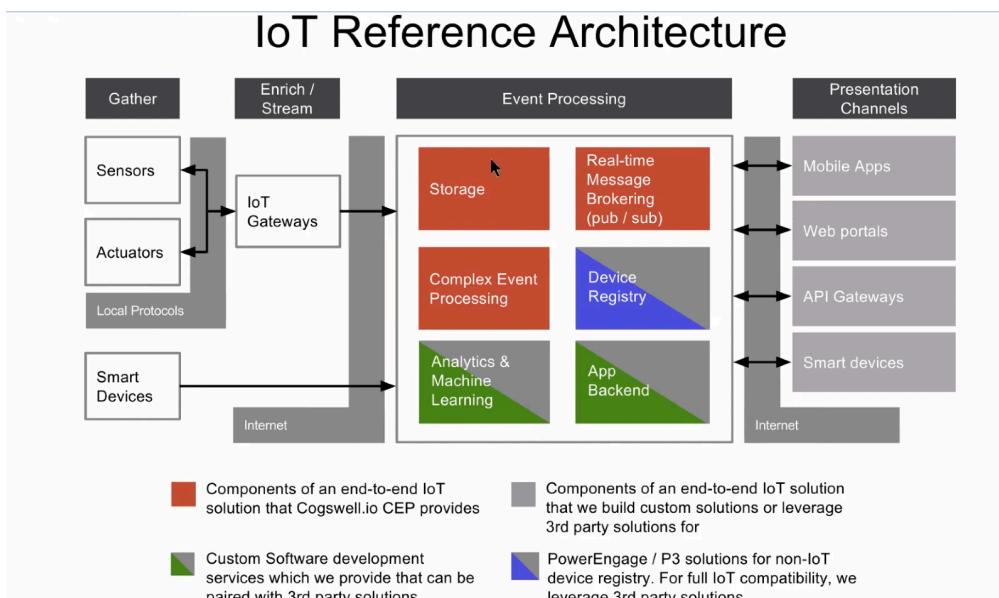


1. Architektura przetwarzania informacji i standardy komunikacji w systemach internetu rzeczy (ang. Internet of Things - IoT).

Architektura referencyjna IoT (z wykładu z Hukiem)



Architektura referencyjna to standardowy szablon lub model, który służy jako punkt odniesienia dla projektowania i budowy systemów informatycznych. Jest to szczegółowy zestaw wytycznych, wzorców, najlepszych praktyk i komponentów, które mogą być stosowane do tworzenia nowych systemów lub oceniania istniejących.

W kontekście Internetu Rzeczy (IoT), architektura referencyjna może definiować, jak różne elementy, takie jak czujniki, siłowniki, bramki IoT, przetwarzanie danych i interfejsy użytkownika, powinny być projektowane i integrowane, aby tworzyć spójny i efektywny system IoT. Dzięki takiemu podejściu można łatwiej tworzyć skalowalne, bezpieczne i wydajne rozwiązania IoT, które mogą być wykorzystywane w różnych branżach, od przemysłu po inteligentne domy.

Opis architektury referencyjnej IoT:

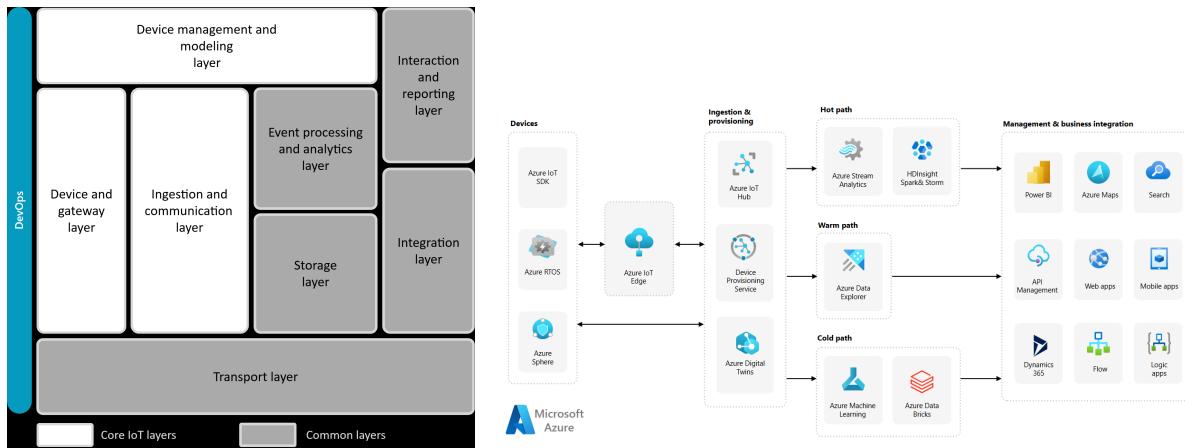
- 1. Zbieranie danych (Gather)**
 - Czujniki (Sensors):** Urządzenia zbierające dane z otoczenia (np. temperatura, wilgotność).
 - Siłowniki (Actuators):** Urządzenia wykonujące działania na podstawie otrzymanych poleceń (np. silniki, światła).
 - Inteligentne urządzenia (Smart Devices):** Urządzenia z wbudowanymi czujnikami i siłownikami, które mogą łączyć się z internetem.
- 2. Wzbogacanie / Strumieniowanie (Enrich / Stream)**

- **Bramki IoT (IoT Gateways):** Działają jako pośrednicy między czujnikami/siłownikami a warstwą przetwarzania zdarzeń. Zbierają dane z lokalnych protokołów i przesyłają je przez internet.
3. **Przetwarzanie zdarzeń (Event Processing)**
- **Magazynowanie (Storage):** Komponent do przechowywania zebranych danych.
 - **Brokerowanie wiadomości w czasie rzeczywistym (Real-time Message Brokering, pub/sub):** Zarządza wymianą danych w czasie rzeczywistym między urządzeniami i aplikacjami za pomocą modelu publikacji/subskrypcji.
 - **Złożone przetwarzanie zdarzeń (Complex Event Processing):** Analizuje i przetwarza strumienie danych w czasie rzeczywistym, aby wykrywać wzorce i wyciągać znaczące informacje.
 - **Rejestr urządzeń (Device Registry):** Utrzymuje bazę zarejestrowanych urządzeń i ich status.
 - **Analiza i uczenie maszynowe (Analytics & Machine Learning):** Analizuje przechowywane dane, aby wyciągać wnioski i tworzyć prognozy.
 - **Zaplecze aplikacji (App Backend):** Wspiera logikę aplikacji i współpracuje z komponentami przetwarzania zdarzeń.
4. **Kanały prezentacyjne (Presentation Channels)**
- **Aplikacje mobilne (Mobile Apps):** Aplikacje na urządzenia mobilne, które współpracują z systemem IoT.
 - **Portale internetowe (Web Portals):** Strony internetowe lub aplikacje webowe zapewniające interfejs użytkownika.
 - **Bramki API (API Gateways):** Interfejsy umożliwiające zewnętrznym systemom współpracę z systemem IoT.
 - **Inteligentne urządzenia (Smart Devices):** Mogą również służyć jako kanały prezentacyjne, wyświetlając dane lub bezpośrednio współpracując z użytkownikiem.

Przepływ danych:

1. Dane przepływają od czujników, siłowników i inteligentnych urządzeń do bramek IoT za pomocą lokalnych protokołów.
2. Bramki IoT przesyłają dane przez internet do warstwy przetwarzania zdarzeń.
3. W warstwie przetwarzania zdarzeń dane są magazynowane, przetwarzane i analizowane.
4. Przetworzone dane i wnioski są następnie udostępniane przez różne kanały prezentacyjne, do których użytkownicy mogą uzyskać dostęp poprzez aplikacje mobilne, portale internetowe, bramki API i inteligentne urządzenia.

Architektura IoT przedstawiona przez Microsoft



[Omówienie obciążeń IoT - Microsoft Azure Well-Architected Framework | Microsoft Learn](#)

[Azure IoT reference architecture - Azure Reference Architectures | Microsoft Learn](#)

<https://azure.microsoft.com/pl-pl/solutions/iot/iot-technology-protocols/>

Większość systemów IoT używa wzorca architektury *połączonych produktów lub połączonych operacji*. Każdy wzorzec ma określone wymagania i ograniczenia w obszarach projektowych IoT.

- Architektury *połączonych produktów* koncentrują się na *ścieżce gorącej (przetwarzanie danych w czasie rzeczywistym)*.
 - Użytkownicy końcowi zarządzają produktami i korzystają z niej za pomocą aplikacji w czasie rzeczywistym.
 - Przykłady obejmują inteligentne ekspresy do kawy, inteligentne telewizory i inteligentne maszyny produkcyjne.
 - W tych rozwiązańach IoT konstruktory produktów udostępniają użytkownikom produktów połączone usługi.
- Architektury *operacji połączonych* koncentrują się na *cieplej(jak wyżej) lub zimnej ścieżce(przetwarzanie danych z opóźnieniem lub w partiach)*
 - Te rozwiązania analizują dane z wielu źródeł, zbierają szczegółowe informacje operacyjne, tworzą modele uczenia maszynowego i inicjują dalsze akcje dotyczące urządzeń i chmury.
 - zastosowanie do przedsiębiorstw i dostawców usług inteligentnych łączących istniejące maszyny i urządzenia. Przykłady obejmują inteligentne fabryki i inteligentne budynki.
 - Wykorzystanie urządzeń brzegowych, alertów i przetwarzania w chmurze.

1. Warstwa Urządzeń (Edge Layer)

1.1 Czujniki i Aktuatory

Czujniki są podstawowymi elementami IoT, które zbierają dane z otoczenia. Mogą mierzyć temperaturę, wilgotność, ciśnienie, ruch, światło, dźwięk i inne parametry fizyczne. Aktuatorzy, z kolei, wykonują określone akcje na podstawie zebranych danych, np. włączają oświetlenie, regulują temperaturę, otwierają zamki.

1.2 Mikrokontrolery i Mikroprocesory

Urządzenia IoT są zazwyczaj wyposażone w mikrokontrolery lub mikroprocesory, które przetwarzają dane z czujników. Mogą one wykonywać podstawowe obliczenia i podejmować decyzje bez konieczności komunikacji z centralnym serwerem, co redukuje opóźnienia i obciążenie sieci.

2. Warstwa Komunikacji (Network Layer)

2.1 Technologie Bezprzewodowe

- **Wi-Fi:** Używane w środowiskach domowych i biurowych do komunikacji o wysokiej przepustowości.
- **Bluetooth:** Popularne w urządzeniach noszonych (wearables) i krótkiego zasięgu.
- **Zigbee:** Energooszczędna technologia o krótkim zasięgu, używana w systemach automatyki domowej.
- **LoRa (Long Range):** Technologia o dużym zasięgu i niskim zużyciu energii, stosowana w rozległych sieciach IoT.
- **NB-IoT (Narrowband IoT):** Technologia komórkowa zaprojektowana dla urządzeń IoT o niskim zużyciu energii i dużym zasięgu.
- **5G:** Oferuje wysoką przepustowość i niskie opóźnienia, co jest kluczowe dla zaawansowanych zastosowań IoT, takich jak autonomiczne pojazdy i przemysł 4.0.

2.2 Protokoły Komunikacyjne

- **MQTT (Message Queuing Telemetry Transport):** Lekki protokół publikacji/subskrypcji, idealny do komunikacji z urządzeniami o ograniczonych zasobach.
- **CoAP (Constrained Application Protocol):** Protokół oparty na HTTP, zaprojektowany do urządzeń o ograniczonych zasobach.
- **HTTP/HTTPS:** Standardowy protokół internetowy, szeroko stosowany, ale mniej efektywny dla IoT.
- **AMQP (Advanced Message Queuing Protocol):** Protokół przesyłania wiadomości, zapewniający niezawodną komunikację.

3. Warstwa Przetwarzania Danych (Processing Layer)

3.1 Edge Computing

Przetwarzanie danych bezpośrednio na urządzeniach IoT lub w ich pobliżu. Pozwala to na szybkie reagowanie na zdarzenia i redukcję obciążenia sieci. Edge computing jest szczególnie

użyteczny w aplikacjach wymagających niskich opóźnień, takich jak autonomiczne pojazdy czy systemy bezpieczeństwa.

3.2 Fog Computing

Fog computing rozciąga przetwarzanie danych do lokalnych serwerów pośredniczących, które są bliżej urządzeń IoT niż chmura. To podejście balansuje między edge computing a cloud computing, oferując korzyści obu technologii, takie jak niskie opóźnienia i możliwość przetwarzania dużych ilości danych.

