowej synchronizacja procesów odbywa się bez wymiany danych pomiędzy procesami. Pełna wersja pozwala na wymianę danych pomiędzy procesami w punktach synchronizacji. Wyróżnia się też wersję rozszerzoną, gdzie uwzględnia się upływ czasu.

**Zastosowanie LOTOS:**

Wykorzystany w tworzeniu systemów informatycznych, jako narzędzie pozwalające modelować systemy z wykorzystaniem algebry procesów – wyrażenie wymagań funkcjonalnych systemu w postaci sekwencji akcji.

Pierwotny obszar zastosowań: specyfikacja usług, specyfikacja protokołów sieci komputerowych.

*Z wykładów Huzara, zastosowanie m. formalnych w IO:*

Programowanie:

* Definicja składni i semantyki j programowania (m operacyjne, denotacyjne, aksjomatyczne)
* Specyfikacja programów (specyfikacje wykonywalne – algorytmiczne, specyfikacje logiczne – warunki wstępne i końcowe)
* Testowanie programów
* Weryfikacja programów (badanie poprawności częściowej, badanie poprawności całkowitej względem specyfikacji, automatyczne dowodzenie twierdzeń)

Modelowanie:

* Definicja składni semantyki języków modelowania (m operacyjne, np. sieci Petriego, aksjomatyczne np. algebry procesów [LOTOS])
* Analiza modelowa (badanie własności modeli: realizowalność modeli i spełnianie zadanych własności [m logiczne i algebraiczne])
* Definicja składni i semantyki j transformacji modeli (m logiczne i operacyjne)
* Konstrukcja modeli
* Weryfikacja modeli

# Problemy transformacji i spójności modeli

Cały problem transformacji i spójności modeli zaczyna się od pojęcia MDE (Model Driven Engineering) które oznacza specyfikację projektowania i rozwoju projektów informatycznych, opracowaną przez OMG(Object Management Group). MDA jest podejściem, w którym UML jest traktowany jako język programowania.

Głównym celem MDA jest tworzenia oprogramowania w oparciu o modele biznesowe oraz separacja modelu na zależny oraz niezależny od platformy. Dzięki MDA te same rozwiązania mogą być realizowane na wielu różnych platformach. Ponadto stworzone na bazie MDA systemy mogą być łatwo integrowane oraz łączone z innymi systemami. Nic nie stoi też na przeszkodzie aby ponownie używać opracowane rozwiązania.

Podejście MDA zakłada następujące etapy budowy systemu informatycznego:

•Specyfikacja systemu w sposób niezależny od platformy implementacyjnej

•Wybranie najbardziej odpowiedniej platformy dla rozwijanego systemu

•Transformacja (odwzorowanie) systemu do wybranej platformy implementacyjnej

W skład MDA wchodzą następujące modele:

•CIM (computation independent model) – odpowiednik modelu biznesowego oraz modelu przypadków użycia

•PIM (platform independent model) – odpowiednik modelu analizy

•PSM (platform specific model) – odpowiednik modelu projektowego

**CIM** – Computation Independent Model - Widok na system z poziomu, niezależnego obliczeniowo. CIM nie pokazuje szczegółów struktury systemu. CIM czasami jest zwany modelem dziedzinowym. Jego użytkownikami, są konsultanci i eksperci dziedzinowi, nie muszą oni znać modeli, ani artefaktów potrzebnych do realizacji konstrukcji, stosują się tylko do wymagań. CIM pokazuje zachowanie zewnętrzne systemu i funkcje, bez wchodzenia w szczegóły. Pełni ważną rolę w łączeniu przestrzeni pomiędzy ekspertami dziedzinowymi, a architektami struktury systemu.

**PIM** – Platform Independent Model – spojrzenie na system z widoku niezależnego od platformy. PIM powinien być tak pokazany, aby można go było użyć na wielu platformach.

Konstrukcja systemu jest opisana, bez szczegółów implementacyjnych. Mówi się w tym przypadku o modelu na poziomie ogólnym. System jest na pokazany na poziomie ogólnym jeżeli posiada następujące właściwości:

· Kompozycja: przedstawia zbiory elementów z danej kategorii

· Środowisko: posiada elementy pogrupowane w kategorie. Rozłączne z kompozycją

· Produkcja: Elementy kompozycyjne produkują elementy używane w środowisku

· Struktura: Łączenie elementów kompozycyjnych a także środowiskowych

Tak jest zbudowany UML

· Kompozycja: Klasy, asocjacje itd.

· Środowisko: Połączenie dwóch klas asocjacją

· Produkcja: Z elementów kompozycyjnych utworzony element środowiskowy

· Struktura: Dalsze połączenia pomiędzy konstrukcjami

Następnym krokiem jest zamiana PIM na model oprogramowania.

**PSM** – Platfrom Specific Model – jest spojrzeniem na system z poziomu platformy implementacyjnej. PSM łączy pojęcia z PIM ze szczegółami jak system będzie używał elementów specyficznych dla platformy dla której został stworzony. Innymi słowy PSM jest bardziej szczegółowym PIM.

Transformacja w MDA polega na automatycznej generacji modelu wyjściowego z modelu wejściowego wykonanej w oparciu o definicję transformacji. Definicja transformacji jest to zbiór reguł transformacji, czyli opisów przekształceń jednego lub więcej elementów języka modelu wejściowego w jeden lub więcej elementów modelu wyjściowego.

*Transformując modele, niezależnie jakie na jakie, należy pamiętać, że każda transformacja ma 8 różnic o których należy pamiętać:*

*1. Reguły transformacji – definiują jak element z modelu źródłowego będzie przedstawiony w modelu docelowym.*

*2. Reguła zakresu aplikacji – pozwala zdefiniować docelowy zakres w docelowym modelu.*

*3. Relacje pomiędzy źródłem a celem – po prostu czy model docelowy, będzie całkowicie zmienionym modelem źródłowym, czy też jego ewolucją*

*4. Reguła strategii aplikacji – określa strategię jaka zostanie przyjęta przy transformacji*

*5. Reguły harmonogramu - określają kolejność przekształceń i elementów do przekształcenia*

*6. Organizacja reguł – określenie ważności i priorytetów reguł przekształcania*

*7. Śledzenie – transformacje muszą dać się, śledzić tak aby było wiadomo, która z czego wynika*

*8. Kierunek – transformacje muszą mieć określony kierunek, choć mogą być wykonywane tylko w jednym lub w obu kierunkach.*

**Przykład transformacji PIM na PSM:** Przystępując do transformacji musi być znana i wybrana platforma. W pierwszym kroku wybiera się elementy z PIM, które będą musiały ulec transformacji. Taki oznaczony PIM następnie przekształca się tak aby wszystkie oznaczone elementy pasowały do wybranej platformy. Elementy specyficzne dla wybranej platformy mapują potem elementy z PIM.

W UML transformacje takie będą pojawiać się np. przy usuwaniu klas asocjacji, wielodziedziczenia czy asocjacji n-arnych, których przykładowo w Javie nie da się zaimplementować.

**Spójnosc semantyczna** jest pojęciem dotyczącym diagramu jako takiego i moze byc rozstrzygnieta bez odwoływania sie do zewnetrznych załozen. Rozpatruje sie w tym przypadku, czy własnosci przedstawione na diagramie nie sa wzajemnie sprzeczne; innymi słowy, czy moze istnieć dziedzina problemu o tych własnosciach, a w konsekwencji implementacja diagramu. Naturalnie, implikacja braku spójnosci semantycznej jest jednoczesny brak poprawnosci semantycznej. W oparciu o formalna semantyke pojeciowego diagramu klas jego spójność semantyczna moze byc zdefiniowana i weryfikowana formalnie. Spójnosc semantyczna definiuje sie dalej najpierw w odniesieniu do pojedynczego klasyfikatora, a nastepnie dla całego diagramu.

Klasyfikator jest spójny semantycznie jezeli własnosci pokazane na diagramie pozwalaja na istnienie instancji tego klasyfikatora.

Z definicji spójnosci klasyfikatora wynikaja — ujete dalej w postaci twierdzeń — proste własnosci:

• Jezeli dany klasyfikator jest niespójny semantycznie w kontekscie danego diagramu, to niespójne semantycznie sa równiez jego specjalizacje na tym diagramie. I równowaznie, jezeli klasyfikator jest spójny semantycznie, to spójne semantycznie sa równiez klasyfikatory, które specjalizuje.

• Jezeli dana klasa jest niespójna semantycznie w kontekscie danego diagramu, to niespójne semantycznie sa równiez asocjacje, w których bierze udział na tym diagramie. I równowaznie, jezeli dana asocjacja jest spójna semantycznie, to spójne semantycznie sa równiez klasy na jej koncach.

Diagram jest spójny semantycznie jezeli własnosci pokazane na diagramie w żaden sposób nie wykluczaja sie wzajemnie. Wyraza to ponizsza definicja: *Diagram D jest spójny semantycznie wtedy i tylko wtedy, gdy dowolny klasyfikator cf jest spójny semantycznie w kontekście tego diagramu. W przeciwnym wypadku mówi sie, ze diagram jest niespójny semantycznie.*

# Walidacja i weryfikacja modeli

**Walidacja** – zapewnienie konstrukcji właściwego produktu, spełniającego potrzeby użytkownika systemu.

**Weryfikacja** – zapewnienie właściwej konstrukcji produktu, konstrukcji zgodnej ze specyfikacją.

Walidacja jest to proces wyznaczania stopnia, w jakim model jest wiernym odzwierciedleniem rzeczywistego systemu z przyjętego punktu widzenia. Ma na celu określenie, czy symulacja daje wiarygodne wyniki, w założonym stopniu zgodne z odpowiedziami rzeczywistego systemu na takie same dane wejściowe. O ile dzięki weryfikacji projektant uzyskuje informacje o zgodności systemu symulacyjnego z jego założeniami, o tyle walidacja weryfikuje zgodność jego wizji z realnym światem. Obie te fazy wzajemnie się uzupełniają i jako takie czasami przedstawiane są wspólnie jako faza oceny adekwatności modelu.

**Weryfikacja produktu:** patrz wyżej: MDA - Weryfikacja odbywa się podczas transformacji

**Walidacja produktu programowego**

* Wymagania funkcjonalne –konieczne
* Wymagania niefunkcjonalne dotyczące produktu (jakość produktu), dotyczące procesu wytwarzania
* Modele jakości ISO, IEEE, …
  + Perspektywy oceny, Charakterystyki jakości (atrybuty jakości) i ich miary, Kryteria oceny

# Różnice między wyszukiwaniem informacji a wyszukiwaniem danych

*Dane są reprezentacjami obiektów świata zewnętrznego wybranymi ze względu na potrzeby zapamiętania określonych faktów, zdarzeń, prawidłowości itp. lub ze względu na łatwość ich przetwarzania (np. reprezentacja cyfrowa, a nie literowa liczb). Aby uzyskać informację, tj. dane wraz z ich znaczeniem (sensem), dane muszą być zinterpretowane, odniesione do tego co reprezentują. Rozpatrując odwrotny proces zauważa się, że aby zapisać informację lub wiedzę należy dobrać odpowiednią jej reprezentację, tzn. przedstawić ją w postaci danych. Proces interpretacji danych wymaga posiadania określonej wiedzy o opisywanym świecie i o języku, w którym dane są zapisane. Wiedza pełni zatem aktywną rolę w procesie interpretacji danych, nadawania im znaczenia (sensu).*

Dane to liczby, fakty, pojęcia lub rozkazy przedstawione w sposób wygodny do przesyłania, interpretacji lub przetwarzania metodami ręcznymi lub automatycznie. Ale najbardziej wyczerpujące objaśnienie tego pojęcia brzmi następująco: Dane (ang. data) — to litery, słowa, teksty, liczby, znaki, symbole, kombinacje liter, liczb, symboli i znaków. Przedstawione np. za pomocą binarnego kodu ASCII, mogą być przechowywane w komputerze i poddawane obróbce przez różne programy. Powszechnie przez dane rozumie się dane dla programu komputerowego. Szczególnym przypadkiem zorganizowanego zbioru danych jest baza danych, w której w sposób uporządkowany są zgromadzone dane mające podobną postać i dotyczące określonej tematyki. Jak wynika z przytoczonych opisów dane nie są uporządkowane, przetworzone ani połączone zgodnie z celem i z potrzebami odbiorcy. Dane stanowią podstawę, surowy materiał do przetwarzania. Dane przetworzone otrzymują postać informacji gdy otrzymane wyniki z przetwarzania umożliwiają ich interpretację.

**Przetwarzanie danych** Dane przetwarza się po to by otrzymać informację, wiadomość. Informacja jest produktem finalnym przetwarzania danych. Wszystkie cząstkowe wyniki przetwarzania są półfabrykatem informacyjnym, czyli danymi do dalszego przetwarzania. Proces powstawania informacji jest podobny do etapów wytwarzania dóbr materialnych i trwa dotąd aż uzyska się wyrób finalny - wiadomość. A oto schemat tego procesu:

pozyskiwanie danych (surowca) przechowywanie danych przetwarzanie danych i poddanie wyników interpretacji uzyskanie informacji (produkt finalny) przechowywanie informacji przekazywanie informacji

**Własności danych i informacji**

Istotną cechą danych jest brak uporządkowania; jest to zbiór nieuporządkowany. Ale jednocześnie dana jest wiarygodna i pewna (przy odpowiednim przechowywaniu). Dane w każdej chwili mogą być weryfikowane pod względem poprawności i aktualizowane. Dane pełnią także rolę nośników przepływu informacji.

Informacja jest zbiorem uporządkowanym wg określonego kryterium i poddanym interpretacji. Cechą każdej informacji (jako wyniku interpretacji) jest jej niepewność i ograniczona w czasie trwałość (wiarygodność). W popularnym pojmowaniu informacji zwracamy uwagę na ilość informacji (dużo, mało). Z informacją wiążemy także pojęcie jakości, na którą składa się wierność przekazu, wiarygodność, szybkość i sposób archiwizowania.

Z powyższych rozważań wynika, że dane i informacje różnią się własnościami. Okazuje się, że z "morza danych" możemy uzyskać niewiele informacji i odwrotnie — z niewielkiej ilości danych można otrzymać znaczące informacje. Tak więc ilość informacji nie jest zależna do ilości danych.

**Wyszukiwanie danych:**

Wyszukiwanie danych można prowadzić na dwa sposoby:

•wyszukiwanie poprzez indeks – wybieramy pole po którym chcemy wyszukiwać i wpisujemy litery szukanego terminu

•wyszukiwanie bezpośrednie

**Systemy Wyszukiwania Informacji** Celem systemu wyszukiwania informacji jest dostarczenie użytkownikowi poszukiwanej przez niego informacji. Użytkownik, który ma szereg pytań, powinien na nie otrzymać odpowiedź w jak najkrótszym czasie

**Systemy Wyszukiwania Informacji** w skrócie SWI, to systemy informacyjne, których głównym zadaniem jest optymalizacja indeksowania zbioru termów w celu jak najszybszego i najbardziej trafnego (relatywnego) udzielenia odpowiedzi na zadane pytanie.

Wyszukiwanie Informacji w Systemie zależy od zastosowanej metody wyszukiwania. Stosowanie odpowiedniej metody zależy od odpowiedniego przygotowania (przetworzenia) bazy danych. Przetworzenie bazy danych polega na tworzeniu dokumentów w formacie wyszukiwawczym - dokumentów, które przechowują lub przygotowują informacje do procesu. Bazę danych można więc nazwać zbiorem dokumentów wyszukiwawczych czyli kartoteką danych wyszukiwawczych.

# Działanie systemu informacyjnego w sieci komputerowej

***System informacyjny*** *to zestaw współdziałających składników w przechowywania celu i gromadzenia, udostępniania przetwarzania, informacji, aby wspomagać podejmowanie decyzji, koordynowanie, sterowanie, analizowanie i wizualizację informacji.*

***System informatyczny*** *to taki system informacyjny, który wspomaga funkcjonowanie firm i instytucji z wykorzystaniem infrastruktury teleinformatycznej:*

**System informatyczny** To taki system informacyjny, który wspomaga funkcjonowanie firm i instytucji z wykorzystaniem infrastruktury teleinformatycznej. Zadanie systemu informacyjnego (SI):

● Zaspokojenie potrzeb informacyjnych organizacji

● Wsparcie komunikacji poziomej pomiędzy różnymi komórkami organizacyjnymi

● Wsparcie komunikacji pionowej pomiędzy przełożonymi a podwładnymi

● Ułatwienie komunikacji z otoczeniem

Rozwój technologii telekomunikacyjnych i komputerowych w ostatniej dekadzie, zwłaszcza niezwykła ekspansja WWW – stworzyły nowe możliwości rozpowszechniania i wymiany informacji, a tym samym przyczyniły się do powstania „nowoczesnych technik informacyjnych”. Sam System WWW, po wprowadzeniu standardów i metod pozwalających na opis i organizację multimedialnych danych, zawartych na stronach WWW i opracowaniu systemów wspomagających procesy wyszukiwawcze – ma szansę stać się globalnym systemem informacyjnym, efektywnie wykorzystywanym przez użytkowników końcowych do wyszukiwania informacji o wysokiej trafności. Jedną z cech powstającego społeczeństwa informacyjnego jest rozwój pojęcia telematyka, a więc niejako połączenia telekomunikacji i informatyki. Inaczej mówiąc są to sieciowe cyfrowe multimedia. Obecnie ważniejsze jest dotarcie do informacji niż jej gromadzenie, użytkownik ma możliwość bezpośredniego dostępu do informacji bez instytucji pośredniczących. Czas i przestrzeń dzięki połączeniom sieciowym nie odgrywają już takiej roli, jak niegdyś. Ważna jest także sprawa aktualności danych. W przypadku dokumentów drukowanych istnieje niebezpieczeństwo ich dezaktualizacji już w momencie opublikowania. Dokumenty elektroniczne w sieciach dostępne są w trybie czasu rzeczywistego. Poza tym istnieje możliwość aktualizowania danych na bieżąco.

# Technologie multimedialne stosowane w systemach informacyjnych

Podstawowe technologie multimedialne stosowane w systemach informacyjnych to:

• Elementy graficzne, takie jak wykresy, schematy, ilustracje, diagramy

• Elementy dźwiękowe: sygnały, podkłady muzyczne, głos ludzki (lektor)

• Elementy animowane: video, animacje

Zaawansowane systemy umożliwiają łaczenie tych metod, np w postaci webinariów (głos prowadzącego konferencję dociera do każdego użytkownika, który ma dostęp też do prezentowanych danych wizualnych – slajdów, animacji, obrazów, tekstu, etc.)

**Multimedialny system informacyjny** ma na celu integrację różnych narzędzi potrzebnych do pozyskiwania, przechowywania, indeksowania i wyszukiwania, edycji,zarządzania, przetwarzania oraz rozpowszechniania dużej ilości informacji(obiektów) multimedialnych.

Aby mówić o obiekcie multimedialnym lub po prostu o multimediach, należy zacząć od pojęcia media.

**Media** są rozumiane jako środki, za pomocą których informacja może być wyrażana, przekazywana, prezentowana lub przechowywana. Do mediów możemy zaliczyć tekst, dźwięk, obrazy, animację, wideo oraz hipertekst. **Hipertekst** jest to nielinearny sposób wyświetlania zbioru dokumentów tekstowych, którego istotą są zautomatyzowane odwołania noszące nazwy hiperłączy. **Multimedia** są często mylone z mediami, jest to dowolna kombinacja **dwóch lub więcej** mediów połączona w jeden obiekt.

Według Steinmetza kluczowe właściwości mediów to:

● ciągłe i dyskretne w zależności od czasu (ciągłe - film, dyskretne - obraz)

● niezależne od siebie nawzajem

● w przeciwieństwie do mediów, multimedia są tworzone i zarządzane przez aplikację komputerową

● multimedia są integralne, tworzą jedną całość ze zbioru mediów. Różnorodność typów mediów jest ważna cechą nowoczesnych systemów informacyjnych, jednak wiąże się z tym kluczowy problem - integracja mediów.

**Integracja mediów** jest to proces scalania różnych mediów prowadzący do uzyskania elementów multimedialnych. Systemy pozwalające na integrację mediów noszą nazwę systemów integracji mediów. Są to programy komputerowe pozwalające na tworzenie multimediów w oparciu o wcześniej zaprogramowane elementy. Każdy system integracji mediów działa w oparciu o pewną metodologię (model). Możemy wyróżnić następujące modele integracji mediów:

# Efektywność systemów informacyjnych

Efektywność opisuje stopień wykorzystania zasobów sprzętowych i programowych stanowiących podstawę działania systemu informacyjnego.

Zła organizacja i niska efektywność systemu skutkują wzrostem kosztów działalności organizacji, spadkiem jakości obsługi, utrudnionym dostępem do informacji na różnych szczeblach, ich znaczną niekompletnością, wydłużonym czasem generowania.

Jedną ze strategii zwiększenia efektywności pracy zespołów projektujących systemy informacyjne jest wielokrotne używanie raz opracowanych fragmentów projektu lub modułów programowych. Ponowne użycie wcześniej opracowanego fragmentu projektu albo modułu programu ma wiele zalet, wśród których na plan pierwszy wysuwają się: obniżenie kosztów, przyspieszenie realizacji projektu oraz minimalizacja błędów

# Zadania projektowania sieci komputerowej

**Warunki, które powinien spełniać dobry projekt sieci komputerowej:**

1. Realizacja oczekiwań zleceniodawcy,

2. Fachowa dokumentacja,

3. Możliwość rekonfiguracji i rozbudowy sieci,

4. Łatwość rekonfiguracji w przypadku awarii,

5. Niezależność uszkodzeń w różnych segmentach sieci,

6. Bezpieczeństwo danych i serwerów.

Etapy projektowe

**(1) Treść zadania i założenia ogólne** Dokładna treść zadania wraz ze wszystkimi uzgodnionymi założeniami oraz rysunkami poglądowymi; w razie konieczności uzupełniona planami schematycznymi budynków i pomieszczeń.

**(2) Założenia dotyczące przepustowości sieci** Założenia dotyczące wymaganej przepustowości wszystkich segmentów sieci (w tym łączy WAN), ze szczególnym uwzględnieniem charakterystyki ruchu i koncentracji serwerów. W ramach założeń warto uwzględnić telefonię Voice-Over-IP (VoIP) dla założonej maksymalnej ilości rozmów jednoczesnych.

**(3) Koncepcja rozwiązania i proponowane technologie** Propozycja wykonania sieci wraz z dokładnym wskazaniem technologii sieciowej wykonania każdego segmentu sieci. Rozwiązanie musi spełniać wszystkie wymagania etapów poprzednich oraz zapewniać właściwy poziom nadmiarowości. Należy krótko scharakteryzować zaproponowane technologie oraz uzasadnić ich wybór; warto także zaprezentować poglądowy schemat rozwiązania.

**(4) Proponowana topologia sieci** Prezentacja proponowanych topologii wszystkich segmentów sieci z uwzględnieniem wymagań proponowanych technologii oraz wymagań projektu. Propozycje powinny dotyczyć także połączeń dla sieci WAN oraz rozmieszczenia punktów dostępowych dla lokalnych sieci bezprzewodowych.

**(5) Plan adresacji sieci** Podział domen rozgłoszeniowych na VLAN-y zgodnie ze strukturą organizacyjną firmy lub instytucji. Propozycja adresacji IP projektowanej sieci ze wskazaniem numerów wszystkich sieci i ich masek; bez konieczności wskazywania adresów IP poszczególnych komputerów.

**(6) Dobór sprzętu** Szczegółowe wskazania dotyczące niezbędnego sprzętu aktywnego sieci dla realizacji poczynionych wcześniej założeń i propozycji. Dobór powinien uwzględniać możliwość wzrostu ilości węzłów sieci; całość należy zaprezentować także w postaci graficznej. Specyfikacja urządzeń Dokładna specyfikacja wybranych urządzeń aktywnych, wraz z ich krótkim opis i charakterystyką oraz szczegółowym numerem katalogowym producenta (i specyfikacją modułów dla urządzeń modułowych lub niezbędnymi konfiguracjami sprzętowymi). Wskazania urządzeń powinny umożliwiać ich jednoznaczną identyfikację.

**(7) Projekt okablowania** Dokładny projekt okablowania strukturalnego sieci z uwzględnieniem wszystkich norm i wymogów. Projekt powinien obejmować wskazanie traktów kablowych z obliczonymi długościami kabli (w ramach budynków i pomiędzy nimi), rozmieszczenie punktów dystrybucyjnych oraz gniazdek abonenckich. Punkty dystrybucyjne Projekt wszystkich punktów dystrybucyjnych, obejmujący dokładne wskazania rozmieszczenia wszystkich urządzeń oraz paneli i innych wymaganych oraz zalecanych elementów w szafach dystrybucyjnych. W projektach szaf należy uwzględnić wymagania producentów urządzeń aktywnych. Wszystkie szafy punktów dystrybucyjnych powinny być przedstawione w postaci rysunków schematycznych.

**(8) Plan zarządzania siecią** Podstawowa koncepcja zarządzania siecią ze wskazaniem preferowanej platformy zarządzania, właściwych protokołów oraz uzasadnieniem wykonanych wyborów.

**(9) Kosztorys** Szacunkowy kosztorys wszystkich elementów projektowanej sieci komputerowej: elementów aktywnych sieci, elementów okablowania strukturalnego oraz wszelkich opłat dotyczących połączeń WAN.

Zasady:

1. Nie wolno osiągać granic możliwości sieci/sprzętu.

2. Maksymalna długość kabla dla 5 kategorii: 3m do komputera, 90m kabla poziomego, 6m kabla krosującego.

3. Zgodnie z ISO: 10 metrów kwadratowych na miejsce pracy (nie oznacza to 1 gniazdka na 10m2, ale co najmniej 1 gniazdko na 10m2).

4. Każda kondygnacja musi być wyposażona w minimum 1 punkt dystrybucyjny. W przypadku pomieszczeń o powierzchni większej od 1000m2 lub w przypadku, kiedy okablowanie poziome przekracza 90 m należy wprowadzić dodatkowy punkt dystrybucyjny.

5. Powierzchnia punktu dystrybucyjnego PD (SPD) w zależności od obsługiwanej powierzchni (S):

- S 1000 m2 -> SPD min. 3.0x3.4m,

- S 800m2 -> SPD min. 3.0x2.8m,

- S 500m2 -> SPD min. 3.0x2.3m.

6. Maksymalna dozwolona ścieżka sygnału obejmuje 5 segmentów kabla połączonych 4 hubami. W takim przypadku 2 z tych segmentów mogą być użyte wyłącznie jako połączenia między hubami. Jeśli się nie da inaczej, należy podzielić sieć na domeny kolizyjne (podsieci) i wprowadzić switche.

7. Maksymalnie 1024 urządzenia na podsieć.

8. Maksymalna całkowita odległość w podsieci 500m.

# Klasyfikacja ruchu teleinformatycznego

Ruchem telekomunikacyjnym nazywamy przepływ zgłoszeń, połączeń i wiadomości. W przypadku sieci pakietowych, w zależności od warstwy modelu OSI, na którym poziomie dokonujemy obserwacji ruchu, definicję tę można rozszerzyć na przepływ innych jednostek transmisji danych (pakietów – np. pakietów TCP, datagramów– np. datagramów IP, ramek – np. ramek Ethernet, komórek – np. komórek ATM (Asynchronous Transfer Mode), itd.). Każda aplikacja przesyłająca dane generuje wsieci pewien ruch. W przypadku sieci jednousługowej (np. sieci telefonii analogowej), ruch generowany przez poszczególnych użytkowników ma charakter homogeniczny. W przypadku sieci wielousługowej (np. Internet) ruch telekomunikacyjny jest ruchem heterogenicznym.

**W sieci Internet składa się na niego m.in.:**

● ruch generowany podczas transmisji danych masowych (np. z wykorzystaniem usługi ftp przesyłania plików) – charakteryzuje się on długim czasem trwania połączenia transportowego), związanym z przesyłaniem długich wiadomości (plików zawierających dane masowe), odstępy pomiędzy połączeniami transportowymi realizowanymi przez tego samego użytkownika są stosunkowo długie;

● ruch generowany podczas przesyłania poczty elektronicznej – charakteryzuje się on stosunkowo krótkimi wiadomościami tekstowymi (procentowy udział długich wiadomości, zawierających załączniki w postaci plików z danymi czy skompresowanych obrazów jest niewielki), przesyłanymi w bardzo dużych odstępach czasu;

● ruch generowany przez usługę zdalnego terminala (np. telnet), charakteryzujący się krótkimi wiadomościami, przesyłanymi w relatywnie krótkich odstępach czasu;

● ruch generowany przez usługę WWW charakteryzuje się krótkimi wiadomościami (tekst, niewielkie obrazy – np. logo firmy, małe elementy graficzne), przesyłanymi w bardzo krótkich odstępach czasu (związanymi z przesyłaniem zawartości strony WWW), po których następują relatywnie długie okresy braku aktywności (związane z przeglądaniem zawartości stron przez użytkownika); podobnie, jak w przypadku poczty elektronicznej, procentowy udział długich wiadomości jest niewielki;

● ruch generowany przez aplikacje realizujące transmisję informacji multimedialnej w czasie rzeczywistym – charakteryzuje się przesyłaniem bardzo krótkich (np. transmisja głosu) lub bardzo długich wiadomości (np. transmisja ramek wideo), lub ich złożenia (np. równoczesna transmisja głosu, obrazu, wiadomości tekstowych); cechą charakterystyczną źródła ruchu multimedialnego jest generowanie wiadomości w stałych odstępach czasu (np. generowanie ramek wideo co 40 ms lub co ok. 33 ms).

# Zarządzanie zasobami sieci komputerowej

**Zarządzanie zasobami sieci jest ważnym** elementm sterowania ruchem w sieci komputerowej. Szczególnie związane jest z mechanizmami zapewnienia jakości oraz przeciwdziałania przeciążeniom. Sposób zarządzania różni się w zależności od konkretnej koncepcji realizacji uslug w sieci. Np. w koncepcji usług zintegrowanych, zarządzanie zasobami związane jest z udostępnianiem, rezerwacją i przydzialem zasobów. Może również polegać na szeregowaniu pakietów (ang. scheduling) oraz zarządzanie pamięcią buforową i kolejkami (ang. queue/buffer management).

**Szeregowanie pakietów** Zadaniem szeregowania w lokalnych mechanizmach przeciwdziałania przeciążeniom jest ustalanie kolejności obsługi (transmisji) pakietów. Dyscyplina szeregowania steruje alokacją pojemności łącza przez wybieranie do obsługi określonej liczby pakietów każdego połączenia w ustalonym przedziale czasu. Szeregowanie jest jedynym efektywnym sposobem zarządzania zasobami transmisyjnymi i najczęściej implementuje się wraz z różnymi strategiami zarządzania pamięcią buforową lub zarządzania kolejkami.

Wybór dyscypliny szeregowania pakietów, implementowanej w mechanizmie przeciwdziałania przeciążeniom, zależy od kilku czynników, w tym m.in. od:

- liczby poziomów priorytetów, którym odpowiada liczba oddzielnych kolejek pakietów w routerze,

- kolejności obsługi pakietów o tym samym poziomie priorytetu,

- stopnia agregacji przepływu.

**Zarządzanie pamięcią buforową i kolejkami** Celem metod zarządzania pojemnością pamięci buforowej jest podział dostępnej puli buforów pomiędzy przepływy współzawodniczące o ten zasób sieci. Opracowano i wdrożono wiele różnych strategii podziału pamięci buforowych charakteryzujących się różną dynamiką alokacji, różnymi kryteriami efektywności alokacji i zajętości pamięci, granulacją przepływów itd. Podstawowym kryterium podziału metod alokacji pojemności pamięci buforowej jest sposób udostępniania pamięci poszczególnym przepływom: współdzielenie całkowitej pojemności przez wszystkie przepływy (ang. shared buffer pool) lub dedykowanie pojemności dla przepływów (ang. per-flow allocation). Celem metod zarządzania kolejkami jest sterowanie długością kolejek i przepływami, które kolejkę zajmują. Sterowanie to polega na wyborze pakietów do odrzucenia i czasu ich  odrzucania. Metody zarządzania kolejkami są komplementarne dla dyscyplin szeregowania (decydujących o kolejności obsługi pakietów oczekujących w kolejkach) i metod zarządzania pamięcią buforową (decydujących o liczbie organizowanych kolejek i agregacji przepływów w kolejkach). Kryterium ogólnego podziału metod zarządzania jest zdolność do zapobiegania przeciążeniom lub regeneracji przeciążenia; z tego względu wyróżnia się dwie obszerne grupy metod zarządzania kolejkami:

- **reaktywne**, tzn. regenerujące przeciążenia,

- **prewencyjne**, tzn. zapobiegające powstawaniu przeciążeń.

Najczęściej stosowane rozwiązania z grupy metod **reaktywnych** polegają na **odrzucaniu pakietów w razie przepełnienia kolejki**. Różne przykładowe odmiany metod odrzucania pakietów to:

- odrzucanie „ogonów” (ang. tail drop) – przepełnienie kolejki powoduje odrzucanie pakietów w kolejności ich napływania do routera,

- odrzucanie z czoła kolejki (ang. drop-from-front) – przepełnienie kolejki powoduje odrzucanie pakietów znajdujących się na czele kolejki,

- odrzucanie losowe (ang. random drop) – nadejście nowego pakietu i przepełnienie kolejki uruchamia procedurę losowania i odrzucania wylosowanego pakietu.

Metody **prewencyjne** to tzw. aktywne metody zarządzania kolejkami. Cechą charakterystyczną tej grupy metod jest to, że – w przeciwieństwie do metod grupy m. reaktywnych  – odrzucanie pakietów nie jest jedynym sposobem powiadamiania źródła o przeciążeniu lub wystąpieniu symptomów przeciążenia; funkcje takie może pełnić **znakowanie pakietów**.

Do zadań metod prewencyjnych, oprócz zapobiegania przeciążeniom, zalicza się także:

eliminację skutków „wybuchowości” źródeł, gwarancję zachowania górnych ograniczeń długości kolejek, nawet w przypadku obecności tzw. niewspółpracujących przepływów (ang. noncooperative flows), karanie (ang. penalize) tzw. zachłannych przepływów (ang. aggresive flows), redukcję liczby odrzucanych pakietów, gwarancję ograniczonych z góry opóźnień pakietów.