3.3 Cloud Computing

Centralne przetwarzanie danych w dużych centrach danych. Chmura oferuje ogromne zasoby obliczeniowe i przechowywania, co jest idealne do analizy dużych zbiorów danych, uczenia maszynowego i skomplikowanych obliczeń. Przykłady usług chmurowych to AWS, Google Cloud, Microsoft Azure.

4. Warstwa Aplikacji (Application Layer)

4.1 Interfejsy Użytkownika	przeglądarki internetowe czy interfejsy głosowe.	monitorowanie, kontrolę i optymalizację systemów IoT.
Interfejsy użytkownika umożliwiają użytkownikom końcowym interakcję z systemem IoT. Mogą to być aplikacje mobilne, panele kontrolne na komputerach,	4.2 Analiza Danych i Wizualizacja Oprogramowanie do przetwarzania zebranych danych i generowania wartościowych wniosków. Umożliwia to	4.3 Integracja z Innymi Systemami Zintegrowane z innymi systemami informatycznymi, takimi jak ERP (Enterprise Resource Planning).

5. Bezpieczeństwo w IoT

5.1 Szyfrowanie

- **TLS/SSL:** Szyfrowanie komunikacji internetowej.
- **DTLS (Datagram Transport Layer Security):** Szyfrowanie komunikacji opartej na UDP.

5.2 Uwierzytelnianie i Autoryzacja

- **OAuth 2.0:** Protokół autoryzacji umożliwiający bezpieczny dostęp do zasobów.
- **X.509:** Certyfikaty cyfrowe do uwierzytelniania urządzeń.

2. Architektura REST (Representational State Transfer). Charakterystyka, właściwości i zastosowania.

Kontekst (to wystarczy raz sobie przypomnieć)

Współczesne sieci komputerowe muszą zapewniać współpracę różnorodnego sprzętu komputerowego, technologii sieciowych oraz oprogramowania. Ze względu na stopień skomplikowania działania sieci komputerowych, różne funkcje sieci komputerowych należy podzielić na grupy realizowane w różnych warstwach. Liczbę warstw ustala się w ten sposób, aby **separować** zasadniczo różne funkcje i zadania.

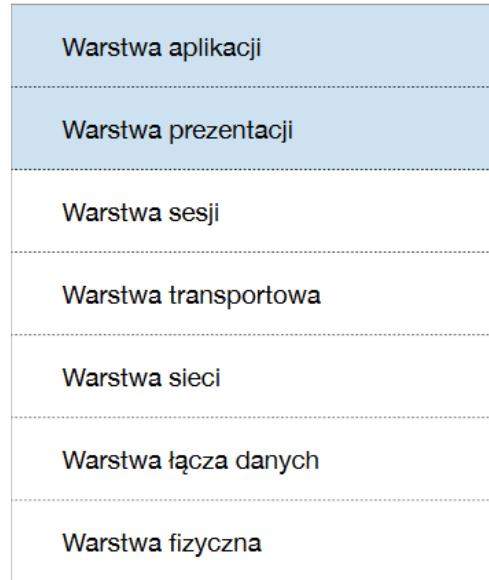
Zadaniem N-tej warstwy jest oferowanie usług warstwie wyższej (N+1) z jednoczesnym **izolowaniem** warstwy N+1-wszej od sposobu realizacji tych usług. Obiekty tej samej warstwy nawiązują połączenie **logiczne**. W rzeczywistości dane **nie** są przesyłane bezpośrednio między tymi samymi warstwami, ale za pomocą warstw niższych połączonych ze sobą fizycznymi interfejsami.

Protokół HTTP

HyperText Transfer Protocol to protokół warstwy aplikacji, początkowo zaprojektowany do przesyłania żądań udostępnienia strony www: dokumentu HTML. Określa formę **żądań** klienta dotyczących danych oraz formę **odpowiedzi** serwera na te żądania. Jest **bezstanowy**, czyli nie zachowuje żadnych informacji o poprzednich transakcjach z klientem.

Struktura żądania HTTP

- Metoda
GET, POST, PUT itp.
- Adres
np. *http://restapi.com/user/john*
- Dane nagłówkowe
User-agent, accept, keep-alive, if-modified-since, itp.



Czym jest REST?

REST (Representational State Transfer) styl architektury oprogramowania wywiedziony z doświadczeń przy pisaniu specyfikacji protokołu HTTP dla systemów rozproszonych.

REST opisuje sposób wykorzystania **istniejącego** protokołu HTTP. Termin został wprowadzony w 2000 roku przez Roya Fieldinga w jego pracy doktorskiej.

"Transfer Stanu Reprezentacyjnego ma wywoływać obraz tego, jak zachowuje się dobrze zaprojektowana aplikacja internetowa: sieć stron internetowych (wirtualna maszyna stanów), gdzie użytkownik przechodzi przez aplikację, wybierając linki (przejście stanów), co skutkuje

przeniesieniem do użytkownika następnej strony (reprezentującej następny stan aplikacji) i jej renderowaniem do użytku."

- Roy Fielding

Centralnym elementem REST są **zasoby**. Zasobem jest dowolna **identyfikowalna** informacja, która wskazuje na konkretny obiekt ze świata rzeczywistego (**rzecznik**). Zasób posiada zdefiniowaną i opisywalną **strukturę** wewnętrzną i może wchodzić w relacje z innymi zasobami (tak samo jak encje konceptualne w bazie danych)

Filozofia REST

Filozofia REST

REST wykorzystuje zapytania GET, POST, PUT, DELETE, które odwzorowują akcje CRUD (ang. Create, Read, Update and Delete)

- GET – pobranie obiektu
- POST – utworzenie obiektu
- PUT/PATCH – aktualizacja obiektu
- DELETE – skasowanie obiektu

REST opiera się na kilku kluczowych zasadach:

1. **Bezstanowość (Statelessness)**: Każde żądanie klienta do serwera musi zawierać wszystkie informacje niezbędne do zrozumienia i przetworzenia tego żądania. Serwer nie przechowuje żadnego kontekstu między różnymi żądaniami od tego samego klienta.
2. **Jednolity interfejs (Uniform Interface)**: REST definiuje ograniczony zestaw operacji (takich jak GET, POST, PUT, DELETE) i konwencji adresowania zasobów, co zapewnia jednolitość w komunikacji między komponentami systemu.
3. **Rozdział klienta i serwera (Client-Server Separation)**: Serwer i klient są oddzielnymi bytami, co umożliwia ich niezależny rozwój. Klient nie musi wiedzieć nic o logice serwera, a serwer nie musi wiedzieć nic o stanie klienta.
4. **Możliwość buforowania (Cacheability)**: Odpowiedzi serwera mogą być oznaczone jako buforowalne lub niebuforowalne. Dzięki temu klient może przechowywać kopie odpowiedzi i ponownie je wykorzystać, co poprawia wydajność.
5. **Warstwowość (Layered System)**: Architektura REST może być zorganizowana w warstwy, gdzie każda warstwa ma określoną funkcję. Klient nie musi znać ani rozróżniać warstw pośrednich.
6. **Kod na żądanie (Code on Demand)**: Opcjonalnie, serwer może dostarczać wykonujący się kod (np. skrypty JavaScript) do klienta, co pozwala na dynamiczne rozszerzanie funkcjonalności aplikacji.

Właściwości

- **Skalowalność**: Bezstanowość i możliwość buforowania wspierają wysoką skalowalność aplikacji RESTful.

- **Elastyczność:** Jasne rozdzielenie klienta i serwera pozwala na niezależny rozwój i modyfikacje poszczególnych komponentów.
- **Niezależność technologiczna:** REST opiera się na standardach HTTP, co sprawia, że może być używany niezależnie od technologii stosowanej po stronie klienta lub serwera.
- **Wysoka dostępność:** Dzięki możliwości buforowania odpowiedzi, RESTful API mogą zapewniać lepszą dostępność danych.

Zastosowania

REST znalazł szerokie zastosowanie w różnych dziedzinach:

1. **API dla aplikacji webowych:** RESTful API są powszechnie stosowane do komunikacji między frontendem a backendem aplikacji webowych.
2. **Usługi mobilne:** REST jest również popularny w tworzeniu backendów dla aplikacji mobilnych, gdzie zasoby są często dostępne przez internet.
3. **Integracje systemów:** RESTful API ułatwiają integrację różnych systemów i usług, umożliwiając wymianę danych między nimi.
4. **Internet of Things (IoT):** REST jest używany do komunikacji między urządzeniami IoT a serwerami, pozwalając na zarządzanie i monitorowanie urządzeń zdalnie.
5. **Chmura obliczeniowa:** Wiele usług chmurowych udostępnia swoje funkcje za pomocą RESTful API, co umożliwia łatwą automatyzację i integrację z innymi systemami.

3. Cele, konstrukcja i zastosowania Internetu rzeczy. Potencjalne korzyści i zagrożenia.

Cele IoT

1. **Zwiększenie efektywności operacyjnej:** Integracja urządzeń i systemów pozwala na zoptymalizowanie procesów przedsiębiorstw.
 - a. przykład: Jednym z pierwszych zastosowań Internetu rzeczy było podłączenie do internetu automatów z Coca-Colą. Maszyny informowały centralę o zasobach napoju.
2. **Poprawa jakości produktów i usług:** Wykorzystując urządzenia IoT można monitorować jakość produkcji na każdym etapie. Pozwala to na ciągłą kontrolę produkcji i ewentualne eliminowanie defektów w jak najszybszym tempie.
 - a. przykład: Kamery inspekcyjne w zakładzie produkującym płytki PCB. W razie wykrycia defektów, można przekierować płytki do utylizacji.
3. **Zbieranie i analiza danych:** Gromadzenie danych z różnorodnych źródeł w celu ich analizy i wyciągania wniosków.
 - a. przykład: Śledzenie pojazdów dostawczych. W ciężarówkach, które przewożą produkty w niskiej temperaturze, montowane są czujniki temperatury, GPS oraz poziomu paliwa. Dzięki temu można wcześnie wykryć problemy z dostawą, błędy kierowcy lub problemy z silnikiem maszyn transportowych.
4. **Innowacja i rozwój:** Wykorzystanie nowych technologii do wprowadzania innowacji produktowych i procesowych.
 - a. przykład: Ciągła aktualizacja oprogramowania. Poprzez połączenie z Internetem, pojazdy mogą otrzymywać regularne aktualizacje oprogramowania, które poprawiają ich działanie i bezpieczeństwo. Aktualizacja map w nawigacjach, naprawa sterowników.

Konstrukcja IoT

<https://azure.microsoft.com/pl-pl/solutions/iot/iot-technology-protocols/>

1. **Urządzenia i sensory:** Podstawowe elementy IoT, które zbierają dane z otoczenia (np. czujniki temperatury, wilgotności, ciśnienia).
2. **Komunikacja:** Urządzenia IoT łączą się za pomocą różnych protokołów komunikacyjnych (np. Wi-Fi, Bluetooth, Zigbee).
3. **Przetwarzanie danych:** Dane zbierane przez urządzenia są przetwarzane lokalnie lub w chmurze.
4. **Analiza danych:** Zebrane dane są analizowane w celu wyciągania wniosków i podejmowania decyzji.
5. **Interfejs użytkownika:** Użytkownicy mogą zarządzać i monitorować systemy IoT za pomocą aplikacji mobilnych, komputerowych lub specjalnych paneli kontrolnych.

Zastosowania IoT

1. **Inteligentne domy:** Automatyzacja urządzeń domowych, systemy zarządzania energią, inteligentne oświetlenie i systemy bezpieczeństwa.
2. **Przemysł 4.0:** Monitorowanie maszyn, zarządzanie produkcją, predykcyjne utrzymanie ruchu.
3. **Zdrowie:** Monitorowanie stanu zdrowia pacjentów, inteligentne urządzenia medyczne, systemy zarządzania lekami.
4. **Transport:** Systemy zarządzania flotą, inteligentne systemy transportowe, autonomiczne pojazdy.
5. **Miasta intelligentne:** Zarządzanie ruchem, monitorowanie jakości powietrza, zarządzanie odpadami.

Potencjalne korzyści i zagrożenia IoT

Potencjalne korzyści

1. **Efektywność i jakość:** Automatyzacja procesów pozwala na oszczędność czasu, zasobów i przekłada się na zwiększenie jakości.
 - a. Monitorowanie zasobów w czasie rzeczywistym.
2. **Lepsze decyzje biznesowe:** Analiza danych w czasie rzeczywistym umożliwia lepsze planowanie i podejmowanie decyzji.
 - a. Większe możliwości zbierania danych umożliwiają skuteczniejszą inteligencję biznesową.
3. **Bezpieczeństwo przesyłanych danych w rozwiązaniach przewodowych oraz przechowywanie części danych lokalnie:**
 - a. Przesyłanie danych w rozwiązaniach przewodowych chroni przed ich wykradzeniem w sieci.
 - b. Przechowywanie danych lokalnie.
4. **Oszczędność energii:** Inteligentne zarządzanie zasobami naturalnymi prowadzi do bardziej zrównoważonego wykorzystania energii.

Potencjalne zagrożenia

1. **Bezpieczeństwo i prywatność:** Dane z urządzeń IoT są podatne na ataki hakerów, a zbierane dane mogą być wykorzystywane w sposób nieuprawniony.
 - a. Wykradzenie danych z połączeń bezprzewodowych.
 - b. Magazynowanie danych w rozwiązaniach chmurowych producentów urządzeń.
2. **Złożoność systemów:** Skala i złożoność systemów IoT mogą prowadzić do trudności w zarządzaniu i utrzymaniu. Mogą pojawić się problemy z przetwarzaniem danych.
 - a. Przesyłanie zbyt dużej ilości danych -> problemy z pamięcią
 - b. Przesyłaniem małej ilości danych w bardzo krótkim czasie -> za mała moc obliczeniowa lub przesył sieci, lub zbyt mała pamięć podręczna
3. **Zależność od technologii:** Nadmierne poleganie na technologii może prowadzić do problemów w przypadku jej awarii.
 - a. Przy awarii kluczowego komponentu, system może przestać działać prawidłowo

-
-
-
4. **Koszty wdrożenia i utrzymania:** Wdrażanie zaawansowanych systemów IoT może być kosztowne, a utrzymanie i aktualizacje wymagają dodatkowych zasobów.
 - a. Szybka implementacja urządzeń bezprzewodowych, jednakże problemy z konserwacją tych urządzeń.
 - b. Kosztowna (czasowa i pieniężna) implementacja rozwiązań przewodowych.
5. **Regulacje i standardy:** Brak spójnych regulacji i standardów może prowadzić do problemów z integrowaniem urządzeń i systemów.
 - a. przykładowo dedykowane technologie do urządzeń danego producenta, takie jak standardy ładowania lub kompatybilności systemów wbudowanych.

4. Charakterystyka dobrze zaprojektowanej gry (ang. Gameplay).

Jasny cel i motywacja:

- Gra powinna mieć wyraźnie określone cele, które gracz musi osiągnąć. Cele te mogą być krótkoterminowe (np. ukończenie poziomu, zdobycie przedmiotu) lub długoterminowe (np. pokonanie głównego złoczyńcy, poznanie zakończenia historii).
- Motywacja do grania powinna być zróżnicowana i angażująca, np. ciekawa fabuła, rozwój postaci, zdobywanie nowych umiejętności lub nagród, czy rywalizacja z innymi graczami.

Dobrze określony poziom trudności:

- Trudność gry powinna stopniowo wzrastać, wraz z przewidzianym wzrostem umiejętności gracza, uczącego się w czasie mechanik gry. Zbyt łatwa gra szybko staje się nudna, a zbyt trudna może frustrować.

Intuicyjne i responsywne sterowanie:

- Sterowanie w grze powinno być łatwe do opanowania, ale jednocześnie oferować głębię, która pozwala na rozwijanie umiejętności.
- Gra powinna reagować na działanie gracza bez opóźnień, aby gracz czuł kontrolę nad swoją postacią

Różnorodność mechanik i zawartości:

- Gra powinna oferować różnorodne mechaniki, które utrzymują zainteresowanie gracza. Może to obejmować różne tryby rozgrywki czy nowe umiejętności.
- Zróżnicowana zawartość w postaci różnych wrogów, lokacji i przedmiotów, wzbogaca atrakcyjność gry.

Immersja:

- **Immersyjność** – proces „zanurzania” albo „pochłaniania” osoby przez rzeczywistość wirtualną
- Dobre napisana fabuła i ciekawi bohaterowie dodają głębi i emocjonalnego zaangażowania.
- Spójny i szczegółowy świat przedstawiony.

Plynność, wydajność, optymalizacja, stan techniczny:

- Gra powinna działać płynnie, bez długich czasów ładowania czy błędów (bugów), które mogą wybić gracza z rytmu czy nawet uniemożliwić dalsze korzystanie z gry.
- Gra (w miarę możliwości) powinna być zoptymalizowana pod daną platformę aby zapewnić jak najlepszy stan techniczny na określonych urządzeniach

Gra wieloosobowa:

- Umożliwienie interakcji z innymi graczami poprzez tryby wieloosobowe (współpracy lub rywalizacyjne), rankingi zwiększą zaangażowanie.

Wsparcie długoterminowe:

- Regularne aktualizacje, naprawa błędów, wydarzenia specjalne i dobra komunikacja z klientami może przyczynić się do długofalowych zysków i polepszenia renomii firmy.

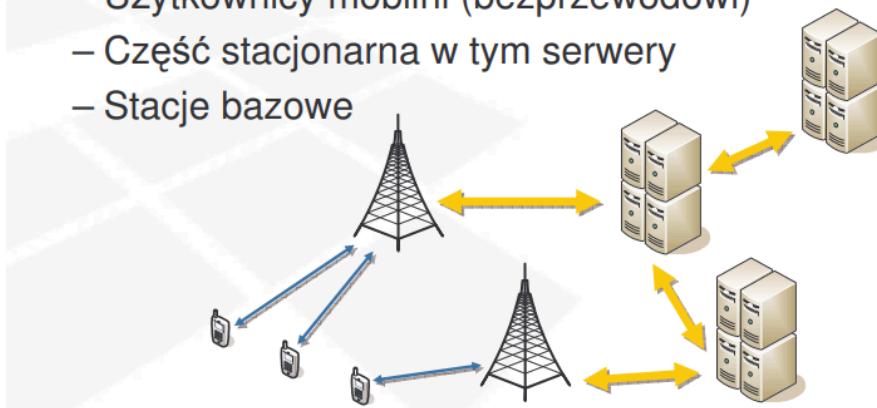
5. Charakterystyka informatycznych systemów mobilnych.

Główna idea przetwarzania mobilnego sprowadza się do dwóch angielskich słów „**ANYTIME, ANYWHERE**” (gdziekolwiek jesteś, o każdej porze masz dostęp do danych).

Na realizację tej koncepcji złożył się między innymi olbrzymi postęp technologiczny, który spowodował dostępność urządzeń nawigacyjnych i komputerów przenośnych, wyposażonych w różnorakie systemy łączności bezprzewodowej.

Powstaje duża liczba aplikacji mających zapewnić użytkownikowi mobilnemu dostęp do informacji porównywalny ze stałym miejscem pracy. Pracownicy firm nie są już przywiązani do swoich stanowisk, ich biuro jest tam, gdzie aktualnie się znajdują.

- Komponenty systemu mobilnego
 - Użytkownicy mobilni (bezprzewodowi)
 - Część stacjonarna w tym serwery
 - Stacje bazowe



System mobilny w ogólności składa się z elementów stałych i ruchomych:

- **elementy ruchome (MOBILE UNITS lub MOBILE HOSTS),**
 - Użytkownik najczęściej będący w ruchu
 - Korzysta najczęściej z połączenia bezprzewodowego
 - Wysoka różnorodność użytkowników
- **komputery stacjonarne (FIXED HOSTS)** wchodzące w skład systemu mobilnego, połączone z tradycyjną siecią przewodową zapewniającą duże przepustowości,
- **stacje bazowe (BASE STATIONS lub MOBILE SUPPORT STATIONS)**. Komputery stacjonarne nie mogą bezpośrednio łączyć się z użytkownikami mobilnymi, wobec tego w sieć tą włączono odpowiedzialne za łączność z nimi stacje bazowe.

Stacje bazowe są interfejsem pomiędzy komputerami stacjonarnymi (FIXED HOSTS) a użytkownikami w terenie, którzy tworzą bezprzewodową sieć.

System mobilny, podobnie jak klasyczny system rozproszony, składa się z węzłów, na których odbywa się przetwarzanie, połączonych siecią komunikacyjną – najczęściej bezprzewodową. Różni się od tradycyjnego systemu rozproszonego tym, że użytkownik zmienia swoje położenie, co powoduje ciągłą zmianę topologii sieci.

Cechy systemów mobilnych:

- **Brak wspólnej pamięci** - Każdy węzeł dysponuje własną, lokalną pamięcią.
- **Tylko wymiana wiadomości (message passing)** - Komunikacja między węzłami odbywa się wyłącznie poprzez wymianę wiadomości. Oznacza to, że węzły wysyłają sobie nawzajem komunikaty zawierające dane i instrukcje, zamiast odwoływać się bezpośrednio do wspólnej pamięci.
- **Brak globalnego zegara** - Nie istnieje centralny zegar; podejmowane są jedynie próby synchronizacji zegarów lokalnych.
- **Asynchronizm przetwarzania** - Przetwarzanie odbywa się w sposób całkowicie asynchroniczny, co oznacza, że węzły wykonują swoje zadania niezależnie od siebie, bez potrzeby czekania na zakończenie operacji innych węzłów.
- **Asynchronizm komunikacji** - Komunikacja między węzłami jest również asynchroniczna. Węzły wysyłają i odbierają wiadomości niezależnie od siebie, bez potrzeby synchronizacji czasowej.
- **Przetwarzanie rozproszone** - Przetwarzanie danych odbywa się w sposób rozproszony. Zadania są podzielone między różne węzły, a każdy węzeł przetwarza swoją część danych. Węzły współpracują ze sobą poprzez wymianę wiadomości.
- **Problematyczna prezentacja danych** - Małe ekranы, różne interfejsy sprzętowe (brak myszki, klawiatury), specyficzne warunki pracy (las, pole bitwy, bocznica kolejowa) itd. wymuszają inne podejście do zobrazowania danych użytkownikowi.
- **Pozycjonowanie użytkowników** - Jednym z poważniejszych problemów w systemach mobilnych jest dokładne pozycjonowanie uzytkownika, a takie jego nawigacja w oparciu o dane przestrzenne zapisane w odpowiednich formatach:
 - systemy satelitarne (GPS, GLONAS, GALILEO),
 - nawigacja zliczeniowa,
 - pozycjonowanie w sieciach GSM,
 - nawigacja w budynkach

Zalety systemów mobilnych:

- 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.

Bezprzewodowość nie jest tożsama z mobilnością - Tak naprawdę możliwe są wszystkie cztery przypadki (stacjonarny/mobilny, przewodowy/bezprzewodowy):

Rodzaje klientów systemów mobilnych:

Cienki klient:

- Aplikacja zainstalowana na komputerze użytkownika mobilnego
- Brak lokalnego przetwarzania i lokalnie zapisanych danych
- Tylko interfejs użytkownika
- Wymaga ciągłej interakcji z serwerem
- Przykłady aplikacji „cienkich klientów”
- Definicja sprzętowa - uboga platforma

Gruby klient

- Aplikacja zainstalowana na komputerze użytkownika mobilnego
- Definicja sprzętowa - bogata platforma
- Lokalne przetwarzanie i lokalnie zapisywane dane
- Nie wymaga ciągłej interakcji z serwerem

6. Definicja i modele przetwarzania dużych danych (ang. Big Data).

Big Data odnosi się do niezwykle dużych, złożonych i różnorodnych zbiorów danych, które są generowane z różnych źródeł i których tradycyjne metody przetwarzania i analizy danych nie są w stanie efektywnie obsługiwać. Wykorzystanie technologii Big Data umożliwia przechwytywanie, zarządzanie i analizowanie tych danych w celu uzyskania wartościowych informacji i wspierania podejmowania decyzji.

Cechy Big Data:

- **Volume (Objętość):** Ogromna ilość danych generowanych każdego dnia. Przykłady to dane pochodzące z mediów społecznościowych, czujników IoT, transakcje online.
- **Velocity (Szybkość):** Szybkość generowania i przetwarzania danych. Dane napływają z dużą prędkością i muszą być przetwarzane niemal w czasie rzeczywistym.
- **Variety (Różnorodność):** Różne typy danych – strukturalne, półstrukturalne i niestrukturalne. Przykłady to tekst, obraz, video, logi serwerów.
- **Veracity (Wiarygodność):** Zmiennaść i niepewność danych, co wymaga mechanizmów weryfikacji i czyszczenia danych, aby uzyskać dokładne wyniki.
- **Value (Wartość):** Możliwość uzyskania wartościowych informacji z danych poprzez odpowiednie analizy

Modele przetwarzania dużych danych:

Wersja 1:

MapReduce - Jeden z najbardziej znanych modeli przetwarzania dużych danych, zaprojektowanym przez Google. Jest to model do przetwarzania problemów możliwych do zrównoleglenia na dużych zbiorach danych, z wykorzystaniem dużej liczby komputerów (węzłów), wspólnie nazywanych klastrem (jeśli wszystkie węzły znajdują się w tej samej sieci lokalnej i używają podobnego sprzętu) lub gridem (jeśli węzły są rozproszone geograficznie i administracyjnie oraz używają bardziej zróżnicowanego sprzętu).