W metodach prewencyjnych powszechnie stosowanym mechanizmem detekcji przeciążenia lub jego symptomów jest znakowanie pakietów. Właściwości poszczególnych metod zależą od stosowanej miary przeciążenia i funkcji znakowania pakietów, których parametrami są wartości miar przeciążenia.

Metody prewencyjne są podstawowymi rozwiązaniami zarządzania pamięcią w sieciach z różnicowaniem usług.

**Planowanie i udostępnianie zasobów** (ang. resource provisioning) można zaliczyć zarówno do prewencyjnych, jak i do reakcyjnych metod zarządzania zasobami i ich udostępniania. W zasadzie są metodami planowania zasobów. Zadania zarządzania zasobami mają charakter prewencyjny wtedy, gdy sieć i jej zasoby są planowane. Na podstawie rozpoznanych i zidentyfikowanych wymagań aplikacji, liczby użytkowników i charakterystyk przewidywanego ruchu definiowana jest struktura topologiczna sieci, liczba kanałów transmisyjnych i ich pojemności, liczba węzłów komutacyjnych i wartości ich parametrów, liczba węzłów dostępowych, nadmiarowość zasobów transmisyjnych i komutacyjnych itd. Zmiana aplikacji, liczby użytkowników, wymagań ilościowych, jakościowych itp. powoduje, że niezbędne są zmiany w zasobach sieci. Zmiany te polegają na zmianach struktury topologicznej, liczby kanałów transmisyjnych i ich przepustowości, liczby i rodzaju węzłów komutacyjnych, liczby i parametrów węzłów dostępowych itd.

\*\*\*

**Typy sieci ze względu na zarządzanie zasobami:**

Sieci równorzędne (każdy-z-każdym) P2P (od ang. peer-to-peer – równy z równym) – model komunikacji w sieci komputerowej, który gwarantuje obydwu stronom równorzędne prawa (w przeciwieństwie do modelu klient-serwer). W sieciach P2P każdy komputer może jednocześnie pełnić zarówno funkcję klienta, jak i serwera. Każdy węzeł sieci (czyli komputer użytkownika) odgrywa rolę serwera przyjmując połączenia od innych użytkowników danej sieci, jak i klienta, łącząc się i pobierając dane z innych maszyn działających w tej samej sieci. Wymiana danych jest zawsze prowadzona bez pośrednictwa centralnego serwera.

Sieci oparte na serwerach (klient-serwer) W sieciach klient-serwer zasoby często udostępniane gromadzone są w komputerach odrębnej warstwy, zwanych serwerami. Serwery zwykle nie mają użytkowników bezpośrednich. Są one raczej komputerami wielodostępnymi, które regulują udostępnianie swoich zasobów szerokiej rzeszy klientów. W sieciach tego typu zdjęty jest z klientów ciężar funkcjonowania jako serwery wobec innych klientów.

**Narzędzia do zarządzania zasobami**: procesory, pamięć, dyski, połączenia sieciowe itp. Narzędzia pozwalające na równoważenie obciążenia w środowisku heterogenicznym systemów komputerowych: algorytmy rozdziału zasobów, języki i sposoby (narzędzia) specyfikacji zasobów, mapowanie zasobów do aplikacji. Funkcjonalność zarządzania zasobami jest ograniczona i charakteryzuje się takimi wadami jak:

1. niewykorzystanie zasobów obliczeniowych w czasie oczekiwania na wszystkie żądane zasoby;
2. konieczność upewnienia się, czy komponenty aplikacji nie rozpoczną działania zanim moduł przydzielający zasoby określi czy żądanie się powiedzie.

# Metody naprawiania błędów w systemach teleinformatycznych

W systemach teleinformatycznych poszczególne komponenty komunikują się ze sobą przesyłając wiadomości. Komunikat zostaje zakodowany przez nadawcę do postaci umożliwiającej jego transport poprzez kanał komunikacyjny a następnie zdekodowany u odbiorcy. Kanałem komunikacyjnym może być linia telefoniczna, przewód światłowodowy, itp. Medium to jest nie jest niezawodne i w czasie transportu kanałem może dojść do zniekształcenia komunikatu na skutek zakłóceń. Aby system teleinformatyczny działał w sposób niezawodny błędy w wiadomościach muszą być wykrywane i naprawiane.

* **Detekcją błędu** nazywamy zdolność do zdecydowania czy otrzymany komunikat jest poprawny bez posiadania oryginalnej wiadomości,
* **Korekcją błędu** nazywamy rekonstrukcję komunikatu do jego poprawnej, pozbawionej błędów postaci.

**Korekcja błędu** jest zawsze poprzedzona detekcją błędu, ponadto usunięcie błędu jest zadaniem trudniejszym niż jego wykrycie. Korekcję błędu można osiągnąć **na dwa sposoby**:

* **(Automatic repeat request)** Ponowne żądanie i przesłanie na nowo całego komunikatu,
* **(Forward error correction)** Wykorzystanie dodatkowych (nadmiarowych) danych wysyłanych wraz z komunikatem. Dane te umożliwiają korekcję u odbiorcy, wówczas nie ma potrzeby informowania nadawcy o błędzie, żądania retransmisji i ponownego wysyłania komunikatu.

**Wykrywanie błędów:**

* **Kodowanie powtarzające** – polega na kilkukrotnym, np. 3-krotnym, wysłaniu bloku 4 bitów. Odbiorca porównuje wszystkie otrzymane zestawy. Jeśli jeden z nich różni się od pozostałych, oznacza to wystąpienie błędu w transmisji. Metoda ta jest bardzo prosta, ale nieefektywna a także nieodporna na błędy, np. wszystkie zestawy mogą ulec tej samej zmianie a błąd pozostanie niewykryty.
* **Kontrola parzystości** (bit parzystości) – najstarsza i najprostsza metoda. Bit parzystości dodawany jest do wiadomości i wskazuje czy liczba bitów o wartości 1 w komunikacie jest parzysta czy nie. Jeśli liczba ta jest parzysta, bit parzystości ma wartość równą 1, w przeciwnym wypadku 0. Odbiorca po odebraniu komunikatu sprawdza parzystość wiadomości i porównuje z wartością w bicie kontrolnym determinując poprawność wiadomości. Bit *nieparzystości* jest komplementarny do bitu parzystości, tj. przyjmuje odwrotne wartości – 1 w przypadku nieparzystej liczby jedynek, 0 w przeciwnym przypadku. Metoda ta nie jest odporna na przekłamania w bicie kontrolnym, np. wiadomość dotarła nieuszkodzona, ale bit parzystości uległ zmianie.
* **Sumy kontrolne** – rozwinięcie idei kontroli parzystości. Blok danych przed wysłaniem staje się argumentem pewnej funkcji, która w wyniku zwraca liczbę. Wartość ta jest przed transmisją dołączana do danych. Odbiorca wylicza wartość funkcji dla odebranych danych i porównuje z wartością oryginalną. Jest wiele algorytmów i funkcji używanych do wyliczania sum kontrolnych, wśród nich wymienić należy: funkcje skrótu MD5, SHA-1, SHA-2 i SHA-3 oraz algorytm Luhna weryfikujący poprawności numerów, np. kart kredytowych.
* **Cykliczny kod nadmiarowy (CRC)** - blok danych traktowany jest jako wielomian, dzielony modulo 2 przez pewien wielomian, zwykle szesnastego stopnia (np. x16 + x15 + x5 + 1). Otrzymana reszta z dzielenia stanowi 16-bitową sekwencje kontrolną, przesyłaną na końcu bloku.
* **Kody korekcyjne** (umożliwiają zarówno detekcję jak i korekcję)

Przy kodach korekcyjnych wprowadza się pojęcie odległości Hamminga dla słów kodujących, przykładowo binarnie wyrażone liczby 15 (01111) i 16 (10000) mają odległość Hamminga równą 5, a zestaw liczb 8, 15 i 16 mają *minimalną* odległość Hamminga równą 3  - pomiędzy 8 (01000) a 15 (01111).

* + **Kody Hamminga**

Kody Hamminga również adaptują ideę bitu parzystości, jednak w kodowaniu tym bitów kontrolnych jest więcej niż jeden. W ogólności kodowanie odbywa się za pomocą słów kodowych o długości n bitów, w tym m bitów danych i r bitów kontrolnych. Przy użyciu zestawu słów kodowych o *minimalnej* odległości Hamminga równej 3 można **wykryć** 3 – 1 = **2** błędy oraz **poprawić** (3 – 1)/2 = **1** błąd (błąd w pojedynczym bicie).

**Korekcja błędów**:

* **Automatyczna retransmisja**
  + Retransmisja pojedyncza (Stop-and-wait ARQ) – odbiorca wysyła komunikat i oczekuje na przysłanie potwierdzenia (ACK). Po otrzymaniu potwierdzenia wysyła kolejny komunikat. Jeśli w wyznaczonym czasie potwierdzenie nie dotrze (timeout - komunikat od nadawcy nie dotarł do odbiorcy lub potwierdzenie zostało zagubione) nadawca ponawia wysłanie wiadomości.
  + Retransmisja ciągła grupowa (Go-Back-N ARQ) – Wykorzystuje się przesuwne okno oraz numerowanie kolejnych komunikatów. Komunikaty nie muszą być potwierdzane indywidualnie. Przykładowo przy oknie długości 8 można przesłać maksymalnie 8 wiadomości bez otrzymania potwierdzenia. Odbiorca oczekuje na wiadomości o kolejno wzrastających numerach, tzn. po odebraniu wiadomości o nr 4 oczekuje na wiadomość nr 5 ignorując wszystkie duplikaty o wcześniejszych numerach i wiadomości „przyszłe”. W przypadku przedłużającego się oczekiwania wysyła prośbę o wiadomość nr 5 przesyłając **ACK o nr 4**. Wówczas nadawca, jeśli wysłał już wszystkie 8 wiadomości z okna, wraca do ostatniego potwierdzonego i wysyła ponownie wiadomości od nr 5 do nr 8. Dopiero po potwierdzeniu wiadomości nr 8 nadawca przesuwa okno i kontynuuje wysyłanie kolejnych 8 komunikatów.
  + Retransmisja ciągła selektywna (Selective Repeat ARQ) – odbiorca posiada bufor na komunikaty i przyjmuje je niezależnie od kolejności nadejścia oraz potwierdza odbiór każdego z osobna. Ponawia prośbę o wysłanie jedynie tych komunikatów, które nie dotarły w określonym czasie.
* **Kody korekcyjne –** zwykle dzielone na kody konwolucyjne i blokowe:
  + Kody konwolucyjne – w podejściu tym wiadomości koduje się bit po bicie, często realizowane sprzętowo,
  + Kody blokowe – przetwarzanie odbywa się blok po bloku, przykładem są wspomniane już kody Hamminga oraz np. kodowanie Reeda-Solomona.
* **Podejście hybrydowe –** łączy automatyczną retransmisję z danymi korekcyjnymi, istnieją dwa podstawowe podejścia:
  + Wiadomości są wysyłane razem z danymi do detekcji  błędów i korekcji danych. Odbiorca po zdekodowaniu wiadomości prosi o retransmisję tylko wtedy, gdy dane naprawcze są niewystarczające do odtworzenia wiadomości.
  + Wiadomości są wysyłane tylko z danymi do detekcji błędów. Jeśli błąd jest wykryty, odbiorca prosi o transmisję danych naprawczych i na ich podstawie odtwarza komunikat.

\*\*\*

Do detekcji i korekcji pojedynczych błędów transmisji stosuje się blokowe sekwencje znaków kontrolnych. Powszechnie stosowaną korekcją jest sekwencja BCC (Block Check Character) przedstawiająca znak lub sekwencję znaków generowaną przez algorytm kontrolny przed wysłaniem wiadomości w łącze transmisji danych. Urządzenie odbiorcze porównuje odtworzoną sekwencję kontrolną z sekwencją odebraną, aby stwierdzić, czy wystąpiły błędy transmisji.

Przy korekcji CRC blok informacyjny traktuje się jako wielomian, który w nadajniku dzieli się modulo 2 przez wielomian CRC, zwykle szesnastego stopnia (CCITT zaleca kilka, popularnym jest x16+x12+x5+1). Otrzymana reszta tworzy 16-bitową sekwencję kontrolną FCS (Frame Check Sequence) transmitowaną na końcu bloku. W odbiorniku odebrany blok informacyjny również dzieli się przez taki sam wielomian. Przez porównanie otrzymanej reszty z dzielenia z odebraną sekwencją kontrolną można stwierdzić wystąpienie błędu transmisji. Brak zgodności sekwencji wymusza przesłanie odpowiedniej informacji kanałem sprzężenia powrotnego i retransmisję błędnych bloków.

# Koncepcje dostarczania jakości usług w sieciach teleinformatycznych

**Jakość usług** (ang. Quality of Services – QoS) w sieciach teleinformatycznych dotyczy klasyfikacji generowanego ruchu (ang. traffic) lub przepływów (ang. flow) w celu umożliwienia traktowania ich w szczególny – w porównaniu z innymi klasami ruchu – sposób. Pozwala na różnicowanie klas ruchu i usług dostarczanych różnym klasom ruchu w sieci. Jest to zdolność sieci (lub jej elementów) do gwarantowania konsekwentnego dostarczania usług dla wyróżnionych klas ruchu na określonym poziomie.

**Uzyskanie i utrzymanie wymaganej jakości aplikacji** w sieciowym systemie informatycznym, cechującym się dynamicznymi zmianami ilości dostępnych zasobów, jest możliwe dzięki:

* **Gwarancji jakości usług sieci dla aplikacji** (ang. quality of service assurance) - rezerwacja zasobów sieciowych dla aplikacji. Redukuje zmienność dostępnych zasobów z punktu widzenia aplikacji.
* **Dopasowaniu aplikacji do zmienności dostępnych zasobów sieci** (ang. network-aware applications) – zmiana jakości usług sieci powoduje zmianę zapotrzebowania aplikacji na zasoby. Wymaga posiadania przez aplikacje zdolności do pozyskiwania informacji o dostępności żądanych zasobów i stanie sieci (sieci muszą udostępnianiać takie informacje ich węzłom).

**Jakość aplikacji** w sieciowym systemie informatycznym zależy od:

* **Węzły i ich liczba** – realizacja aplikacji w systemie sieciowym wymaga wyboru węzłów, spełniających specyficzne wymagania, oraz liczby węzłów.
* **Architektura sieci** – specyfika aplikacji może powodować, że różne rozwiązania sieciowe są w rozmaitym stopniu przydatne do jej obsługi. Żądania dystrybucji danych w trybie rozgłoszeniowym, obsługi procesów czasu rzeczywistego, obsługi transakcji itp. dyskryminują jedne, a preferują inne architektury sieciowe.
* **Równoważenie obciążeń** – zmiany środowiska sieciowego mogą powodować konieczność z zastosowanie innych warunków realizacji aplikacji, tj. zmiany węzłów i ich liczby oraz wynikającej z tego reorganizacji połączeń komunikacyjnych, jeżeli aplikacje nie jest elastyczna w tym względzie.
* **Miary jakości aplikacji** – warunkiem realizacji różnych aplikacji jest spełnienie charakterystycznych dla nich wymagań (wrażliwość na zmienność opóźnienia w aplikacjach odtwarzania, synchronizacja przekazu w aplikacjach multimedialnych itd.). W danej sieci można wieloma różnymi sposobami spełniać specyficzne wymagania aplikacji.
* **Lokalne i zdalne przetwarzanie** – w zależności od dostępności usług sieci oraz mocy obliczeniowych dla ustalonego modelu kosztów część funkcjonalności aplikacji można w różnym stopniu przetwarzać zdalnie oraz lokalnie.

**Podstawowe koncepcje dostarczania jakości usług** w sieciach teleinformatycznych, ze względu na występujące **mechanizmy**:

* **rezerwacji zasobów (usługi zintegrowane)** – dostępne zasoby sieci komputero wej są dzielone pomiędzy aplikacje (i ruch generowany przez nie) żądające usług o określonej jakości; zasoby sieci są rozdzielane pomiędzy źródła ruchu aż do chwili wykorzystania wszystkich dostępnych zasobów (w ramach ograniczeń wynikających z procedur zarządzania przepustowością),
* **priorytetowania ruchu (usługi zróżnicowane)** – ruch generowany i przekazywany w sieci komputerowej jest klasyfikowany zgodnie z przyjętą (określoną) w sieci hierarchią ruchu; przydział zasobów odbywa się zgodnie z przyjętymi procedurami zarządzania przepustowością, pozwalającymi na preferowanie klas ruchu charakteryzowanych wyższymi wymaganiami co do jakości usług.

Koncepcje te związane są z dwiema kategoriami sterowania ruchem: prewencyjny (ang. preventive control) i reakcyjny (ang. reactive control). W systemach jakości usług opartych na mechanizmach rezerwacji zasobów stosowane są długoterminowe procedury sterowania napływem ruchu, zawierające metody sterowania zarówno prewencyjnego, jak i reakcyjnego. W systemach jakości usług opartych na priorytetowaniu ruchu główne zastosowanie mają krótkoterminowe mechanizmy kształtowania ruchu, które zalicza się do metod sterowania reakcyjnego.

Realizację koncepcji **usług zintegrowanych** charakteryzują następujące założenia:

* gwarancja jakości usług sieci jest możliwa dzięki **rezerwacji zasobów sieci dla każdego z przepływów** przyjętych do obsługi w sieci,
* efektywne wykorzystanie zasobów sieci jest możliwe przez **przydzielanie zasobów przepływom na najniższym możliwym poziomie granulacji**, tj. elementarnym przepływom pomiędzy źródłami i ujściami sieci,
* **dostępność zasobów jest warunkiem koniecznym, ale nie wystarczającym**, gwarancji jakości usług sieci.

Model funkcjonalny obsługi wywołań w takiej sieci składa się z dwóch poziomów:

* **poziom sterowania** - przygotowanie i **obsługa żądania ustanowienia rezerwacji,** konfiguracja procedury sterowania dostępem, konfiguracje parametrów modułów identyfikacji (klasyfikacji) jednostek przepływół
* **poziom przepływów** - identyfikacja przepływów, polegającej na **klasyfikacji (filtrowaniu) jednostek danych** należących do obsługiwanego wywołania; szeregowanie przepływów jednostek danych, polegającego na wymuszaniu **alokacji zarezerwowanych zasobów** do obsługiwanych przepływów.

Podstawę koncepcji **usług zróżnicowanych** stanowi założenie, że przedmiotem sterowania jest **ograniczona**, w porównaniu z liczbą wszystkich przepływów elementarnych, **liczba klas ruchu** w sieci. Sterowanie, polegające na **ograniczaniu ilości wprowadzanego ruchu**, eliminuje potrzebę stosowania procedur rezerwacji zasobów sieci. Ograniczona i stała lub wolnozmienna liczba klas ruchu uniezależnia rozmiar zadania sterowania ruchem w sieci od liczby aktualnie obsługiwanych przepływów elementarnych. Architektura usług zróżnicowanych opiera się na prostym modelu, w którym ruch wprowadzany do sieci jest odpowiednio **klasyfikowany** i następnie **obsługiwany według określonego kontraktu ruchowego** w każdym z węzłów sieci. Niezależne strumienie ruchu obsługiwane w tej samej klasie są agregowane i otrzymują ten sam poziom jakości usług.

**Warunkami efektywnej implementacji koncepcji usług zróżnicowanych** jest rozwiązywanie, w czasie obsługi jednostek danych należących do wyróżnionych klas ruchu, następujących zadań:

* **klasyfikacji jednostek danych** – kwalifikowanie jednostek danych do agregatów ruchu oznacza konieczność identyfikacji każdej jednostki danych (zadanie klasyfikacji),
* **wymuszania ograniczeń na ilość ruchu w sieci dla każdej z klas** – jakość usług dla każdej z klas ruchu zależy od ilości ruchu danej klasy w sieci oraz od całkowitej ilości ruchu w sieci (zadania monitorowania, kształtowania i odrzucania ruchu),
* **różnicowania sposobów obsługi klas różnych klas ruchu w różnych domenach** administracyjnych sieci – zadania klasyfikacji i nadzorowania ruchu różnią się w zależności od miejsca w sieci, w którym są realizowane.

\*\*\*

**Quality of Service – w skrócie QoS** – czyli gwarantowana jakość usług i kształtowanie ruchu w sieciach komputerowych są pojęciami nierozerwalnymi. Kształtowanie ruchu (ang. traffic shaping) jest bowiem jednym z podstawowych mechanizmów mających za zadanie poprawić działanie sieci.

**Qulity of services** – jest gwarancją jakości usług. Mówiąc o QoS mamy najczęściej na myśli zbiór parametrów dzięki którym możemy określić czy oferowane nam usługi są na wysokim poziomie czy nie i w konsekwencji czy daną sieć będziemy mogli wykorzystać do pewnych szczególnych zastosowań.

**Prędkość transmisji** (przepustowość)- oznacza ile danych można przesłać między dwoma punktami w jednosce czasu. Mierzona jest w bit na sekundę.

**Straty pakietów** – kiedy współdzielone łącze jest w coraz większym stopniu wykorzystywane, kolejki w urządzeniach sieciowych zaczynają się wypełniać pakietami w konsekwencji nawet odrzucać kolejne pakiety. Straty pakietów mogą być wyrażone procentowo – ilość zgubionych paczek danych przez ilość wysłanych. Strata pakietów często ma miejsce, kiedy węzeł łączy dwa segmenty sieci o różnej prędkości transmisji.

**Opóźnienie –** Oznacza czas w jakim pakiety wędrują pomiędzy dwoma wyznaczonymi węzłami sieci, czyli czas przez jaki pakiet znajduje się w sieci zanim dotrze do punktu przeznaczenia.Parametr ten jest szczególnie istotny dla aplikacji interaktywnych taki jak Telnet, a także służących do transmisji głosu np. w telefonii IP czy wideokonferencjach. Opóźnienia pojawiają się przede wszystkim, gdy pakiety oczekują na obsługę wewnątrz kolejek napotykanych na swej drodze routerów.

**Zmienność opóźnienia** Główną przyczyną powstawania zmienności opóźnień przy transmitowaniu pakietów jest przechowywanie ich w kolejkach routerów. Gdy kolejki te są puste wówczas router przesyła pakiety natychmiast. Przy zapełnianiu się kolejek opóźnienia rosną. Wzrosty opóźnień są szczególnie zauważalne dla długich kolejek, jednak ustalanie zbyt krótkich buforów może prowadzić do zwiększenia współczynnika gubienia pakietów. Zmienność opóźnienia jest też konsekwencją faktu, że pakiety mogą przechodzić przez sieć różnymi drogami, a tak zawsze może się zdarzyć gdy topologia sieci pozwala połączyć w niej

**Mechanizmy quality of services –** mechanizmy gwarancji usług

**Zapobieganie zatorom** – najczęściej zatory tworzą się na węzłach np. routerach. Zadaniem takiego węzła jest wczesne odrzucanie pakietów w celu przeciwdziałania temu zjawisku. **Kształtowanie ruchu** - polega na wygładzeniu ruchu który wypływa z węzła do sieci, tak żeby odbiorą transmisji nie został zalewany dużą ilością pakietów. Odpowiednie algorytmy kolejkowania – który w sposób równomierny wyprowadzi pakiety z routera jest dobrym rozwiązaniem.

**Klasyfikacja i kolejkowanie** – ściśle łączy się z kształtowanie ruchu.

**Rezerwowanie łącza** – istnieją protokoły które potrafią zamówić w poszególnych węzłach pewne poziomy wymaganych parametrów. Technika ta jest trudna do implementacji w sieci Internet.

**Znakowanie pakietów –** dołączanie do pakietów informacji określających ścieżki którymi pakiet ma wędrować jak również priorytety pakietów

**OGÓLNE W PKT CO SKŁADA SIE NA KONCEPCJE:**

* kształtowanie i ograniczanie przepustowości
* zapewnienie sprawiedliwego dostępu do zasobów
* nadawanie odpowiednich priorytetów poszczególnym pakietom wędrującym przez sieć
* zarządzanie opóźnieniami w przesyłaniu danych
* zarządzanie buforowaniem nadmiarowych pakietów: DRR, WFQ, WRR
* określenie charakterystyki gubienia pakietów
* unikanie przeciążeń: Connection Admission Control (CAC),

# Pojęcie systemu decyzyjnego oraz komputerowego systemu wspomagania decyzji

W problematyce systemów decyzyjnych oraz systemów wspomagania decyzji wykorzystuje się następujące podstawowe pojęcia: **DECYZJA** – świadomy (nielosowy) wybór jednego z wariantów przyszłego działania; wynik działania czynności nazywanej „podejmowaniem decyzji”.

**PODEJMOWANIE DECYZJI** – proces polegający na zbieraniu i przetwarzaniu informacji o przyszłym działaniu (definicja wg. Koźmińskiego).

**DECYDENT** – podmiot podejmujący decyzję.

**PRZEDMIOT DECYZJI** – rzeczywistość (proces, system, obiekt, zjawisko), której decyzja dotyczy (obiekt podejmowania decyzji).

**TEORIA DECYZJI –** całokształt zagadnień związanych z wyznaczaniem decyzji. Jest interdyscyplinarnym obszarem badawczym, łączącym szczegółowe zagadnienia, m.in. matematyki, statystki, psychologii, socjologii, ekonomii, zarządzania, filozofii, kognitywistyki oraz informatyki. Proces podejmowania decyzji można rozpatrywać z perspektywy każdego z wymienionych obszarów.

**SYSTEMEM DECYZYJNYM** nazywamy odpowiednie połączenie *obiektu decyzyjnego* oraz *decydenta*.

Wyróżnia się cztery struktury systemu decyzyjnego:

1. Struktura otwarta bez wykorzystania informacji o obiekcie decyzyjnym

Brak wykorzystania informacji o obiekcie decyzyjnym dotyczy tylko braku aktualnych obserwacji tego obiektu (jego aktualnego stanu). Decydent posiada jednak aprioryczną (zadaną ‘odgórnie’) informację o obiekcie decyzyjnym, na podstawie której podejmuje decyzje. Określenie ‘struktura otwarta’ dotyczy braku wykorzystania przez decydenta informacji zwrotnej dotyczącej skutku decyzji – w podejmowaniu decyzji wykorzystywana jest tylko aprioryczna informacja o obiekcie.

PRZYKŁAD: Decydentem jest osoba operująca wiertarką. Obiektem decyzyjnym natomiast wiertarka oraz obiekt, w którym wykonywane będą nawierty. Wykorzystując tylko informację aprioryczną wiedziałaby, że wciskając coraz mocniej spust wiertarki zwiększa obroty, a zwalniając go zmniejsza je. Nie wie natomiast jaki jest aktualny stan wiertarki (przegrzanie wiertarki i wiertła).

1. Struktura otwarta z wykorzystaniem informacji o obiekcie decyzyjnym

Decydent podejmując decyzję bierze również pod uwagę aktualną obserwacje obiektu decyzyjnego.

PRZYKŁAD: Poza zasadą działania wiertarki, użytkownik posiada informacji dotyczące aktualnego jej stanu – wie, że w danym momencie urządzenie jest przegrzane i w ten sposób może podjąć decyzję o zwolnieniu obrotów i zapobiegnięciu jej awarii.

1. Struktura zamknięta

W strukturze zamkniętej decydent wykorzystuje zwrotną informację z obiektu decyzyjnego. Znając skutek poprzednio podjętych decyzji, decydent może korygować aktualnie podejmowane decyzje, aby szybciej/lepiej osiągać zamierzony cel.

PRZYKŁAD: Załóżmy, że decydent chce wykonać wiertarką kilka takich samych nawiertów. Biorąc pod uwagę jak głębokie były nawierty i jak szybko powstały (przy użyciu jakiej siły nacisku oraz obrotów), decydent może kontrolować kolejne wiercenia poprzez dostosowywanie parametrów wiertarki.

1. Struktura mieszana

Struktura ta jest połączeniem struktury zamkniętej oraz struktury otwartej z wykorzystaniem obserwacji obiektu. Podejmując decyzje, decydent korzysta zarówno z obserwacji obiektu decyzyjnego, jak i skutków poprzednio podjętych decyzji.

PRZYKŁAD: Decydent (robotnik) wykonuje serie nawiertów obserwując zarówno stan wiertarki, jak i efekty poprzednich decyzji i podejmując optymalne decyzje dotyczących kolejnych nawiertów w oparciu o te dwa czynniki.

Kluczowym czynnikiem w podejmowaniu decyzji jest **użytkownik**, który formułuje wymagania, akceptuje/odrzuca decyzję – jest czynnym (interaktywnym) uczestnikiem procesu decyzyjnego.

**System wspomagania decyzji:**

System decyzyjny, którego wygenerowane decyzje podlegają akceptacji przez użytkownika

SWD to interaktywny system komputerowy, które pomaga użytkownikowi wykorzystać dane i modele w rozwiązywaniu problemów (A. Kwiatkowska). Powyższa definicja dotyczy perspektywy systemu informatycznego, czyli technicznej (informatycznej) realizacji SWD.

Systemy Wspomagania Decyzji są zorganizowanym zbiorem ludzi, procedur, baz danych i urządzeń wykorzystywanych w celu wspomagania podejmowania decyzji na wszystkich eta-pach tego procesu, poczynając od rozpoznania czyli zdefiniowania problemu i zaklasyfiko-wania go do określonej grupy standardowej, następnie poprzez wybór odpowiednich danych stworzenie i analizę modelu informacyjnego opisującego rzeczywistość, dalej pomagając w generowaniu wariantów dopuszczalnych rozwiązań oraz w wyborze najlepszego rozwiązania.

# Czynności techniki systemów

Teoria i technika systemów zajmuje się wspólnymi problemami, metodami i technikami dotyczącymi opisu, własności i sposobów rozwiązywania zadań, których przedmiotem są systemy o różnej naturze. W szczególności t.s. zajmuje się kreowaniem modeli i modelowaniem, identyfikacją i rozpoznawaniem, analizą i projektowaniem, sterowaniem (kierowaniem, zarządzaniem).

**Problem regulacji** – Należy zaprojektować mechanizm zapewniający utrzymanie pewnych sygnałów z obiektu sterowania na stałym, zadanym poziomie, pomimo obecności zewnętrznych zakłóceń oddziaływujących na obiekt.

**Sprzężenie zwrotne** – fundamentalna zasada sterowania. Sterownik akceptuje mierzone sygnały wyjściowe obiektu sterowania jako swoje wejścia i z kolei generuje sygnały sterujące podawane na obiekt sterowania.

Modele ze zmiennymi stanu. Systemy dynamiczne nie są postrzegane jedynie jako transformacja wejścia na wyjście. Modele ze zmiennymi stanu wyrażają tą transformację za pośrednictwem transformacji stanu.

Równania różniczkowe są bardziej pokrewne klasycznym modelom matematycznym używanym w fizyce, chemii, ekonomii dostarczają bardziej wszechstronny język, pozwalają dołączyć efekty nieliniowe, są bardziej przyjazne obliczeniom.

**Obiekt sterowania** – czarna skrzynka napędzana przez sygnały sterujące generująca sygnały wyjściowe

**Sterownik** – z czujników pobierany sygnał, z którego wydobyte są informacje o aktualnej dynamice obiektu sterowania o nieznanych parametrach i o wewnętrznym stanie obiektu. Na bazie tych informacji, stosownie do zadania sterowania, sterownik wyznacza sygnał sterujący, który należy zastosować. Za pośrednictwem organów wykonawczych jest on podawany na wejście obiektu sterowania.

Zadania sterowania: stabilizacja, tłumienie zakłóceń, śledzenie, odporność.

Sterownik PID – P, podstawowa kompensacja, I – duży sygnał korekcji gdy błąd nie maleje, D – działeni wyprzedzające. Szczególnie stosowany w procesach chemicznych.

Identyfikacja systemów lub procesów to termin opisujący zespół metod i narzędzi i algorytmy, które mają na celu zbudować dynamiczny model systemu lub procesu na podstawie danych pomiarowych zebranych z wejścia i wyjścia. Model taki może opisywać:

● właściwości wejściowo-wyjściowe systemu - jeżeli jest tworzony w oparciu o sekwencje sygnałów wejściowych i towarzyszące im sekwencje sygnałów wyjściowych,

● przebieg wyjścia systemu o wejściach pomiarowo niedostępnych - jeżeli jest tworzony jedynie w oparciu o mierzoną sekwencję sygnału wyjściowego.

Analiza - Przeciwstawną metodą do identyfikacji jest modelowanie analityczne. Polega ono na tym, że system dzielony jest na podsystemy, których właściwości oraz prawa fizyczne nimi rządzące dają się opisać modelami matematycznymi. Metoda ta jest zależna od skali problemu, może być bardzo czasochłonna i prowadzić do uzyskania modeli matematycznych zbyt skomplikowanych, by nadawały się do dalszego wykorzystania.

\*\*\*

Teoria i technika systemów zajmuje się wspólnymi problemami, metodami i technikami dotyczącymi opisu, własności i sposobów rozwiązywania zadań, których przedmiotem są systemy o różnej naturze. W szczególności t.s. zajmuje się:

* Kreowaniem modeli, modelowaniem
* Identyfikacją i rozpoznawaniem
* Analizą i projektowaniem
* Sterowaniem (kierowaniem, zarządzaniem)

**Identyfikacja systemów lub procesów** to termin opisujący zespół metod i narzędzi i algorytmy, które mają na celu zbudować dynamiczny model systemu lub procesu na podstawie danych pomiarowych zebranych z wejścia i wyjścia.

**Analiza** - Polega na tym, że system dzielony jest na podsystemy, których właściwości oraz prawa fizyczne nimi rządzące dają się opisać modelami matematycznymi. Metoda ta jest zależna od skali problemu, może być bardzo czasochłonna i prowadzić do uzyskania modeli matematycznych zbyt skomplikowanych, by nadawały się do dalszego wykorzystania.

**Sterowanie** - Stworzony model pozwala na syntezę układu regulacji poprzez wprowadzenie regulatora sterującego danym obiektem lub procesem tak, by ten zachowywał się w pożądany sposób.

# Problemy decyzyjne dla kompleksu operacji

Jedną z klasyfikacji obiektów, będących przedmiotem badań w szeroko rozumianych badaniach systemowych, jest podział na obiekty wejściowo – wyjściowe oraz kompleksy operacji. **Kompleks operacji** jest obiektem złożonym, którego elementami są operacje powiązane ze sobą na zasadzie kolejności czasowych, to znaczy rozpoczęcie wykonywania niektórych operacji może się rozpocząć po zakończeniu wykonywania innych operacji. Graficznym sposobem przestawienia kompleksu operacji może być graf, w którym łuki to operacje, a wierzchołki – momenty czasu. Dla przypomnienia, matematycznie graf opisujemy uporządkowaną parą G = (V, E), w której V = { v1, v2,...,vn } jest zbiorem n ponumerowanych wierzchołków, E = { e1, e2, ... em} jest zbiorem ponumerowanych krawędzi. Każda krawędź jest parą wierzchołków grafu połączonych tą krawędzią.