MapReduce zazwyczaj składa się z trzech kroków (mimo że jest ogólnie uważany za kombinację operacji/funkcji Map i Reduce).

1. **Map:** Każdy węzeł roboczy stosuje funkcję map do lokalnych danych i zapisuje wyniki do tymczasowej pamięci. Węzeł główny zapewnia, że tylko jedna kopia redundantnych danych wejściowych jest przetwarzana.
2. **Shuffle:** Węzły robocze ponownie rozdzielają dane na podstawie kluczy wyjściowych (wygenerowanych przez funkcję map), tak aby wszystkie dane należące do jednego klucza znajdowały się na tym samym węźle roboczym.
3. **Reduce:** Węzły robocze teraz przetwarzają każdą grupę danych wyjściowych, według klucza, równolegle.

W skrócie:

MapReduce składa się z procedury mapowania, która wykonuje filtrowanie i sortowanie (takie jak sortowanie uczniów według imienia w kolejkach, po jednej kolejce dla każdego imienia), oraz metody redukcji, która wykonuje operację podsumowania (taką jak zliczanie liczby uczniów w każdej kolejce, uzyskując częstotliwość imion).

<http://informatyka.wroc.pl/node/424?page=0.0>

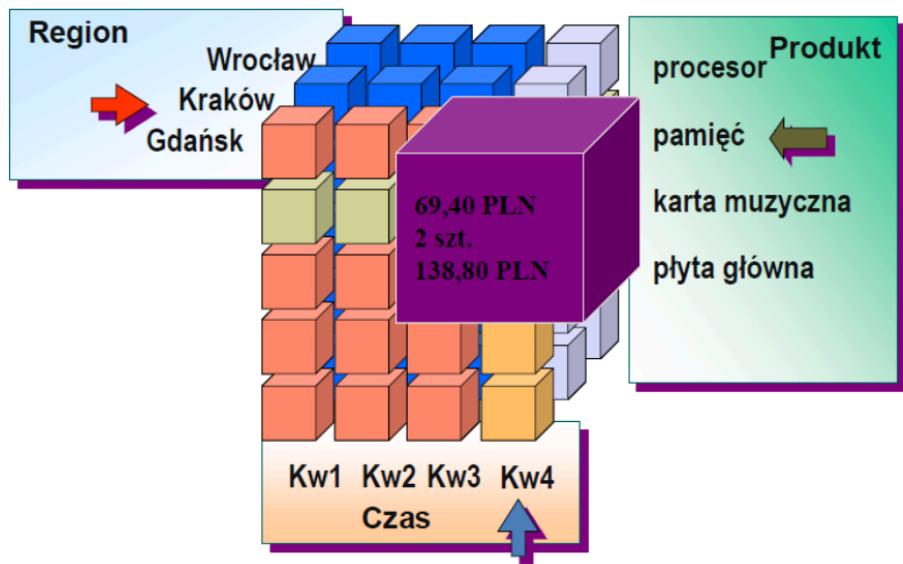
Wersja 2:

- **Modele przetwarzania wsadowego (Batch Processing)** zajmują się ogromnymi ilościami danych historycznych, wykonując równolegle obliczenia zgodnie z określonym harmonogramem (wiąże się to z opóźnieniem od minut do godzin).
 - **Apache Hadoop:** Ekosystem oprogramowania do rozproszonego przechowywania i przetwarzania dużych zbiorów danych (Oparty na **MapReduce**).
- **Modele przetwarzania strumieniowego** zajmują się danymi w czasie rzeczywistym, które powinny być przetwarzane natychmiast po ich nadejściu (wiąże się to z opóźnieniem od milisekund do sekund).
 - **Apache Storm:** Otwarty system obliczeniowy czasu rzeczywistego. Storm ułatwia przetwarzanie nieograniczonych strumieni danych.
 - **Apache Spark Streaming:** Rozszerzenie podstawowego Apache Spark umożliwiające przetwarzanie strumieni danych na żywo.

Wersja 3 (Maleszka):

1. **OLTP (Online Transaction Processing):** system przetwarzania transakcji online, który obsługuje codzienne operacje biznesowe w czasie rzeczywistym. Jest zaprojektowany do szybkiego i efektywnego przetwarzania dużej liczby krótkich transakcji, takich jak dodawanie, aktualizowanie i usuwanie rekordów w bazie danych. OLTP zapewnia dostęp do aktualnych danych i wspiera operacyjne funkcje organizacji.
2. **OLAP (Online Analytical Processing):** technologia analityczna stosowana do przetwarzania i analizy dużych ilości danych z różnych perspektyw. Jest przeznaczona głównie do wykonywania złożonych zapytań analitycznych, eksploracji danych i

raportowania. Systemy OLAP umożliwiają użytkownikom dokładne badanie danych poprzez wielowymiarowe widoki danych (kostki), co wspiera podejmowanie decyzji biznesowych.



OLTP

- Przechowywane dane są zorientowane procesowo (np. wystawianie faktury)
- Stosunkowo niewielkie rozmiary baz danych (kilka gigabajtów)
- Przechowywane są dane bieżące bez konieczności gromadzenia danych historycznych
- Realizowana jest duża ilość prostych zapytań
- Przechowywane są dane elementarne
- Realizowane są operacje wstawiania, modyfikowania i usuwania danych

OLAP

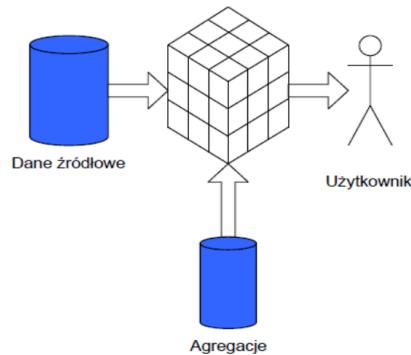
- Przechowywane dane są zorientowane tematyczne (np. sprzedaż produktów, stany zapasów)
- Bardzo duże ilości gromadzonych danych (wiele terabajtów)
- Przechowywane są dane bieżące i historyczne
- Bardzo złożone zapytania operujące na wielkich ilościach danych
- Przechowywane są dane elementarne i zagregowane (sumy, średnie)
- Wykonywane są głównie operacje dopisywania nowych danych – praktycznie nie wykonuje się operacji modyfikowania danych

Rodzaje OLAP:

- **ROLAP (Relational OLAP)**: Wykorzystuje bazę danych relacyjną do przechowywania i zarządzania danymi wielowymiarowymi. Dane są

ROLAP

- dane i agregacje są przechowywane w RSZBD
- najwolniejsze odpowiedzi na zapytania
- zwykle najwolniejsze przetwarzanie
- można tworzyć indeksowane widok
- najbardziej użyteczny tryb dla dużej liczby danych
- wspomaga rozwiązania real-time OL

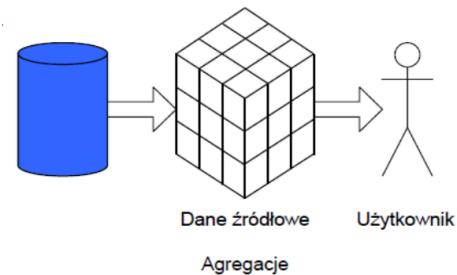


przechowywane w postaci tabelarycznej, a zapytania OLAP są tłumaczone na zapytania SQL do bazy danych relacyjnej.

- **MOLAP (Multidimensional OLAP)**: Dane wielowymiarowe są przechowywane w specjalnie zaprojektowanych, optymalizowanych dla OLAP, strukturach danych, takich jak kostki danych (cube) czy też wielowymiarowe tabele.

MOLAP

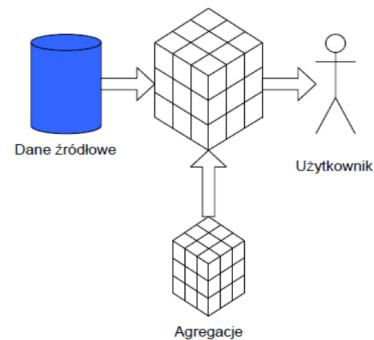
- wielowymiarowe OLAP
- wielowymiarowy format danych i agregacji
- zapytania wykonują się szybko
- wymaga największej przestrzeni na d



- **HOLAP (Hybrid OLAP):** Kombinuje cechy ROLAP i MOLAP. Dane szczegółowe przechowywane są w bazie danych relacyjnej, podczas gdy dane zagregowane lub często używane są przechowywane w strukturach MOLAP dla szybszego dostępu.

HOLAP

- dane zarządzane przez RSZDB
- agregacje tworzone w formacie wielowymiarowym
- dobry wybór, gdy przestrzeń na dysku jest wąskim gardem
- dobra wydajność dla częstych odwołań do agregacji



3. **Hurtownia danych (Data Warehouse):** tematycznie zorientowana, zintegrowana, chronologiczna i trwała kolekcja danych do wspomagania procesów podejmowania decyzji.

Główne cechy hurtowni danych to:

- Optymalizacja dla zapytań analitycznych i raportowania.
- Integracja danych z różnych źródeł, często pochodzących z systemów operacyjnych przedsiębiorstwa.
- Czasowa konsolidacja danych (np. dane historyczne).
- Zastosowanie ETL (Extract, Transform, Load) do ładowania i aktualizacji danych.

Aspekt	HD	OLTP
Tematycznie zorientowana	Ścisły podział na obszary informacyjne (tematyczne); lokalizacja danych zależy od tematyki	Aplikacje projektowane są wokół procesów oraz funkcji, które wymagają różnych danych
Zintegrowana	Spójność danych, kluczy, synonimy, homonimy, rekordy logiczne...	Jedno źródło, bez integracji
Chronologiczna	Dane są znakowane czasowo, szerszy horyzont czasowy np. 5 lat, historyzacja zmian	Dane są znakowane czasowo w wąskim horyzoncie np. 30-90 dni
Trwała	Dane nie są modyfikowane przez użytkowników, odczyt danych	Transakcje (insert, delete, update)
Cel	Wspomaganie procesów podejmowania decyzji (raczej na poziomach: taktycznym i strategicznym)	Ewidencja transakcji, wspieranie operacyjnego poziomu zarządzania (również podejmowania decyzji na tym poziomie)

Typy architektur hurtowni danych:

Architektura scentralizowana:

- Materialna centralna hurtownia danych.
- Dane fizycznie przechowywane w jednym centralnym miejscu.
- Potencjalnie niska rozproszenie magazynów danych operacyjnych.

Architektura federacyjna:

- Wirtualna centralna hurtownia danych jako niezmaterializowana perspektywa.
- Wspólny schemat logiczny i pojęciowy.
- Dane fizycznie przechowywane w różnych magazynach danych operacyjnych.
- Możliwe problemy z wydajnością z powodu rozproszenia danych.

Architektura warstwowa:

- Wiele warstw hurtowni tematycznych zawierających różne poziomy agregacji danych.
- Każda warstwa zmaterializowana, co może poprawić wydajność.
- Optymalizacja wielkości danych w poszczególnych warstwach tematycznych.

4. **Jeziorko danych (Data Lake):** Elastyczna, skalowalna i bezstrukturalna przestrzeń do przechowywania dużych ilości danych w oryginalnej formie, nieprzetworzonej i niewykrytej wcześniej. Główne cechy jeziora danych to:
- Przechowywanie różnorodnych typów danych: strukturalnych, półstrukturalnych i nieustrukturyzowanych.
 - Brak ściśle zdefiniowanej struktury danych przed ładowaniem.
 - Możliwość przechowywania danych w oryginalnej postaci przed przetworzeniem lub analizą.

- Umożliwienie eksploracji danych oraz analizy za pomocą różnych narzędzi i technologii analitycznych.

7. Definiowanie schematów dokumentów XML. Różnice między DTD, a XML-Schema

XML (ang. Extensible Markup Language, rozszerzalny język znaczników) – uniwersalny język znaczników przeznaczony do reprezentowania różnych danych w strukturalizowany sposób. To język znaczników i format pliku do przechowywania, przesyłania i rekonstrukcji dowolnych danych.

DTD (ang. document type definition) – rodzaj dokumentu definiujący formalną strukturę dokumentów XML, HTML, XHTML lub innych pochodzących z rodziny SGML lub XML. Definicje DTD mogą być zawarte w pliku dokumentu, którego strukturę definiują, przeważnie jednak zapisane są w osobnym pliku tekstowym, co pozwala na zastosowanie tego samego DTD dla wielu dokumentów. Zazwyczaj DTD definiuje każdy dopuszczalny element dokumentu, jego zbiór atrybutów i dopuszczalne wartości. DTD określa także zagnieżdżanie i wymagalność poszczególnych elementów w dokumencie. W praktyce DTD przeważnie składa się z definicji ELEMENT i definicji ATTLIST.

W praktyce ze względu na małe możliwości obecnie DTD jest wypierane przez nowocześniejsze XML Schema, które posiada znacznie większe możliwości oraz nie wymaga stosowania dodatkowej, nie-XML-owej składni.

XML Schema stanowi alternatywę dla DTD, przy czym posiada znacznie większe możliwości.

XML Schema jest strukturą XML, w odróżnieniu od DTD niebędącego częścią standardu XML. Dokumenty zawierające definicje XML Schema zapisuje się zwykle w plikach z rozszerzeniem.xsd.

W schemacie na najwyższym poziomie, tzn. jako dzieci elementu głównego schema, występują definicje:

- elementów i atrybutów,
- typów (prostych i złożonych),
- grup (elementów i atrybutów),
- notacji.

Ponadto na początku dokumentu mogą wystąpić elementy związane z łączeniem wielu schematów, a w dowolnym miejscu anotacje, które powinny stanowić dokumentację schematu i, jeśli nie ma tego w osobnym dokumencie, określać znaczenie poszczególnych elementów i atrybutów.

Jakie są problemy z DTD i ograniczenia DTD:

- brak mechanizmu 'typ danych' - tekstowa zawartość elementów nie może być ograniczona,

- brak możliwości określania typów i wartości atrybutów, a także kolejności atrybutów,
- niemożliwa jest zmiana kolejności elementów,
- deklaracja atrybutów i elementów nie mogą zależeć od kontekstu: często potrzebny jest możliwość użycia takiej deklaracji, która będzie zależna od wystąpienia lub braku atrybutu,
- dane znakowe nie mogą być związane z wyrażeniami regularnymi,
- DTD nie udostępnia mechanizmów związanych z modułami, ponownym użyciem typów - trudne jest operowanie dużymi DTD,
- mechanizm ID i IDREF jest zbyt prosty i nie wystarcza w pewnych sytuacjach,
- DTD nie wspiera przestrzeni nazw,
- DTD nie jest dokumentem XML

Przykładowo dla tak skonstruowanego DTD umieszczonego w pliku o nazwie pracownicy.dtd:

```
<!ELEMENT pracownicy (pracownik+)
<!ELEMENT pracownik (daneOsobowe, dzia&ł?)>

<!ELEMENT daneOsobowe (imie, drugieimie?, nazwisko)
<!ELEMENT imie (#PCDATA)
<!ELEMENT drugieimie (#PCDATA)
<!ELEMENT nazwisko (#PCDATA)

<!ELEMENT dzia&ł (#PCDATA)
```

prawidłowym dokumentem XML jest:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE pracownicy SYSTEM "pracownicy.dtd">
<pracownicy>
  <pracownik>
    <daneOsobowe>
      <imie>Izabela</imie>
      <nazwisko>Nowakowska-Jasnorzębska</nazwisko>
    </daneOsobowe>
  </pracownik>
  <pracownik>
    <daneOsobowe>
      <imie>Zygflyd</imie>
      <drugieimie>Zenobiusz</drugieimie>
      <nazwisko>Wawrzyniak</nazwisko>
    </daneOsobowe>
    <dzia&ł>Public Relations</dzia&ł>
  </pracownik>
</pracownicy>
```

Przykład 1. Plik XML.

```
<?xml version="1.0" encoding="UTF-8"?>
<pracownicy xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="pracownicy.xsd">
    <pracownik>
        <imie>Alojzy</imie>
        <nazwisko>Filipski</nazwisko>
        <rocznik>1967</rocznik>
    </pracownik>
    <pracownik>
        <imie>Dawid</imie>
        <nazwisko>Raskolnikow</nazwisko>
        <rocznik>1982</rocznik>
    </pracownik>
</pracownicy>
```

Atrybuty `xmlns:xsi` oraz `xsi:noNamespaceSchemaLocation` informują, że w pliku `pracownicy.xsd` znajduje się zbiór zasad dla tego pliku XML i jest on w formacie XML Schema.

Przykład 2. Plik XML Schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<xss:schema xmlns:xss="http://www.w3.org/2001/XMLSchema">
    <xss:element name="pracownicy">
        <xss:complexType>
            <xss:sequence>
                <xss:element ref="pracownik" maxOccurs="unbounded"/>
            </xss:sequence>
        </xss:complexType>
    </xss:element>
    <xss:element name="pracownik">
        <xss:complexType>
            <xss:sequence>
                <xss:element name="imie" type="xs:string"/>
                <xss:element name="nazwisko" type="xs:string"/>
                <xss:element name="rocznik" type="xs:short"/>
            </xss:sequence>
        </xss:complexType>
    </xss:element>
</xss:schema>
```

<https://nowaker.net/post/xml-instance-xml-schema.html>

S.NO.	DTD	XSD
1.	DTD are the declarations that define a document type for SGML.	XSD describes the elements in a XML document.
2.	It doesn't support namespace.	It supports namespace.
3.	It is comparatively harder than XSD.	It is relatively more simpler than DTD.
4.	It doesn't support datatypes.	It supports datatypes.
5.	SGML syntax is used for DTD.	XML is used for writing XSD.
6.	It is not extensible in nature.	It is extensible in nature.
7.	It doesn't give us much control on structure of XML document.	It gives us more control on structure of XML document.
8.	It specifies only the root element.	Any element which is made global can be done as root as markup validation.
9.	It doesn't have any restrictions on data used.	It specifies certain data restrictions.
10.	It is not much easy to learn .	It is simple in learning.
11.	File here is saved as .dtd	File in XSD is saved as .xsd file.
12.	It is not a strongly typed mechanism.	It is a strongly typed mechanism.
13.	It uses #PCDATA which is a string data type.	It uses fundamental and primitive data types.

8. Harmonogramowanie projektów informatycznych

[z wykładu 4 ZPI Trawińskiego]

Harmonogramowanie to wybór i zastosowanie najbardziej odpowiednich technik dla stworzenia programu lub sekwencji działań nakierowanych na dotrzymanie kluczowych terminów i celów projektu. Obejmuje ono również rozpoznanie i uwzględnienie w programie:

- etapów (faz), dat, punktów kontrolnych (kamieni milowych),
- zapotrzebowania i dostępności zasobów,
- sekwencji zadań i zależności pomiędzy zadaniami,
- limitów czasowych,
- ograniczeń wewnętrznych i zewnętrznych.

Techniki harmonogramowania:

- wykres Gantt'a lub wykresy paskowe,
- diagramy sieciowe (zadanie na wierzchołku/łuku),
- metoda ścieżki krytycznej,
- harmonogramowanie zasobów,
- harmonogramowanie punktów kontrolnych (kamieni milowych).

Etapy opracowania harmonogramu

- Podział projektu na czynności
- Określanie sekwencji czynności
- Przydział wykonawców i innych zasobów
- Szacowanie czasu trwania czynności
- Tworzenie diagramu sieciowego
- Tworzenie harmonogramu
- Kontrola harmonogramu

1. **Planowanie zarządzania harmonogramem** obejmuje określenie zasad, procedur i dokumentacji, które będą używane do planowania, wykonywania i kontrolowania harmonogramu projektu. Głównym rezultatem tego procesu jest plan zarządzania harmonogramem.
2. **Definiowanie czynności** obejmuje identyfikację konkretnych czynności, które członkowie zespołu projektowego i interesariusze muszą wykonać, aby uzyskać rezultaty projektu. Czynność lub zadanie to element pracy zwykle znajdujący się w strukturze podziału pracy (WBS), który ma oczekiwany czas trwania, koszty i wymagania dotyczące zasobów. Główne wyniki tego procesu to lista czynności, atrybuty czynności, lista punktów kontrolnych i aktualizacje planu zarządzania projektem.
3. **Sekwencjonowanie czynności** obejmuje identyfikację i dokumentowanie relacji między czynnościami w projekcie. Główne wyniki tego procesu obejmują sieci działań i aktualizacje dokumentów projektowych.

- 4. Szacowanie zasobów** obejmuje oszacowanie, ile zasobów - ludzi, sprzętu i materiałów - powinien zespół projektowy użyć do wykonania czynności. Głównymi wynikami tego procesu są wymagania dotyczące zasobów, struktura podziału zasobów i aktualizacje dokumentów projektu.
- 5. Szacowanie czasu trwania czynności** obejmuje oszacowanie liczby jednostek czasu potrzebnych do wykonania poszczególnych czynności. Wyniki obejmują szacunki czasu trwania czynności i aktualizacje dokumentów projektowych.
- 6. Opracowanie harmonogramu** obejmuje analizę sekwencji czynności, oszacowania zasobów i oszacowania czasu trwania czynności. Dane wyjściowe obejmują plan bazowy harmonogramu, harmonogram projektu, dane harmonogramu, kalendarze projektów, aktualizacje planu zarządzania projektem i aktualizacje dokumentów projektu.
- 7. Kontrolowanie harmonogramu** polega na kontrolowaniu i zarządzaniu zmianami w harmonogramie projektu. Dane wyjściowe obejmują informacje o wydajności pracy, prognozy harmonogramu, żądania zmian, aktualizacje planu zarządzania projektem, aktualizacje dokumentów projektu i aktualizacje zasobów procesu organizacyjnego.

Tworzenie harmonogramu przedsięwzięcia:

- podziel przedsięwzięcie na zadania i oszacuj czas i zasoby konieczne do wykonania każdego z nich,
- zorganizuj zadania równolegle aby najlepiej wykorzystać ludzi i zasoby,
- minimalizuj zależności pomiędzy zadaniami, aby najlepiej wykorzystać czas przypadający na każde z nich,
- wszystko zależy od umiejętności i intuicji zarządzającego projektem.



Proces tworzenia harmonogramu przedsięwzięcia



9. Heurystyki użyteczności Nielsena

Analiza heurystyczna to metoda oceny interfejsów, której celem jest identyfikacja problemów w ich użyteczności. Heurystyki Nielsena opisują elementy i zasady projektowania interfejsów, które najczęściej generują błędy.

1. Pokazuj status systemu

System powinien zawsze informować użytkownika w jakim miejscu/sekcji się znajduje, a także o swoim statusie czy akcji, którą może następnie wykonać za pomocą odpowiedniego feedbacku w odpowiednim czasie. Na przykład, podczas ładowania strony powinna być wyświetlana animacja ładowania. Do podstawowych elementów, które informują użytkownika o miejscu, w którym się znajduje oraz statusie podejmowanych akcji należą np. znacznik title, nagłówek H1, oznaczenia aktywnych kategorii menu, breadcrumbs.

2. Zachowaj zgodność pomiędzy systemem a rzeczywistością

System powinien używać języka i terminologii zrozumiałej dla użytkowników, opartych na rzeczywistych konwencjach i realiach. Ikony, słownictwo i przepływy pracy powinny być intuicyjne. Przykładem mogą być tutaj choćby muzyczne serwisy streamingowe, w których przyciski funkcji odtwarzacza odpowiadają tym, jakie są doskonale znane z używanych od dekad magnetofonów. Warto trzymać się przyjętych standardów, np. umieszczenia wewnętrznej wyszukiwarki po prawej stronie menu czy oznaczania linków niebieskim kolorem czcionki i podkreśleniem.

3. Daj użytkownikowi pełną kontrolę

Użytkownicy serwisów internetowych czują się pewniej, jeśli mają pełną kontrolę nad tym, co robią w czasie wizyty na stronie. Dlatego powinni mieć możliwość swobodnych działań, np. **łatwiej zmiany kategorii sklepu, usunięcia produktów z koszyka czy rezygnacji z dalszych kroków na ścieżce zakupowej**, czyli innymi słowy rezygnacji z zakupu.

4. Trzymaj się standardów i zachowaj spójność

Użytkownicy nie powinni się zastanawiać, czy różne słowa, sytuacje lub działania oznaczają to samo. Poprawne projektowanie musi opierać się na **spójnych standardach**, które pozwolą użytkownikowi łatwo uczyć się i zrozumieć, jak poruszać się po naszej witrynie.

Spójność zewnętrzna to ogólnie przyjęte standardy, które sprawdzają się niemal we wszystkich serwisach. Jej elementy to m.in.:

- podlinkowanie do strony głównej logotypu, który umieszczony jest w lewym górnym rogu lub u góry na środku strony;
- wyszukiwarka i symbol koszyka w prawym górnym rogu strony;
- główne menu, umieszczone w górnej lub bocznej części strony, a na urządzeniu mobilnym po kliknięciu ikony trzech poziomych pasków.