*Problemy decyzyjne dla kompleksu operacji:*

1.**Problem alokacji (deterministyczny)**

a. **Problem rozdziału zasobów** – Zasobem może być np. pamięć dostępna do wykonywania procedur obliczeniowych, energia elektryczna a także realizator operacji. Zasoby dzielimy na odnawialne i nieodnawialne. Najczęściej stosowanym kryterium przydziału zasobów jest czas całkowitego wykonania kompleksu operacji (moment, w którym kończy się ostatnie zadanie, czyli długość uszeregowania). Inne kryteria: koszt wykonania operacji z ograniczeniem czasu jego wykonania, zysk związany z realizacją kompleksu itp.

b. **Problem rozdziału zadań** – zadanie jest efektem wykonania operacji, operacje wykonuje się po to, aby osiągnąć pewien zamierzony cel, wyrażony w postaci zadania.

2.**Problem szeregowania (deterministyczny)**

Zarówno realizatorzy jak i zadania są elementami zbirów dyskretnych i skończonych. Problem szeregowania można określić jako wyznaczanie takiego dopuszczalnego zbioru przyporządkowania elementów jednego zbioru elementom drugiego zbioru, które jest najlepsze ze względu na przyjęte kryterium szeregowania. Jeżeli w zbiorze zadań występuje co najmniej jedno ograniczenie kolejnościowe, to o zadaniach mówimy, że są zależne, w przeciwnym przypadku są niezależne. Wygodną reprezentacją graficzną szeregowania zadań jest tak zwany wykres Gantta.

W przypadku problemu szeregowania mamy najczęściej do czynienia z problemem NP-trudnym [ale nie zawsze!] (*przypomnienie: NP trudne nie są rozwiązywalne w czasie wielomianowym, ale których rozwiązanie można w czasie wielomianowym zweryfikować)*. Stosuje się najczęściej rozwiązania: programowanie dynamiczne oraz zasadę podziału i ograniczeń.  Algorytmy  efektywne o złożoności wielomianowej dają rozwiązanie nieoptymalne (alg. przybliżony, gdy jakość otrzymanego rozwiązania można oszacować w stosunku do rozwiązania optymalnego, gdy nie można to heurystyka).

Wśród problemów szeregowania wyróżniamy przepływowy problem szeregowania zadań, omawiany w ramach wykładu.Założenie: zadań w ramach każdej operacji mamy tyle samo, co dostępnych realizatorów. Należy określić sekwencję wykonywania zadań.

a)      **Permutacyjny problem przepływowy** – z punktu widzenia każdego realizatora, kolejność wykonywania zadań jest taka sama

b)      **Niepermutacyjny p. p.** – kolejność wykonywania zadań jest inna dla różnych realizatorów.

3.       **Probabilistyczne problemy podejmowania decyzji (niederministyczne)**

Informacja o niektórych wartościach właściwych dla kompleksu operacji nie jest pełna albo pewna (np. informacja o operacjach, zasobach, zadaniach, realizatorach, strukturze itp.).

a)      **Rozdział zasobów i szeregowanie zadań dla losowych parametrów** – brak determinizmu polega na niedokładnej znajomości modeli operacji.

b)      **Obsługa zadań –** w problemach szeregowania zadań należało wyznaczać kolejność wykonywania zadań na poszczególnych realizatorach, zbiór zadań był z góry określony przed podjęciem i realizacją decyzji. W problemie niederministycznym obsługi zadań zbiór zadań nie jest znany priori lub zadania pojawiają się na bieżąco, w pewnych odstępach czasu. System taki określamy systemem obsługi, a problem obsługą zadań, teorią kolejek lub teorią masowej obsługi.

# Podstawowe problemy, metody i algorytmy optymalizacji dyskretnej

Większość deterministycznych problemów planowania i sterowania w dyskretnych systemach wytwarzania jest formułowana jako zagadnienia optymalizacji, w których wszystkie zmienne decyzyjne (bądź ich część) przyjmują wartości dyskretne, całkowitoliczbowe lub binarne. Zadania takie często nazywa się problemami optymalizacji dyskretnej lub dyskretno-ciągłej, należą do klasy problemów wyjątkowo kłopotliwych z obliczeniowego punktu widzenia. Zagadnienia te sprowadzają się do zadania minimalizacji funkcji celu K(x) na zbiorze rozwiązań dopuszczalnych X, określonym przez zestaw warunków ograniczających. Głównymi powodami tych kłopotów są: częsty brak „klasycznych”, analitycznych własności (różniczkowalność, liniowość, itp.), wielo-ekstremalność ze znaczną liczbą ekstremów lokalnych. NP-trudność większości problemów pochodzących z praktyki oraz przekleństwo wielowymiarowości.

Wielokrotnie, w celu uniknięcia kłopotów, próbuje się zamiast rozwiązywać problem dokładnie, wyznaczyć pewne jego rozwiązanie przybliżone. Dokładność tego przybliżenia posiada tendencję przeciwstawną do czasu obliczeń, tzn. uzyskanie dokładniejszego rozwiązania wymaga dłuższego czasu trwania algorytmu, przy czym ta ostatnia zależność posiada charakter silnie nieliniowy. Jest to czynnikiem powstania wielu rodzajów zarówno modeli jak i metod rozwiązywania, zwykle dedykowanych dla wąskich klas zagadnień. Często dla tego samego problemu NP.-trudnego występuje w literaturze kilka, kilkanaście różnych algorytmów o istotnie różnych cechach numerycznych.

Wprowadzenie dyskretnych zmiennych decyzyjnych umożliwia niekiedy względne łatwe rozwiązanie zadań, w których zbiór rozwiązań dyskretnych nie jest zbiorem wypukłym lub też jest zbiorem niespójnym.

Metody rozwiązania zadań optymalizacji dyskretnej można podzielić na dwie grupy:

• metody płaszczyzn tnących,

• metody heurystyczne.

Pierwsze dwie metody są metodami dokładnymi, metody heurystyczne dostarczają rozwiązań przybliżonych.

**METODA PODZIAŁU I OGRANICZEŃ (branch & bound)**

Do rozwiązywania dyskretnych zadań decyzyjnych stosuje się tzw. metodę podziału i ograniczeń. Idea metody polega na tym, że tzw. przegląd zupełny (pełny) zbioru ograniczeń D zastępujemy przeglądem ukierunkowanym. Pozwala to ocenić pośrednio pewne podzbiory rozwiązań i ewentualnie je odrzucić lub czasowo pominąć, bez utraty rozwiązania optymalnego, co znacznie przyspiesza uzyskanie rozwiązania

**Metoda AHP**

Analitycal Hierarchy Process „szkoła amerykanska” – Thomas L. Saaty

Działania w metodzie AHP da sie ujac w trzech etapach.

I. Budowa macierzy porównań parami dla n obiektów osobno w ramach każdego kryterium (macierze (1) A , (2) A , ... , (K ) A ) oraz dla samych kryteriów (macierz (0) A ). Porównania te prowadza do powstania K+1 macierzy porównan parami ( (0) A , (1) A , (2) A , ... , (K ) A ). Ważnym uzupełnieniem etapu I jest badanie spójnosci ocen decydenta.

II. Wyznaczanie rankingów indywidualnych dla każdej z macierzy etapu I.

III. Wyznaczanie rankingu wielokryteriowego n obiektów.

**[Metody przybliżone]**

Metoda przybliżona dostarcza pewnego rozwiązania bliskiego optymalnemu, to znaczy takiego rozwiązania, dla którego wartość funkcji celu niewiele różni się od wartości optymalnej. Metod przybliżonych jest znacznie więcej niż dokładnych, są one zwykle zorientowane na problem, który rozwiązują. Jakość metody przybliżonej ocenia się na podstawie złożoności obliczeniowej algorytmu oraz dokładności przybliżenia.

**Poszukiwanie lokalne** Wiele algorytmów przybliżonych opiera się na iteracyjnym polepszaniu bieżącego rozwiązania poprzez lokalne przeszukiwanie. Przeszukiwanie to rozpoczyna się od pewnego rozwiązania startowego. Następnie generowane jest jego otoczenie (sąsiedztwo), z którego wybiera się najlepsze rozwiązanie, które staje się rozwiązaniem startowym w kolejnej iteracji.

**Symulowane wyżarzanie** Metoda symulowanego wyżarzania (odprężania) posiada pewne analogie z termodynamicznym procesem studzenia. Stany ciała są postrzegane jako analogiczne do rozwiązań, a energia ciała - analogiczna do wartości funkcji celu.

**Algorytmy ewolucyjne** W algorytmach ewolucyjnych przy rozwiązywaniu danego problemu korzysta się z mechanizmu opartego na zjawisku naturalnej ewolucji gatunków.

**Procedura (algorytm) (Tabu search - TS)** stosowana do rozwiązywania problemów optymalizacyjnych. Wykorzystywana do otrzymywania rozwiązań optymalnych lub niewiele różniących się od niego dla problemów z różnych dziedzin (np. planowanie, planowanie zadań). Podstawową ideą algorytmu jest przeszukiwanie przestrzeni, stworzonej ze wszystkich możliwych rozwiązań, za pomocą sekwencji ruchów. W sekwencji ruchów istnieją ruchy niedozwolone, ruchy tabu. Algorytm unika oscylacji wokół optimum lokalnego dzięki przechowywaniu informacji o sprawdzonych już rozwiązaniach w postaci listy tabu (TL). Twórcą algorytmu jest Fred Glover.

**Algorytm zachłanny (*ang. greedy algorithm*)** – algorytm, który w celu wyznaczenia rozwiązania w każdym kroku dokonuje zachłannego, tj. najlepiej rokującego w danym momencie wyboru rozwiązania częściowego. Innymi słowy algorytm zachłanny nie patrzy czy w kolejnych krokach jest sens wykonywać dane działanie, dokonuje decyzji lokalnie optymalnej, dokonuje on wyboru wydającego się w danej chwili najlepszym, kontynuując rozwiązanie podproblemu wynikającego z podjętej decyzji. Typowe zadanie rozwiązywane metodą zachłanną ma charakter optymalizacyjny. W dziedzinie sztucznej inteligencji zachłanna odmiana przeszukiwania lokalnego jest nazywana "podchodzeniem pod wzgórze".

**[Metody dokładne]**

Metoda dokładna wyznacza rozwiązanie globalnie optymalne. Wśród metod dokładnych można wymienić m. in.

a) efektywne algorytmy dedykowane,

b) przegląd zupełny,

c) metody oparte o schemat podziału i ograniczeń,

d) metody oparte o schemat programowania dymanicznego (PD),

e) metody oparte na programowaniu liniowym całkowitoliczbowym (PLC).

Metody (b)-(e) są kosztownymi obliczeniowo metodami wykorzystywanymi do rozwiązywania problemów silnie NP-trudnych. Są to metody czaso- i pamięciochłonne oraz rozmiar rozwiązywanych przez nie problemów jest ciągle zbyt mały. Metody te są na przykład wykorzystywane do poszukiwania minimalnego czasu cyklu dla powtarzalnego procesu produkcyjnego z niewielkim repertuarem wyrobów. Dla małej liczby zadań można wyznaczyć rozwiązanie którego nawet mały zysk otrzymany w jednym cyklu zwielokrotni się o liczbę cykli w całym procesie produkcyjnym. Metody dokładne są wykorzystywane także do wyznaczania rozwiązań referencyjnych, do których następnie porównuje się wyniki algorytmów przybliżonych oceniając ich jakość.

# Podstawowe metody obliczeń miękkich (inteligentnych)

Wiele problemów w optymalizacji nie ma dokładnego algorytmów rozwiązania, albo ten algorytm ma tak dużą złożoność obliczeniową, że w praktyce i tak go nie ma. Wtedy stosuje się heurystyki, które nie ma gwarancji znalezienia rozwiązania optymalnego. Nie ma nawet gwarancji uzyskania rozwiązania proporcjonalnego do optymalnego, jak w algorytmach przybliżonych (aproksymacyjnych).

Metody używa się też często do znajdowania rozwiązań przybliżonych, na podstawie, których później wylicza się ostateczny rezultat pełnym algorytmem. To ostatnie zastosowanie szczególnie dotyczy przypadków, gdy heurystyka jest wykorzystywana do nakierowywania pełnego algorytmu ku optymalnemu rozwiązaniu, aby zmniejszyć czas działania programu w typowym przypadku bez poświęcania jakości rozwiązania (np. algorytm A\* w odnajdywaniu najkrótszej ścieżki w grafie).

**Sztuczne systemy** **immunologiczne** – stanowią odpowiednik realizacji procesu adaptacji i dywersyfikacji naturalnego systemu immunologicznego. Ich zadaniem jest, poprzez sterowanie populacją przeciwciał, doprowadzenie do otrzymania rozwiązania. Algorytmy immunologiczne można podzielić na populacyjne (selekcja klonalna, selekcja negatywna) i sieciowe (sieć idiotypowa).

**Techniki** **rojowe** (mrówkowe, pszczele, świetlikowe, kukułcze) - wywodzące się z algorytmu optymalizacji kolonii cząstek - patricle swarm optimization PSO. W zależności od stworzenia, na którego obserwacjach bazuje metoda, przeszukiwanie dziedziny i wybór najlepszych rozwiązań następuje w różny sposób, wszystkie mają jednak jedną bazę i w zapisie formalnym różnią się nieznacznie.

**Oparte o teorię chaosu** - bazujące na deterministycznych układach, tak wrażliwych na drobne zmiany warunków początkowych, że podczas obserwacji uchodzą za działające losowo (niedeterministycznie)

**Logika rozmyta typu 2** - wprowadzająca stany pośrednie w logice, umożliwiające określenie stopnia przynależności obiektu do zbioru.

Dzielą się na przedziałowe i uogólnione:

>przedziałowe: współczynniki przynależności są przedziałami ostrymi, umożliwiają modelowanie niepewności w przeciwieństwie do logiki rozmytej typu 1, a operacje and i or są proste i szybkie,

>uogólnione: współczynniki przynależności nie są ostre, należą do zbioru rozmytego. Podejście na razie raczej teoretyczne ze względu na słabo poznane zasady matematyczne i złożoność obliczeniową (zamiast operacji and i or - meet i join, bardzo wymagające obliczeniowo).

Przykładowo: - typ-1: Karol jest w 0.72 wysoki,

- typ-2 przedziałowy: Karol jest w [0.62-0.82] wysoki, rozkład przynależności równomierny

-typ-2 uogólniony: Karol jest w N(0.72, 0.1) wysoki, rozkład przynależności normalny (mam nadzieję że nie zakręciłęm) czy jakikolwiek inny niż równomierny

**Techniki agentowe** - system oparty o autonomiczne byty zwane agentami, rozproszone podejście do rozwiązania problemu.

Agent: oddziałuje na środowisko, komunikuje się z innymi agentami, działa realizując wyznaczone cele, ma dostęp i dysponuje zasobami, posiada jakiś zbiór umiejętności, posiada ograniczoną percepcję, posiada wiedzę nt środowiska lub ją gromadzi, czasami może się rozmnażać (pewnie jak spotka ładną agentkę :) )

**Zbiory przybliżone -** rough sets, dla zbiorów o nieregularnych zakresach definiujemy przybliżenie górne i dolne o zakresach regularnych. Dzięki temu możemy określić nieostre pojęcie w ścisły sposób. Przynależność sprawdza się na podstawie klas równoważności R, zwanych atomami. Obiekty należące do tej samej klasy równoważności są nierozróżnialne.

- aproksymacja dolna: składa się z obiektów, które z całkowitą pewnością należą do zbioru X

- aproksymacja górna: składa się z obiektów, które MOGĄ należeć do zbioru X

- obszar brzegowy: różnica między aproksymacją dolną i górną

- zbiór dokładny (crisp): obszar brzegowy nie zawiera żadnych obiektów

- zbiór przybliżony (rough): obszar brzegowy zawiera jakieś obiekty

**Hybrydy** - rozwiązania powstałe poprzez połączenie powyższych z innymi rozwiązaniami, np. rozmyte sieci neuronowe, transformata falkowa połączona z sieciami lub też połączenie między sobą itd.

**Algorytm ewolucyjny** jest algorytmem przeszukującym przestrzeń rozwiązań. Jest podobny do algorytmów iteracyjnych i losowych, z tą różnicą, że operuje na zbiorze rozwiązań, a nie na poszczególnym jednym rozwiązaniu, przeprowadzając na nim równoległe obliczenia. Inspiracją algorytmów genetycznych dla J.H.Hollanda był proces ewolucji w przyrodzie. Przez to połączenie stosuje on specyficzną nomenklaturę jak na algorytm optymalizacji.

Optymalizacja = Alg genetyczny; Zmienna optymalizacyjna = Osobnik (chromosom)

Zbiór rozwiązań dopuszczalnych = Populacja; Funkcja celu = Funkcja przystosowania

Iteracja = Pokolenie; Kroki (fazy) algorytmu optymalizacji = Operatory

**Tabu search** (TS) jest algorytmem do optymalizacji dyskretnej. Przeszukuje on wszystkie rozwiązania, sąsiadujące do tego, w którym się znajduje. Pomysł polega na tym, by zabronić powrotu do już odwiedzonych rozwiązań (na co najmniej kilka iteracji), czyli algorytm może wybrać gorsze rozwiązanie, by podążyć inną ścieżką. Możliwe jest zastosowanie tego algorytmu, także do przestrzeni ciągłych poprzez ich dyskretyzację.

Do zastosowania TS do konkretnego problemu, należy określić: reprezentację rozwiązania, funkcję przystosowania oraz generowanie sąsiedztwa dla rozwiązania.

# Podejmowanie decyzji w warunkach niepewności

**Niepewność** jako pojęcie teoriidecyzji oznacza sytuację, w której określone decyzje mogą spowodować różne skutki, w zależności od tego, który z możliwych stanów rzeczy zajdzie, przy czym nie są znane prawdopodobieństwa wystąpienia poszczególnych z nich.

Formalnie, **decyzjami podejmowanymi w warunkach niepewności** nazywamy taką klasę problemówdecyzyjnych, w której dla przynajmniej jednej decyzji nie jest znany rozkładprawdopodobieństwa konsekwencji.

W praktyce prawie zawsze w wypadku niepewności określamy prawdopodobieństwosubiektywne zajścia danej konsekwencji.

Ze względu na posiadane informacje, możemy podzielić problemy decyzyjne na trzy grupy:

**-decyzja podejmowana w warunkach pewności** – każda decyzja pociąga za sobą określone, znane konsekwencje

**-decyzja podejmowana w warunkach ryzyka** – każda decyzja pociąga za sobą więcej niż jedną konsekwencję, znamy zbiór możliwych konsekwencji i prawdopodobieństwa ich wystąpienia

**-decyzja podejmowana w warunkach niepewności** – nie znamy prawdopodobieństw wystąpienia konsekwencji danej decyzji.

Metody podejmowania decyzji w warunkach niepewności:

- zbiory przybliżone

- zbiory rozmyte

- sieci bayesowskie

**Teoria zbiorów przybliżonych** – Zbiór przybliżony (ang. *rough set*) to obiekt matematyczny zbudowany w oparciu o logikę trójwartościową. W swym pierwotnym ujęciu zbiór przybliżony to para klasycznych zbiorów: przybliżenie dolne i przybliżenie górne. Dany element może należeć do obydwu przybliżeń, do żadnego lub tylko do przybliżenia górnego. Ten ostatni przypadek jest o tyle ciekawy, że pozwala na modelowanie niepewności.

**Logika rozmyta** (ang. fuzzy logic), jedna z logik wielowartościowych (ang. multi-valued logic), stanowi uogólnienie klasycznej dwuwar-tościowej logiki. Została zaproponowana przez Lotfi Zadeha, jest ściśle powiązana z jego teorią zbiorów rozmytych. W logice rozmytej między stanem 0 (fałsz) a stanem 1 (prawda) rozciąga się szereg wartości pośrednich, które określają stopień przynależności elementu do zbioru.

Logika rozmyta okazała się bardzo przydatna w zastosowaniach inżynierskich, gdzie klasyczna logika klasyfikująca jedynie według kryterium prawda/fałsz nie potrafi skutecznie poradzić sobie z wieloma niejednoznacznościami i sprzecznościami. Znajduje wiele zastosowań, między innymi w elektronicznych systemach sterowania (maszynami, pojazdami i automatami), zadaniach eksploracji danych czy też w budowie systemów ekspertowych.

**Sieć Bayesowska** to acykliczny graf (DAG, Directed Acyclic Graph), składający się z:zbioru wierzchołków odpowiadających zbiorowi zmiennych zbioru skierowanych krawędzi łączących pary węzłów – intuicyjne znaczenie połączenia od węzła A do B oznacza, że A bezpośrednio wpływa na B

* graf nie zawiera cykli
* Węzły, od których dochodzą krawędzie do danego węzła to węzły rodzicielskie
* Każdy węzeł zawiera tabelę prawdopodobieństw warunkowych, określających wpływ węzłów ‘rodzicielskich’ na dany węzeł

Cechy BN

* łatwiej ekspertom określić bezpośrednie zależności warunkowe w dziedzinie, niż podawać aktualne prawdopodobieństwa
* po zbudowaniu topologii BN, należy określić prawdopodobieństwa warunkowe dla węzłów, które są ze sobą bezpośrednio połączone; te prawdopodobieństwa są wykorzystywane do obliczania każdego innego prawdopodobieństwa

# Metody i algorytmy rozpoznawania

Rozpoznawanie obiektów – przypisanie do klasy na podstawie cech (algorytm rozpoznawania)

schemat: obiekt opisany cechami → pomiar cech → przetwarzanie cech → klasyfikacja

Rozpoznawanie w oparciu o ciąg uczący – rozpoznawanie z nauczycielem

Algorytmy odległościowe:

• Algorytm NM (najbliższej średniej) – obliczamy wartość średnią dla każdej klasy. Obliczamy odlległość x od każdego elementu średniego, bierzemy min odległość i przypisujemy do klasy odpowiadającej tej wartości średniej

• Algorytm KNN (k najbliższych sąsiadów) – liczymy odległość x od każdego elementu ciągu uczącego, sortujemy w porządku niemalejącym, bierzemy k pierwszych pozycji i wybieramy najczęściej występującą klasę

Algorytmy probabilistyczne

• Naiwny Bayes – zakładamy, że zmienne losowe – cechy klasy są niezależne. Dla każdej cechy Xk wyznaczamy wartości prawdopodobieństwa p(J=j|Xk=xk) – czyli podliczamy liczbę obiektów należących do klasy j i mających Xk=xk w stosunku do wszystkich obiektów klasy j w zbiorze uczącym. Wyznaczamy też prawdopodobieństwo p(J=j). Przyporządkowujemy obiekt do tej klasy, dla której iloczyn powyższych prawdopodobieństw jest największy.

Drzewa klasyfikacyjne

algorytm (ogólny) - 2 kroki:

1) reguła podziału wierzchołków

2) reguła uznania wierzchołka za końcowy oraz wiążąca się z tym reguła przypisania wierzchołka

Algorytm ID.3 (alg. budowy drzewa klasyfikacyjnego)

1) Dla każdej cechy liczymy entropię warunkową (w tym celu wyliczamy prawdopodobieństwa, że przykład przybierze daną wartość cechy oraz prawdopodobieństwa warunkowe takie że dla danej wartości cechy przykład będzie należał do klasy oraz dla danej wartości cechy przykład nie będzie należał do klasy)

2) Wybieramy cechę o najmniejszej wartości entropii i umieszczamy jako wierzchołek. Od wierzchołka odchodzą krawędzie etykietowane wartościami cechy (teraz będziemy brać pod uwagę tylko przykłady zawierające wartość cechy z tej krawędzi) itd.

Inne istotne zagadnie powiązane z tematem do redukcja wymiaru/selekcja cech. Pozwala ono na ograniczenie zbioru cech tylko do niezbędnych do rozpoznawania i eliminację cech redundantnych, o niskiej entropii (niewielkiej informacji o klasie obiektu) lub silnie skorelowanych. Usunięcie niepotrzebnych cech pozwala na przyśpieszenie procesu nauki, uproszczenie algorytmu i potencjalnie poprawę wyniku rozpoznawania – szczególnie widoczną np. w przypadku algorytmów naiwnego Bayesa.

# Postulaty metodologii nauk

**Aksjomat** (postulat, pewnik) – jedno z podstawowych pojęć logiki matematycznej. Od czasów Euklidesa uznawano, że aksjomaty to zdania przyjmowane za prawdziwe, których nie dowodzi się w obrębie danej teorii matematycznej.

**Metodologia nauk** – nauka zajmująca się metodami stosowanymi przy formułowaniu twierdzeń i teorii naukowych. Metodologia nauk analizuje nie tylko procedury badawcze, lecz także jej wytwory: pojęcia, hipotezy, twierdzenia.

**Metodologia nauk**, w aspekcie pragmatycznym – nauka o metodach działalności naukowej i stosowanych w nauce procedurach badawczych; w aspekcie teoretycznym – nauka o elementach i strukturze systemów naukowych

**Postulaty:**

- falsyfikowalność - jakkolwiek żadne doświadczenie nie może wykazać prawdziwości teorii, to istnieją doświadczenia, które mogą wykazać jej fałszywość

- sprawdzalność - wiedza naukowa poddaje się sprawdzeniu, kontroli; powinno być powiedziane w jaki sposób doszliśmy do takich, a nie innych stwierdzeń, wniosków, przy użyciu jakich metod, na podstawie jakich danych, a także jak inni mogą sprawdzić nasze wyniki

- prostota - wiedza naukowa powinna być zaprezentowana w sposób możliwie najprostszy, dostępny dla jak najszerszej grupy odbiorów (???)

- bezstronność - przeciwstawienie tendencyjności, obiektywna sprawdzalność, niezależność od podmiotu sprawdzającego

# Współczesne metody naukometrii

**Naukometria** jest dziedziną naukoznawstwa zajmującą się badaniem rozwoju nauki jako procesu informacyjnego. W oparciu o metody statystyczno-ilościowe (liczba publikacji, przyznanych stopni naukowych i nagród, placówek naukowych) określa aktualny stan danej dyscypliny naukowej oraz prognozuje perspektywy jej rozwoju.

**Ocena dorobku naukowego**

·         **Ekspercka:**

o   ocena wartości naukowej osiągnięcia na tle osiągnięć nauki z danego obszaru, wykonana przez kompetentnych specjalistów

·         **Naukometryczna:**

o    prestiż czasopism (IF)

o    pozycja w gronie współautorów (pierwszy, korespondencyjny)

o    cytowania ( w tym bez autocytowań)

o    wskaźnik Hirscha

o    punkty MNiSzW

**Ocena parametryczna** oparta jest o wskaźniki nauko metryczne

**Ocena projektów NCN** oparta jest o wskaźniki naukometryczne i ocenę ekspercką

**Prestiż czasopism (IF – impact factor)**

Impact factor ("czynnik wpływu") wyznaczany jest co roku dla tytułu czasopisma.

Impact factor  jest ustalany według wzoru: IF= B/C, gdzie:

B – to łączna lista cytowań w danym roku kalendarzowym, wszystkich publikacji, które ukazały się w danym czasopiśmie w ciągu ostatnich dwóch lat, odejmując od tej liczby autocytowania – czyli cytowania publikacji autora w jego publikacjach.

C – to liczba wszystkich publikacji, które ukazały się w danym czasopiśmie, w ciągu ostatnich dwóch lat.

IF sprawdzamy w bazie ISI Web of Knowledge, w części Journal Citation Reports. Baza JCR opracowywana jest przez Institiute for Scientific Information ISI w Filadelfii (stąd dawna nazwa: lista filadelfijska).

**Współczynnik h (Hirsch index)**

Współczynnik h jest sposobem mierzenia osiągnięć naukowych z uwzględnieniem liczby publikacji i liczby cytowań.

Np. współczynnik h = 10 oznacza, że autor ma 10 publikacji cytowanych co najmniej 10 razy. Wielkość h zależy więc od dwóch czynników: liczby publikacji i ich popularności. Wzrost współczynnika h można osiągnąć publikując prace, które znajdą znaczny oddźwięk.

Cechy współczynnika h:

·         Zaletą tego współczynnika jest możliwość jego szybkiego obliczenia, jeśli znana jest liczba cytowań każdej publikacji danego autora. Współczynnik ten jest też proponowany do rankingu instytucji naukowych, grup uczonych lub oceny osiągnięć np. krajowych w określonej dziedzinie wiedzy.

·         Wadą współczynnika h  jest konieczność obserwowaniu osiągnięć naukowych w długim okresie czasu, bo wówczas pojawiają się cytowania i istnieje możliwość opublikowania wielu prac, zwłaszcza często cytowanych. Z tego powodu współczynnik h został przez Adama Pronia i Halinę Szatyłowicz nazwany żartobliwie, lecz słusznie, „współczynnikiem dostojeństwa naukowego”

·         Zdaniem Hirscha współczynnik h może być wykorzystany do prognozowania indywidualnych osiągnięć naukowych, gdyż jest w tym przypadku skuteczniejszy, niż liczba publikacji, liczba cytowań lub średnia cytowań w przeliczeniu na publikację

·         Zasięg cytowań jest różny w różnych dziedzinach nauki. Nauki biologiczne są dziedziną, w której publikacje są szeroko cytowane i stąd największy wskaźnik h podany przez Hirscha w cytowanej wyżej pracy wynosi h=191 dla neurologa  Solomona H. Snydera z Baltimore. Najwyższy wynik w fizyce osiągnął  E. Witten z h=110.

TRZY SPOSOBY WYLICZANIA INDEKSU H

Indeks H możemy wyliczyć na co najmniej trzy sposoby, używając:

·         bazy ISI Web of Knowledge;

·         bazy SCOPUS;

·         programu „Publish or Perish”.

# Algorytmy ewolucyjne w systemach informacyjnych

Algorytmy ewolucyjne są wzorowane idei ewolucji biologicznej i najczęściej wykorzystuje się je do zadań **optymalizacji** heurystycznej. Algorytmy takie korzystają z takich mechanizmów biologicznych, jak reprodukcja, mutacja, rekombinacja i selekcja.

Najpopularniejszą odmianą algorytmów ewolucyjnych są algorytmy **genetyczne.** Są to algorytmy przeszukujące przestrzeń rozwiązań problemu w sposób heurystyczny (nie przez przegląd zupełny).

Problem definiuje środowisko, w którym istnieje pewna **populacja osobników**. Każdy z osobników ma przypisany pewien zbiór informacji stanowiących jego **genotyp**, a będących podstawą do utworzenia fenotypu. **Fenotyp** to zbiór cech podlegających ocenie **funkcji przystosowania** modelującej środowisko. Innymi słowy - **genotyp** opisuje **proponowane rozwiązanie problemu**, a **funkcja przystosowania** ocenia, **jak dobre jest to rozwiązanie**.

Wspólnymi cechami algorytmów ewolucyjnych, odróżniającymi je od innych, tradycyjnych metod optymalizacji, są:

1. stosowanie operatorów genetycznych, które dostosowane są do postaci rozwiązań,
2. przetwarzanie populacji rozwiązań, prowadzące do równoległego przeszukiwania przestrzeni rozwiązań z różnych punktów,
3. w celu ukierunkowania procesu przeszukiwania wystarczającą informacją jest jakość aktualnych rozwiązań,
4. celowe wprowadzenie elementów losowych.

**Najczęściej działanie algorytmu przebiega następująco**

1. Losowana jest pewna populacja początkowa.
2. Populacja poddawana jest ocenie (selekcja). Najlepiej przystosowane osobniki biorą udział w procesie reprodukcji.
3. Genotypy wybranych osobników poddawane są operatorom ewolucyjnym:
   1. są ze sobą kojarzone poprzez złączanie genotypów rodziców (krzyżowanie),
   2. przeprowadzana jest mutacja, czyli wprowadzenie drobnych losowych zmian.
4. Rodzi się drugie (kolejne) pokolenie. Aby utrzymać stałą liczbę osobników w populacji te najlepsze (według funkcji oceniającej fenotyp) są powielane, a najsłabsze usuwane. Jeżeli nie znaleziono dostatecznie dobrego rozwiązania, algorytm powraca do kroku drugiego. W przeciwnym wypadku wybieramy najlepszego osobnika z populacji - jego genotyp to uzyskany wynik.

**Funkcja przystosowania** (fitness) - funkcja pozwalająca dla danego osobnika określić jego jakość (z punktu widzenia rozwiązywanego problemu). Zakładamy, że jej wartości są rzeczywiste nieujemne oraz że wyższa wartość funkcji oznacza zawsze, że dany osobnik jest lepszy. W przypadku ewolucji naturalnej odpowiednikiem takiej funkcji jest ogólna ocena przystosowania osobnika do danego środowiska. W praktyce funkcja ta jest zwykle niewielką modyfikacją funkcji celu rozwiązywanego problemu.

W systemach informacyjnych algorytmy ewolucyjne są wykorzystywane do rozwiązywania zadań, które do niedawna były niemożliwe do rozwiązania (lub przynajmniej ich rozwiązanie było niesatysfakcjonujące ze względu na długi czas lub niską precyzję). Poniżej wymieniono przykładowy zakres wykorzystania algorytmów ewolucyjnych w procesach biznesowych.

**Obszar zastosowań + przykłady zastosowań decyzyjnych:**

Transport - rozwiązanie problemu komiwojażera w firmie transportowej / spedycyjnej / dostawczej / cateringowej,

Sprzedaż - symulacja zapasów; problem transportowy w hurtowni; określenie pierwszego nakładu książki, rozmieszczenie punktów sprzedaży samouczący się model konsumenta

Organizacja - harmonogramowanie pracy; sporządzanie tras pociągów towarowych; sporządzanie tras pojazdów kołowych; kolejność lądowań samolotów; kolejność jazdy pociągów po linii jednotorowej; projektowanie sieci telekomunikacyjnych; optymalizacja sieci komputerowej; układanie podziału godzin w szkole; planowanie systemu dostaw wody;

Finanse - szacowanie ryzyka ubezpieczeniowego; prognozowanie bankructwa; określanie zdolności kredytowej; przewidywanie opóźnień w płatnościach; analiza cyklów czasowych; symulacja rynku akcji; optymalizacja portfela akcji; model gracza na rynku akcji; modelowanie rynków finansowych; wykrywanie oszustw finansowych; alokacja nakładów dla różnych programów społecznych;

Produkcja - równoważenie linii produkcyjnej; określanie wielkości partii; sekwencyjność produkcji; proces planowania; magazynowanie; zarządzanie obciążeniami; optymalizacja działania reaktorów jądrowych; programowanie symulatorów lotniczych

# Jakość oprogramowania i jakość danych w systemach informacyjnych

**Jakość oprogramowania** (ang. *software quality*) – ogół cech produktu programowego, które wpływają na zdolności spełniania przez niego określonych wymagań, m.in. elastyczność, funkcjonalność, integralność, niezawodność, efektywność, użyteczność, wydajność.