Z kolei spójność wewnętrzna to **utrzymanie w całym serwisie jednej konwencji stylistycznej**, np. jednakowych fontów, spójnej kolorystyki, podświetleń przycisków po najechaniu na nie kursem, umiejscowienia opisów czy sposobie informowania o następstwach czynności i błędach.

5. Zapobiegaj błędom

Staraj się, aby możliwie najsłuszniej zapobiec popełnieniu błędu przez użytkownika strony. W tym celu stosuje się m.in. **informacje o wyprzedaniu produktu lub braku dostępności wybranego wariantu** (np. informacja „Wyprzedane” lub przycisk „Powiadom mnie o dostępności”). Bardzo ważne jest też ułatwienie w czasie wpisywania przez użytkownika wymaganych danych, np. poprzez przygotowanie formatek na wpisanie adresu, kodu pocztowego czy numeru telefonu, a także wyświetlanie instrukcji i komunikatów o błędach w przejrzysty, dokładny i zrozumiały sposób.

6. Pokaż, zamiast zmuszać do zapamiętania

Ogranicz obciążenie pamięci użytkownika, czyniąc obiekty, akcje i opcje widocznymi. Użytkownik nie powinien musieć pamiętać informacji z jednej części dialogu do drugiej. Warto umieszczać sekcje tj. „Ostatnio przeglądane”, czy opcje podglądu koszyka.

7. Elastyczność i efektywność

System powinien służyć zarówno nowym, jak i doświadczonym użytkownikom. Skróty, akceleratory i dostosowania mogą zwiększyć efektywność dla bardziej zaawansowanych użytkowników. Jest to też ułatwienie i skrócenie wykonywania wszystkich czynności w obrębie serwisu. Kilka funkcjonalnych przykładów to m.in.:

- checkbox „Zaznacz wszystkie” przy wyrażeniu zgód;
- możliwość skopiowania koszyka z ostatnich zakupów;
- możliwość wybrania wielu filtrów i kliknięcia przycisku „Filtruj” zamiast automatycznego ładowania się filtrów po wybraniu każdego z osobna.

8. Dbaj o estetykę i umiar

„Mniej znaczy więcej” - formularze zamówienia czy rejestracji powinny zawierać jedynie najważniejsze informacje i niezbędne opcje, podane w estetycznej i przejrzystej formie.

9. Zapewnij skutecną obsługę błędów

Nawiązanie do piątej heurystyki. Jeśli pomimo starań użytkownik popełnił błąd, należy mu go jasno wskazać i podpowiedzieć rozwiązanie. Przykładem będą tu jasne komunikaty, np. „**Login XXX jest już zajęty**” i sugerowanie wyboru loginu XXX11. Podobnie w przypadku haseł – jeśli hasło proponowane przez użytkownika nie spełnia warunków, warto jasno mu podpowiedzieć, co powinien do niego dodać, np. „**Hasło musi zawierać dużą literę**”.

10. Zadbaj o pomoc i dokumentację

Niektóre systemy ze swojej natury będą bardziej złożone, w związku z czym użytkownik będzie potrzebował pomocy, np. samouczka, sekcji pomocowej czy rozbudowanej sekcji FAQ (najczęściej zadawanych pytań). Ważne jest, żeby sekcje te były przejrzyste i w klarowny sposób wyjaśniały problem.

10. Indeksowanie i wyszukiwanie informacji multimedialnych

KUKLA MONKEY - oczywiście nic nie ma w jej wykładach sensownego - to niżej opracuję

<https://esezam.okno.pw.edu.pl/mod/book/view.php?id=71&chapterid=1534>

https://link.springer.com/referenceworkentry/10.1007/978-0-387-39940-9_1037

https://link.springer.com/referenceworkentry/10.1007/978-1-4614-8265-9_80631

<https://dou.eu/community/posts/what-are-multimedia-information-retrieval-systems>

https://en.wikipedia.org/wiki/Multimedia_information_retrieval

<https://ebooks.inflibnet.ac.in/lisp7/chapter/multimedia-information-retrieval/>

11. Inteligencja obliczeniowa - metody i obszary zastosowań

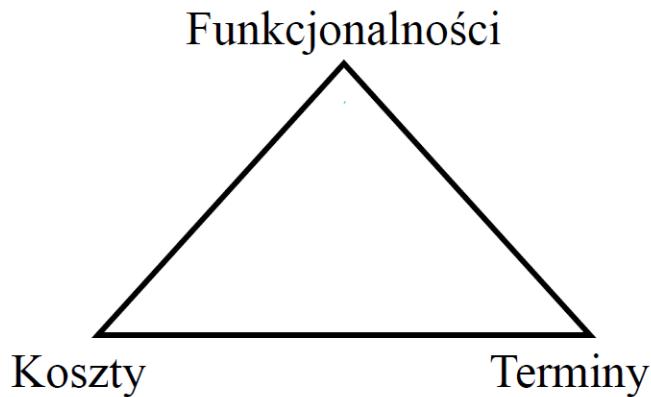
TIO z Kozierką

12. Inżynieria ontologii w przestrzeni Sieci Semantycznej

ONKOLOGIA

13. Koszty uwzględniane w kosztorysie projektu informatycznego

Co charakteryzuje system:



Kosztorys projektu informatycznego to kompleksowy dokument, który określa wszystkie koszty związane z realizacją projektu IT. Należy pamiętać, że balansowanie między funkcjonalnościami, kosztami i terminami jest kluczowe dla sukcesu każdego projektu informatycznego.

Koszty można podzielić na:

Koszty realizacji systemu - koszty jednorazowe, np. pomieszczenia biurowe, sprzętanie, telefony, zakup sprzętu, zatrudnienie pracowników

Koszty eksploatacji systemu - koszty ciągłe, zwykle ile rocznie, zużycie prądu, tonery, papier, łącze internetowe, licencje na oprogramowanie, obsługa techniczna systemu (administrator, konserwator)

Na różnych etapach projektu mogą pojawić się następujące koszty:

→ **Koszt przygotowania oprogramowania:** Dotyczy całego procesu tworzenia oprogramowania, w tym projektowania, kodowania, testowania i dokumentowania systemu.

Przykłady: zatrudnienie programistów, zakup narzędzi programistycznych, testowanie i debugowanie aplikacji.

→ **Koszt wdrożenia:** Związany z instalacją systemu, konfiguracją środowisk produkcyjnych, migracją danych oraz szkoleniem użytkowników końcowych.

Przykłady: koszty sprzętu IT, konfiguracja serwerów, szkolenia dla personelu.

→ **Koszt utrzymania i rozbudowy:** Dotyczy wszelkich kosztów związanych z codziennym użytkowaniem systemu oraz jego dalszą rozbudową i aktualizacjami.

Przykłady: koszty licencji, wsparcia technicznego, aktualizacje oprogramowania, rozbudowa funkcjonalności systemu.

Koszt zależny jest od rodzaju systemu, realizowanych funkcjonalności, liczby użytkowników systemu, liczby pracowników, którzy będą przygotowywali system, dostępności zasobów IT, czasu wdrożenia i liczby zmian w firmie, reorganizacji procesów.

14. Metody i narzędzia badania doświadczeń użytkownika

1. Badania etnograficzne

Przebieg: respondent jest obserwowany podczas wykonywania działań w naturalnym kontekście, obserwator zapisuje wnioski oraz utrwała zdarzenia i artefakty, nagrywając filmy i robiąc zdjęcia

Cel badania: poznanie naturalnych wzorców zachowań, kontekstu działań, przebiegu procesów oraz wpływu środowiska

2. Badania dzienniczkowe

Przebieg: respondent samodzielnie wypełnia przygotowany wcześniej dziennik, informując o zdarzeniach, ich częstości, miejscu oraz towarzyszących emocjach

Cel badania: poznanie naturalnych wzorców zachowań, kontekstu użycia produktu, obserwacja procesów długofalowych, zebranie inspiracji, testowanie funkcjonalności produktu

3. Badanie kwestionariuszowe

Przebieg: respondent samodzielnie i anonimowo wypełnia kwestionariusz

Cel badania: identyfikacja użytkowników pod kątem demograficznym, badania opinii i satysfakcji, testowanie hipotez z badań jakościowych, rozkład cech w populacji

4. Indywidualne wywiady pogłębione

Przebieg: rozmowa badacza i respondenta na zadany temat według scenariusza, pogłębiając ważne zagadnienia

Cel badania: poznać grupę odbiorców, ich język, wzorce zachowań i dotychczasowe doświadczenia

5. Badania fokusowe

Przebieg: kilkuosobowa grupa uczestników wspólnie dyskutuje o problemach poruszanych przez moderatora, dzieli się opiniemi, doświadczeniami i emocjami

Cel badania: poznanie grupy odbiorców, wzorców ich zachowań, doświadczenia, punkty widzenia; odkrycie potrzeb, badanie konkurencyjnych rozwiązań

6. Analiza danych zastanych

Przebieg: badacz wnioskuję na podstawie istniejących, dostępnych mu danych

Cel badania: poznanie kontekstu biznesowego, identyfikacja szans, poznanie grupy docelowej i istniejących potrzeb, poznanie języka użytkownika, terminologii w kontekście biznesowym

7. Sortowanie kart

Przebieg: uczestnik badania otrzymuje kilkadziesiąt etykiet, które łączy w grupy i nadaje im nazwy

Cel badania: opracowanie zrozumiałej struktury serwisu, intuicyjne pogrupowanie treści, poznanie terminologii użytkownika

Narzędzia:

1. Google Analytics - Narzędzie do analizy ruchu na stronie internetowej, które dostarcza dane o zachowaniu użytkowników, takie jak liczba odwiedzin, czas spędzony na stronie, ścieżki nawigacyjne itp.
2. Google Forms - narzędzie Googla do tworzenia ankiet.
3. Hotjar - Narzędzie do analizy zachowań użytkowników, które oferuje mapy cieplne, nagrania sesji, ankiety i analizy ścieżek.
4. Miro - Narzędzie do tworzenia map myśli, journey maps i innych wizualnych reprezentacji, które wspomagają proces projektowania UX.
5. Figma - Narzędzie do projektowania interfejsów, które umożliwia prototypowanie i testowanie projektów z użytkownikami.

15. Narzędzia analizy dużych danych (ang. Big Data)

Big Data to termin używany do opisu zbiorów danych, które są zbyt duże, złożone lub szybko zmieniające się, aby można je było efektywnie przetwarzać, analizować i zarządzać za pomocą tradycyjnych narzędzi i technik baz danych. Big Data charakteryzuje się kilkoma kluczowymi cechami, często określonymi jako **3V**:

1. **Volume (Wolumen):**

- Ogromna ilość danych generowana codziennie przez różne źródła, takie jak media społecznościowe, urządzenia IoT, transakcje handlowe, czujniki, logi systemowe itp. Przykładem może być ilość danych generowanych przez platformy takie jak Facebook, które przetwarzają miliardy zdjęć i wiadomości dziennie.

2. **Velocity (Prędkość):**

- Szybkość, z jaką dane są generowane, przetwarzane i analizowane. W dzisiejszych czasach dane są generowane w czasie rzeczywistym lub prawie rzeczywistym, co wymaga szybkiego przetwarzania. Przykładem może być analiza strumieniowa danych z mediów społecznościowych, która musi być przeprowadzona w czasie rzeczywistym, aby zrozumieć aktualne trendy.

3. **Variety (Różnorodność):**

- Różnorodność typów danych, które mogą być analizowane, w tym dane strukturalne (np. bazy danych), niestrukturalne (np. teksty, obrazy, filmy) i półstrukturalne (np. JSON, XML). Przykładem jest analiza treści e-maili, tweetów, zdjęć i filmów jednocześnie w celu zrozumienia sentymentu użytkowników.

Big Data ma szerokie zastosowania w różnych branżach, w tym:

- **Biznes:** Personalizacja ofert, optymalizacja łańcucha dostaw, analiza sentymentu klientów.
- **Medycyna:** Predykcja chorób, personalizowana medycyna, analiza genomów.
- **Finanse:** Wykrywanie oszustw, analiza ryzyka, optymalizacja inwestycji.
- **Transport:** Optymalizacja tras, analiza ruchu, przewidywanie awarii pojazdów.
- **Marketing:** Analiza kampanii marketingowych, targetowanie reklam, analiza trendów rynkowych.

Power BI

Power BI to zestaw narzędzi do analizy biznesowej opracowany przez firmę Microsoft, który umożliwia tworzenie interaktywnych raportów i wizualizacji danych. Pozwala użytkownikom na łatwe łączenie się z różnymi źródłami danych, przetwarzanie ich oraz udostępnianie wyników. Kluczowe cechy Power BI obejmują możliwość łączenia się z wieloma źródłami danych, w tym bazami danych SQL, plikami Excel, usługami chmurowymi (Azure, Google Analytics), plikami CSV i danymi strumieniowymi. Użytkownicy mogą tworzyć interaktywne raporty i dashboardy, które umożliwiają dynamiczną eksplorację danych i łatwe zrozumienie złożonych informacji.

Microsoft BI

Microsoft BI, często nazywane Microsoft Business Intelligence, to zbiór narzędzi i usług do analizy danych, które umożliwiają organizacjom zbieranie, przetwarzanie i analizowanie danych w celu wspierania decyzji biznesowych. W skład Microsoft BI wchodzą takie komponenty jak SQL Server Analysis Services (SSAS), SQL Server Integration Services (SSIS), SQL Server Reporting Services (SSRS) oraz Power BI. SSAS umożliwia zaawansowaną analizę danych i tworzenie modeli wielowymiarowych, SSIS służy do integracji danych i tworzenia procesów ETL (Extract, Transform, Load), a SSRS pozwala na generowanie raportów i ich udostępnianie.

Tableau

Tableau to narzędzie do wizualizacji danych, które umożliwia użytkownikom tworzenie interaktywnych i intuicyjnych wizualizacji oraz dashboardów. Tableau jest znane z łatwości obsługi, pozwalając użytkownikom na przeciąganie i upuszczanie elementów w celu tworzenia wizualizacji bez potrzeby pisania kodu. Tableau obsługuje szeroki zakres źródeł danych, w tym bazy danych, pliki, usługi chmurowe i strumienie danych. Oferuje również zaawansowane funkcje analityczne, takie jak kalkulacje na poziomie wiersza, analizy trendów i predykcje. Tableau Server i Tableau Online umożliwiają współpracę i udostępnianie wizualizacji w całej organizacji, zapewniając jednocześnie bezpieczeństwo i zarządzanie dostępem do danych.

17. Podstawowe metody i narzędzia inteligencji biznesowej.

System BI

- generuje raporty i wylicza kluczowe wskaźniki KPI
- na podstawie raportów stawia się hipotezy
- weryfikacja hipotez na podstawie szczegółowych analiz (OLAP, data mining)

Eksploracja danych:

- Analiza i poznanie dziedziny zastosowania, identyfikacja dostępnej wiedzy i celów użytkownika,

- Wybór danych związanych z celami procesu,
- Czyszczenie i wstępne przetwarzanie danych oraz ich redukcja,
- Wybór zadań i algorytmów eksploracji danych,
- Pozyskiwanie wiedzy z danych (krok eksploracji danych),
- Interpretacja i ocena odkrytej wiedzy,
- Przygotowanie wiedzy do użycia.

1. Metody Statystyczne:

- Regresja (liniowa, logistyczna) – szacowanie wartości wynikowej na podstawie zmiennych niezależnych, dostępne nawet w pakietach biurowych .
- Analiza szeregów czasowych – prognozowanie na podstawie danych historycznych, oddzielanie szumu od trendu .
- Drzewa decyzyjne

2. Data Mining (Eksploracja Danych):

- Pozyskiwanie wiedzy z danych przez identyfikację wzorców i reguł asocjacyjnych
- Wykorzystanie klasyfikatorów, takich jak drzewa decyzyjne i SVM (Support Vector Machines) .

3. Techniki Machine Learning i AI:

- Sieci neuronowe – modele do przewidywania i klasyfikacji na podstawie dużych zbiorów danych .
- Metody uczenia maszynowego – automatyczna analiza i prognozowanie na podstawie historycznych danych .

Narzędzia Inteligencji Biznesowej:

1. Narzędzia ETL (Extract, Transform, Load):

Wyodrębnienie (Extract)

Podczas wyodrębnienia ETL identyfikuje dane i kopiuje je ze źródeł, dzięki czemu może je przenieść do docelowego magazynu danych. Dane mogą pochodzić z ustrukturyzowanych oraz nieustrukturyzowanych źródeł, w tym z dokumentów, e-maili, aplikacji biznesowych, baz danych, wyposażenia, czujników, innych firm itp.

Transformacja (Transform)

Ponieważ wyodrębnione dane są nieprzetworzone w swej pierwotnej formie, konieczne jest ich odwzorowanie i przekształcenie w celu późniejszego przechowania. W procesie transformacji ETL weryfikuje, uwierzytelnia, deduplikuje i/lub agreguje dane w sposób, który daje uzyskanym danym wiarygodność i pozwala wysyłać zapytania na ich temat.

Ładowanie (Load)

ETL przenosi przekształcone dane do docelowego magazynu danych. Ten etap może wiązać się z początkowym załadunkiem wszystkich danych źródłowych lub z ładowaniem przyrostowych zmian w danych źródłowych. Dane można ładować w czasie rzeczywistym lub w zaplanowanych partiach.

2. Systemy OLAP (Online Analytical Processing):

- OLAP: OLAP to technologia służąca do szybkiego przetwarzania i analizy danych z różnych perspektyw. Jest kluczowym narzędziem w inteligencji biznesowej, umożliwiającym użytkownikom łatwe przeglądanie, analizowanie i modelowanie dużych ilości danych w celu wsparcia podejmowania decyzji.

TUTAJ TO SAMO CO W PYTANIU 6. OLAP i jego rodzaje itp, nie wrzucam, żeby nie duplikować 15 stron i nie straszyć ilością materiału

3. Hurtownie Danych (Data Warehouses):

Hurtownia danych to:

- tematycznie zorientowana
- zintegrowana
- chronologiczna
- trwała

kolekcja danych do wspomagania procesów podejmowania decyzji. Są miary i wymiary.

- **Architektura zcentralizowana i federacyjna – integracja danych z różnych źródeł w celu analizy biznesowej .**

- Infrastruktura hurtowni danych – skalowalność, dostępność, bezpieczeństwo, zarządzanie i współdziałanie .

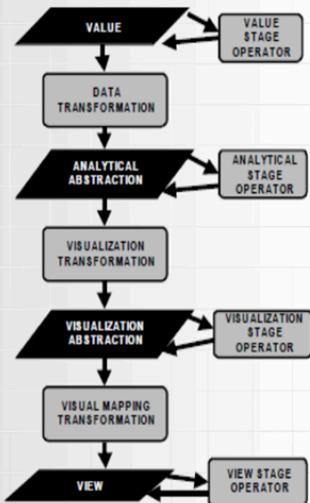
4. Narzędzia do wizualizacji danych:

- Wykresy statystyczne (boxplot, histogram, scatter plot, Dystrybuanta) .
- Wizualizacje oparte na ikonach i hierarchiczne – przedstawianie danych w formie graficznej .

Tutaj można wspomnieć o Power BI czy Tableau, czyli narzędziach do wizualizacji i analizy danych. Wygodne, proste, są darmowe wersje. Power BI ma kilka wariantów (desktopowe, mobilne, chmurowe, embedded).

Wizualizacja danych (Data State Model, ca. 2000)

- Value – dane
- Data transformation – najczęściej ekstrakcja danych
- Analytical abstraction – metadane
- Visual transformation – przekształcanie metadanych do postaci graficznej
- Visualization Abstraction – (meta)dane w postaci do wyświetlenia
- Visual Mapping Transformation – przekształcenie danych na podstawie metadanych
- View – ostateczna „grafika” dla użytkownika



18. Rodzaje diagramów projektowych - główne elementy strukturalne diagramu i jego przeznaczenie.

Jeśli chodzi o UML:

Lp	Rodzaj diagramu	Charakterystyka
1	Diagram klas	Diagram klas to graficzne przedstawienie statycznych, deklaratywnych elementów dziedziny przedmiotowej oraz związków między nimi
2	Diagram obiektów	Diagram obiektów to wystąpienie diagramu klas, odwzorowujące strukturę systemu w wybranym momencie jego działania
3	Diagram pakietów	Diagram pakietów to graficzne przedstawienie logicznej struktury systemu w postaci zestawu pakietów połączonych zależnościami i zagnieżdżeniami
4	Diagram struktur połączonych	Diagram struktur połączonych to graficzne przedstawienie wzajemnie współpracujących części dla osiągnięcia pożądanej funkcjonalności współdziałania
5	Diagram komponentów	Diagram komponentów to rodzaj diagramu wdrożeniowego, który wskazuje organizację i zależności między komponentami
6	Diagram rozlokowania	Diagram rozlokowania to rodzaj diagramu wdrożeniowego, który przedstawia sieć połączonych ścieżkami komunikowania węzłów z ulokowanymi na nich artefaktami
7	Diagram przypadków użycia	Diagram przypadków użycia to graficzne przedstawienie przypadków użycia, aktorów oraz związków między nimi, występujących w danej dziedzinie przedmiotowej

8	Diagram czynności	Diagram czynności to graficzne przedstawienie sekwencyjnych i/lub współbieżnych przepływów sterowania oraz danych pomiędzy uporządkowanymi ciągami czynności, akcji i obiektów
9	Diagram maszyn stanowej	Diagram maszyn stanowej to graficzne odwzorowanie dyskretnego, skokowego zachowania skończonych systemów stan-przejście
10	Diagram sekwencji	Diagram sekwencji jest rodzajem diagramu interakcji, opisującym interakcje pomiędzy instancjami klasyfikatorów systemu w postaci sekwencji komunikatów wymienianych między nimi
11	Diagram komunikacji	Diagram komunikacji jest rodzajem diagramu interakcji, specyfikującym strukturalne związki pomiędzy instancjami klasyfikatorów biorącymi udział w interakcji oraz wymianę komunikatów pomiędzy tymi instancjami
12	Diagram harmonogramowania	Diagram harmonogramowania jest rodzajem diagramu interakcji, reprezentującym na osi czasu zmiany dopuszczalnych stanów, jakie może przyjmować instancja klasyfikatora uczestnicząca w interakcji
13	Diagram sterowania interakcją	Diagram sterowania interakcją jest rodzajem diagramu interakcji, dokumentującym przepływ sterowania pomiędzy logicznie powiązanymi diagramami i fragmentami interakcji z wykorzystaniem kategorii modelowania diagramów czynności

Jeśli chodzi o wykład Pietranika:

BPMN (Business Process Modeling Notation)-Graficzna notacja do modelowania procesów biznesowych.

Podstawowe elementy

Elementy aktywne procesów

- Zdarzenia
- Działania

CO?

Przepływy

- Bramki **JAK?**
- Przepływ pracy
- Przepływ komunikatu
- Powiązanie/asocjacja

Wykonawcy zadań

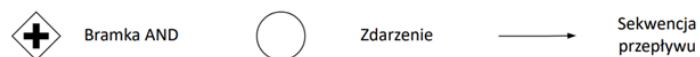
- Jednostki organizacyjne (pule)
- Jednostki składowe (tory)

KTO?