**Metryki jakości oprogramowania** stanowią szybki i wygodny sposób oceny jakości oprogramowania. Znajomość wartości poszczególnych metryk i wielkości, które mierzą, pozwala oszacować złożoność systemu, koszt jego pielęgnacji czy elastyczność.

**Metryki obiektowe** stanowią odpowiedź na zmianę paradygmatu programowania, która nie znalazła odzwierciedlenia w metodach pomiaru i szacowania dotyczących programowania strukturalnego. Zaproponowane metryki pozwalają mierzyć charakterystyczne dla obiektowości cechy: dziedziczenie, polimorfizm, powiązania i spójność.

**>>Depth of Inheritance Tree** (DIT, z ang. głębokość drzewa dziedziczenia) określa maksymalną **liczbę poziomów nadklas**, z których rozpatrywana klasa dziedziczy. Wysoka wartość DIT wskazuje, że klasa ta jest specjalizowana, o niskim poziomie abstrakcji, oraz prawdopodobnie posiada wiele metod odziedziczonych po przodkach. Z jednej strony wskazuje to na wysoki stopień powtórnego użycia kodu poprzez dziedziczenie, ale z drugiej może ograniczać wykorzystanie innych mechanizmów, np. kompozycji i wiązania klas przez referencje, dlatego zalecane wartości tej metryki znajdują się w przedziale 2-3.

**>>Lack of Cohesion Of Methods** (LCOM, z ang. brak spójności metod) to metryka określająca spójność metod wewnątrz klasy. **Spójność** jest jedną z pozytywnych cech związanych z klasą i oznacza, że metody klasy są silnie związane ze sobą nawzajem i polami zdefiniowanymi w tej klasie. Spójne klasy mają precyzyjnie określony obszar odpowiedzialności i zwykle komunikują się ze światem zewnętrznym przez stosunkowo wąskie interfejsy. Spójność klas należy do najważniejszych czynników decydujących o poprawności projektu obiektowego. Klasyczna metryka LCOM zdefiniowana przez C&K (oznaczana często jako LCOM1) mierzy brak odwołań do wspólnych atrybutów klasy i przyjmuje wartość równą **różnicy liczby metod odwołujących się do wspólnego atrybutu** klasy oraz **liczby metod o rozłącznych zbiorach wykorzystywanych atrybutów**. Posiada ona jednak pewne wady (m.in. zliczanie jedynie wspólnych pól a nie wywoływanych metod oraz podawanie tej samej wartości dla całkowicie odmiennych klas), które powodują jej niewrażliwość na pewne własności  danego projektu. Dlatego zaproponowano kilka innych definicji, wśród nich m.in. metrykę LCOM3, która przyjmuje wartości od 0 do 2. Za anomalię uznaje się wartości przekraczające 1 – klasy o takim wskaźniku powinny zostać podzielone na mniejsze.

**>>Coupling Between Objects** (CBO, z ang. powiązania pomiędzy obiektami) jest drugą, obok LCOM, ważną metryką oceniającą jakość projektu obiektowego. Niski stopień powiązań klas między sobą (ang. coupling) wskazuje na ścisłe określenie granic pomiędzy klasami i sposobu ich komunikowania się. Ponadto zwiększa stopień abstrakcji projektu, ponieważ łatwiej modyfikować fragmenty kodu w nieznacznym stopniu zależne od pozostałych. Metryka CBO zlicza **liczbę klas związanych** z klasą rozpatrywaną w sposób **inny niż poprzez dziedziczenie** (a więc jako atrybuty klasy, parametry metod, wartości zwracane, klasy wyjątków etc.). Wartość tej metryki powinna być utrzymana na możliwie niskim poziomie (oczywiście, wartość 0 oznaczałaby, że klasy w ogóle nie odwołują się do siebie, co jest niemożliwe). Typowo wynosi ona do 5.

>>**Weighted Method per Class** (WMC, z ang. ważona liczba metod w klasie) jest miarą złożoności klasy. W podstawowej wersji jest **liczbą wszystkich metod** zdefiniowanych w klasie (czyli waga każdej metody jest równa 1), w wersji rozszerzonej wagi przypisywane metodom są obliczane na podstawie **złożoności cyklomatycznej** CC McCabe'a). Metryka WMC pozwala na precyzyjniejszą ocenę niż RFC, ponieważ uwzględnia złożoność każdej metody. Typowe wartości to 0-100.

**>>Number of Children** (NOC, z ang. liczba klas pochodnych) jest liczbą bezpośrednich potomków danej klasy lub **liczbą klas implementujących** dany interfejs. Wysoka wartość tej metryki wskazuje na nieprawidłową faktoryzację nadklasy lub interfejsu (jest ona wysoce abstrakcyjna i prawdopodobnie obejmuje zbyt duży obszar odpowiedzialności) i może powodować trudności z jej pielęgnacją.

**>>Coupling Factor** (CF, z ang. współczynnik powiązań) ocenia **stopień zależności poszczególnych klas od siebie**. Pełni on podobną rolę jak CBO z zestawu C&K, pozwalając określić stopień wypełnienia grafu zależności. Niska wartość wskazuje, że odpowiedzialność jest nieprawidłowo podzielona pomiędzy klasy i kilka z nich realizuje większość funkcjonalności. Wysoka wartość CF też sugeruje złe zaprojektowanie systemu, z wieloma zależnościami pomiędzy klasami, co znacznie ogranicza możliwości rozbudowy i podwyższa koszt pielęgnacji kodu. Wartości powinny należeć do przedziału 5 do 20%.

**>>Affarent coupling** (Ca, z ang. powiązania do wewnątrz) to **liczba typów, które zależą** od rozpatrywanego modułu. Klasa o wysokiej wartości metryki Ca jest obciążona dużą odpowiedzialnością wobec innych modułów, co oznacza, że zmiany w niej powodują konieczność modyfikacji zależnych od niej typów.

**>>Efferent coupling** (Ce, z ang. powiązania na zewnątrz) określa zależność rozpatrywanego modułu od modułów zewnętrznych. Wartością tej metryki jest **liczba typów, od których zależy dany typ**. Zmiany w klasach o wysokiej wartości tej metryki w niewielkim stopniu wpływają na zewnętrzne moduły, natomiast zmiany w nich z dużym prawdopodobieństwem będą miały wpływ na rozpatrywany moduł.

**>>Abstractness** (A, z ang. abstrakcja) to miara abstrakcyjności pakietu i jest mierzona stosunkiem **liczby typów abstrakcyjnych** wchodzących w skład tego pakietu **do wszystkich typów**. Klasy o wysokim wskaźniku A trudno podlegają zmianom, ponieważ są odpowiedzialne wobec typów zależnych od nich.

**>>Instability** (I, z ang. niestabilność) jest miarą niestabilności pakietu, czyli wpływu zewnętrznych typów na rozpatrywaną klasę. Jest ona obliczana jako stosunek wartości metryki Ce do sumy wszystkich zależności (Ce + Ca). Wartości I bliskie jedynce sugerują, że klasa jest podatna na zmiany w zewnętrznych modułach (powiązania na zewnątrz stanowią niemal 100% wszystkich powiązań), natomiast wartości bliskie zeru charakteryzują klasę trudno poddającą się zmianom z uwagi na ich wpływ na pozostałe typy. Metryka ta powinna przyjmować właśnie wartości ekstremalne: albo bliskie zeru (czyli rdzeniowe klasy typu kontroler), albo bliskie jedynce (klasy korzystające z funkcjonalności dostarczanej przez inne moduły).

**>>Normalized distance from main sequence** (Dn, z ang. znormalizowana odległość od ciągu głównego) określa odległość od optymalnej linii łączącej punkty (1, 0) i (0, 1) w układzie współrzędznych metryk A i I. Klasy o niskiej wartości A powinny jednocześnie być niestabilne – i na odwrót, ponieważ to świadczy o dobrej kompensacji tych dwóch cech. Odchylenie od tej linii wskazuje na niepoprawne przypisanie odpowiedzialności do danej klasy.

**>>Złożoność cyklomatyczna McCabe'a** Złożoność cyklomatyczna, zaproponowana przez McCabe'a w 1976 roku jest metryką strukturalną, która odzwierciedla złożoność grafu przepływu sterowania w metodzie. Jej wartość jest równa liczbie niezależnych ścieżek wykonawczych w procedurze.

Podstawą do wyliczeń jest liczba dróg w schemacie blokowym danego programu, co oznacza wprost liczbę punktów decyzyjnych w tym programie.

Ponieważ obiektem pomiaru jest właśnie procedura, dlatego metryka ta nie może być bezpośrednio zastosowana w środowisku obiektowym (nie uwzględnia ona dziedziczenia i pokrywania metod), jednak jest z powodzeniem stosowana jako element bardziej obiektowych miar (np. WMC).

**JAKOŚĆ DANYCH** należy rozumieć jako kwalifikację poprawności danych, ale także ich przydatność. Pojęcie jest subiektywne, zależne od punkty widzenia zainteresowanych grup.

Popularna w informatyce maksyma „śmieci weszły, śmieci wyszły”, przypomina, że wyniki przetwarzania błędnych danych będą błędne nawet wtedy, gdy procedura przetwarzania jest poprawna. Niska jakoś danych stanowi główną przyczynę niepowodzeń rozwiązań BI. Zespoły projektowe, które zapominają o konieczności zapewnienia jakości danych, z czasem boleśnie się przekonują, że ich rozwiązanie nie dostarcza wiarygodnych informacji.

Czas i zasoby wydatkowane na wielokrotne przetwarzanie tych samych informacji nie stanowią bynajmniej najwyższej ceny, jaką przychodzi zapłacić za kiepską architekturą danych. Dane złej jakości skutkują niedokładnym spojrzeniem na wyniki działalności biznesowej i prowadzą do złych lub zbyt późnych decyzji. W rezultacie, przedsiębiorstwo może zostać narażone na utratę szans sprzedażowych oraz sposobności do redukcji kosztów, liczonych w milionach złotych. Problemy z jakością informacji mogą również doprowadzić do nieprzestrzegania przepisów i skutkować grzywnami.

Poprawnie wdrożone zarządzanie danymi, to nie tylko sposoby postępowania, normy i procesy. Zarządzanie danymi prowadzi do ścisłej integracji działalności biznesowej z technologią. W rezultacie przedsiębiorstwo jest w stanie bardzo szybko reagować na zmieniające się wymagania rynkowe.

Zasadniczym elementem skutecznego zarządzania danymi jest zarządzanie metadanymi (ang. meta data management), które silnie wpływa na jakość prezentowanych informacji.

* brak danych, bądź pojedynczych wartości
* dużo NULL – jak je interpretować?
* jak przetwarzać złe dane? pomijać? naprawiać?
* powtórzenia i duplikaty rekordów
* błędy interpretacji

# Metoda COCOMO szacowania kosztów projektów informatycznych

Pierwsza wersja powstała w 1981 i nosiła nazwę COCOMO-81. Aktualna wersja to COCOMO2. Autor metody doszedł do wniosku, że w celu oszacowania kosztów osobowych tj. liczby osobmiesięcy( liczba miesięcy  potrzebnych do realizacji projektu przez jedną osobę), konieczne jest podanie w przybliżeniu, z ilu linijek kodu będzie się składać gotowy produkt. Dalej to oszacowanie trzeba uściślić w kontekście określonych rodzajów projektów. Aby metoda była skuteczna, rozmiar oprogramowania powinien być szacowany przynajmniej na kilkanaście tysięcy linii kodu.

Model ten jest modelem trójpoziomowym, który pozwala na szacowanie kosztów z wrastającą dokładnością wraz z postępem cyklu inżynierii oprogramowania. Bazuje na rozmiarze oprogramowania.

* **Poziom prototypowania (application composition model)**

Faza prototypowania, poprzedzająca systematyczne modelowanie i projektowanie systemu. Występuje prototypowanie, wykorzystanie istniejących komponentów, podstawą oszacowań są punkty obiektowe. Jest przeznaczony dla projektów wykorzystujących nowoczesne narzędzia do tworzenie (generowania) interfejsów użytkownika (czyli na przykład dla prototypów)

* **Poziom wczesnego projektowania (early-design model)**

Odpowiada wczesnym fazom modelowania i projektowania systemu. Jest to etap projektowania architektury, najbliższy oryginalnego COCOMO, podstawą oszacowań są punkty funkcyjne przeliczane na linie kodu.

* **Poziom post-architektoniczny (post-architecture model)**

Odpowiada późnym fazom cyklu rozwoju systemu. Dla etapu rozwoju i utrzymywania, najbardziej szczegółowy model COCOMO II, podstawą oszacowań jest rozmiar kodu .

Model COCOMO stosuje proste wzory matematyczne z szeregiem parametrów. W modelu tym koszt wyraża się jako funkcję miary kodu (miarą może być liczba linii lub liczba tzw. punktów funkcyjnych). COCOMO pozwala także szacować pracochłonność wyrażoną w w osobo-miesiącach i czas realizacji projektu. Podstawowym wzorem dla oszacowania kosztów jest: **nakład[osobomiesiące]=A\*Kb**, gdzie K to rozmiar kodu źródłowego mierzony w tysiącach linii. A i b, to stałe, które są zależne od klasy przedsięwzięcia (klasy: 1. przedsięwzięcie ograniczone (łatwe) – wykonywane przez stosunkowo mały zespół przy pomocy dobrze znanych metod, 2. przedsięwzięcie pół-oderwanych (pośrednie) – członkowie zespołu różnią się stopniem zaawansowania, niektóre aspekty dziedziny nie są dobrze znane, 3. przedsięwzięcie osadzonych (trudne)– realizacja systemu o bardzo złożonych wymaganiach).

**Wady COCOMO:**

* Liczba linii kodu znana jest dokładnie dopiero wtedy, gdy system jest napisany. Szacunki są zwykle obarczone bardzo poważnym błędem.
* Określenie ‘linii kodu źródłowego’ inaczej wygląda dla każdego języka programowania

**Zalety:**

* Pozostawia miejsce na subiektywne oceny
* Utrzymany na podstawie doświadczenia
* Łatwy i szybki do uzyskania wyników
* Ma dobrą dokumentację

**Podsumowując** informacje na temat metody *COCOMO II*:

* Jest to metoda bazująca na **rozmiarze oprogramowania**, mierzonym w tysiącach źródłowych linii kodu. Aby pozyskać tę wartość można wykorzystać metodę punktów funkcyjnych lub specjalnie powstałą na potrzeby *COCOMO II* metodę *Object Points*.
* **Kalibracji** **współczynników** metody dokonano na podstawie 161 projektów. Niestety nie były to projekty realizowane w Polsce, wiec trudno powiedzieć czy użycie metody w rodzimych projektach jest także skuteczne.
* Problem ten można zniwelować dokonując kalibracji współczynników w oparciu o **dane historyczne** organizacji. W praktyce jednak, mało która organizacja posiada wystarczającą bazę danych zakończonych projektów.
* Aby metoda była skuteczna, rozmiar oprogramowania powinien być szacowany przynajmniej na kilkanaście tysięcy linii kodu.

# Metody automatycznej identyfikacji

Automatyczna identyfikacja polega na samoczynnym zebraniu danych o obiekcie, a następnie wprowadzeniu ich do systemu komputerowego bez ingerencji człowieka. Istnieje wiele metod automatycznej identyfikacji, jednak najbardziej znanymi są

* RFID
* Optyczne rozpoznawanie znaków (ang. OCR)
* Technologie biometryczne: identyfikacja głosowa, daktyloskopia
* Technologie kart inteligentnych: karty pamięciowe, karty mikroprocesorowe
* Kody kreskowe

**RFID** Określenie identyfikacja radiowa, "RFID", ang. Radio Frequency Identification, jest używane w odniesieniu do różnych technik komunikacji radiowej między urządzeniem służącym tylko do odczytu danych lub do odczytu i zapisu danych, nazywanym czytnikiem a urządzeniem przechowującym unikalne dane, nazywanym transponderem. System RFID składa się z systemu komputerowego (z bazą danych) i dwóch rodzajów urządzeń identyfikacji radiowej: czytników i transponderów. System komputerowy obsługuje aplikacja, która steruje urządzeniami RFID.

Transponder jest przeznaczony do umocowania do obiektu, który ma być identyfikowany. Może przechowywać i po uaktywnieniu wysyłać tylko swój indywidualny niepowtarzalny numer identyfikacyjny (ID), albo także inne dane odnoszące się do samego transpondera (np. kod wytwórcy) lub dane odnoszące się do obiektu, z którym transponder jest skojarzony. W niektórych systemach te dodatkowe dane mogą być wielokrotnie modyfikowane. Czytnik jest urządzeniem, które komunikuje się z transponderami znajdującymi się w zasięgu jego komunikacji radiowej. Czytnik nadzoruje protokół transmisji, odczytuje informacje z transponderów, może polecać transponderom przechowywanie informacji.

Działanie systemów RFID polega na komunikacji w zakresie częstotliwości radiowych pomiędzy transponderem i czytnikiem.

Transponder

Transponder (znacznik, tag) RFID jest radiowym urządzeniem nadawczym lub nadawczo-odbiorczym, wysyłającym sygnał zawierający kodowane dane identyfikacyjne tylko w odpowiedzi na pobudzenie sygnałem radiowym o określonej częstotliwości. Każdy transponder jest zbudowany z układu scalonego i anteny umieszczonych na odpowiednim podłożu lub w odpowiedniej obudowie. Układ scalony transpondera zawiera procesor, pamięć i nadajnik radiowy. W obudowie transpondera mogą znajdować się również dodatkowe elementy, takie jak np. czujnik temperatury i bateria stanowiąca źródło zasilania. Transponder może przechowywać różne dane i wskutek pobudzenia przesyłać je, np. w najprostszym przypadku tylko wpisany na stałe numer seryjny.

Transpondery mogą być również klasyfikowane jako:

– bierne (passive), tj. takie, które uzyskują energię tylko z pola elektromagnetycznego wytwarzanego przez czytnik;

– aktywne (active), tj. takie, których nadajnik RF jest zasilany z baterii;

– półaktywne (semi-active), tj. takie, w których wbudowana bateria służy do ciągłego zasilania czujnika (np. temperatury) zintegrowanego z transponderem, ale nie jest wykorzystywana do zasilania nadajnika RF.

Klasyfikacja:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Źródło energii zasilania transpondera | |
| Bateria (aktywny) | Pole RF czytnika (bierny) |
| Sposób komunikacji | Bierny (przesyłanie danych przez modulację pola RF) | Transpondery nazywane również półaktywnymi | Prawie wszystkie transpondery powszechnie stosowane |
| Aktywny (własny nadajnik) | Transpondery o dużym zasięgu | Nie istnieją |

Czytnik

Czytnik RFID jest radiowym urządzeniem nadawczo-odbiorczym. Jego nadajnik za pośrednictwem anteny emituje energię wykorzystywaną do uaktywniania transponderów, a w niektórych systemach również sygnały poleceń sterujących transponderami i/lub modyfikujących dane zapisane w pamięci transponderów. Jego odbiornik demoduluje i dekoduje dane nadawane przez transponder.

Czytnik odbierający sygnał transpondera dekoduje dane i może je przesyłać do komputera systemowego za pośrednictwem łącza kablowego (tylko czytniki stacjonarne) lub radiowego.

Podział:

-cz stacjonarne – w celu tworzenia stref odczytu np. w bramie magazynu w celu inwentaryzacji przyjmowanych i wydawanych produktów, są stale w trybie czuwania, aby wykryć każdy transponder, który znajdzie się w ich strefie. Czytnik stacjonarny wymaga dołączenia przewodów zasilania i połączenia z siecią komputerową. Jego anteny powinny być chronione przed wpływami środowiska (kurz, zawilgocenie, wibracje) i uszkodzeniami mechanicznymi.

-cz ruchome – zainstalowane np. na wózku widłowym. Czytniki tego rodzaju są zasilane z baterii pojazdu. Ich anteny są szczególnie narażone na uszkodzenia mechaniczne i środowiskowe

-cz noszone – zasilane z wbudowanej baterii i zwykle są zintegrowane z anteną. Często noszony czytnik RFID jest skojarzony z czytnikiem kodów kreskowych. Połączenie z komputerem systemu identyfikacji jest realizowane drogą radiową np. w radiowej sieci lokalnej (WLAN). W tym przypadku czytnik jest zabierany do miejsca, gdzie znajdują się oznakowane obiekty, jest uaktywniany tylko na czas pojedynczego odczytu. Zatem użytkownik czytnika noszonego w pełni kontroluje gdzie, kiedy i jak czytnik będzie wykorzystany. Tego rodzaju czytniki są narażone na uszkodzenia wskutek upadku

Wady i zalety

+Odczytywanie danych na odległość, nawet gdy transponder nie jest widoczny znacznie usprawnia identyfikację obiektów oznaczonych RFID

+Transpondery przechowują o wiele więcej informacji niż kody kreskowe i etykiety zajmując przy tym mniej miejsca

+Transpondery RFID są bardziej odporne na uszkodzenia

+Technologia RFID znajduje zastosowanie w niemalże każdej dziedzinie życia, pozwalając użytkownikom oszczędzić czas i pieniądze

+Znaczniki RFID mogą być odczytane znacznie szybciej niż kody kreskowe, a do tego nie jest istotne ich położenie względem czytnika

+Znaczniki są trudniejsze do zniszczenia lub usunięcia

-Brak jednolitego standardu protokołu RFID

-Dane mogą zostać przechwycone przez niepowołaną osobę, jeśli są to dane osobowe przechowywane np. w E-paszportach grozi to poważnymi konsekwencjami. To główny argument przeciwników RFID

-Nie zapewnia 100% pewności odczytu

**OPTYCZNE ROZPOZNAWANIE ZNAKÓW** OCR (ang. Optical Character Recognition) – zestaw technik lub oprogramowanie służące do rozpoznawania znaków i całych tekstów w pliku graficznym o postaci rastrowej. Zadaniem OCR jest zwykle rozpoznanie tekstu w zeskanowanym dokumencie (np. papierowym formularzu lub stronie książki).

Niegdyś termin oznaczał samo rozpoznawanie ciągów znaków, głównie drukowanych, które są łatwiejsze do rozpoznania, dziś również pisma odręcznego oraz cech formatowania, jak krój pisma, stopień pisma, interlinia (techniki służące do tego typu zaawansowanego rozpoznawania nazywane są terminem ICR), a nawet układów tabelarycznych, np. formularzy.

Techniki OCR wykorzystywane są m.in. przy digitalizacji zasobów bibliotek, a także jako ułatwienie przy odczytywaniu danych z formularzy wypełnianych pismem odręcznym. W obu przypadkach oprogramowanie OCR nie jest tak skuteczne jak człowiek, zatem w przypadkach wątpliwych (trudności z odczytaniem fragmentu) oraz w celu uniknięcia błędów nieodzowna jest weryfikacja wyniku OCR przez człowieka.

Nowoczesną metodą współpracy oprogramowania OCR z ludźmi jest technika reCAPTCHA. Nie wymaga ona zatrudniania osób specjalnie do weryfikacji OCR, lecz wykorzystuje rozproszoną aktywność milionów użytkowników Internetu, którzy wchodząc na strony internetowe zabezpieczone przez CAPTCHA rozpoznają fragmenty tekstu wymagające weryfikacji.

Zasada działania

Rozpoznawanie pisma jest możliwe dzięki zastosowaniu metod z dziedziny rozpoznawania wzorców zaliczanej do sztucznej inteligencji. Oprogramowanie OCR wykorzystuje różne metody segmentacji obrazu, np. progowanie, aby wyodrębnić poszczególne znaki z obrazu, które następnie są najczęściej osobno klasyfikowane jako poszczególne litery. Zwykle w tym procesie wykorzystywane są sieci neuronowe. Zazwyczaj, by wyeliminować pomyłki, program sprawdza całość rozpoznanego tekstu lub poszczególne wyrazy pod kątem poprawności ortograficznej i gramatycznej danego języka.

**TECHNOLOGIE BIOMETRYCZNE** Opierają się na metodach identyfikacji osób na podstawie indywidualnych cech anatomicznych lub behawioralnych takich jak: linie papilarne, tęczówka i siatkówka oka, kształt twarzy, głos oraz charakter pisma. Rozwiązania wykorzystujące technologie biometryczne są wykorzystywane głównie w systemach wymagających dużego bezpieczeństwa.

Biometryczne cechy fizyczne:

**Tęczówka oka** uznawana obecnie za najdoskonalszy identyfikator człowieka. Zostaje ona ukształtowana już pod koniec 15 tygodnia życia płodowego i pozostaje dalej w niezmienionej postaci (pomijając uszkodzenia mechaniczne i poważne choroby). Jej dodatkową zaletą jest fakt, że jest chroniona przez inne narządy przed ewentualnymi uszkodzeniami. Potwierdzeniem posiadania przez nią wielu indywidualnych cech jest fakt, iż do tej pory nie znaleziono dwóch identycznych tęczówek – pozwala ona na rozróżnienie nawet bliźniąt jednojajowych. Systemy biometryczne dokonujące identyfikacji na podstawie tęczówki oka porównują badaną tęczówkę z zapamiętanym wcześniej wzorcem. Zdjęcie tęczówki wykonywane jest w wysokiej rozdzielczości i na jego podstawie określane są cechy charakterystyczne każdej tęczówki. Okulary i soczewki kontaktowe, a także przypadkowe mrugnięcia czy ruchy głowy nie przeszkadzają podczas przeprowadzania identyfikacji.

**Linie papilarne**, znajdujące się na palcach (maleńkie bruzdy, których zupełnie przypadkowy układ jest wynikiem marszczenia się skóry w czasie rozwoju płodowego), mają zastosowanie w roli identyfikatora człowieka już od dawnych lat, jednak szersze zastosowanie znalazły dopiero w dobie informatyzacji. Jak do tej pory nie znaleziono dwóch identycznych wzorców linii papilarnych (nawet u bliźniaków jednojajowych), co uzasadnia wykorzystanie ich w roli identyfikatora. Największą wadą jest to, że linie papilarne nie mają osłony przed ewentualnym ich uszkodzeniem. Drobne rany zostają zregenerowane, ale poważniejsze mogą spowodować powstanie trwałych zmian we wzorze. Do odczytania służą czytniki linii papilarnych, do których przykłada się palec. System biometryczny porównuje sprawdzane linie z zapamiętanym wcześniej wzorcem. Weryfikacja następuje w samym czytniku lub na serwerze z bazą danych zapamiętanych wzorców.

**Geometria twarzy**. Twarz każdego człowieka ma cechy charakterystyczne, które mogą być wykorzystane do identyfikacji. Elementem pobierającym próbki jest aparat fotograficzny lub kamera, a z uzyskanego w ten sposób obrazu tworzony jest wzór matematyczny opisujący geometrię badanej twarzy. Dąży się do tego, aby w procesie tworzenia wzorca wybrać cechy niezmienne (np. wielkość oczu) tak, aby móc go wykorzystywać w zmieniających się warunkach (np. zmiana uczesania).

**Rozkład temperatury na twarzy.** Dopiero kilka lat temu odkryto, że u każdego z nas układ temperatur na twarzy jest niepowtarzalny. Można go dojrzeć dzięki specjalnej kamerze termowizyjnej wychwytującej zmiany temperatury w dalekiej podczerwieni. W przeciwieństwie do skanowania tęczówki nie jest to metoda inwazyjna. Można też identyfikować osobę w całkowitych ciemnościach. Są też jednak wady – temperatura twarzy zależy częściowo od temperatury na zewnątrz, trudniej też ocenić tożsamość osoby w okularach, bo odbijają promieniowanie. Jednak Amerykanie, którzy prowadzą badania w tym zakresie, identyfikują osoby już z dokładnością 99%.

**Geometria dłoni**. Dłoń ludzka ma wiele indywidualnych cech. Podczas pomiaru umieszcza się ją w specjalnym czytniku, który wykonuje trójwymiarowe zdjęcie, rejestrując długość, szerokość, grubość czterech palców (bez kciuka) oraz wielkość obszarów pomiędzy kostkami. Łącznie dokonywanych jest ponad 90 pomiarów różnych cech charakterystycznych. Wynik tych pomiarów jest przechowywany w pamięci urządzenia w formie 9 bajtowego wzorca (co w teorii daje ponad 1024 możliwych kombinacji). Identyfikator ten jest praktycznie unikatowy dla każdego człowieka. Co więcej nowoczesne systemy biometryczne wykorzystujące geometrię dłoni uwzględniają także jej zmiany spowodowane upływem czasu (np. starzenie).

Inne cechy fizyczne: DNA, zapach, kształt ust, kształt uszu, barwa głosu, siatkówka oka (dno oka), kształt linii zgięcia wnętrza dłoni, układ naczyń krwionośnych na dłoni lub przegubie ręki.

Biometryczne cechy behawioralne:

**Weryfikacja głosu** polega na nagraniu przez sprawdzaną osobę próbki jej głosu, wyszczególnieniu w niej cech charakterystycznych oraz porównaniu ich z wcześniej wprowadzonym wzorcem. Na opis charakterystycznych cech głosu składa się m.in. brzmienie oraz szybkość wymawiania wyrazów czy sposób stawiania akcentu. Identyfikacja z wykorzystaniem głosu jest najczęściej uzupełniana innymi metodami identyfikacji (np. badaniem geometrii twarzy).

**Sposób wykonywania odręcznego podpisu**. Biometria podpisu odręcznego tym różni się od innych biometrii, że nie polega na mierzeniu określonych cech wrodzonych, a pewnego wyuczonego zachowania. Podczas badania podpisu wykorzystuje się nie tylko charakterystykę wizualną (dwuwymiarowy obraz), ale także sposób, w jaki podpis został złożony, tj. dynamikę ruchu pióra. Podpisy są pobierane przy pomocy tabletu graficznego. Każdy pobrany podpis jest reprezentowany przez sekwencje wektorów, z których każdy zawiera współrzędne końcówki pióra, nacisk oraz kąty: azymut i nachylenie. Liczba wymiaru wektora zależy od liczby cech, jakie można pobrać za pomocą konkretnego tabletu.

Inne cechy behawioralne: sposób chodzenia, sposób pisania na klawiaturze, sposób reakcji mózgu – fala P300 – na pewne znane informacje-bodźce (fala P300 jest reakcją mózgu powstającą w czasie 300 msek od bodźca; jest ona charakterystyczna dla każdego człowieka).

**KARTY INTELIGENTNE** to małe plastikowe karty z mikroukładem, umożliwiającym przechowywanie różnego rodzaju informacji (klucze prywatne, hasła itp.). Karty inteligentne wykorzystywane są w systemach PKI, w bankowości oraz w innych instytucjach i firmach IT.

Główną zaletą kart inteligentnych jest ochrona przechowywanych danych przed niepowołanym dostępem oraz modyfikacją. Wykorzystanie kart inteligentnych pozwala budować systemy łatwe w użyciu, bezpieczne i tanie. Dlatego rynek ten sięga setek milionów rocznie wyprodukowanych kart i ciągle rośnie. Największa wadą kart jest ich podatność na uszkodzenia. Dotyczy to głównie układu scalonego, a także styków. Układ scalony można uszkodzić wyginając kartę, a styki ulegają wytarciu, zabrudzeniu lub zardzewieniu.

Wyróżniamy:

-Karty tłoczone

-Karty magnetyczne

-Karty elektroniczne (inteligentne, smart card)

--Pamięciowe, mikroprocesorowe

--Stykowe, bezstykowe (contactless), hybrydowe

Obszary zastosowań:

-Transakcje płatnicze

-Programy lojalnościowe

-Zapis danych biometrycznych (elektroniczny dowód osobisty)

-Infrastruktura klucza publicznego (podpis cyfrowy)

-System GSM

-System dostępu

-Systemy transportu publicznego (elektroniczny bilet)

-Telefonia komórkowa (karta SIM)

Ze względu na budowę układu scalonego możemy wyróżnić dwa typy kart inteligentnych:

K pamięciowe – są to karty elektroniczne posiadające pewien obszar pamięci służący do przechowywania danych. Zapis i odczyt odnosi się do struktury fizycznej karty. Przy czym sam sposób komunikacji, zapisu/odczytu z/do pamięci nie jest ustandaryzowany. Wyposażone są zazwyczaj w pamięć typu EEPROM bądź ROM. Czytnik kart pamięciowych odczytuje zapisane dane, traktując kartę jako maszynę stanów. Dodatkowo można zastosować metody kryptologiczne w celu zwiększenia bezpieczeństwa danych przechowywanych w pamięci. Największą zaletą tego rodzaju kart jest ich stosunkowo niska cena. Przykłady: k telefoniczna.

K procesorowe – wyposażone są w system operacyjny. Nazywane są też kartami mikroprocesorowymi. Zaopatrzone są w mikroprocesor podłączony do pamięci typu ROM, RAM i EEPROM. Pamięć ROM zawiera system operacyjny wgrywany podczas produkcji karty. Jest on identyczny dla wszystkich kart danego typu i nie może podlegać zmianom. Pamięć EEPROM wypełniana jest kodem oraz danymi aplikacji. Zapis danych w tym obszarze kontrolowany jest przez system operacyjny. Pamięć RAM pełni natomiast rolę pamięci roboczej. W pamięci pojedynczej kart mikroprocesorowej można umieścić dowolną aplikację (ograniczeniem jest tylko pojemność), przez co zastosowania tego typu kart są ograniczone tylko przez twórczą inwencję projektantów. Przykłady: SIM, VITAY

**KODY KRESKOWE** Kod kreskowy jest kombinacją jasnych i ciemnych kresek, ułożonych równolegle do siebie, o zróżnicowanych szerokościach, odzwierciedlającą ciąg ściśle określonych znaków. Występują kody jednowymiarowe – liniowe, dwuwymiarowe – piętrowe i matrycowe oraz kody hybrydowe. Symbolem kodu kreskowego nazywa się kompletny kod kreskowy składający się z jasnego marginesu poprzedzającego kod, znaku startu, znaków odwzorowujących dane, znaku stopu i ponownie jasnego marginesu. W skład symbolu wchodzą również znaki pisma, które zostały odzwierciedlone tym kodem kreskowym, umieszczone bezpośrednio nad lub pod kodem.

Zastosowanie m.in.

-W magazynach – do przyspieszenia przygotowywania remanentu i innych operacji magazynowych, np. przyjmowania i wydawania przesyłek, sortowania, konfekcjonowania, zarządzania magazynem oraz spisem z natury

-W zakładach produkcyjnych – do rejestrowania przepływu surowców i półproduktów w procesie produkcyjnym

-W firmach wysyłkowych - do oznaczania wysłanych paczek

-W firmach kurierskich – do oznaczania przesyłek

-W dużych zakładach – do identyfikowania środków trwałych i szybkiego sporządzania spisów, do sporządzania kart klienta, kart rabatowych, kart identyfikacyjnych

-W przedsiębiorstwach spedycyjnych, logistycznych, kurierskich i na poczcie – śledzenie przesyłek

-W sklepach – odczytywanie cen przesuwając produkty z kodami nad czytnikiem

-Znakowanie rachunków, listów, kopert itp.

Podział kodów:

* Linearne - symboliki tego rodzaju składają się z pionowych linii różnej szerokości, rozdzielonych białymi przestrzeniami. Przykłady: Code 128, Code 39, EAN-13, EAN-8, ITF, UPC-A, UPC-E
* Dwuwymiarowe: mają one dużo większą pojemność danych, jeden kod dwuwymiarowy może pomieścić do 3,5 tys. znaków, gdy kod liniowy do 50 znaków, dzielimy na *macierzowe* – wyglądają jak macierze punktów, przykłady: Aztec, Data Martix, MaxiCode, QRCode, *wielowierszowe liniowe (piętrowe)* – linearne kody kreskowe z bardzo krótkimi paskami ułożone jeden na drugim, przykłady: Codablock F, Code 49, PDF417, SuperCode.
* Trójwymiarowe (wypukłe) - jest to tak naprawdę kod liniowy, tylko zamiast czarnych i białych linii wykorzystuje wgłębienia i wypukłości

**KODY KRESKOWE A RFID**

Technologia RFID ma wiele zalet, których brakuje kodom kreskowym, między innymi:

-Możliwość zmiany danych zawartych w znaczniku,

-Znacznik nie musi być widoczny w trakcie odczytu,

-Większy zasięg odczytu (zależnie od rodzaju znacznika),

-Możliwość zapisania większej ilości danych,

-Możliwość równoczesnego odczytu wielu znaczników,

-Większa odporność na zabrudzenie i uszkodzenia,

-Możliwość innych zastosowań poza przechowywaniem danych.

Mimo tak wielu zalet znaczników RFID, kody kreskowe prawdopodobnie nie wyjdą nigdy z użycia, z takich przyczyn jak:

-Niższy koszt jednostkowy,

-Zbliżona a nawet wyższa dokładność odczytu,

-Neutralny wpływ materiału, na którym znajduje się kod kreskowy na jego odczyt,

-Brak międzynarodowych ograniczeń związanych z częstotliwościami pracy urządzeń RFID,

-Brak zastrzeżeń ze strony obrońców prywatności,

-Dojrzała i powszechnie używana technologia.

# Metody szacowania wielkości projektów informatycznych

Uruchomienie dowolnego projektu informatycznego wymaga estymacji podstawowych parametrów tego projektu. Proces estymacji obejmuje cztery podstawowe etapy:

* estymacja rozmiaru,
* kosztów,
* pracochłonności oraz
* czasu realizacji.

Estymacja kosztów przedsięwzięcia informatycznego pozwala na:

* podjęcie decyzji o dokonaniu inwestycji lub innych decyzji finansowych obejmujących pracochłonność wytwarzania oprogramowania
* ustalenie budżetu i harmonogramu projektu jako podstawy planowania i kontroli
* szacowanie kosztów oprogramowania oraz decyzje związane z zarządzaniem ryzykiem podjęcie decyzji, które części systemu są przeznaczone do wytwarzania, ponownego użycia, zakupu.

**Metody szacowania kosztów**

**1.  Estymacja przez analogię**

Estymacja projektu odbywa się na podstawie doświadczeń z **innych projektów** realizowanych przez wykonawcę. Niezbędne w tej metodzie informacje dotyczące zakończonych projektów są zbierane i przechowywane w specjalnych katalogach. Informacje te muszą  obejmować specyfikę projektów pozwalającą na ustalenie **podobieństw i różnic** między poszczególnymi projektami oraz dane dotyczące ostatecznego rozmiaru, nakładu pracy, czasu realizacji i kosztu.

**2.   Metoda bottom- up**

Estymacja jest wykonywana dla pojedynczych składowych komponentów oprogramowania i poszczególnych etapów ich realizacji: projektu, kodowania, testowania itd., za pomocą jednej lub kilku pozostałych metod. Wyniki poszczególnych estymacji są sumowane.

**3.  Modele parametryczne**

Estymacja jest wykonywana  na podstawie funkcji wiążących rozmiar oprogramowania i inne dodatkowe czynniki kosztu z nakładem pracy i czasem realizacji. Ogólny wzór w modelach parametrycznych wygląda następująco:

**Nakład = A\*RozmiarB,**

gdzie:

A – reprezentuje zbiór czynników mających wpływ na pracochłonność przedsięwzięcia

B – określa nieliniowy wpływ skali rozmiaru przedsięwzięcia na wymagany nakład pracy.

Przykładowe modele z tej dziedziny to: **COCOMO**, **SLIM**, **SoftCost**.

**4.  PERT**

W metodzie tej estymacje są dokonywane na podstawie rozkładu prawdopodobieństwa dla pesymistycznego **l**, optymistycznego **h** oraz średniego **m** przypadku kosztu jako średniej ważonej tych wartości z tym, że najbardziej prawdopodobna wartość ma wagę 4 a pozostałe 1

Koszt = (l+4\*m+h)/6

**5.  Metoda top-down**

Estymacja jest wykonywana dla komponentu jako całości na podstawie jego ogólnych własności odwołując się do dotychczasowych doświadczeń. Wyniki estymacji są następnie dzielone dla między elementy składowe komponentu.

**6.   Metoda Delphi(delficka)**

Do realizacji tej metody niezbędny jest zespół ekspertów. Eksperci równolegle estymują podstawowe parametry projektu. Wyniki estymacji są następnie konfrontowane. Proces estymacji postępuje dalej iteracyjnie, aż do uzgodnienia wyników.

# Mobilność w systemach informacyjnych

Mobilność to ogólny termin określający zdolność do używania technologii w ruchu (w przeciwieństwie do przenośności, gdzie urządzenia mogą być wykorzystywane dopiero po ich rozłożeniu w konfiguracji stacjonarnej).

**Charakterystyka mobilności:**

* ograniczone zasoby urządzeń mobilnych
* ograniczona energia
* obniżone bezpieczeństwo
* częste zmiany warunków pracy urządzeń mobilnych

**Przyczyny rozwoju**

• Rozwój technologiczny

• Miniaturyzacja

• Niski pobór prądu – długa praca na bateriach

• Nowoczesne łącza bezprzewodowe

• Spadek cen

• Powszechna dostępność

Nowe technologie, miniaturyzacja i coraz większa dostępność nowoczesnych urządzeń, mobilnych i bezprzewodowych pozwala na budowę systemów o nieznanych dotąd możliwościach. Mały rozmiar tych urządzeń, ich niski pobór prądu, programowalność oraz odporność na zmienne warunki środowiska umożliwia szerokie ich zastosowanie w trudnych warunkach terenowych. Bezprzewodowe interfejsy nie wymagają budowy złożonej i kosztownej infrastruktury, co znacznie przyspiesza wdrożenie i ewentualne zmiany w projektowanych systemach.

**Zalety**

• Dostęp do danych niezależnie od miejsca i czasu,

• Skrócony czas dostępu do danych (czas reakcji),

• Skalowalność– możliwość ciągłego rozwoju systemu w zależności od zadań,

• Efektywność wykorzystania zasobów – możliwość współdzielenia zasobów niezależnie od fizycznej lokalizacji użytkownika i zasobu,

• Łatwość realizacji nowych usług, w tym multimedialnych.

Stosowanie systemów mobilnych w firmie ma wiele zalet. Podstawowa z nich jest nie podlegający dyskusji **dostęp do danych** niezależnie od miejsca i czasu. Związany jest z tym **skrócony czas dostępu** do danych (czas reakcji), co może mieć olbrzymie znaczenie dla administratorów systemów komputerowych czy produkcyjnych jak i podejmującej **błyskawiczne decyzje**, przebywającej poza firma kadry kierowniczej. Systemy mobilne są w pełni **skalowalne**, czyli potrafią dynamicznie dostosowywać sie do ciągłego wzrostu potrzeb i zadań, jakie przed nim stoją. Inna ważna cecha jest efektywność wykorzystania zasobów, czyli możliwość **współdzielenia** zasobów niezależnie od fizycznej lokalizacji użytkownika i zasobu. Łatwość realizacji nowych usług, w tym **multimedialnych** jest najważniejsza chyba, z punktu widzenia użytkownika końcowego, ich zaleta.

**Zastosowania:** pracownicy handlowi, agenci ubezpieczeniowi,personel medyczny, ekipy remontowo-budowlane, zarządzanie flotą, kierowcy(zarówno informacja o ruchu jak i mapy), inteligentne budynki, RFID, sieci sensorowe, bezpilotowce

# Modelowanie funkcji systemu informacyjnego - techniki modelowania, hierarchie funkcji, korzyści z modelowania funkcji i tworzenia hierarchii funkcji

Modelowanie funkcji systemu odbywa się za pomocą diagramów przepływu danych służących do prezentowania sposobu, w jaki dane przepływają oraz są przetwarzane w systemie. Opisują też procesy przetwarzające dane.

* Zbiornik danych (magzyn danych- data store)- składowa systemu przechowująca dane trwałe lub ulotne.
* Proces(process)- operacja(czynność) wykonywana przez system
* System zewnętrzny(external system)- system zewnętrzny z którym system wymienia dane
* Przepływ danych(data flow)- opisuje dane przesyłane między procesami, systemami zewnętrznymi i zbiornikami danych. Może być jedno lub dwukierunkowy. Przepływ pomiędzy procesami oznacza fakt przekazywania danych pomiędzy tymi procesami Nie jest przy tym określone jak system przekazuje dane. Przepływ pomiędzy procesem a zbiornikiem danych oznacza fakt zapisywania, odczytywania lub modyfikowania przez proces danych zawartych w zbiorniku. Przepływ pomiędzy procesem i systemem zewnętrznym opisuje fakt wymiany danych z tym systemem. Niedopuszczalne są przepływy pomiędzy dwom zbiornikami oraz systemem zewnętrznym i zbiornikiem danych.

**Diagram Hierarchii Funkcji**

Wymagania funkcjonalne opisują funkcje (czynności, operacje) wykonywane przez system. Najprostszą i najbardziej przydatną metodą modelowania funkcji jest konstruowanie hierarchii funkcji w postaci **Diagramów Hierarchii Funkcji** (ang. Function Hierarchy Diagram - FHD). Metoda ta jest bardzo szybka w realizacji, umożliwia analitykowi dokładne zdefiniowanie zakresu przyszłego systemu, który można łatwo kontrolować, uzgadniać oraz wykorzystywać w dalszej pracy np. przy tworzeniu diagramów przepływów danych i diagramów związków encji.

Modelowanie funkcji ma służyć zrozumieniu, co firma robi lub powinna robić w przyszłości, aby osiągnąć cele.

Konstruowanie hierarchii funkcji polega na tym, że:

>>każda funkcja na diagramie określa, co system ma robić, a nie w jaki sposób;

>>funkcje tworzą hierarchię - pierwsza funkcja ogólna opisuje główną funkcję systemu, dzieli się następnie na funkcje bardziej szczegółowe, dochodząc aż do funkcji elementarnych;

>>każda funkcja nadrzędna powinna całkowicie opisywać funkcje szczegółowe oraz wszystkie funkcje szczegółowe muszą opisywać funkcję nadrzędną;

>>wykorzystuje się metody z góry na dół (*top-down*) od ogółu do szczegółu;

>>w celu przeprowadzenia weryfikacji diagramu wykorzystuje się metody od dołu do góry (*bottom-up*);

>>każda funkcja powinna być opisana w języku naturalnym na tzw. formularzu opisu zawierającym: nazwę funkcji, opis, dane wejściowe, źródło danych wejściowych, wynik - rezultat wykonania funkcji, warunek wstępny - co musi być wcześniej zrealizowane, warunek końcowy - do czego wykorzystuje się efekty realizacji tej funkcji, cel realizacji funkcji.

Tworząc diagram hierarchii funkcji należy uwzględnić następujące zasady:

>>funkcję przedstawia prostokąt (o ostrych lub zaokrąglonych rogach);

>>każda funkcja jest identyfikowana nazwą (a nie skrótem); nazwa to czasownik lub rzeczownik (odczasownikowy);

>>każda funkcja powinna mieć jednoznaczną etykietę;

FHD powinien mieścić się na jednej kartce (jeśli nie to stawiamy kropeczki oznaczające rozwinięcie funkcji w innym miejscu, np.: na nowej stronie).

Aby uznać diagram za poprawny należy sprawdzić, czy jest: spójny, kompletny (obejmuje całość zagadnień), łatwy w modyfikacji, zwięzły i jednoznaczny w nazewnictwie oraz czytelny.

***Diagram Przepływu Danych***

Jedną z metod wykorzystywanych na etapie analizy i projektowania służących do modelowania funkcji systemu jest **Diagram Przepływu Danych** (ang. Data Flow Diagram - DFD). Przedstawia on, w jaki sposób dane przepływają w systemie oraz opisuje procesy przetwarzające dane. Tworzenie diagramu DFD opiera się na następujących kategoriach pojęciowych: proces, przepływ danych, magazyn danych, terminator i odpowiadających im symbolach graficznych. Szczegółowy opis wszystkich kategorii występujących na DFD zawiera słownik danych omawiany w podrozdziale 3.2.

**Proces** (ang. process) oznacza transformację danych wejściowych w wynikowe i odpowiada tym składnikom systemu, które przetwarzają dane. Procesy otrzymują i przesyłają dane za pośrednictwem przepływów danych. Kojarzą się one z procedurą, której specyfikacja jest przedstawiona przy użyciu innych technik strukturalnych. Nazwa procesu powinna opisywać czynność wykonywaną na określonym obiekcie, jak np.: *aktualizacja konta klienta, rejestracja danych klienta, wyliczenie stawki amortyzacji sprzętu.*

**Przepływ danych** (ang. data flow) - opisuje zbiór danych przepływający pomiędzy dwoma obiektami w systemie. Przedstawia się go za pomocą linii ze strzałkami określającymi kierunek przesyłania informacji. Linie są skierowane najczęściej jednostronnie. Jeśli przekazywana informacja jest zwrotna używa się kolejnych linii lub strzałek dwukierunkowych. Nazwa przepływu to rzeczownik w liczbie pojedynczej. Przykładowe przepływy danych: *kwestionariusz osobowy*, *umowa*, *faktura dla klienta*.

**Magazyn danych** (ang. data store) - inaczej składnica danych służy do przechowywania danych w postaci jednorodnych kolekcji. Zaistnienie magazynu danych w diagramie ma sens, gdy przechowywane dane służą do realizacji, co najmniej dwóch procesów. Charakter magazynu danych zależy od stopnia szczegółowości diagramu. Nazwa magazynu danych to rzeczownik w liczbie mnogiej, jak np.: *pracownicy, towary, wydawnictwa, faktury klienta*.

**Terminator** (ang. terminator) - obiekt zewnętrzny w stosunku do systemu reprezentujący źródła lub miejsca przeznaczenia informacji. Terminatorami są obiekty, z którymi system komunikuje się. Nazwa terminatora, to rzeczownik liczby pojedynczej. Przykładowe terminatory to: *Szef* (odbierający raporty o niezapłaconych fakturach klienta), *Klient* (składający zamówienie na towary), *Moduł finansowo-księgowy systemu informatycznego*(korzystający z zestawień sprzedaży projektowanego systemu ewidencji sprzedaży, a nie będący przedmiotem projektu).

***Dekompozycja funkcji***

Modelowanie funkcji systemu polega na stworzeniu hierarchicznej grupy diagramów DFD. Zazwyczaj pierwszy diagram DFD nazywa się diagram kontekstowy systemu albo diagram środowiska (otoczenia systemu).

**Diagram kontekstowy** (inaczej diagram DFD poziomu zerowego) jest specjalnym graficznym schematem przepływu danych, który definiuje zakres i granice systemu. System przedstawiony jest na diagramie jako pojedynczy proces powiązany bezpośrednio przepływami danych z terminatorami. Całość ma na celu przedstawienie powiązań systemu ze środowiskiem zewnętrznym.

Następnie tworzony jest **diagram systemowy**, ukazujący główne funkcje systemu. Każda funkcja jest przedstawiona w postaci hierarchicznej grupy diagramów niższego poziomu.

Diagramy niższych poziomów to **diagramy szczegółowe**. Zbiór diagramów DFD dla systemu wraz z opisem elementów występujących na diagramie w słowniku danych stanowi model funkcji systemu. Jest to graficzna mapa procesów ukazująca przepływ danych miedzy procesami w systemie oraz między światem zewnętrznym a systemem.

Proces tworzenie DFD opiera się na analizie obiegu dokumentów w organizacji i składa się z następujących kroków:

identyfikacja kluczowych dokumentów wykorzystywanych w systemie oraz wytwarzanych przez system;

zbudowanie zintegrowanego fizycznego modelu przepływu dokumentów ze wskazaniem źródła i odbiorcy informacji.

# Modelowanie struktur wymiany danych

Przy analizie danych powszechnie stosowane są dwa podejścia: „od dołu do góry” i „od góry do dołu”

Analiza danych według podejścia „od góry do dołu” oznacza zastosowanie technik diagramowych odzwierciedlających to, co uważamy za obiekty zainteresowania przedsiębiorstwa i związki między tymi obiektami. Analiza „od góry do dołu” jest często uważana za modelowanie koncepcyjne, ponieważ jest to działanie na wysokim poziomie, często abstrakcyjne. Efektem końcowym takiego modelowania jest diagram związków encji, który w prosty sposób może być przekształcony w relacyjny schemat struktur tablicowych.

Analiza danych „od dołu do góry” nie zajmuje się abstrakcyjnymi pojęciami, lecz danymi konkretnymi. Aby wykonać analizę „od dołu do góry” musimy mieć pulę danych, uzyskanych np. przez badanie istniejących dokumentacji przedsiębiorstwa. Do tej puli danych stosujemy szereg reguł. Podejście „od dołu do góry” w analizie danych nosi nazwę normalizacji.

**Diagram związków obiektów**

Model danych służy do wyrażenia struktury danych projektowanego lub istniejącego systemu. Określa on:

* typy danych występujących w systemie,
* wzajemne powiązania między danymi,
* ograniczenia nałożone na dane.

**Podział modeli danych:**

* Modele konceptualne – opisują dane za pomocą pojęć z których korzystają użytkownicy. Taki opis jest całkowicie niezależny od rozważań na temat pamięci, efektywności i innych szczegółów implementacyjnych.
* Modele fizyczne – opisują dane w kategoriach sposobu ich przechowywania w pamięci komputera i nie są zrozumiałe dla użytkowników.
* Modele implementacyjne - operują pojęciami w zasadzie zrozumiałymi dla użytkowników, ale bliżej związanymi z konkretną strukturą (logiczną) danych w pamięci komputera. Modele: relacyjny, sieciowy i hierarchiczny.

Podstawowe koncepcje i definicje:

* Encja jest rzeczą lub obiektem mającym dla nas znaczenie, rzeczywistym bądź wyobrażonym, o którym informacje muszą być znane.
* Każda encja musi być jednoznacznie identyfikowana – każda instancja (wystąpienie) musi być wyraźnie odróżniana od wszystkich innych instancji tego typu encji.
* Związek jest nazwanym istotnym powiązaniem istniejącym między dwiema encjami . Wyróżniamy związki: wiele do jeden, jeden do jeden, wiele do wiele; wymagane lub opcjonalne. Związki pomiędzy obiektami reprezentują powiązania świata rzeczywistego np. encja WYRÓB *składa się* z encji CZĘŚCI. Obiekt ZAMÓWIENIE *ma* wiele obiektów POZYCJA\_ZAMÓWIENIA.
* Atrybut jest dowolnym szczegółem służącym do kwalifikowania, identyfikowania, klasyfikowania, określania ilości lub wyrażania stanu encji – jest dowolnym opisem mającym znaczenie dla encji. Musi opisywać encję przy której występuje. (Atrybut jest funkcją, przypisującą obiektowi wartość cechy ze zbioru wartości tej cechy, czyli z dziedziny).
* Aby każdą instancję encji można było odróżnić od wszystkich innych instancji tego typu encji , każda musi być jednoznacznie identyfikowalna. Jednoznacznym identyfikatorem może być atrybut, kombinacja atrybutów, kombinacja związków, lub kombinacja atrybutów i związków.

**Zasady tworzenia diagramów związków encji**

Model danych systemu zbudowany na podstawie modelu encja - atrybut –związek reprezentuje wszelkie obiekty oraz powiązania między nimi istotne z punktu widzenia funkcjonowania systemu.

Pięć kroków budowy modelu danych:

1. Identyfikacja (wydzielenie) zbioru typów obiektów wraz z ich atrybutami kluczowymi;
2. Identyfikacja powiązań bezpośrednich między obiektami (ewentualnie z wykorzystaniem tablicy krzyżowej powiązań ) oraz ich rodzaju;
3. Przekształcenie tablicy krzyżowej powiązań w pojęciowy model danych i identyfikacja pozostałych atrybutów obiektów.
4. Przekształcenie każdego z powiązań typu wiele do wiele na dwa powiązania jeden do wiele i identyfikacja dodatkowych atrybutów charakterystycznych dla nowo powstałych obiektów.
5. Sprawdzenie poprawności otrzymanej struktury poprzez porównanie z wymaganiami systemu.

\*\*\*

Diagram związków encji (z ang. ERD - entity relationship diagram) jest jednym ze sposobów przedstawienia modelu logicznego bazy danych. Jest to rysunek encji i związków pomiędzy nimi, który ułatwia analizę wybranego obszaru świata rzeczywistego modelowanego w bazie danych. Do wspomagania procesu projektowania systemu informatycznego służy m.in. pakiet aplikacji Oracle o nazwie Designer/2000.

# Narzędzia integracji systemów informacyjnych

EDI, CORBA, DCOM, RPC, RMI, Web Service, ESB

**EDI (Electronic Data Interchange)** – ustandaryzowana metoda wymiany danych pomiędzy komputerami bądź sieciami komputerowymi, z wykorzystaniem standardowych, zaakceptowanych formatów komunikatów. EDI to sposób wymiany informacji pomiędzy kontrahentami, charakteryzujący się przesyłaniem dokumentów wyłącznie w formie elektronicznej („bezpapierowej”), wymianą dokumentów spełniających uzgodnione wcześniej standardy (definicja pól dokumentu i ich zawartości), wymianą danych bezpośrednio pomiędzy systemami informatycznymi partnerów handlowych; dane będące przedmiotem elektronicznej wymiany są odpowiednikami tradycyjnych dokumentów handlowych takich jak: faktura, zamówienie, awizo dostawy, harmonogram produkcji itp. Kluczowym aspektem EDI jest standaryzacja przesyłanych informacji oraz bezpieczeństwo komunikacji.

Celem EDI jest wyeliminowanie wielokrotnego wprowadzania danych oraz przyspieszenie i zwiększenie dokładności przepływu informacji dzięki połączeniu odpowiednich aplikacji komputerowych między firmami uczestniczącymi w wymianie. Użycie EDI pozwala poprawić czasową dostępność informacji logistycznej, poszerzyć i uściślić dane, a także zmniejszyć pracochłonność procesu. Aby w pełni wykorzystać zalety EDI, uczestnicy kanału logistycznego powinni się komunikować za pośrednictwem komputera. Innymi słowy, efektywne wdrożenie EDI wymaga bezpośredniej komunikacji między systemami komputerowymi, zarówno nabywców jak i sprzedawców produktu.

EDI nie określa sposobu przesyłania komunikatów – mogą one być przesyłane przez dowolne medium, którym posługują się obie strony transmisji. Może to być transmisja modemowa, poprzez FTP, HTTP, AS1, AS2.

Wady i zalety:

+Eliminujemy błędy klawiatury,

+Automatycznie wielokrotnie wykorzystuje wprowadzane dane,

+Pozwala na szybką transmisję danych,

+Szybki czas wypełniania typowych formularzy,

+Eliminuje dokumentację papierową,

+Usprawnia księgowość,

+Pozwala przesunąć personel do innych zadań,

+Obniża koszty operacji administracyjnych

+systemy EDI nie wymagają skomplikowanego sprzętu, korzystają z sieci transmisji danych

-Poziom skomplikowania komunikatów EDI

-Mnogość standardów oraz wersji katalogów i komunikatów

-Problemy z utrzymywaniem i aktualizacją standardów

-Wysokie koszty, trudny i długotrwały proces wdrażania

-Brak pełnego standardu wymiany informacji

-Często zbyt kosztowne dla małych i średnich przedsiębiorstw

**CORBA (Common Object Request Broker Architecture)** – technologia zapewniająca komunikację pomiędzy obiektami pracującymi w heterogenicznych (różnorodnych) systemach komputerowych. Obiekty pełniące dowolne funkcje mogą być zaimplementowane w różnych językach programowania, na dowolnej platformie sprzętowej, pod kontrolą różnych systemów operacyjnych.

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

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

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

Standardy tworzone są według zasady "najpierw standard, potem implementacje" (czyli tak samo jak w W3C i zupełnie inaczej niż w IETF).

Jedną z wad architektury CORBA jest brak standardowego i szeroko zaimplementowanego mechanizmu bezpieczeństwa.

Popularne implementacje standardu CORBA to np. ORBit, OMNIOrb.

Wady i zalety:

+Architektura zapewnia bardzo wygodny sposób zapewniania współpracy różnorodnych systemów

+Daje możliwość zamaskowania systemów odziedziczonych w pełni wykorzystując ich możliwości

+Zapewnia dostęp do obiektów niezależnie od platformy implementacji obiektu dzięki wykorzystaniu języka opisu styków IDL zamiast języka programowania, co jest podstawowym wymaganiem przy współpracy systemów dostarczonych przez różnych dostawców

+Otwarty standard

+Obsługa wielu platform

+Obsługa wielu języków

+Skalowalność

-Duże koszty uruchomienia

-Słaba wydajność, mało efektywne działania w czasie rzeczywistym

-Zapory (?)

-Brak standardowego i szeroko zaimplementowanego mechanizmu bezpieczeństwa

**DCOM (Distributed Component Object Model)** – interfejs programistyczny realizujący rozproszony obiektowy model składników. Jest opatentowaną technologią firmy Microsoft służącą do budowania składników programowych i zapewniania komunikacji między nimi w małej sieci komputerowej. Skonstruowany został również zestaw bazowy takich składników.

Rozwinięty z COM jako odpowiedź na CORBA, stał się później częścią COM+. Został przez Microsoft uznany za przestarzały na rzecz platformy .NET Framework.

Dodanie litery "D" do COM spowodowane było szerokim wykorzystaniem technologii DCE/RPC (Distributed Computing Environment / Remote Procedure Call), a dokładniej udoskonalonej przez Microsoft wersji, znanej jako MSRPC.

*COM - Standard definiowania i tworzenia interfejsów programistycznych na poziomie binarnym dla komponentów oprogramowania wprowadzony przez firmę Microsoft wraz z bibliotekami zapewniającymi podstawowe ramy i usługi dla współdziałania komponentów COM i aplikacji. Nowe środowisko programistyczne Visual Studio oparte na frameworku .NET umożliwia komunikację ze starszymi aplikacjami z interfejsem typu COM. Na bazie standardu COM została zdefiniowana znaczna część niskopoziomowego API dla produktów firmy Microsoft m.in. dla DirectX, SQL Server, MS Access, MSHTML, MSXML.*

Stosownie do tego określenia, DCOM miał rozwiązać problemy:

-Serializacja argumentów i wartości zwracanych wywoływanych metod, oraz reprezentację ich podczas połączenia.

-Rozproszone zbieranie nieużytków - zapewnienie, że połączenie nawiązane przez program klienta zostanie zwolnione, gdy na przykład proces klienta ulegnie awarii lub połączenie sieciowe zostanie przerwane.

Jednym z kluczowych czynników w rozwiązywaniu tych problemów jest zastosowanie DCE / RPC jako mechanizmu RPC wykorzystanego w DCOM. DCE / RPC ma ściśle określone zasady dotyczące serializacji i kto jest odpowiedzialny za zwolnienie pamięci.

Wady i zalety:

+Wsparcie

+HYPED MARKET (?)

-Wsteczna kompatybilność

-Interoperacyjność (zdolność sieci telekomunikacyjnych do efektywnej współpracy w celu zapewnienia wzajemnego dostępu użytkowników do usług świadczonych w tych sieciach) (?)

**RPC (Remote Procedure Call)** – 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.

RPC zdefiniowany jest w RFC 1057. RPC używa do kodowania danych formatu XDR (eXternal Data Representation) zdefiniowanego w RFC 1832.

Protokoły tego typu (jak RPC, CORBA, DCOM, czy XML-RPC) mają na celu ułatwienie komunikacji pomiędzy komputerami. Na typowy scenariusz użytkowania składają się:

-Serwer (czyli program oferujący usługi, np. drukowania) przez cały czas nasłuchuje na wybranym porcie, czy ktoś się z nim nie łączy.

-Klient (czyli program który potrzebuje jakiejś usługi od serwera na innym komputerze) nawiązuje z nim łączność poprzez sieć komputerową.

Klient wysyła swoje dane we wcześniej ustalonym przez programistów klienta i serwera formacie. Serwer realizuje usługę i odsyła potwierdzenie lub kod błędu. Powyższe protokoły same zapewniają cały powyższy mechanizm, ukrywając go przed klientem. Może on nawet nie wiedzieć, że łączy się z innym komputerem - z punktu widzenia programisty zdalne wywołanie procedury serwera wygląda jak wywołanie dowolnej innej procedury z programu klienta.

Wady i zalety:

+prosty interfejs

+popularny język programowania (proceduralny)

+obsługuje modułowe i hierarchiczne modele systemów rozproszonych

+Łatwość pisania przenośnych programów rozproszonych / sieciowych. Implementacja komunikacji client–server sprowadza się do prostego wywoływania odpowiednich funkcji.

-Problemy z przekazywaniem argumentów do funkcji (np. przekazywanie wskaźników), procedury nie mogą korzystać ze zmiennych globalnych

-Różne systemy, języki, kompilatory, architektury sprzętowe wymuszają określony format wbudowanych typów danych (zakres liczb, kolejność bajtów), a także wprowadzają różnorakie ograniczenia na rzutowanie jednego typu na drugi itp.

-Trudno zrealizować zdalne funkcje o zmiennej liczbie argumentów, oraz takie w których zmieniają się typy tych argumentów (np. standardowa funkcja printf).

-Odpowiednie zabezpieczenie systemu obsługującego RPC staje się wręcz krytyczne.

-Trudno kwestią jest zagwarantowanie wywołania procedury na serwerze, sprawdzenie czy procedura została rzeczywiście wykonana oraz czy została wykonana tylko raz.

-Technologia RPC generuje spore narzuty.

**RMI (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.

RMI dostarcza zestaw narzędzi umożliwiających pisanie programów rozproszonych przy użyciu takiej samej syntaktyki i semantyki jak dla programów pracujących na jednym komputerze (jednej maszynie javy, JVM).

RMI oddziela definicję zachowania (interfejs) od jego implementacji i pozwala na uruchamianie implementacji na oddzielnej maszynie javy. Odpowiada to programowaniu typu klient-serwer, gdzie klient wykorzystuje zdefiniowany interfejs, a serwer dostarcza jego implementację. RMI dba o szczegóły dotyczące protokołu, przekazywania parametrów zwracania wyników.

Interfejs javy nie zawiera wykonywalnego kodu, więc RMI używa dwóch klas implementujących ten sam interfejs. Pierwsza klasa implementuje działanie (*service implementation*) i jest uruchamiana na serwerze. Druga klasa funkcjonuje jako pośrednik (*service proxy*) dla wywołań klienta. Program klienta wywołuje metody w klasie pośredniczącej, a ta przekazuje je do zdalnego serwera oraz pobiera zawracane wyniki.

Implementacja RMI oparta jest na trzech warstwach abstrakcji. Pierwsza to warstwa szkieletów i trzonów (*stubs and skeletons*). Warstwa ta jest bezpośrednio używana przez programistę. Przyjmuje ona wywołania metod skierowane do interfejsu i przekazuje do zdalnych wywołań RMI. Kolejną warstwą jest warstwa referencji (*remote reference layer*). Warstwa ta odpowiada za interpretację i zarządzanie referencjami do obiektów przekazywanych między klientem, a serwerem. Dokonuje automatycznej serializalizacji oraz deserializacji obiektów lub przekazuje zdalne referencje do nich. Warstwa transportowa (*transport layer*) bazuje na protokole TCP/IP i dostarcza podstawowego połączenia między systemami. Może być zastąpiona implementacją bazującą np. na protokole UDP bez zmiany wyższych warstw.

Warstwa szkieletów i trzonów

Szkielet jest klasą pomocniczą generowaną przez RMI, która potrafi komunikować się z trzonem poprzez łącze RMI. Szkielet jest klasą działającą na serwerze. Odbiera parametry z połączenia RMI, przekazuje je do wywoływanej klasy serwera. Otrzymany wynik przekazuje z powrotem do klienta poprzez łącze RMI. Od wersji 2 JDK klasy szkieletów nie są wymagane. Trzon łączy się bezpośrednio z obiektami na zdalnym serwerze. Trzon jest klasą generowaną przez RMI, służy jako pośrednik między programem klienta, a RMI. Przekazuje parametry wywołania do łącza RMI, odbiera wynik i zwraca do programu klienta.

Warstwa zdalnych referencji

Warstwa to dostarcza obiekt klasy RemoteRef, który reprezentuje łącze do zdalnej implementacji. Trzon wykorzystuje metodę invoke() do przekazywania wywołań do serwera.

W wersji JDK 1.1 znajduje się tylko jedna metoda zdalnego wywoływania: unicast. Zanim klient wywoła zdalną metodę, zdalna usługa musi zostać utworzona na serwerze i wyeksportowana do mechanizmu RMI.

JDK 1.2 dostarcza nowa metoda zdalnego wywoływania: za pomocą aktywowania. RMI samo zajmuje się tworzeniem zdalnych obiektów oraz odtwarzaniem ich stanu z dysku.

Możliwe jest dodawanie nowych rodzajów wywołań, np. skierowanych do wielu redundantnych serwerów usług.

Warstwa transportowa

Tworzy połączenia między wirtualnymi maszynami javy. Wszystkie połączenia są oparte na strumieniach i używają TCP/IP, nawet jeśli obie JVM są uruchomione na tej samej maszynie.

RMI używa protokołu JRMP (Java Remote Method Protocol) opartego na bazie TCP/IP. Jest to strumieniowy protokół występujący w dwóch wersjach. Pierwsza wersja użyta w JDK 1.1 wymaga klas szkieletów na serwerze. Druga wersja pojawiła się wraz z JDK 2, zastała zoptymalizowana oraz nie wymaga klas szkieletów. Istnieją implementacje RMI, które nie używają JRMP (np. BEA Weblogic).

Wady i zalety:

+Prosty w implementacji i używaniu, która prowadzi do bardziej wytrzymałych, elastycznych aplikacji i lepszego ich utrzymania

+Zapewnia niezależność programów od procesora (wystarczy działająca maszyna Java VM).

+Troszczy się o szczegóły przesyłania obiektów, ich serializację i deserializację

-Wsparcie tylko dla programów napisanych w javie, zarówno klient jak i serwer muszą być aplikacjami lub apletami javy i nie można komunikować się z programami napisanymi w żadnym innym języku programowania za pomocą tego mechanizmu

-Nie może używać kodu spoza zakresu Javy

-Zarówno klient jak i serwer potrzebują dostępu do najświeższych identycznych definicji klas obiektów

**WEBSERWIS** To metoda komunikacji dwóch elektronicznych urządzeń poprzez WWW. WS jest systemem utworzonym w celu wspierania interoperacyjnej interakcji komputer-komputer poprzez sieć. Posiada interfejs opisany w formacie przetwarzanym maszynowo (WSDL). Inne systemy współdziałają z webserwisem w sposób opisany w jego opisie (WSDL) przy pomocy wiadomości SOAP, przeważnie przekazywanej przy użyciu protokołu http. Webserwisy zwykle nie tworzą samodzielnych aplikacji, a jedynie stanowią interfejs komunikacyjny pomiędzy klientami z sieci a systemami biznesowymi.

Wyjątkową siłą webservices jest wykorzystanie istniejących i szeroko stosowanych technologii tj. protokołu HTTP i języka XML. HTTP jest jednym z najbardziej rozpowszechnionych protokołów w sieci WEB, co umożliwia natychmiastowe wykorzystanie tej platformy do przesyłania komunikatów. XML dostarcza metajęzyk za pomocą, którego porozumiewają się klienci z usługami oraz poszczególne komponenty. Idąc dalej, przy budowie i wykorzystaniu webservices nie ma znaczenia w jakiej technologii jest napisana usługa, na jakim systemie operacyjnym pracuje. Pełną interoperatywność zapewnia protokół SOAP, który ma aspiracje stać się następcą technik typu CORBA, RMI czy DCOM.

Dzięki wykorzystywaniu nowego protokołu SOAP usługa opracowana w technologii Web Services ma możliwość współpracy (wymiany komunikatów) z dowolną inną usługą. Powinny zniknąć problemy występujące przy konwersjach realizowanych pomiędzy protokołami architektur DCOM i CORBA. Web Services mogą być pisane w dowolnych językach programowania, więc twórcy aplikacji będą mogli pisać usługi bez zmiany środowiska tworzenia aplikacji, w którym wcześniej pracowali. Protokół SOAP jest obecnie wspierany przez wszystkich podstawowych dostawców systemów oraz oprogramowania narzędziowego.

**SOAP – Simple Object Access Protocol (Prosty Protokół Dostępu do Obiektów).**

*Czym jest SOAP:*

* Jest protokołem, a zatem mechanizmem transportu informacji;
* Przenoszone informacje są uporządkowane (posiadają strukturę i typy);
* Dane zapisane są w języku rozszerzalnych znacznikÛw (XML);
* W tym protokole można przenosić wszelkiego rodzaju dane (w razie potrzeby są one kodowane do postaci wyrażalnej w XML);
* SOAP posiada bardzo szeroki zakres zastosowań: od prostych zastosowań komunikacyjnych po zdalne wywoływanie procedur (RPC).

*Czym nie jest SOAP:*

* Nie jest tym samym, co XML, chociaż jest na nim oparty;
* Nie ma związku z protokołami SMTP czy HTTP, chociaż komunikaty SOAP często są przesyłane z wykorzystaniem mechanizmów typowych dla tych protokołów (np. transmitowane przez TCP/IP na portach 25 lub 80, często spotyka się również enkapsulowanie wewnątrz komunikatów SMTP lub HTTP);
* Nie jest związany z żadnym konkretnym modelem programowania czy jakąś specyfiką implementacji - przeciwnie: jest uniwersalnym sposobem komunikacji w środowiskach rozproszonych oraz heterogenicznych

Model komunikacji

Komunikacja za pomocą SOAP jest zasadniczo komunikacją jednokierunkową: od nadawcy do odbiorcy. Łatwo sobie jednak wyobrazić inne scenariusze bazujące na tym podstawowym: np. schemat wywołanie ñ odpowiedź (request - response) lub dialog dwóch stron. Wybrana do przenoszenia komunikatów warstwa transportowa może ułatwiać budowanie takich bardziej rozbudowanych scenariuszy, np. przy łączeniu za pomocą TCP/IP po przesłaniu komunikatu nadawca z odbiorcą zamieniają się rolami, a odpowiedź jest transmitowana tym samym połączeniem, którym poszło wywołanie.

Struktura komunikatu SOAP

Komunikat SOAP jest dokumentem XML, który składa się z trzech elementów:

1. Koperty SOAP (SOAP envelope) – obowiązkowo: jest głównym, najbardziej zewnętrznym elementem dokumnetu XML, który reprezentuje komunikat. Nazwa tego elementu musi brzmieć „Envelope”, ten element musi być obecny w komunikacie SOAP, może zawierać deklaracje przestrzeni nazw oraz dodatkowe atrybuty.

2. Nagłówka SOAP (SOAP header) – niekoniecznie: służy do przekazywania informacji o tym kto i w jaki sposób powinien przetwarzać dane zawarte w ciele komunikatu. Zawartość nagłówka może nieść także informację dla ogniw pośredniczących w transporcie komunikatu (np. serwerów proxy), może być także przez nie zmieniana. Nazwa tego elementu musi brzmieć „Header”, jeśli ten element występuje w komunikacie SOAP, musi być pierwszym bezpośrednim potomkiem elementu „Envelope”.

3. Ciała SOAP (SOAP body) – obowiązkowo: w tym elemencie przenoszona jest właściwa informacja. Nazwa elementu musi brzmieć „Body”, element ten musi wystąpić w komunikacie i musi być bezpośrednim potomkiem elementu „Envelope”, jeśli w komunikacie występuje nagłówek, to ciało musi następować bezpośrednio po nim, w przeciwnym wypadku musi być pierwszym potomkiem elementu „Envelope”, ten element może zawierać serię wpisów, z których każdy jest jego bezpośrednim potomkiem.

Kodowanie danych w komunikatach SOAP

Kodowanie danych w SOAP wykorzystuje hierarchię typów danych, która jest generalizacją podobnych hierarchii spotykanych we współczesnych językach programowania czy bazach danych. Najbardziej podstawowy podział typów to podział na typy proste (skalarne) oraz na typy złożone, które są połączeniem pewnej liczby części, z których każda posiada swój typ. XML dopuszcza bardzo elastyczne kodowanie danych, SOAP w tej dziedzinie jest bardziej ograniczony (ma ściślejsze reguły kodowania).

Transmisja komunikatów SOAP protokołem HTTP

Bardzo często do transportu komunikatów SOAP używany jest protokół HTTP. Przyczyn jest kilka:

-jest to protokół bardzo elastyczny;

-jest zarazem pewny - posiada mechanizmy raportowania błędów;

-jest niezwykle rozpowszechniony (co wiąże się z bogactwem narzędzi);

-umożliwia przekazywanie komunikatów SOAP nawet przez zapory (firewall) analizujące zawartość transmitowanego strumienia danych (ponieważ zazwyczaj zapory takie przepuszczają transmisję HTTP).

Ze względu na charakter protokołu HTTP najbardziej naturalne jego wykorzystanie zachodzi podczas scenariusza transmisji wywołanie - odpowiedź (request - response). Podczas transmisji komunikatów SOAP wewnątrz wywołań/odpowiedzi HTTP należy używać typu „text/xml”. Pomimo iż SOAP można powiązać z różnymi typami wywołań HTTP, najczęściej jednak jest używane wywołanie POST.

Wywoływanie procedur (RPC)

Jednym z celów projektowych SOAP było umożliwienie transportowania za jego pomocą wywołań i odpowiedzi RPC. Choć nie jest to jedyne rozwiązanie, jednak naturalne staje się zastosowanie jako protokołu transportującego HTTP, gdzie wywołanie HTTP staje się wywołaniem zdalnej procedury, natomiast odpowiedź HTTP reprezentuje odpowiedź RPC. Reprezentacja RPC opisana w specyfikacji protokołu SOAP zakłada, że dane niezbędne do wywołania procedury (URI obiektu docelowego, nazwa metody, opcjonalna sygnatura metody i parametry metody) przekazywane są jako struktura (złożony typ danych) wewnątrz ciała komunikatu SOAP. Oczywiście nie jest to jedyna możliwość.

Wady i zalety

+niezwykła elastyczność protokołu, który pozwala przenosić właściwie dowolne informacje;

+możliwość definiowania struktury i semantyki przenoszonych informacji;

+możliwość łączenia z różnymi protokołami transportowymi (np. HTTP);

+możliwość realizacji różnych scenariuszy komunikacji;

+akceptowalność protokołu przez właściwie wszystkie systemy komputerowe i środowiska systemowe;

+niezawodność protokołu dzięki ścisłemu zdefiniowaniu sytuacji wystąpienia błędu oraz zachowania aplikacji w takich okolicznościach.

-duży narzut samego języka XML (rozmiar komunikatu jest znacząco większy niż sumaryczny rozmiar danych w nim zawartych);

-jest jeszcze dość młodym protokołem, podlega rozwojowi i modyfikacjom, chociaż jest już dość dobrze i ściśle zdefiniowany.

**WSDL (Web Services Description Language)** – prosty język pozwalający opisać usługę sieciową - jej sposób wywołania, parametry i formaty wyników, lecz bez żadnych informacji biznesowych czy parametrów dostępności. Dzięki WSDL mamy informacje jakie metody udostępnia Web Service. Usługę można odpytać, jakie funkcje udostępnia. WSDL umożliwia twórcom usługi Web opisanie w zrozumiałym formacie co potrafi usługa, gdzie się znajduje i jak ją wywołać. Język WSDL jest standardowym formatem opisu sieciowych usług XML. Język WSDL bazuje na standardzie XML. Jego zadaniem jest tworzenie opisu funkcji usług Web Services oraz sposobu ich wywoływania. Język WSDL jest przydatny przy automatyzacji komunikacji pomiędzy usługami Web Services, umożliwiając współdziałanie usług.

WSDL nie wprowadza nowego języka definicji typu. Rozpoznaje jedynie typy w opisywanym formacie wiadomości i wspiera specyfikację XML Schema. WSDL definiuje również wspólny binding mechanism. Jest on używany do dołączania właściwego protokołu lub formatu danych lub struktury do abstrakcyjnej wiadomości, operacji czy punktu końcowego.

Wady i zalety stosowania WSDL

+standardowy format opisu usług

+możliwość wykorzystania w wielu różnych technologiach i systemach komputerowych

+zrozumiały interfejs wykorzystywany przy definiowaniu usług

+łatwy w użyciu

+automatyzacja komunikacji między usługami

+szybsze udostępnianie i aktualizacja oprogramowania

-problemy z bezpieczeństwem (to dodatkowy protokół obsługi, który trzeba zabezpieczyć)

-problemy z autoryzacją

-niezbyt duża wydajność

-jest jeszcze dość młodym protokołem, podlega rozwojowi i zmianom

**UDDI (Universal Description Discovery and Integration)** – głównym zadaniem protokołu jest stworzenie niezależnego od platformy, otwartego rozwiązania do odnajdywania usług sieciowych. UDDI to globalny rejestr usług udostępniający klientom mechanizmy dynamicznego wyszukiwania innych usług Web. UDDI stanowi interfejs umożliwiający dynamiczne połączenie się z usługą udostępnianą przez innego usługodawcę. Projekt UDDI często określany jako DNS dla biznesu. UDDI jest bowiem usługą katalogową przechowującą opisy i lokalizacje Web Services zarejestrowanych przez dostawców. Wyszukiwanie usług jest podobne do działania wyszukiwarek internetowych.

UDDI posiadają dwa rodzaje klientów: usługodawców - publikujących swoje usługi, klientów - pragnących skorzystać z tych usług.

Specyfikacja UDDI opisuje również sposób wymiany danych pomiędzy serwerami, które mogą tworzyć sieć węzłów. Rejestry UDDI zawierają: informacje o webservices na bazie nazwy usługodawcy, jego adresu, kategorii biznesowej czy informacji technicznej itp., operacje dotyczące usługi Web tj. rejestracji, wyszukiwania i korzystania z usługi szczegóły udostępniane przez niskopoziomowe API

Wady i zalety

UDDI jest rozległą inicjatywą umożliwiającą systemom biznesowym odkrywanie siebie nawzajem oraz definiowanie sposobów interakcji poprzez Internet. Standard UDDI definiuje również sposoby replikacji pomiędzy repozytoriami. Obsługa UDDI powinna być integralną częścią systemów biznesowych, dzięki czemu będą one potrafiły łatwo, w sposób dynamiczny i przede wszystkim szybko, znaleźć inne systemy i współdziałać z nimi.

**PODSUMOWANIE SOAP:**

Protokół SOAP umożliwia wywoływanie i wykonywanie usług Web Services, publikowanych za pomocą WSDL i wyszukiwanych za pośrednictwem rejestru UDDI. Wykorzystywanie protokołu SOAP do wywoływania i wykonywania aplikacji poprzez Internet i korporacyjne zapory ogniowe umożliwia przedsiębiorstwom bezpieczną integrację własnych procesów biznesowych z procesami partnerów i dostawców. Korzystanie z usług przebiega w następujący sposób:

• dostarczyciele usług - rejestrują się u pośredników (UDDI).

• klient - pyta pośrednika, kto mu da daną usługę. (SOAP).

• pośrednik (broker) - lokalizuje mu tę usługę i podaje opis jej użycia (WSDL).

• klient podłącza się do dostarczyciela i korzysta z usługi (SOAP)

**Bezpieczeństwo w WS**

Problemy z bezpieczeństwem jest to największa przeszkoda w komercyjnym wykorzystaniu usług sieciowych. Usługi sieciowe wprowadzają bowiem trzy lub cztery dodatkowe warstwy pośrednie, które wymagają odpowiednich zabezpieczeń. Więcej warstw oznacza więcej możliwości dla hakerów.

Jako rozwiązanie tego problemu proponuje się stworzenie pojedynczego standardu. Najbardziej istotnym elementem bezpieczeństwa jest identyfikacja użytkownika, bowiem udostępniając usługi w Internecie, zezwalamy na korzystanie z nich w dowolny sposób. Aby ograniczyć dostęp do konkretnych usług, wystarczy zastosować kod (np. w postaci parametru wywołania usługi) umożliwiający dostęp tylko autoryzowanemu użytkownikowi. Bezpieczeństwo transmisji zapewnia SSL użyty w niższych warstwach protokołu. W przypadku powszechnie dostępnych usług, gdy nie ma możliwości przekazania kodu dostępu, konieczne jest zastosowanie innych metod uwierzytelniających.

**ESB (Enterprise Service Buses)** – Architektura ESB zakłada model centralnej szyny danych jako model połączeń między serwisami. Do szyny ESB podłączane są wszystkie serwisy związane z systemem i wszelkie wiadomości przesyłane od i do serwisów przekazywane są za jej pośrednictwem. Zaletą tego rozwiązania w stosunku do rozwiązań alternatywnych (np. bezpośrednie połączenia między serwisami) jest mała ilość połączeń między usługami.

ESB zakłada jeden, spójny model dla danych wewnątrz szyny. Nie wynika z tego jednak, że zewnętrzne serwisy integrujące się z nią też muszą mieć taki sam model danych. Wymaganie takie byłoby trudne do spełnienia i często uniemożliwiałoby integrację dwóch serwisów o różnych modelach danych. Architektura ESB rozwiązuje ten problem za pomocą adapterów, które umieszczane są na styku pomiędzy szyną a serwisami i służą do konwersji danych pomiędzy poszczególnymi modelami danych: Serwis1----Adapter-Szyna-Adapter----Serwis2

Zastosowanie wzorca ESB do integracji aplikacji daje m.in. następujące korzyści:

+Uniezależnienie od lokalizacji usług. Strona korzystająca z danej usługi nie musi znać jej konkretnej lokalizacji, w szczególności każdorazowo usługa może być dostarczana przez inny podmiot,

+Uniezależnienie od protokołu interakcji. Usługi potraﬁące komunikować się za pomocą jednego protokołu (np. HTTP/SOAP) mogą współpracować z serwisem używającym zupełnie innego protokołu (np. RMI),

+Uniezależnienie od zgodności interfejsów. Wywołujący nie jest zobligowany do dostosowania parametrów wywołania do wymagań serwisu, z którego korzysta, ale odpowiednia konwersja wywołania może zostać dokonana w ramach szyny ESB.

Zastosowanie ESB nie ma (a przynajmniej nie powinno mieć) wpływu na sposób budowy komunikujących się z jego pomocą serwisów. Z tego powodu w szczególności znajduje zastosowanie tam, gdzie nie ma możliwości ingerencji w kod źródłowy aplikacji (np. w przypadku systemów zastanych, które są kosztowne w modyﬁkacji lub systemów o nieznanym kodzie źródłowym), która jednak musi zostać przystosowana do współpracy z innym modułem. Wprowadzenie ESB jest decyzją na poziomie integracji poszczególnych części systemu rozproszonego (ang. deployment), a nie ich projektowania. Sam proces dostosowania szczegółów integracji usługobiorców i usługodawców, polegający na zmianie ﬁzycznej reprezentacji wywołania ”w locie” (ang. in-ﬂight), zwany jest mediacją wiadomości

Mediacja wiadomości

Szyna ESB nie jest tylko warstwą transportową dla wiadomości wysyłanych między serwisami. Oprócz zwykłego łączenia usług, posiada jeszcze następujące funkcje:

1. Mapowanie żądań usług z konkretnego protokołu i adresu na inny protokół/adres,

2. Transformacja danych na inny format,

3. Zarządzanie wieloma modelami transakcji i bezpieczeństwa i łączy różne modele integrowanych serwisów,

4. Agregacja żądań do serwisów,

5. Obsługa protokołów sieciowych między różnymi platformami z zachowaniem jakości usług (QoS).

Oprócz wyżej wymienionych własności, szyna ESB musi posiadać wiele sposobów integracji zewnętrznych aplikacji, które często posiadają własny, unikalny sposób dostępu, własne protokoły, modele bezpieczeństwa itd. Szyna powinna dostarczać w tym celu różne rodzaje adapterów, obsługujące wymagane protokoły komunikacji i formaty danych. Adaptery takie ukrywają detale związane z wykorzystaniem wymaganych rodzajów zasobów udostępniając interfejsy umożliwiające korzystanie z zasobów w jednolity sposób w ramach ESB. Stopień skomplikowania nie powinien być widoczny dla użytkownika serwisów, szczegóły implementacji powinny być zasłonięte za dobrze zdeﬁniowanym interfejsem adapteru. W ten sposób, przy wykorzystaniu szyny ESB, można skupić się na integracji serwisów na poziomie logicznym, pomijając szczegóły niskopoziomowego połączenia.

Wzorce mediacyjne Wyróżnia się kilka wzorców, według których dokonywana jest transformacja komunikatów przesyłanych poprzez ESB:

* Wzbogacenie (ang. enrich) - wzbogacenie zawartości wiadomości poprzez dołączenie dodatkowych parametrów pochodzących z mediacji lub np. z bazy danych,
* Przełączenie protokołu (ang. protocol switch) - zmiana protokołu komunikacji pomiędzy usługodawcą a usługobiorcą, może mieć miejsce w dowolnym punkcie na drodze pomiędzy współpracującymi stronami,
* Transformacja - zmiana formatu wiadomości z postaci rozumianej przez usługobiorcę do formatu akceptowanego przez usługodawcę; może mieć postać enkapsulacji, dekapsulacji, szyfrowania itp.
* Przekierowanie (ang. routing) - Wybór usługodawcy na podstawie określonych kryteriów. Mogą być one rozpatrywane na podstawie zawartości komunikatu, jego kontekstu, ale także na podstawie cech dostępnych usługodawców,
* Dystrybucja (ang. distribute) - rozsyła wiadomość do wielu zainteresowanych stron, rejestrujących się zwykle na zasadzie subskrypcji,
* Monitoring - umożliwia szeroko pojętą obserwację przepływających wiadomości w celach logowania, rozwiązywania problemów, pomiaru wykorzystania zasobów, w celach billingowych i innych,
* Korelacja - łączy komunikaty lub strumienie zdarzeń; zawiera reguły identyﬁkacji i rozpoznawania wzorców w komunikatach.

Powyższe wzorce można zestawiać w dowolne konﬁguracje tworząc złożone łańcuchy dokonujące pożądanej mediacji.

# Narzędzia typu CASE

Narzędzia CASE (czyli Computer Aided Software Engineering lub Computer Aided System Engineering) to systemy komputerowe, przeznaczone do wspomagania rutynowych czynności procesu tworzenia oprogramowania. Dzięki nim projekty tworzy się dokładniej, a praca nad diagramami, sprawdzanie ich poprawności oraz śledzenie wykonanych testów jest prostsze i szybsze.

**Podział narzędzi CASE** Systemy CASE można podzielić według faz cyklu życia systemu na: Upper-CASE i Lower-CASE, a także według zakresu zastosowań na pakiety narzędziowe oraz pakiety zintegrowane.

**Upper-CASE**

* wspomaganie wczesnych faz prac nad oprogramowaniem,
* w szczególności fazy analizy (potrzeby analityków i projektantów),
* narzędzia te nie są związane z konkretnym środowiskiem implementacyjnym.

**Lower-CASE**

* wspomaganie faz projektowania i implementacji (potrzeby programistów)
* narzędzia te są zazwyczaj związane z konkretnym środowiskiem implementacji (np. Java)
* notacje graficzne są bardziej naturalnym sposobem prezentacji dużych programów niż tradycyjny zapis tekstowy
* dzięki nim nie jest konieczne zapisywanie w całości kodu programu ręcznie

**Istnieje także inny podział:**

* narzędzia - wspierające tylko jedną funkcję,
* pakiety zintegrowane - podchodzące do projektowania kompleksowo.

**Funkcje**

Od systemów CASE wymagamy bardzo wiele. Wspomaganie w każdej fazie cyklu projektu jest inne i wymaga różnych funkcjonalności. Można jednak wyróżnić kilka standardowych modułów, których istnienie świadczy o zaawansowaniu danego systemu i spełnieniu wymagań użytkownika.

* **Business process engineering tools (BPMN) –** modelowanie procesów biznesowych.
* **Planowanie, Harmonogramowanie projektu**
* **Definiowanie wymagań na system**
* **Szacowanie kosztów**
* **Narzędzia oceny ryzyka projektu**
* **Słowniki danych (repozytoria)** – bazy wszelkich danych o tworzonym systemie wraz z narzędziami edytującymi, zarządzającymi i wyszukującymi te dane.
* **Edytor Notacji Graficznych** – program graficzny, umożliwiający tworzenie i edycję diagramów (np. UML) dla faz określania wymagań systemu, analizy i projektowania. Powinien też umożliwiać powiązania między symbolami w modelu a innymi, zdekomponowanymi modelami, oraz wydruk tych diagramów.
* **Moduł Kontroli Poprawności** – narzędzie do wykrywania i poprawiania błędów w diagramach i repozytoriach. Bardzo często działa w czasie rzeczywistym, co znacząco wpływa na komfort pracy.
* **Moduł Kontroli Jakości** – narzędzie do oceny pewnych ustalonych miar jakości projektu – np. stopnia złożoności lub powiązań składowych modelu (metryki kodu).
* **Generator testów jednostkowych**
* **Generator Raportów** – narzędzie tworzące dowolny raport na podstawie danych z repozytorium.
* **Generator Kodu** – narzędzie transformujące projekt na szkielet kodu w wybranym języku programowania. Usprawnia pracę programistów, pozwala na zautomatyzowanie pewnych fragmentów kodu, a także na uzupełnienie kodu o dodatkowe informacje ze słownika danych.
* **Generator Bazy danych** – narzędzie transformujące projekt na szkielet kodu w języku SQL lub automatycznie tworzące strukturę bazy na wybranym serwerze bazodanowym.
* **Generator Dokumentacji Technicznej** – generator ustandaryzowanych dokumentów, zawierających specyfikację, opisy faz projektu, diagramy oraz wybrane raporty.
* **Moduł Projektowania Interfejsu Użytkownika** – narzędzie do projektowania menu, okien dialogowych oraz innych elementów interfejsu użytkownika.
* **Moduł refaktoryzacji kodu**
* **Moduł Inżynierii Odwrotnej** – narzędzie pozwalające odtworzyć słownika danych oraz diagramów, na podstawie kodu źródłowego lub struktury bazy danych.
* **Moduł Importu/Eksportu Danych** – narzędzie służące do wymiany danych z innymi CASE'ami czy też innymi programami.
* **Moduł Zarządzania Pracą Grupową** – narzędzie umożliwiające współpracę grupy osób podczas pracy nad projektem (np. SVN)

Gdy chcemy zacząć pracę nad projektem, możemy to zrobić tworząc dowolny model od podstaw, ale możemy też za pomocą inżynierii odwrotnej stworzyć model, opierając się o istniejące struktury bazy danych, kod źródłowy z klasami, czy też struktury w XMLu. Gdy już będziemy mieli modele i będziemy równolegle pracować nad kilkoma etapami, może się okazać, że potrzebujemy wprowadzić zmiany w kilku modelach. Dobry system CASE potrafi powiązać zmiany w tych modelach z koniecznymi zmianami w innych modelach oraz dokonać automatycznie odpowiednich korekt.

**Co brać pod uwagę przy wyborze narzędzi:**

* Ergonomia pracy. Diagramy graficzne są jednym z podstawowych narzędzi pracy w fazach analizy i projektowania. Powinny one pozwalać analitykom i projektantom skupić się na pracy, a nie na “zmaganiach” z edytorem.
* Możliwość kontrolowania ilości informacji prezentowanej w sposób graficzny.
* Jakość i możliwość formatowania wydruków.
* Wykrywanie na bieżąco konstrukcji niepoprawnych.
* Zapewnienie spójności informacji umieszczonych na różnych diagramach.

**Popularne narzędzia**

* **Eclipse** – darmowe, otwarte środowisko programistyczne dla Javy, które za pomocą platformy modelowania Eclipse (Eclipse Modeling Framework) może posłużyć do budowania oprogramowania, wykorzystując także UML. EMF posiada także generator kodu.
* **NetBeans** – otwarty projekt zawierający wiele narzędzi wspomagających tworzenie oprogramowania. Dodatkowo "Enterprise Pack" umożliwia modelowanie UML oraz użycie schematów XML.
* **StarUML** – Zapewnia forward i reverse engineering kodu w Javie, C# i C++.
* **Borland Together** – rodzina programów integrujących środowisko IDE Javy z narzędziami do UMLa. Posiada m.in. funkcje modelowania danych, szablony kodu, generator dokumentacji, czy też moduł weryfikacji kodu.
* **Enterprise Architect** generowanie kodu: ActionScript, C, C#, C++, Delphi, Java, PHP,  Python, Visual Basic, Visual Basic .NET, DDL, EJB, XML Schema, Ada, VHDL, Verilog, WSDL, BPEL, Corba IDL ActionScript, C, C#, C++, Delphi, Java, PHP, Python, Visual Basic, Visual Basic .NET, DDL, XML Schema, WSDL (dla wielu istnieje także wsparcie reverse engineeringu) Integracja z: Eclipse, Visual Studio, TcSE , Modelowanie: UML 2.4.1, SysML, BPMN, SoaML, SOMF, WSDL, XSD, ArchiMate. Frameworks: UPDM, Zachman, TOGAF. Forward and Reverse Engineering for code and Database. Model Driven Integrated Developement (Edit/Build/Debug) for Java, .Net, PHP & GNU compilers. Simulates Activity, State Machine, Sequence and BPMN diagrams.
* **Visual Paradigm** - Generowanie kodu: Java, C#, C++, PHP, Ada, Action Script Reverse engineering: Java, C# (binary), C++, PHP Integracja z: Eclipse, NetBeans and IntelliJ Full UML 2.1, SysML, ERD, BPMN, data modeling,business modeling and reverse engineering of source code and dtabases

# Podejścia do zarządzania zespołami realizującymi projekty informatyczne

**NA STARYCH NOTATKACH**

# Podpis elektroniczny

Podpis **elektroniczny** (electronic signature) to operacja podpisywania konkretnych danych (dokumentu) przez osobę fizyczną. Podpisywanie to **dołączanie** **danych** w postaci elektronicznej, do konkretnego! **dokumentu**. Razem, taki zestaw może posłużyć do identyfikacji osoby składającej podpis elektroniczny.

Dokument opatrzony podpisem elektronicznym może być – przy spełnieniu dodatkowych przesłanek – równoważny pod względem skutków prawnych dokumentowi opatrzonemu podpisem własnoręcznym. Podpis elektroniczny będzie mógł być uznany za równoważny podpisowi własnoręcznemu jeśli spełnia warunki umożliwiające uznanie go za podpis elektroniczny bezpieczny. Zgodnie z ustawą **bezpieczny podpis elektroniczny** to podpis elektroniczny, który:

* jest **przyporządkowany wyłącznie** do osoby składającej ten podpis,
* jest sporządzany za pomocą podlegających **wyłącznej kontroli osoby** **składającej** **podpis** elektroniczny bezpiecznych urządzeń służących do składania podpisu elektronicznego i danych służących do składania podpisu elektronicznego,
* jest powiązany z danymi, do których został dołączony, w taki sposób, że jakakolwiek późniejsza **zmiana** tych danych **jest** **rozpoznawalna**.

Skutki prawne związane ze złożeniem oświadczenia woli opatrzonego **bezpiecznym** podpisem elektronicznym weryfikowanym przy pomocy ważnego **kwalifikowanego certyfikatu** określa **kodeks** **cywilny**. Zgodnie z tym przepisem **oświadczenie woli** podpisane takim podpisem ma takie same skutki prawne, jak oświadczenie woli podpisane podpisem własnoręcznym.

**Certyfikat kwalifikowany** to taki, który został wystawiony jego właścicielowi z zastosowaniem odpowiednich procedur weryfikacji tożsamości a ponadto klucz prywatny (służący do składania podpisów) jest przechowywany w sposób bezpieczny (np. na karcie elektronicznej).

Zgodnie z prawem polskim tylko podpis kwalifikowany ma automatycznie takie samo znaczenie jak podpis odręczny. Inne rodzaje podpisu mogą być wiążące prawnie jedynie na podstawie umów cywilnych pomiędzy kontrahentami je stosującymi. Podobne prawodawstwo obowiązuje w większości krajów Unii Europejskiej, chociaż występuje między nimi wiele drobnych różnic, na przykład jeśli chodzi o obowiązek korzystania z bezpiecznego urządzenia (*SSCD*).

Bezpieczny podpis elektroniczny nie zastępuje całkowicie podpisu własnoręcznego. W szczególności nie może go zastąpić, gdy ustawa wymaga podpisu własnoręcznego, jak np. przy **testamencie własnoręcznym** lub **podpisie notarialnie uwierzytelnionym**.

Bezpieczny podpis elektroniczny weryfikowany za pomocą kwalifikowanego certyfikatu spełnia w obrocie prawnym i gospodarczym m.in. następujące funkcje: identyfikacyjną, dowodową, kontraktową.

W prawie wspólnotowym skutki prawne podpisu elektronicznego są przedmiotem regulacji *Dyrektywy o Wspólnotowej Infrastrukturze Podpisów Elektronicznych*. Prawo unijne wyróżnia następujące rodzaje podpisu elektronicznego:

* **Podpis elektroniczny** – czyli deklaracja tożsamości autora, złożona w formie elektronicznej pod dokumentem. Dane w postaci elektronicznej, które służą do identyfikacji osoby go składającej.
* **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).

Podsumowując, pojęcie podpis elektroniczny (electronic signature) jest wprowadzone przez unijną Dyrektywę i jednoznacznie określa, że jest to operacja podpisywania konkretnych danych (dokumentu) przez osobę fizyczną.

Od strony technicznej podpis elektroniczny jest realizowany za pomocą mechanizmów **podpisu cyfrowego**.

Podpis cyfrowy służy zapewnieniu między innymi następujących funkcji:

1. **autentyczności**, czyli pewności co do autorstwa dokumentu,
2. **niezaprzeczalności nadania informacji**, nadawca wiadomości nie może wyprzeć się wysłania wiadomości, gdyż podpis cyfrowy stanowi dowód jej wysłania (istnieją także inne rodzaje niezaprzeczalności),
3. **integralności**, czyli pewności, że wiadomość nie została zmodyfikowana po złożeniu podpisu przez autora.

**Podpis cyfrowy** (digital signature) to 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. Skrót sporządzony jest za pomocą funkcji haszującej, np. MD5 (zalecana do 1999 roku), SHA1 (zalecana do 2010 roku), SHA2. W roku 2012 będzie wybrany algorytm SHA3 (spośród tych zgłoszonych do publicznego konkursu organizowanego przez amerykański instytut NIST). Potwierdzenie autentyczności wiadomości jest możliwe po odszyfrowaniu skrótu kluczem publicznym nadawcy i porównaniu go z wytworzonym przez siebie skrótem odebranego dokumentu.

Innymi słowy: Strona uwierzytelniająca wylicza skrót (ang. **hash**) podpisywanej wiadomości. Następnie **szyfruje** ten skrót swoim kluczem prywatnym i jako podpis cyfrowy **dołącza** do oryginalnej wiadomości. Dowolna osoba posiadająca klucz publiczny może sprawdzić autentyczność podpisu, poprzez odszyfrowanie skrótu za pomocą klucza publicznego nadawcy i porównanie go z własnoręcznie wyliczonym na podstawie wiadomości. Działają tak np. certyfikaty SSL na zabezpieczonych stronach WWW (https).

Szyfrowanie/deszyfrowanie odbywa się przy użyciu algorytmów kryptografii (*asymetrycznych, bo jest klucz prywatny i publiczny*) takich, jak RSA czy ELGamal (2 najpopularniejsze). Podpis tej samej wiadomości w RSA jest zawsze identyczny. W ElGamalu i DSA każdy kolejny podpis tej samej wiadomości zwykle jest inny – co ma znaczenie w niektórych zastosowaniach.

*Co do kluczy prywatnych i publicznych*: we wszystkich kryptosystemach (czy to RSA czy Elgamal) uzyskanie klucza **prywatnego** **na podstawie** **publicznego** musi być **obliczeniowo trudne**. Klucz publiczny może być obliczony na podstawie prywatnego, co zresztą ma miejsce podczas generacji kluczy w ElGamal.

Najpopularniejsze standardy pozwalające na złożenie podpisu elektronicznego to **X.509 oraz PGP**.

**PGP** jest systemem zdecentralizowanym, w którym poziom autentyczności danego klucza jest determinowany przez sumę podpisów, złożonych przez różne osoby znające posiadacza klucza (model Sieć zaufania). System ten jest powszechnie wykorzystywany w Internecie oraz w środowiskach korporacyjnych (np. niektóre systemy EDI).

**Sieć zaufania** (ang. web of trust) to zdecentralizowana metoda uwierzytelniania osób, w której nie ma hierarchicznej struktury organizacji uwierzytelniających, a zaufanie do poszczególnych certyfikatów jest sumą podpisów złożonych przez innych uczestników sieci.

Każdy uczestnik sieci podpisuje klucze osób, które osobiście zweryfikował. Podpis stanowi poświadczenie, że osoba podpisująca jest przekonana o autentyczności klucza oraz tym, że faktycznie należy on do osoby, która deklaruje że jest jego właścicielem.

Jeśli ktoś z uczestników sieci chce sprawdzić czy pewien klucz rzeczywiście należy do danej osoby, próbuje ułożyć łańcuch zaufania w postaci: "Ja zweryfikowałem że X to X i podpisałem jego klucz KX, X zweryfikował że Y to Y i podpisał jego klucz KY, itd., Z zweryfikował że sprawdzany klucz rzeczywiście należy do tej osoby."

System **X.509** jest systemem scentralizowanym, w którym autentyczność klucza jest gwarantowana przez hierarchię centrów certyfikacji formalnie poświadczających związek klucza z tożsamością jego właściciela. Ze względu na jednoznaczną odpowiedzialność, łatwiejszą do osadzenia w prawie, X.509 jest obecnie dominującym systemem, na którym opiera się aktualnie obowiązujące prawodawstwo o podpisie elektronicznym.

"Podpis" składany przez **centrum/urząd certyfikacji** pod certyfikatem osoby fizycznej nazywa się "poświadczeniem elektronicznym" *(chyba dlatego, że ‘podpis’ to w polskim prawie słowo zarezerwowane dla osób fizycznych)*, zaś sam certyfikat centrum - "zaświadczeniem certyfikacyjnym".

W Polsce instytucją certyfikującą pozostałych wystawców certyfikatów (centra certyfikacji/urzędy certyfikacji) jest Narodowe Centrum Certyfikacji. Podmioty wydające certyfikaty kwalifikowane to: Krajowa Izba Rozliczeniowa SA, Polska Wytwórnia Papierów Wartościowych SA (Sigillum) i Unizeto Technologies SA (Certum).

*Ważniejsze pojęcia:*

CA - Certification Authority - **urząd certyfikacji** - wystawia certyfikaty, listy CRL, certyfikuje inne CA.

RA - Registration Authority - **urząd rejestracji** - zbiera wnioski o wydanie certyfikatu, weryfikuje tożsamość subskrybentów.

**Subskrybent** - właściciel certyfikatu.

# Specyficzne własności struktur baz danych w systemach informacyjnych

Rodzaje struktur baz danych: relacyjne, obiektowe, hurtownie danych, temporalne, NoSQL

**Bazy relacyjne**

W bazachrelacyjnych wiele tablic danych może współpracować ze sobą (są między sobą powiązane). Bazy relacyjne posiadają wewnętrzne języki programowania, wykorzystujące zwykle SQL do operowania na danych, za pomocą których tworzone są zaawansowane funkcje obsługi danych. Relacyjne bazy danych (jak również przeznaczony dla nich standard SQL) oparte są na kilku prostych zasadach:

1. Wszystkie wartości danych oparte są na prostych typach danych.
2. Wszystkie dane w bazie relacyjnej przedstawiane są w formie dwuwymiarowych tabel (w matematycznym żargonie noszących nazwę "relacji"). Każda tabela zawiera zero lub więcej wierszy (w tymże żargonie – "krotki") i jedną lub więcej kolumn ("atrybutów"). Na każdy wiersz składają się jednakowo ułożone kolumny wypełnione wartościami, które z kolei w każdym wierszu mogą być inne.
3. Po wprowadzeniu danych do bazy, możliwe jest porównywanie wartości z różnych kolumn, zazwyczaj również z różnych tabel, i scalanie wierszy, gdy pochodzące z nich wartości są zgodne. Umożliwia to wiązanie danych i wykonywanie stosunkowo złożonych operacji w granicach całej bazy danych.
4. Wszystkie operacje wykonywane są w oparciu o algebręrelacji, bez względu na położenie wiersza tabeli. Nie można więc zapytać o wiersze, gdzie (x=3) bez wiersza pierwszego, trzeciego i piątego. Wiersze w relacyjnej bazie danych przechowywane są w porządku zupełnie dowolnym – nie musi on odzwierciedlać ani kolejności ich wprowadzania, ani kolejności ich przechowywania.
5. Z braku możliwości identyfikacji wiersza przez jego pozycję pojawia się potrzeba obecności jednej lub więcej kolumn niepowtarzalnych w granicach całej tabeli, pozwalających odnaleźć konkretny wiersz. Kolumny te określa się jako "klucz podstawowy" (ang. *primary key*) tabeli.

**Obiektowa baza danych** to zbiór obiektów, których zachowanie się, stan oraz związki są określone zgodnie z obiektowym modelem danych. Obiektowy system zarządzania bazą danych jest systemem wspomagającym definiowanie, zarządzanie, utrzymywanie, zabezpieczanie i udostępnianie obiektowej bazy danych.

Obiektowe systemy zarządzania bazą danych zapewniają tradycyjną funkcjonalność baz danych, lecz bazują na modelu obiektowym. Ich atutem jest udostępnianie danych w postaci obiektowej, czyli takiej samej w jakiej dane są przechowywane w programach napisanych w obiektowych językach programowania. Znika konieczność mapowania między modelem obiektowym a modelem relacyjnym jak to ma miejsce w przypadku użycia relacyjnej bazy danych.

**Temporalna baza danych** - baza danych posiadająca informację o czasie wprowadzenia lub czasie ważności zawartych w niej danych. Temporalne bazy danych są często administrowane automatycznie, poprzez usuwanie nieaktualnych danych lub ich archiwizowanie.

Użyteczne tam gdzie potrzebujemy przechowywać historię bądź ją analizować (hurtownie danych)

**Hurtownia danych** (ang. data warehouse) – rodzaj bazy danych, która jest zorganizowana i zoptymalizowana pod kątem pewnego wycinka rzeczywistości.

Hurtownia danych jest wyższym szczeblem abstrakcji niż zwykła relacyjna baza danych (choć do jej tworzenia używane są także podobne technologie). W skład hurtowni wchodzą zbiory danych zorientowanych tematycznie (np. hurtownia danych klientów). Dane te często pochodzą z wielu źródeł, są one zintegrowane i przeznaczone wyłącznie do odczytu.

W praktyce hurtownie są bazami danych integrującymi dane z wszystkich pozostałych systemów bazodanowych w przedsiębiorstwie. Ta integracja polega na cyklicznym zasilaniu hurtowni danymi systemów produkcyjnych (może być tych baz lub systemów dużo i mogą być rozproszone).

Architektura bazy hurtowni jest zorientowana na optymalizację szybkości wyszukiwania i jak najefektywniejszą analizę zawartości. Stąd bywa, że hurtownie danych nie są realizowane za pomocą relacyjnych baz danych, gdyż takie bazy ustępują szybkością innym rozwiązaniom.

**NoSQL** to koncepcja budowy baz danych. Specyficzne cechy:

- nie używa SQLa do wykonywania zapytań

- coś na zasadzie działania rozproszonego cache

- świetna skalowalność, ale za to ogromne wymagania sprzętowe zwłaszcza co do ilości pamięci operacyjnej

- nie zapewniają ACID (pewne implementacje zapewniają, ale to wyjątki)

- zoptymalizowane pod kątem danego zastosowania, najczęściej do trzymania par typu klucz-wartość

- nie wymagają JOINów

- wyjebista redudancja danych

- używane przez wszystkich wielki

- w odróżnieniu od relacyjnych dają radę przy dużej ilości insertów czy updatów

# Standardy zapewnienia jakości oprogramowania

**NA STARYCH NOTATKACH**

# Strategie realizacji systemu informacyjnego

**Iteracyjny MODEL KASKADOWY** (ang. waterfall model) – jeden z kilku rodzajów procesów tworzenia oprogramowania zdefiniowany w inżynierii oprogramowania. Jego nazwa wprowadzona została przez Winstona W. Royce w roku 1970, w artykule "Managing the Development of Large Software Systems" (zarządzanie tworzeniem dużych systemów informatycznych). Polega on na wykonywaniu podstawowych czynności jako odrębnych faz projektowych, w porządku jeden po drugim. Każda czynność to kolejny schodek (kaskada):

1. Planowanie systemu (w tym specyfikacja wymagań)
2. Analiza systemu (w tym Analiza wymagań i studium wykonalności)
3. Projekt systemu (poszczególnych struktur itp.)
4. Implementacja (wytworzenie kodu)
5. Testowanie (poszczególnych elementów systemu oraz elementów połączonych w całość)
6. Wdrożenie i pielęgnacja powstałego systemu.

Jeśli któraś z faz zwróci niesatysfakcjonujący produkt cofamy się wykonując kolejne iteracje aż do momentu kiedy otrzymamy satysfakcjonujący produkt na końcu schodków.

Model kaskadowy jest rzadko używany z następujących powodów:

* Nie można przejść do następnej fazy przed zakończeniem poprzedniej
* Model ten posiada bardzo nieelastyczny podział na kolejne fazy
* Iteracje są bardzo kosztowne - powtarzamy wiele czynności
* Tego typu modelu należy używać wyłącznie w przypadku gdy wymagania są zrozumiałe i przejrzyste, ponieważ każda iteracja jest czasochłonna i wymaga dużych wydatków na ulepszanie.

**PROTOTYPOWANIE** – to rozwój oprogramowania poprzez tworzenie kolejnych wersji prototypów oprogramowania przy użyciu prototypowania stało się praktyką na tyle powszechną, że uważa się ją obecnie za standard.

Zastosowanie metody prototypowania umożliwia:

* usprawnienie procesu analizy problemu
* doprecyzowanie wymagań na podstawie doświadczeń z prototypem
* wczesne wykrycie nieporozumień na linii wykonawca-zleceniodawca
* wykrycie brakujących funkcjonalności
* zdiagnozowanie potencjalnych trudności
* szkolenie użytkowników w trakcie tworzenia systemu
* testowanie systemu w trakcie jego rozwoju

Zaczyna się od zbudowania prostego systemu spełniającego część funkcjonalności. Jest on następnie modyfikowany w miarę poznawania wymagań. Ostatecznie prototyp staje się systemem, którego oczekiwano.

**Zalety**

* Przyśpieszone dostarczenie użytkownikowi początkowej wersji systemu
* Włączenie użytkownika w budowę systemu
* wykrycie nieporozumień pomiędzy klientem i projektantami
* wykrycie brakujących funkcji w projekcie (w systemie)
* wykrycie trudnych funkcji
* wykrycie braków w specyfikacji wymagań

**Wady**

* Kłopoty z zarządzaniem – prototypowanie ewolucyjne cechuje się gwałtownym rozwojem, przez co nie nadąża się z tworzeniem dokumentacji;
* Kłopoty z pielęgnacją – ustawiczne zmiany powodują uszkodzenia struktury prototypowanego systemu;
* Kłopoty ze stworzeniem umowy – tradycyjny model umowy klienta z wytwórcą oprogramowania oparty jest na specyfikacji systemu uzgodnionej przed rozpoczęciem prac. W przypadku prototypowania ewolucyjnego zakres prac nie jest znany przed ich rozpoczęciem. Powoduje to problemy z ustaleniem wynagrodzenia.

**Cechy**

* Procesy specyfikowania, projektowania i implementowania przeplatają się
* System jest budowany w postaci ciągu przyrostów. Użytkownicy i inni udziałowcy systemu są włączeni w projektowanie i ocenę każdego przyrostu
* Stosuje się metody błyskawicznego tworzenia systemów. Mogą to być narzędzia CASE i języki czwartej generacji
* Systemowe interfejsy użytkownika są zwykle budowane za pomocą interakcyjnego systemu wytwórczego, który umożliwia szybkie tworzenie projektu interfejsu przez rysowanie i rozmieszczanie ikon

**MODEL SPIRALNY** Proces tworzenia ma postać spirali, której każda pętla reprezentuje jedną fazę procesu. Najbardziej wewnętrzna pętla przedstawia początkowe etapy projektowania, np. studium wykonalności, kolejna definicji wymagań systemowych, itd.

Każda pętla spirali podzielona jest na cztery sektory:

* **Ustalanie celów** - Definiowanie konkretnych celów wymaganych w tej fazie przedsięwzięcia. Identyfikacja ograniczeń i zagrożeń. Ustalanie planów realizacji.
* **Rozpoznanie i redukcja zagrożeń** - Przeprowadzenie szczegółowej analizy rozpoznanych zagrożeń, ich źródeł i sposobów zapobiegania. Podejmuje się odpowiednie kroki zapobiegawcze.
* **Tworzenie i zatwierdzanie** - Tworzenia oprogramowania w oparciu o najbardziej odpowiedni model, wybrany na podstawie oceny zagrożeń.
* **Ocena i planowanie** - Recenzja postępu prac i planowanie kolejnej fazy przedsięwzięcia bądź zakończenie projektu.

**Cechy** Widoczną cechą modelu spiralnego jest szczegółowe potraktowanie zagrożeń realizacji projektu. Dobrze rozpoznane zagrożenia i przedsięwzięte kroki im zapobiegania lub redukcji skutkują m.in. wysoką niezawodnością (dependability) powstającego oprogramowania, bądź pewnością, że projekt ma szanse dalszej realizacji.

W modelu spiralnym nie ma takich faz jak specyfikowanie albo projektowanie. Jeden cykl spirali może przebiegać w oparciu o model kaskadowy procesu tworzenia oprogramowania, w innym można użyć prototypowania lub przekształceń formalnych, w zależności od aktualnego etapu przedsięwzięcia / realizowanej części systemu (np. inny dla tworzenia interfejsu użytkownika, inny dla krytycznych funkcji bezpieczeństwa)

Każdy cykl wymaga formalnej decyzji o kontynuacji projektu.

**Zalety**

* Można wykorzystać gotowe projekty
* Faza oceny w każdym cyklu pozwala uniknąć błędów lub wcześniej je wykryć
* Cały czas istnieje możliwość rozwijania projektu
* Częste kontrole jakości w kolejnych cyklach spirali
* Nastawienie na wykrywanie błędów i działania kontrolne, a nie na zapobieganie
* Orientacja na zarządzanie, czas i budżet.

**Wady**

Model nie do końca dopracowany. Każdy projekt jest inny i powstaje w innych warunkach. Ciężko określić jakie warunki brać pod uwagę.

* Tworzenia w oparciu o model spiralny wymaga doświadczenia w prowadzeniu tego typu projektów oraz często wiedzy ekonomicznej w zarządzaniu
* Wysoki koszt usuwania błędów wykrytych w finalnych etapach projektu

**Zastosowanie**

Model spiralny z racji ogólnego charakteru stosuje się przy dużych projektach.

**MODEL PRZYROSTOWY** jedna z technik pisania oprogramowania, stosowany w przypadkach, w których dopuszczalna jest okrojona funkcjonalność systemu.

Fazy

* określenie całości wymagań (w ramach naszych możliwości, na tyle na ile uda nam się ją sprecyzować), wykonanie wstępnego, ogólnego projektu całości systemu
* wybór pewnego podzbioru funkcji systemu
* szczegółowy projekt (wg modelu kaskadowego) oraz implementacja części systemu realizującej wybrane funkcje
* testowanie zrealizowanego fragmentu i dostarczenie go klientowi
* powtarzanie kolejnych etapów, aż do zakończenia implementacji całego systemu

Zalety

* częste kontakty z klientem (skrócenie przerw w porównaniu z modelem kaskadowym)
* brak konieczności zdefiniowania z góry całości wymagań (na wstępie definiujemy co nam się uda mając nadzieję, że uda nam się wyspecyfikować całość wymagań na etapie testowania zrealizowanych fragmentów)
* wczesne wykorzystanie przez klienta fragmentów systemu (funkcjonalności)
* możliwość elastycznego reagowania na opóźnienia realizacji fragmentu – przyspieszenie prac nad inną/innymi częściami (sumarycznie – bez opóźnienia całości przedsięwzięcia projektowego)

Wady

* dodatkowy koszt związany z niezależną realizacją fragmentów systemu
* potencjalne trudności z wycinaniem podzbioru funkcji w pełni niezależnych
* dlatego: konieczność implementacji szkieletów (interfejs zgodny z docelowym systemem) – dodatkowy nakład pracy (koszt), ryzyko niewykrycia błędów w fazie testowania

**PROGRAMOWANIE WYNALAZCZE ??**

# Systemy biometryczne

Biometryka jest nauką zajmującą się identyfikowaniem i weryfikowaniem tożsamości osoby na podstawie jej cech fizjologicznych lub behawioralnych zwanych biometrykami.

Ważnym elementem biometryki są jej cechy. Pozwalają one scharakteryzować każde żywe stworzenie oraz rozróżnić je od innych przedstawicieli tego samego gatunku. Można podzielić je na dwie grupy: behawioralne oraz fizjologiczne. Elementy behawioralne to te, które dotyczą zachowania w czasie. Są to między innymi: sposób wpisywania na klawiaturze, głos, sposób chodzenia. Druga grupa to cechy fizjologiczne czyli takie cechy, które posiadają informacje o cechach fizycznych i są do zmierzenia w danej chwili. Są to np. odcisk palca, kształt dłoni czy też wygląd tęczówki.

1. **Identyfikacja i weryfikacja.**

Podstawowe wykorzystanie biometryki w systemach informacyjnych to weryfikacja oraz identyfikacja użytkowników. Te dwa pojęcia często są mylone ze sobą dlatego warto je przytoczyć i pokazać jaka jest różnica. Identyfikacja proces ustalający kim jest dana osoba. Mechanizm pobiera próbkę od danej osoby i wyszukuje w dużej bazie danych taką próbkę, która odpowiada pobranym.

Weryfikacja ma na celu natomiast sprawdza czy podana osoba jest tą, za którą się podaje. Tutaj pobierana jest próbka by sprawdzić czy dane wcześniej zebrane od tej osoby są takie same. Różnice między tymi procesami występują w trzech blokach procesu. Pierwszy blok to pobieranie – identyfikacja pobiera jedynie próbki, weryfikacja natomiast oprócz próbki potrzebuje również identyfikatora. Podczas działania procesu identyfikacja sprawdza całą bazę wyszukując rekordu zgodnego z elementem wzorca, natomiast w weryfikacji próbka porównywana jest tylko ze wzorcem posiadającym podany wcześniej identyfikator. Ostatnia róznica zachodzi w wyniku procesów. Identyfikacja daje wynik w postaci rozpoznania lub odrzucenia, weryfikacja akceptuje lub rozróżnia próbkę.

Opis metod indentyfikacji i weryfikacji opisane w pkt 26 **TECHNOLOGIE BIOMETRYCZNE .**

W biometryce możemy wyróżnić dwa podstawowe typy błędów:

**FAR** (ang. False Acceptance Rate) - błąd fałszywej akceptacji występuje, gdy osoba spoza bazy danych (a właściwie jej dane biometryczne) systemu zostaje rozpoznana jako jedna z bazy,

**FRR** (ang. False Rejection Rate) - błąd fałszywego odrzucenia występuje, gdy osoba zarejestrowana w bazie danych (jej dane biometryczne) systemu nie zostaje rozpoznany jest odrzucona przez system.

Większy błąd FRR świadczy o większym bezpieczeństwie systemu. W razie fiasku identyfikacji użytkownik może ponownie podać się temu procesowi. Przyczyny tego błędu są różne. Przeważnie wpływ na to mają warunki zewnętrzne: ubrudzenie czytnika, złe oświetlenie, bądź też brak szkolenia użytkowników.

Istnieje również błąd klasyfikacyjny który rozpoznaje użytkownika jako inną osobę z danej bazy.

Największe szanse na rozwój biometryka posiada tam gdzie wazny jest poziom bezpieczeństwa tj: kontrole dostępu do zakladu penitencjarnego, komputery służb specjalnych itp. Kierunek, w którym ta technologia powinna się rozwijać to standaryzacja i unifikacja metod. Ważne jest by systemy były jednolite, by np. w przypadku sposobu pisania na klawiaturze nie używały różnych cech, gdyż może to spowodować błędy. Wraz z rozwojem technologii przewidziany jest też spadek kosztów czujników, co pozwoli na rozpowszechnienie technologii na szerszym rynku.

**Klasyfikacja metod biometrycznych**

* Statyczne (fizjologiczne)
  + Metoda Bertillona
  + Odciski palców
  + Rozpoznawanie twarzy
  + Skan tęczówki
  + Skan siatkówki
  + Geometria dłoni
  + Dopasowywanie wzorców naczyń krwionośnyc
* Dynamiczne (behawioralne
  + Rozpoznawanie mowy
  + Rozpoznawanie pisma odręcznego
  + Odstępy między kliknięciami klawiszy klawiatury podczas pisania (Keystroke dynamics)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metoda** | **Dokładność** | **Niezawodność** | **Źródło błędów** | **Poziom bezp.** | **Pomyłka pozytywna** | **Pomyłka negatywna** |
| **Odcisk palca** | Wysoka | Wysoka | Wilgoć, zanieczyszczenia, wiek | Wysoki | Bardzo trudno | Bardzo trudno |
| **Geometria dłoni** | Średnia | Średnia | Uszkodzenia dłoni, artretyzm, wiek | Niski | Trudno | Średni |
| **Tęczówka oka** | Wysoka | Wysoka | Oświetlenie | Wysoki | Bardzo trudno | Bardzo trudno |
| **Siatkówka oka** | Wysoka | Bardzo wysoka | Okulary | Wysoki | Bardzo trudno | Bardzo trudno |
| **Geometria twarzy** | Średnia | Średnia | Oświetlenie, okulary, włosy (zarost), wiek | Średni | Trudno | Łatwo |

# Systemy pośredniczące w wymianie danych

**EDI (Electronic Data Interchange)** – ustandaryzowana metoda wymiany danych pomiędzy komputerami bądź sieciami komputerowymi, z wykorzystaniem standardowych, zaakceptowanych formatów komunikatów. EDI to sposób wymiany informacji pomiędzy kontrahentami, charakteryzujący się przesyłaniem dokumentów wyłącznie w formie elektronicznej („bezpapierowej”), wymianą dokumentów spełniających uzgodnione wcześniej standardy (definicja pól dokumentu i ich zawartości), wymianą danych bezpośrednio pomiędzy systemami informatycznymi partnerów handlowych; dane będące przedmiotem elektronicznej wymiany są odpowiednikami tradycyjnych dokumentów handlowych takich jak: faktura, zamówienie, awizo dostawy, harmonogram produkcji itp. Kluczowym aspektem EDI jest standaryzacja przesyłanych informacji oraz bezpieczeństwo komunikacji.

EDI polega na wymianie danych pomiędzy systemami informatycznymi, nie pomiędzy ludźmi. Dane te mogą być automatycznie przetwarzane przez komputer.

EDI nie określa sposobu przesyłania komunikatów – mogą one być przesyłane przez dowolne medium, którym posługują się obie strony transmisji. Może to być transmisja modemowa, poprzez FTP, HTTP, AS1, AS2.

EDI łączy możliwości informatyki i telekomunikacji. Umożliwia eliminację dokumentów papierowych zwiększając efektywność wszystkich działań związanych z handlem. Dokładniej mówiąc wykorzystanie EDI wiąże się z spadkiem kosztów własnych ponoszonych na łączności i wymianie informacji między firmami. Uzyskanie przewagi nad konkurencją, szybsza reakcja na zmiany i potrzeby. Wprowadzenie EDI w węższym zakresie nie wymaga wielu zmian w organizacji ani metodach działania firmy (jedynie przysyłania dokumentów i informacji) Redukcja błędów powstających wskutek tradycyjnych metod wymiany danych np. błędów przy wypełnianiu papierkowych dokumentów. Optymalizacja procesów np. zamówień i zleceń, wszystko odbywa się natychmiastowo bez potrzeby oczekiwania na papierowe zamówienia. Największym minusem jednak samego systemu jest jego wdrażanie, które jest raczej trudnym zadaniem, a błędy nie wykryte na początku mogą propagować w przyszłości.

**Web Servicy -** Są to usługi internetowe śwwiadczone poprzez sięc telekomunikacyjną, a w szczególności przez internet. Same Web Servicy zazwyczaj są tylko składnikiem oprogramowania, który jest niezależny od platformy oraz implementacji. Web Serivcy są opisywane przez język WSDL, język ten opsiuje sposób wywoływania usług, ich parametry, zwracane wyniki oraz udostępniane metody. Ogólnie WSDLe pozwalają twórcom usług internetowych je opisać, napisać gdzie się one znajdują oraz  w jaki sposób można z nich skorzystać. WSDL bazuje na XML.

Komunikaty wysyłane przez Web Servicy są wysyłane przy użyciu protkołu SOAP, jest to protokół, mechanizm transportu informacji, służący zazwyczaj do jednokierunkowej komunikacji czyli w przypadku Web Serviców będzie to komunikacja nadwaca-klient, podobnie jak WSDL SOAP jest oparty na XML.

UDDI jest to rejestr globalny usług, które zawiera ich opis oraz lokalizację. Głównym jego zadaniem jest umożliwienie klientom dynamicznego wyszukiwania usług udostępnianych w internecie. Użytkownikami UDDI są zarówno klienci (osoby chcące skorzystać z Web Servicu) jak i usługodawcy, ponieważ usługodawca udostępnia swoje usługi a informacje przypisane do klienta zawierają szczegóły poszukiwanych przez niego usług.

Najczęsciej używanym przez usługi internetowe protokołem do przekazywania danych jest protkołu HTTP oraz wykorzystanie plików XML (są to zalecenia W3C).

Główną zaletą Web Serviców jest jak wspomniałem ich niezależność od platformy oraz implementacji, co pozwala na udostępnianie różnorodnych usług różnym klientom bez potrzeby zmiany chociażby systemu operacyjnego czy używanego języka programowania.

**CORBA** - technologia ta umożliwia komunikację pomiędzy obiektami pracującymi w różnorodnych systemach. Podobnie jak w przypadku Web Serviców obiekty te mogą być zaimplementowane w różnych językach proramowania na dowolnej platformie sprzętowej pod dowolnym systemm operacyjnym.

Również podobnie jak w przypadku Web Serviców CORBA wymusza opis obiektów (interfejsów do obiektów) w postaci plików IDL, który jest kompilowan na kod zajmujący się przekazywaniem metod.

Również jak przy Web Servicach, obiekty mają jakiś swój adres nazywany IOR, które zawierają dodatkowo wiele informacji o obiekcie m.in. adres komputera, adres programu na komputerze, numer obiektu, typ obiektu, itd.

CORBA jak i Web Servicy nie należą do najbezpieczniejszych rozwiązań, nie mniej w przypadku Web Serviców podobno coś się robi, żeby to bezpieczeństwo zwiększyć nie wiem jak w przypadku CORBY. Największym chyba minusem CORBY jest jej wydajność, która jest mało efektywna w czasie rzeczywistym oraz duże koszty jej wdrożenia. Podobnie jak DCOM CORBA najczęsciej jest opracowywana na sieci prywatnych, dlatego też z uwagi na szerszy zakres Web Servicy mają szanse zastąpić to rozwiązanie.

# Techniki modelowania danych i przepływu danych

**ERD (Entity-relationship diagram)** - Diagram związków encji odzwierciedla statyczną strukturę systemu, a konkretnie związki istniejące pomiędzy poszczególnymi zbiorami danych.

Składa się z:

* typów obiektów - encji - przedstawia się ja za pomocą prostokątów. Typ obiektu reprezentuje zbiór obiektów, reczy ze świata rzeczywistego, które mogą być jednoznacznie identyfikowane oraz opisane przez jeden lub więcej atrybutów.
* związki - przedsawia się za pomocą rombów. Związki reprezentują wszelkie powiązania i asocjacje występujące pomiędzy encjami.

Encja:

* reprezentuje zbiór obiektów opisanych tymi samymi cechami (atrybuty, własności)
* konkretny obiekt świata rzeczywistego jest instancją encji
* każda encja posiada swoją unikalną nazwę oraz zbiór cech
* encje wchodzą w związki z innymi encjami.
* dowolny obiekt może być reprezentowany przez tylko jedną encję
* nazwa encji - rzeczownik w liczbie pojedynczej
* encja musi posiadać unikatowy identyfikator ( naturalni - NIP, PESEL; sztuczny  - numer pozycji katalogowej, id pracownika )
* encja może być silna bądź słaba. Encja jest słaba gdy jej wystąpienie jest uwarunkowane występowaniem innych encji, tzn. zależy od kontekstu. Przykładem może być encja ‘realizacja’, która występuje tylko wtedy gdy występują encje ‘pracownik’ i ‘projekt’. Tylko encje silne mają własny identyfikator.

Związek:

* reprezentuje powiązania pomiędzy obiektami świata rzeczywistego (łączy encje)
* może zawierać krótki opis ułatwiający jej interpretację
* posiada 3 cechy:
  + stopień związku
    - unarny (1 encja)
    - binarny (2 encje)
    - ternarny (3 encje)
    - n-narny (n encji)
  + typ asocjacji
    - jeden-do-jeden ( 1:1 ) - np. jeden zespół może mieć tylko jednego kierownika, natomiast każdy zespół musi mieć kierownika.
    - jeden-do-wiele ( 1:N ) - np. Każdy pracownik pracuje w jednym dziale. Dział może zatrudniać wielu pracowników.
    - wiele-do-wiele ( M:N ) - np. Pracownik może brać udział w wielu projektach. Każdy projekt jest realizowany przez co najmniej jednego pracownika.
  + klasa przynależności
    - obowiązkowa
    - opcjonalna
* może posiadać atrybuty np. jak pracownik bierze udział w jakimś projekcie, to atrybutem może być jego wynagrodzenie

W przypadku diagramów ERD występuje jeszcze generalizacja - mówi ona o tym, że pewne encje o wspołnych cechach można uogólnić do jednej encji wyższego poziomu - encję genrealizacji. Przykładem generalizacji może być pracownik i klient, dla których taką encją będzie encja osoba.

**DFD (DataFlow Diagram)** - Diagram przepływu danych z kolei opisuje dynamiczne właściwości systemu.

Składa się z:

* procesów - proces to pojedyncza funkcja realizowana przez system. Graficznie jest reprezentowana przez kółko.
* przepływów -  przepływ to związek między procesami. Graficznie jest reprezentowany przez strzałkę.
* zbiorów danych - zbiory danych to magazyny danych, które przechowują jak sama nazwa wskazuje dane. Graficznie mogą być reprezentowane w dwojaki sposób albo poprzez elipsy, albo nazwę między dwiema pionowymi kreskami.
* terminatorów -  terminatorem określa się obiekt zewnętrzny, który komunikuje się z systemem, np. klient. Graficznie jest reprezentowany przez prostokąt.

Schematy te wzbogacane są również o tekstowe narzędzia modelowania:

* Specifykację danych - są to słowniki zawierające definicje struktur danych.
* Specyfikację procesów - zawiera idee oraz specyfikacje konkretnych algorytmów.

# Tłumaczenie komputerowe tekstów w językach naturalnych

Przez język naturalny rozumiemy zarówno mówiony jak i pisany język danej spo-łeczności. Język mówiony raczej nie jest wykorzystywany jako język informacyjno-wyszukiwawczy. Jeśli nawet pytania do systemu i jego odpowiedzi przekazywane są w posta-ci mówionej, to i tak przekształcane są one z/na formę tekstu pisanego i wykorzystywane wewnątrz systemu do wyszukiwania właśnie w takiej postaci.

Aby język naturalny mógł być wykorzystywany jako język informacyjno-wyszukiwawczy powinno się zapewnić procedurę automatycznego ustalania równoważności lub wymaganego stopnia zbieżności znaczeniowej (bliskoznaczności) dowolnych dwóch tek-stów w tym języku.

Klasyfikacje alfabetyczno-przedmiotowe, języki deskryptorowe i język syntagmatycz-ny SYNTOL oparte są na wykorzystaniu fragmentów znormalizowanej leksyki jakiegoś języ-ka naturalnego. Nie są one jednak uznawane za wykorzystanie języka naturalnego w charakte-rze języka informacyjno-wyszukiwawczego.

Gdy mówi się, że w jakimś systemie język naturalny jest wykorzystywany jako język informacyjno-wyszukiwawczy, na ogół chodzi o to, że słowa i związki językowe są wykorzy-stywane do indeksowania dokumentów bez normalizacji i eliminowania ich niejednoznacz-ności. Jednak ta właśnie niejednoznaczność jest największą przeszkodą w wykorzystywaniu języka naturalnego w systemach informacyjnych.

**Tłumaczenie automatyczne** albo **tłumaczenie maszynowe** (ang. Machine Translation) jest dziedziną językoznawstwa komputerowego, które zajmuje się stosowaniem algorytmów tłumaczenia tekstu z jednego języka (naturalnego) na drugi. Główne metody, przez które realizowane jest automatyczne tłumaczenie:

● Systemy tłumaczenia bezpośredniego – wyrazy tekstu źródłowego zamieniane są tu wprost na tłumaczenie w oczekiwanym języku. Program zawiera odpowiadające sobie słowa i najczęściej stosowane frazy. Tłumaczenie tego typu daje akceptowalne wyniki tylko w zastosowaniu dla blisko ze sobą spokrewnionych języków. (Coś jak słownik)

● Systemy przekładu składniowego – analizują składniową stronę tekstu. Najczęściej rezultatem jest drzewo składników, do którego następnie stosuje się odpowiednie reguły transferu.

● Systemy oparte na powierzchniowym transferze semantycznym – biorą pod uwagę własności składniowe i częściowo znaczeniowe. Realizowane jest to poprzez dołączenie do drzewa struktury syntaktycznej dodatkowych informacji naprowadzających, np. atrybutów znaczeniowych.

● Systemy międzyjęzykowe – oparte są na uniwersalnym języku reprezentacji znaczenia (tzw. interlingwę), który jest niezależny od języków naturalnych, zawartych w systemie. Proces translacji składa się z dwóch etapów: tłumaczenia z języka źródłowego na interlingwę i tłumaczenia z interlingwy na język wynikowy. (skojarzeniem może być proces komunikacji w EDI, gdzie wysyła się plik z firma A, który jest zamieniany przez XSLT na uniwersalny format, a następnie zamieniany przez inny XSLT na plik w formacie firmy B)

● Tłumaczenie statystyczne – tłumaczenie w oparciu o wielkie zestawy (korpusy) tekstów przetłumaczonych przez człowieka. Dla danego zdania szukane jest jego najbardziej prawdopodobne tłumaczenie. Prawdopodobieństwo tłumaczenia obliczane jest na podstawie współwystępowania wyrazów w zebranym korpusie. Sukcesy w tym podejściu notuje portal Google, gdyż korzysta ze swoich olbrzymich korpusów stron internetowych.

● Tłumaczenie oparte na przykładach – podobnie jak tłumaczenie statystyczne opiera się na istniejących tekstach przetłumaczonych. Dla danego zdania źródłowego system szuka najbardziej podobnego przykładu w swojej bazie danych i na tej podstawie wnioskuje jego tłumaczenie.

Kwestie/problemy semantyczne związane z automatycznym tłumaczeniem:

● Trzy rodzaje sensowności:

* + Sensowność lokucyjna – związana jest ona z językiem naturalnym i jest słownikowym znaczeniem znaków. Nie zależy ona od kontekstu sytuacyjnego. Sensowność ta jest stopniowalna (np. prosimy o pojaśnianie).
  + Sensowność logiczna – nie ujawnia się empirycznie. Sensowność ta jest związana językiem logiki. Jednakże język logiki komunikowalny jest tylko poprzez język naturalny. Powstaje problem przekładu. Nie jest ona stopniowalna, ani nie zależy od kontekstu sytuacyjnego.
  + Sensowność wolicjonalno-emotywna – ujawnia się w kontekstach sytuacyjnych. Wypowiedzi mogą być niedostosowane do konwencji sytuacyjnej.

● Metaforyczność stanowi ważny problem. W języku naturalnym znajdują się metafory (nie ma ich w języku logiki). Kiedy mamy do czynienia z metaforą? Wtedy gdy zostanie złamana zasada kompozycji znaczeniowej, która mówi, że znaczenie wyrażenia całościowego jest funkcją znaczeń wyrażeń składowych. Np. zdanie: „Matematyka jest moją piętą achillesową” oznaczałoby dosłownie, nie metaforycznie, że matematyka jest dla mnie częścią nogi mitycznego herosa. Jednakże nie można pozbyć się metafor, gdyż język straciłby swą moc informacyjną.

● Synonimia. Nawet w jednym języku nastręcza ona trudności przez to, iż wyrazy mają różny zakres pojęciowy. W tłumaczeniu problem ten ulega tylko powiększeniu. Częstokroć bywa tak, że tłumaczony wyraz nie ma swego odpowiednika w drugim języku.

● Homonimy są kolejnymi problemami w automatycznym tłumaczeniu. Właściwa interpretacja jest przy nich bardzo ważna. Np. zdanie: „Podszedł do zamku” można interpretować na różne sposoby i interpretacje są zależne od kontekstu. Pojawia się tu kwestia umiejętności donoszenia się do kontekstu przez automatycznego tłumacza.

● Dużą grupę problemów stanowią różnice w składni pomiędzy językami. Np. w języku angielskim istotne jest miejsce wyrazu w zdaniu (jest to język pozycyjny), natomiast w języku polskim nie.

● Innymi problemami jest występowanie w języku rodzajników określonych i nieokreślonych lub podmiotu domyślnego. Powstaje również pytanie o możliwość stworzenia języka pośredniego w tłumaczeniu (interlingwa z systemu międzyjęzykowego), biorąc pod uwagę dotychczasowe rozważania.

# Własności sieci neuronowych

Jako obiekt badań sieci neuronowe stanowią bardzo uproszczony (przez co łatwiejszy do ogarnięcia myślą lub do zamodelowania na komputerze), ale bogaty i ciekawy model rzeczywistego biologicznego systemu nerwowego.

Składają się one z połączonych ze sobą obiektów (umownie zwanych neuronami). Istotną cechą sieci takich elementów jest możliwość uczenia się - to jest modyfikowania parametrów charakteryzujących poszczególne neurony w taki sposób, by zwiększyć efektywność sieci przy rozwiązywaniu zadań określonego typu.

Sieci neuronowe mogą być bardzo skuteczne jako narzędzia obliczeniowe - i to w rozwiązywaniu takich zadań, z którymi typowe komputery i typowe programy sobie nie radzą. Jest tak z tego powodu, że sieci neuronowe mają w stosunku do typowych systemów obliczeniowych dwie zasadnicze zalety. Po pierwsze obliczenia są w sieciach neuronowych wykonywane równolegle, w związku z czym szybkość pracy sieci neuronowych może znacznie przewyższać szybkość obliczeń sekwencyjnych. Drugą zaletą sieci jest możliwość uzyskania rozwiązania problemu z pominięciem etapu konstruowania algorytmu rozwiązania problemu.

Sieci nie trzeba programować. Istnieją metody uczenia i samouczenia sieci pozwalają uzyskać ich celowe i skuteczne działanie nawet w sytuacji, kiedy twórca sieci nie zna algorytmu, według którego można rozwiązać postawione zadanie.

Zarówno program działania oraz informacje stanowiące bazę wiedzy, a także dane na których wykonuje się obliczenia, jak i sam proces obliczania - są w sieci całkowicie rozproszone.

Sieć działa zawsze jako całość i wszystkie jej elementy mają swój wkład w realizację wszystkich czynności, które sieć realizuje. Jedną z konsekwencji takiego działania sieci jest jej zdolność do poprawnego działania nawet po uszkodzeniu znacznej części wchodzących w jej skład elementów.

Struktura sieci powstaje w ten sposób, że wyjścia jednych neuronów łączy się z wejściami innych. Oczywiście konkretna topologia sieci powinna wynikać z rodzaju zadania, jakie jest stawiane przed siecią. Jednak decyzje dotyczące struktury sieci nie wpływają na jej zachowanie w stopniu decydującym. Zachowanie sieci w zasadniczy sposób determinowane jest przez proces jej uczenia, a nie przez strukturę czy liczbę użytych do jej budowy neuronów.

Znane są doświadczenia, w których strukturę sieci wybierano w sposób całkowicie przypadkowy (ustalając na drodze losowania, które elementy należy ze sobą połączyć i w jaki sposób), a sieć mimo to zdolna była do rozwiązywania stawianych jej zadań.

Sieci neuronowe mogą całą swoją wiedzę zyskiwać wyłącznie w trakcie nauki i nie muszą mieć z góry zadanej, dopasowanej do stawianych im zadań, jakiejkolwiek precyzyjnie określonej struktury. Sieć musi jednak mieć wystarczający stopień złożoności, żeby w jej strukturze można było w toku uczenia "wykrystalizować" potrzebne połączenia i struktury. Zbyt mała sieć nie jest w stanie nauczyć się niczego, gdyż jej "potencjał intelektualny" na to nie pozwala - rzecz jednak nie w strukturze, a w liczbie elementów.

**Modele neuronów**

Na podstawie zasad działania rzeczywistego neuronu stworzono wiele modeli matematycznych, w których uwzględnione zostały w większym lub mniejszym stopniu właściwości rzeczywistych komórek nerwowych. Schemat obwodowy powiązany z większością tych modeli odpowiada modelowi McCullocha-Pittsa.

Jego działanie jest następujące: Do wejść doprowadzane są sygnały dochodzące z neuronów warstwy poprzedniej. Każdy sygnał xj mnożony jest przez odpowiadającą mu wartość liczbową zwaną wagą wij. Wpływa ona na percepcję danego sygnału wejściowego i jego udział w tworzeniu sygnału wyjściowego przez neuron. Zsumowane iloczyny sygnałów i wag stanowią argument funkcji aktywacji neuronu f(si).

**Zastosowanie sieci neuronowych**

Trudno wymienić wszystkie aktualnie spotykane zastosowania sieci neuronowych. Oto niektóre z nich.

**... w technice**

Do dziś najczęstszym spotykanym obszarem zastosowań technicznych sieci neuronowych są zagadnienia rozpoznawania, a zwłaszcza problem rozpoznawania kontekstowego. Sieci neuronowe stosuje się także do zadań klasyfikacji oraz do analizy obrazów i ich przetwarzania. Wyróżnić można tu kompresję, segmentację, odtwarzanie oraz rozumienie obrazów. Aby sklasyfikować i rozpoznawać wzorce, sieć uczy się ich podstawowych cech takich jak: odwzorowania geometryczne, pikselowego układu wzorca, rozkładu składników głównych wzorca, składników transformacji Fouriera czy innych jego właściwości. W uczeniu podkreślane są różnice występujące w różnych wzorcach.

Kolejna dziedziną zastosowań jest wykorzystanie sieci neuronowych do klasycznych zadań przetwarzania sygnałów, takich jak konwersje, filtracje i aproksymacje oraz inne odwzorowania i transformacje. Coraz więcej prac opisuje struktury takich sieci i ich zastosowanie.

Inne, często spotykane, zastosowania sieci neuronowych dotyczą robotyki, automatyki, a także teorii sterowania i zagadnień optymalizacji, percepcji ruchu i jego planowania. W zagadnieniach identyfikacji i sterowania procesami dynamicznymi sieć neuronowa pełni zwykle kilka funkcji. Stanowi model nieliniowy tego procesu, pozwalający na wypracowanie odpowiedniego sygnału sterującego. Pełni również funkcje układu śledzącego i nadążnego, adaptując się do zmiennych warunków środowiskowych. Ważna rolę, zwłaszcza w sterowaniu robotów, odgrywa funkcja klasyfikatora wykorzystywana w podejmowaniu decyzji, co do dalszego przebiegu procesu.

Sieci neuronowe stosowane są do zbudowania pamięci asocjacyjnej, zwłaszcza duże zainteresowanie budzi klasa tak zwanych dwukierunkowych pamięci asocjacyjnych BAM. W zadaniach asocjacji sieć neuronowa pełni role pamięci skojarzeniowej. W przypadku pamięci asocjacyjnej, skojarzenie dotyczy tylko poszczególnych składników wektora wejściowego. W pamięci heteroasocjacyjnej zadaniem sieci jest skojarzenie ze sobą dwóch wektorów. Gdy na wejście będzie podany wektor odkształcony, sieć neuronowa jest w stanie odtworzyć wektor oryginalny, pozbawiony szumów.

**... w ekonomii**

Sieci neuronowe mają liczne i różnorodne zastosowania poza techniką np.w ekonomii. Oto ważniejsze kierunki tych zastosowań.

Sieci neuronowe są wykorzystywane do przewidywania pewnych określonych rozwiązań na podstawie danych początkowych. Przykładami mogą być wszelkiego rodzaju prognozy ekonomiczne np. przewidywania bessy i hossy na giełdzie.

Sieci mogą klasyfikować na podstawie danych bilansowych czy dane przedsiębiorstwo należy do zwyżkujących gospodarczo czy też może przeżywa stagnację lub regres. Można klasyfikować, w które branże warto inwestować lub który region może być zagrożony bezrobociem.

Dzięki zdolnościom adaptacji, sieci wykorzystuje się do procesów wnioskowania na podstawie zgromadzonych danych. Ułatwiają one managerom wykrywanie istotnych powiązań i ważnych aspektów.

Analiza danych prowadzona z wykorzystaniem sieci pozwala na ustalenie np. przyczyn niepowodzeń określonych przedsięwzięć podejmowanych w przeszłości, dzięki czemu łatwiejsze jest unikanie błędów na przyszłość.

Optymalizacja - sieci poszukują rozwiązań prowadzących do optymalnych decyzji gospodarczych. Sieć taka może rozwiązać np. klasyczne zadanie zwane problemem komiwojażera.

Mimo ogromnej liczby zastosowań sieci neuronowych, możliwości ich dalszego wykorzystania w przetwarzaniu sygnałów są jeszcze nie do końca zbadane i wydaje się, że będą one jeszcze przez wiele lat stanowić o postępie w technice informacyjnej. Prace nad dalszymi zastosowaniami trwają...

**2.3 Opis działania neuronu**

Sztuczne neurony charakteryzują się występowaniem wielu wejść i jednego wyjścia. Sygnały wejściowe xi (i = 1, 2, . . . , n) oraz sygnał wyjściowy y mogą przyjmować wartości, odpowiadające pewnym informacjom. W ten sposób zadanie sieci sprowadzone do funkcjonowania jej podstawowego elementu polega na tym, że neuron przetwarza informacje wejściowe xi na pewien wynik y. Neurony traktować można jako elementarne procesory o następujących właściwościach:

• każdy neuron otrzymuje wiele sygnałów wejściowych i wyznacza na ich podstawie swoją odpowiedź to znaczy jeden sygnał wyjściowy;

• z każdym oddzielnym wejściem neuronu związany jest parametr nazywany wagą ( określa stopień ważności informacji docierających tym właśnie wejściem - na wyjście ma wpływ waga przemnożona przez wejście);

• sygnał wchodzący określonym wejściem jest najpierw przemnażany przez wagę danego wejścia, w związku z czym w dalszych obliczeniach uczestniczy już w formie zmodyfikowanej: wzmocnionej (gdy waga jest większa od 1) lub stłumionej (gdy waga ma wartość mniejszą od 1) względnie nawet przeciwstawnej w stosunku do sygnałów z innych wejść gdy waga ma wartość ujemną (tzw. wejścia hamujące);

• sygnały wejściowe (przemnożone przez odpowiednie wagi) są w neuronie sumowane, dając pewien pomocniczy sygnał wewnętrzny nazywany czasem łącznym pobudzeniem neuronu (w literaturze angielskiej net value);

• do tak utworzonej sumy sygnałów dodaje niekiedy (nie we wszystkich typach sieci) pewien dodatkowy składnik niezależny od sygnałów wejściowych, nazywany progiem (w literaturze angielskiej bias);

• suma przemnożonych przez wagi sygnałów wewnętrznych z dodanym (ewentualnie) progiem może być bezpośrednio traktowana jako sygnał wyjściowy neuronu ( w sieciach o bogatszych możliwościach sygnał wyjściowy neuronu obliczany jest za pomocą pewnej nieliniowej zależności między łącznym pobudzeniem a sygnałem wyjściowym);

Każdy neuron dysponuje pewną wewnętrzną pamięcią (reprezentowaną przez aktualne wartości wag i progu) oraz pewnymi możliwościami przetwarzania wejściowych sygnałów w sygnał wyjściowy.

Z powodu bardzo ubogich możliwości obliczeniowych pojedynczego neuronu - sieć neuronowa może działać wyłącznie jako całość. Wszystkie możliwości i właściwości sieci neuronowych są wynikiem kolektywnego działania bardzo wielu połączonych ze sobą elementów (całej sieci, a nie pojedynczych neuronów).

**2.4 Uczenie sieci neuronowych**

Cykl działania sieci neuronowej podzielić można na etap nauki, kiedy sieć gromadzi informacje potrzebne jej do określenia, co i jak ma robić, oraz na etap normalnego działania (nazywany czasem także egzaminem), kiedy w oparciu o zdobytą wiedzę sieć musi rozwiązywać konkretne nowe zadania. Możliwe są dwa warianty procesu uczenia : z nauczycielem i bez nauczyciela.

**2.4.1 Uczenie z nauczycielem**

Uczenie z nauczycielem polega na tym, że sieci podaje się przykłady poprawnego działania, które powinna ona potem naśladować w swoim bieżącym działaniu (w czasie egzaminu). Przykład należy rozumieć w ten sposób, że nauczyciel podaje konkretne sygnały wejściowe i wyjściowe, pokazując, jaka jest wymagana odpowiedź sieci dla pewnej konfiguracji danych wejściowych. Mamy do czynienia z parą wartości - przykładowym sygnałem wejściowym i pożądanym (oczekiwanym) wyjściem, czyli wymaganą odpowiedzią sieci na ten sygnał wejściowy. Zbiór przykładów zgromadzonych w celu ich wykorzystaniu w procesie uczenia sieci nazywa się zwykle ciągiem uczącym. Zatem w typowym procesie uczenia sieć otrzymuje od nauczyciela ciąg uczący i na jego podstawie uczy się prawidłowego działania, stosując jedną z wielu znanych dziś strategii uczenia.

**2.4.2 Uczenie bez nauczyciela**

Obok opisanego wyżej schematu uczenia z nauczycielem występuje też szereg metod tak zwanego uczenia bez nauczyciela (albo samouczenia sieci). Metody te polegają na podawaniu na wejście sieci wyłącznie szeregu przykładowych danych wejściowych, bez podawania jakiejkolwiek informacji dotyczącej pożądanych czy chociażby tylko oczekiwanych sygnałów wyjściowych. Odpowiednio zaprojektowana sieć neuronowa potrafi wykorzystać same tylko obserwacje wejściowych sygnałów i zbudować na ich podstawie sensowny algorytm swojego działania - najczęściej polegający na tym, że automatycznie wykrywane są klasy powtarzających się sygnałów wejściowych i sieć uczy się (spontanicznie, bez jawnego nauczania) rozpoznawać te typowe wzorce sygnałów.

Samouczenie jest też bardzo interesujące z punktu widzenia zastosowań, gdyż nie wymaga żadnej jawnie podawanej do sieci neuronowej zewnętrznej wiedzy , a sieć zgromadzi wszystkie potrzebne informacje i wiadomości.

**2.5 Organizacja uczenia sieci**

Kluczowym pojęciem dla uczenia sieci są wagi wejść poszczególnych neuronów. Każdy neuron ma wiele wejść, za pomocą których odbiera sygnały od innych neuronów oraz sygnały wejściowe podawane do sieci jako dane do obliczeń. Z wejściami tymi skojarzone są parametry nazywane wagami; każdy sygnał wejściowy jest najpierw przemnażany przez wagę, a dopiero później sumowany z innymi sygnałami. Jeśli zmienią się wartości wag - neuron zacznie pełnić innego rodzaju funkcję w sieci, a co za tym idzie - cała sieć zacznie inaczej działać. Uczenie sieci polega więc na tym, by tak dobrać wagi, żeby wszystkie neurony wykonywały dokładnie takie czynności, jakich się od nich wymaga.

Ze względu na rozmiar sieci (w wielu wypadkach mamy do czynienia z b. dużą liczbą neuronów) niemożliwe jest zdefiniowanie potrzebnych wag dla wszystkich wejść w sposób jednorazowy i arbitralny ręcznie. Można jednak zaprojektować i zrealizować proces uczenia polegający na rozpoczęciu działania sieci z pewnym przypadkowym zestawem wag i na stopniowym polepszaniu tych wag. W każdym kroku procesu uczenia wartości wag jednego lub kilku neuronów ulegają zmianie, przy czym reguły tych zmian są tak pomyślane, by każdy neuron sam potrafił określić, które ze swoich wag ma zmienić, w którą stronę (zwiększenie lub zmniejszenie) a także o ile. Oczywiście przy określaniu potrzebnych zmian wag neuron może korzystać z informacji pochodzących od nauczyciela (o ile stosujemy uczenie z nauczycielem), nie zmienia to jednak faktu, że sam proces zmiany wag (będących w sieci jedynym śladem pamięciowym) przebiega w każdym neuronie sieci w sposób spontaniczny i niezależny dzięki czemu może być realizowany bez konieczności bezpośredniego stałego dozoru ze strony osoby sterującej tym procesem.

W praktycznych zastosowaniach korzysta się czasem z dodatkowego mechanizmu "rywalizacji" między neuronami, który w niektórych zastosowaniach pozwala uzyskiwać znacznie lepsze wyniki działania sieci. Zaobserwowanie działania sieci z rywalizacją (competition network) możliwe jest po wprowadzeniu do sieci elementu porównującego ze sobą sygnały wyjściowe wszystkich neuronów i typującego wśród nich "zwycięzcę". Zwycięzcą w tej konkurencji zostaje neuron o największej wartości sygnału wyjściowego. Z wytypowaniem "zwycięzcy" mogą wiązać się różne konsekwencje (na przykład tylko temu jednemu neuronowi można nadać prawo uczenia się (sieci Kohonena), najczęściej jednak wytypowanie zwycięzcy służy do tego, by silniej spolaryzować wyjściowe sygnały z sieci - na przykład tylko neuron będący "zwycięzcą" ma prawo wysłać swój sygnał na zewnątrz, wszystkie pozostałe sygnały są natomiast zerowane. Taka zasada działania sieci, nazywana czasem WTA (Winner Takes All - zwycięzca zabiera wszystko) pozwala łatwiej interpretować zachowanie sieci (szczególnie wtedy, gdy ma ona wiele wyjść), ale niesie ze sobą pewne niebezpieczeństwa (wzmiankowane wyżej).

# Zarządzanie ryzykiem w projekcie informatycznym

**Ryzyko** – to możliwość, szansa wystąpienia **niebezpieczeństwa**, sytuacja **niedeterministyczna**, w której są określone prawdopodobieństwa wystąpienia przypadków, zarówno pozytywnych, jak i negatywnych. Ryzyko jest zjawiskiem permanentnym.

**Ryzyko projektu** oznacza ryzyko niedotrzymania technicznych i/lub ekonomicznych (finansowych) warunków projektu (przedsięwzięcia). Ryzyko jest funkcją, którego atrybuty możemy zdefiniować przez: zdarzenia, prawdopodobieństwo ich wystąpienia oraz konsekwencje – skutki, które mogą nastąpić w wyniku wystąpienia zdarzenia.

**Definicja podstawowa ryzyka**

**Ryzyko = *P*(*z*) ⋅ *S*(*z*)**

gdzie: *z* – element ryzyka, *P*(*z*) – prawdopodobieństwo wystąpienia *z*, *S*(*z*) – miara skutku wystąpienia *z*, wyrażana zazwyczaj szacunkowym kosztem lub wartością z przyjętej skali.

**Zarządzanie ryzykiem** Celem zarządzania ryzykiem jest utrzymanie odpowiedniego stopnia gwarancji odnośnie do sukcesu przedsięwzięcia. Poziom tego ryzyka, który w projekcie jest dopuszczalny, zależny jest od wielu czynników zarówno zewnętrznych, jak i wewnętrznych.  Są projekty, w których różne jego elementy mogą być krytyczne i poziom ryzyka musi być ograniczony do wartości bliskiej zeru.

**Czynnik ryzyka**

* Wydarzenie lub warunek, którego wystąpienie nie jest pewne, a które posiada pozytywny lub negatywny wpływ na cele projektu
* Posiada źródło w niepewności będącej stałym elementem każdego projektu
* Może dotyczyć zarówno zagrożeń (wpływ negatywny) dla projektu, jak i potencjałów (wpływ pozytywny) usprawnienia jego realizacji
* Posiada przyczynę, a jeżeli wystąpi, również konsekwencje wystąpienia

**Czynniki ryzyka w kontekście IT**

* technologia
* sprzęt
* zespół
* koszt
* harmonogram
* oprogramowania

**Kategorie ryzyka**

* **Ryzyko organizacyjne**, które wynika z :
  + nieadekwatność stosowanej technologii informatycznej
  + ograniczone zasoby realizacyjne przedsięwzięcia,
  + niedostateczna umiejętności i doświadczenie realizatorów systemu,
  + wykorzystanie odpowiednich metod, technik i narzędzi informatycznych,
* **Ryzyko psychospołeczne** określane przez:
  + niechęć do wprowadzania zmian organizacyjnych,
  + nieumiejętność celowego zastosowania technologii informatycznej,
  + niska kultura informatyczna.
* **Ryzyko techniczno-technologiczne** określane przez:
  + niski poziom w zakresie infrastruktury informatycznej,
  + wykorzystywanie niewłaściwej technologii informatycznej.

**Analiza ryzyka** Proces oceny prawdopodobieństwa i wpływu na projekt zidentyfikowanych czynników ryzyka. Właściwa ocena ryzyka sprowadza się do jego identyfikacji, a następnie opisu, samo uświadomienie sobie ryzyka nie wystarcza. Analiza musi dotyczyć opisanych zagrożeń – list kontrolnych oraz prognozowanego rozmiaru, skutków jakie dane zagrożenie będzie miało dla projektu oraz w jakiej jego fazie, jak również jakimi symptomami ryzyko będzie się przejawiać. Ważne jest też skupienie się na istotnych zagrożeniach, aby analiza była pomocna w uruchamianiu działań zapobiegawczych.

**Metody szacowania ryzyka**

**Metoda punktowa**

Metoda punktowa przebiega następująco:

1. Ustalenie kategorii ryzyka i ich wag dla projektu
2. Przegląd zadań, identyfikacja ryzyka i wartości ryzyka
3. Obliczenie ryzyka nieznormalizowanego i znormalizowanego w poszczególnych kategoriach oraz ryzyka całkowitego
4. W przypadku, gdy wartość ryzyka przekracza ustalony próg ryzyka akceptowalnego dokonujemy ponownego przeglądu zadań i wprowadzamy akcje zapobiegawcze, które pozwalają na zmniejszenie wartości ryzyka
5. Obliczamy wartość ryzyka po wprowadzeniu akcji zapobiegawczych
6. Powtarzamy kroki aż do uzyskania ryzyka na poziomie ryzyka akceptowalnego

**Metoda PERT**

Technika PERT (ang. *Program Evaluation and Review Techinque*) została stworzona w celu  oszacowania przybliżonych czasów trwania realizacji aktywności/zadań oraz wyznaczenia  prawdopodobieństwa zakończenia tych *aktywności/zadań* w żądanym czasie.

1. Oszacowanie czasu realizacji **t** pojedynczego zadania
2. Obliczyć standardowe **s**  odchylenia zadania
3. Wyznaczyć dla zadania wartość współczynnika *z* ze wzoru:

**z = (T-t)/s**

gdzie:

*T* – żądana data docelowa zakończenia zadania,

*t* – czas oszacowany w punkcie 1.

1. Odwzorować wartość z na prawdopodobieństwo, korzystając z „odpowiednich krzywych” zamieszczonych w np. tablicach matematycznych.

**Strategie zapobiegania (zarządzania) ryzykiem**

Wśród strategii postępowania z ryzykiem

wyróżnia się przede wszystkim następujące podejścia do zarządzania ryzykiem:

1. **Unikanie**
   * Zmiany planu projektu, w celu eliminacji ryzyka lub ochrony celów projektu przed jego wpływem
   * Wyjaśnianie wymagań, pozyskiwanie dodatkowych informacji, ekspertyzy, poprawa komunikacji, etc. (na wstępnym etapie projektu).
   1. Przykłady:
      * 1. Ograniczenie zakresu
        2. Zwiększenie zasobów
        3. Wydłużenie czasu trwania przedsięwzięcia
        4. Unikanie nieznanych podwykonawców
        5. Wykorzystanie jedynie znanych rozwiązań
2. **Przeniesienie**
   * Przeniesienie konsekwencji wystąpienia czynnika ryzyka na inny zespół/firmę wraz z odpowiedzialnością za zarządzanie nim
   * Najbardziej skuteczne w zarządzeniu ryzykiem finansowym. Niemal zawsze wiąże się z  finansowym zyskiem trzeciej strony.
   * Przykłady:
3. Ubezpieczenie
4. Gwarancje
5. Wybór kontraktu, np. kontrakt fix-price przenosi większość ryzyka na sprzedającego, podczas gdy kontakt typu time-and-material – na klienta
6. **Minimalizacja**
   * Działania związane z minimalizacją wpływu i/lub prawdopodobieństwa wystąpienia czynnika ryzyka do akceptowanego poziomu
   * Powinna uwzględniać koszt podejmowanych czynności w kontekście prawdopodobieństwa   ryzyka
   * Wcześnie podjęte działania są bardziej efektywne i tańsze niż radzenie sobie z konsekwencjami
   * Przykłady:
7. Zastosowanie mniej złożonych procesów wytwórczych
8. Przeprowadzenie większej liczby testów
9. Prototypowanie
10. **Akceptacja**
    * świadoma decyzja o niepodejmowaniu działań związanych z zarządzaniem i reagowaniem na ryzyko
    * Często zdarza się, że nie jest możliwe zastosowanie innej strategii postępowania z ryzykiem, poza  jego akceptacją
    * Najczęściej stosowane są rezerwy/plany rezerwowe

**Monitorowanie ryzyka**

* Ustalenie, czy dane ryzyko rzeczywiście miało miejsce
* Aktualizacja sumarycznego ryzyka projektowego
* Zebranie doświadczeń na potrzeby przyszłych przedsięwzięć
* Ustalenie przyczyny (identyfikacja czynników ryzyka, które spowodowały problem)

# Zarządzanie zmianami i konfiguracjami oprogramowania

??

# Zasady tworzenia harmonogramów realizacji systemu

**Etapy wykonywania harmonogramu:**

* Podział projektu na czynności
* Określanie zależności pomiędzy czynnościami
* Szacowanie czasu trwania czynności
* Ustalenie chronologii zadań i wyznaczenie tzw. „kamieni milowych”,
* Wykreślenie tzw. ścieżki krytycznej w postaci rysunku wektorowego - grafów.
* Tworzenie harmonogramu
* Kontrola harmonogramu

**Problemy do uwzględnienia:**

* Szacowanie złożoności problemów i przez to czasu koniecznego na ich rozwiązanie jest trudne
* Produktywność nie jest proporcjonalna do liczby pracowników
* Dodawanie pracowników spowalnia przedsięwzięcie ze względu na czas komunikacji
* Niespodziewane zdarzenia zawsze się pojawiają. Utrzymuj margines czasowy

**Planowanie uwzględnia**

* czas
* zasoby ludzkie
* zasoby materialne (techniczne)
* zasoby finansowe
* zadania do realizacji

**Powodzenie realizacji projektu wymaga znajomości**

* wszystkich zadań wchodzących w skład projektu
* czasu realizacji czynności (zdefiniowanie lub oszacowanie)
* wzajemnych zależności pomiędzy wszystkimi zadaniami (kolejności)

**Podstawowe techniki planowania**

* metoda graficzna Gantta - wiadomo
* metoda CPM - robimy ścieżkę krytyczną
* metoda PERT - szacujemy optymistycznie, średnio i pesymistycznie

# Zastosowanie sztucznej inteligencji w systemach informacyjnych

Obszar zastosowań + Przykłady zastosowań decyzyjnych

* Transport: rozwiązanie problemu komiwojażera w firmie transportowej / spedycyjnej / dostawczej / cateringowej,
* Sprzedaż: symulacja zapasów; problem transportowy w hurtowni; określenie pierwszego nakładu książki; rozmieszczenie punktów sprzedaży; samouczący się model konsumenta
* Organizacja: harmonogramowanie pracy; sporządzanie tras pociągów towarowych; sporządzanie tras pojazdów kołowych; kolejność lądowań samolotów; kolejność jazdy pociągów po linii jednotorowej; projektowanie sieci telekomunikacyjnych; optymalizacja sieci komputerowej; układanie podziału godzin w szkole; planowanie systemu dostaw wody
* Finanse: szacowanie ryzyka ubezpieczeniowego; prognozowanie bankructwa; określanie zdolności kredytowej; przewidywanie opóźnień w płatnościach; analiza cyklów czasowych; symulacja rynku akcji; optymalizacja portfela akcji; model gracza na rynku akcji; modelowanie rynków finansowych; wykrywanie oszustw finansowych; alokacja nakładów dla różnych programów społecznych;
* Produkcja: równoważenie linii produkcyjnej; określanie wielkości partii; sekwencyjność produkcji; proces planowania; magazynowanie; zarządzanie obciążeniami; optymalizacja działania reaktorów jądrowych; programowanie symulatorów lotniczych

O sztucznej inteligencji mówi się dużo. Praktycznie każda osoba studiująca informatykę, czy kierunek pokrewny, słyszała o algorytmach genetycznych, czy systemach ekspertowych. Niektórzy jednak wciąż traktują zagadnienia związane ze sztuczną inteligencją z przysłowiowym przymrużeniem oka. Trzeba powiedzieć, że sztuczna inteligencja jest dziedziną obejmującą wiele zagadnień, również tych niepowiązanych bezpośrednio z programowaniem, czy projektowaniem systemów informatycznych. Poniżej czytelnik może znaleźć kilka zagadnień należących, bądź związanych ze sztuczną inteligencją:

Algorytmy Genetyczne,

Programowanie Genetyczne,

Systemy ekspertowe,

Automatyczne pozyskiwanie wiedzy,

Inżynieria wiedzy.

**Metody sztucznej inteligencji znajdują zastosowanie:** Wmedycynie, Do rozpoznawania obrazów, Jako metoda optymalizacji, Do wspomagania decyzji, Do pozyskiwania wiedzy z baz danych, W zarządzaniu, Do projektowania układów elektronicznych, Do obróbki obrazu, W ekonomii i ekonometrii, W grach logicznych, Do klasyfikacji.

Przydatność algorytmów genetycznych jako metody optymalizacji nie jest kwestionowana np. w przypadku funkcji, których minimum nie można wyznaczyć tradycyjnymi metodami matematycznymi.