Artefakty

- Dane
- Grupy **CZYM?**
- Adnotacje

BPMN - podstawowe symbole



Zdarzenia



początek



zdarzenie pośrednie



koniec

Zdarzenia



wiadomość



wyjątek



licznik czasu

Zdarzenia



link



kompensacja



anulowanie



reguła

Model C4

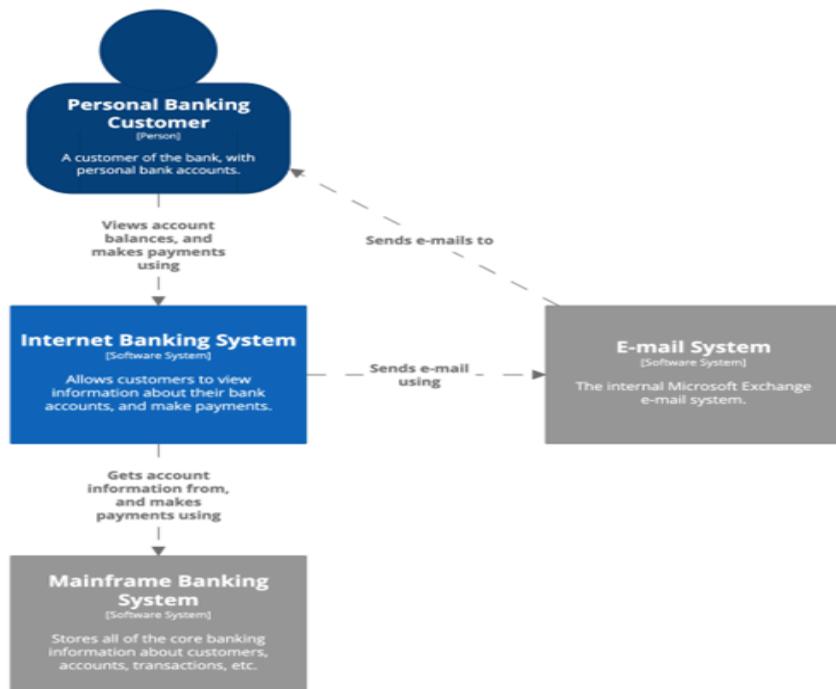
- Uproszczona notacja zaproponowana przez Simona Browna
- Oparta o czterostopniowy model dokumentowania architektury

i w konsekwencji cztery rodzaje diagramów opisujących system

na różnych poziomach szczegółowości :

- Diagram kontekstu systemowego (Context)
 - Punktem wyjściowym do tworzenia architektury jest kontekst, w jakim dany system działa.
 - Projektowany system jest centralnym elementem diagramu.
 - Przedstawione są jego interakcje z zewnętrznymi aktorami lub systemami.
 - Nie pojawiają się stosowane technologie lub protokoły

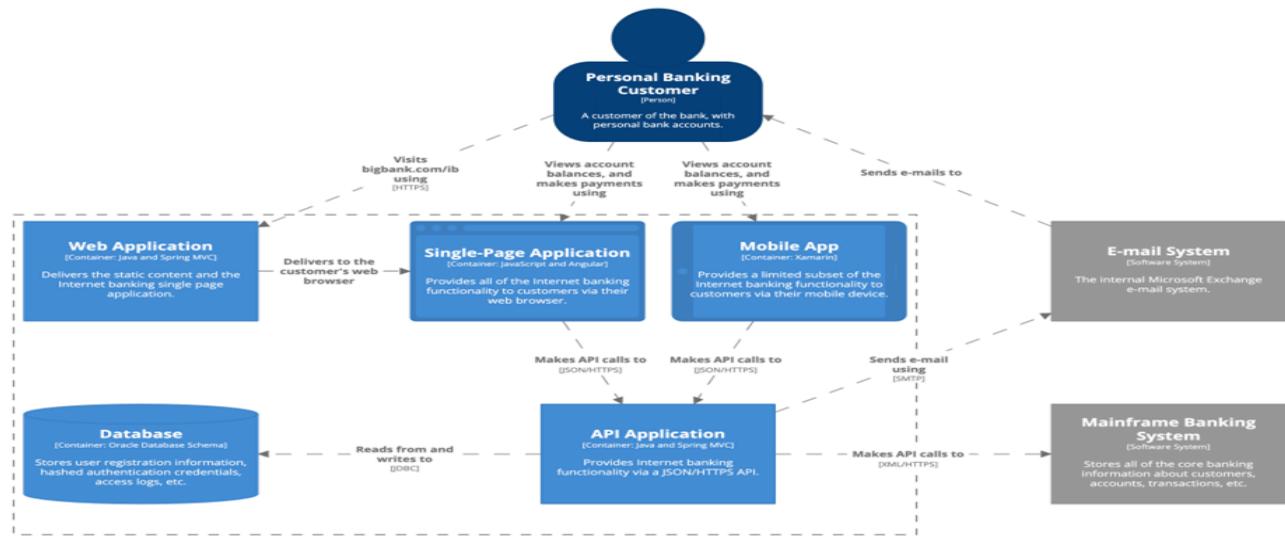
Tutaj tylko rysujemy systemy, ich nazwy i co z czym się komunikuje



- Diagram kontenerów (Container)

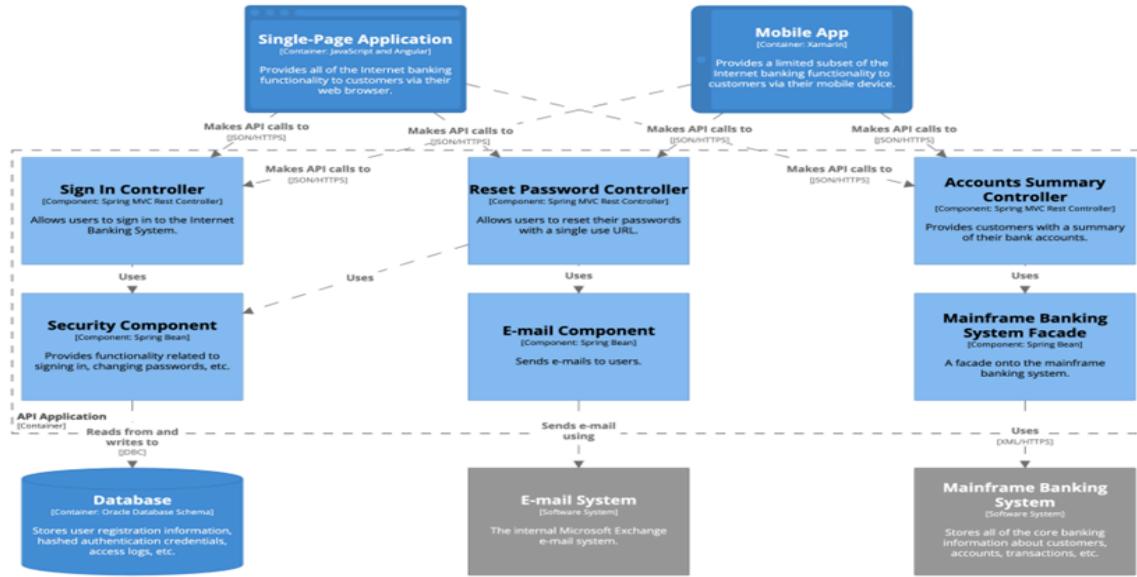
- Kontenery, które odwzorowują rozkład odpowiedzialności elementów systemu realizujących procesy biznesowe
- Często poziom ten jest nazywany „architekturą wdrożeniową”.
- Opisuje jak wdrażamy system, z jakich elementów się składa i na jakich elementach infrastruktury będzie instalowany.
- Typowe elementy to np. aplikacje mobilne, aplikacje webowe, bazy danych, kolejki, protokoły użyte w komunikacji pomiędzy kontenerami itp.

Tutaj rozdrabniamy się bardziej, rysujemy czy system jest mobilna apka, webowka, czy ma bazy danych itp.



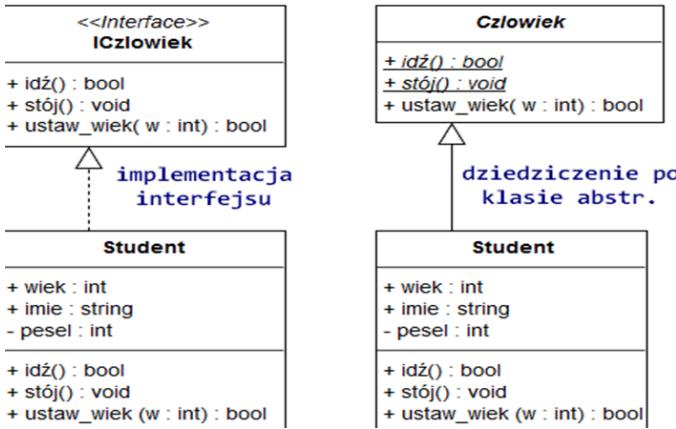
- Diagram komponentów (Component)

- W modelu C4 komponent ilustruje abstrakcję na poziomie kodu.
- W ujęciu Domain Driven Design komponenty można identyfikować z „bounded context”.
- Widoczne są aspekty technologiczne odnoszące się do implementacji komponentu - użyty framework, język programowania itp.



- Diagram klas (Class)

No to tutaj rozpisujemy jakie mamy klasy w kodzie, atrybuty itp., która z którą się łączy (agregacja, asocjacja itp.), dziedziczenie i inne pierdoły



19. Rodzaje dokumentacji systemu informatycznego, tworzonej w trakcie projektowania i realizacji systemu.

Screen z wykładu Chorosia:

DOKUMENTACJA SYSTEMU

1. Raport z założeniami systemu wraz z opracowaniem analitycznym.
2. Projekt techniczny.
3. Dokumentacja techniczna.
4. Dokumentacja użytkowa.
 - Podręcznik instalatora systemu,
 - Podręcznik administratora,
 - Podręczniki użytkowników.

Dodatkowo:

5. Materiały szkoleniowe.
6. Dokumentacja testów.
7. Dokumentacja wdrożeniowa.
8. Materiały informacyjne i reklamowe.

Dokumentacja systemu to wszystkie raporty/dokumenty sporządzane w trakcie realizacji systemu informatycznego na wszystkich etapach projektowych.

1. Raport z założeniami systemu wraz z opracowaniem analitycznym

- Dokument ten zawiera podstawowe założenia dotyczące systemu, jego cele oraz wymagania. Opracowanie analityczne może obejmować analizę potrzeb użytkowników, analizę ryzyka oraz specyfikację funkcjonalną i niefunkcjonalną systemu.

2. Projekt techniczny

- Projekt techniczny to szczegółowy plan techniczny systemu, który opisuje jego architekturę, komponenty, interfejsy, bazy danych oraz inne elementy techniczne. Jest to podstawa do realizacji i implementacji systemu.

3. Dokumentacja techniczna

- Dokumentacja techniczna zawiera szczegółowe opisy dotyczące implementacji systemu, w tym kod źródłowy, instrukcje instalacyjne, konfigurację systemu, oraz sposób integracji z innymi systemami. Może obejmować również schematy blokowe i diagramy.

4. Dokumentacja użytkowa

- Dokumentacja użytkowa to zbiór materiałów przeznaczonych dla różnych grup użytkowników systemu:
 - **Podręcznik instalatora systemu:** Instrukcje dotyczące instalacji systemu dla osób odpowiedzialnych za instalację.

- **Podręcznik administratora:** Przewodnik dla administratorów systemu, zawierający informacje o zarządzaniu, konserwacji i monitorowaniu systemu.
- **Podręczniki użytkowników:** Instrukcje i materiały dla końcowych użytkowników systemu, wyjaśniające, jak korzystać z jego funkcji.

Dodatkowo

5. Materiały szkoleniowe

- Materiały te są przeznaczone do szkolenia użytkowników i administratorów systemu. Mogą to być prezentacje, filmy instruktażowe, tutoriale, a także ćwiczenia praktyczne.

6. Dokumentacja testów

- Zawiera szczegółowe informacje dotyczące testowania systemu, w tym plany testów, scenariusze testowe, wyniki testów oraz raporty z testów. Dokumentacja ta jest kluczowa do zapewnienia jakości i poprawnego działania systemu.

7. Dokumentacja wdrożeniowa

- Opisuje proces wdrożenia systemu w środowisku produkcyjnym. Obejmuje plan wdrożenia, harmonogram, kroki migracji danych, procedury uruchomienia oraz plan awaryjny.

8. Materiały informacyjne i reklamowe

- Materiały te są używane do promocji systemu i informowania potencjalnych użytkowników o jego funkcjonalnościach i korzyściach. Mogą to być broszury, ulotki, prezentacje marketingowe oraz strony internetowe.

20. Rola i zadania wydawcy w procesie tworzenia gier komputerowych.

Wydawca gier komputerowych pełni kluczową rolę jako pośrednik między producentem/deweloperem gry a końcowym użytkownikiem, którym mogą być gracze, sklepy detaliczne oraz platformy cyfrowe takie jak Steam czy inne podobne serwisy.

Głównym zadaniem wydawcy jest maksymalizacja sprzedaży gry, co obejmuje docieranie do jak największej liczby potencjalnych graczy. Aby to osiągnąć, wydawca angażuje się w różnorodne działania marketingowe i promocyjne, takie jak:

- Marketing i reklama: Tworzenie kampanii reklamowych, organizowanie eventów promocyjnych, zarządzanie relacjami z mediami i influencerami.
- Umowy handlowe: Negocjowanie i podpisywanie umów z dystrybutorem oraz platformami sprzedażowymi, zarówno fizycznymi jak i cyfrowymi.
- Media społecznościowe: Prowadzenie kont gry na platformach społecznościowych, angażowanie społeczności graczy, publikowanie aktualizacji i materiałów promocyjnych.
- Podtrzymywanie zainteresowania: Regularne informowanie o postępach w produkcji gry, publikowanie zwiastunów, organizowanie beta testów i innych działań, które utrzymują zainteresowanie graczy.

Dzięki działalności wydawcy, deweloperzy mogą skoncentrować się wyłącznie na procesie tworzenia gry, bez konieczności angażowania się w aspekty sprzedażowe i marketingowe.

Wydawcy również odgrywają istotną rolę w dystrybucji gier na rynkach zagranicznych, w tym w wydawaniu wersji pudełkowych. Mogą oni finansować różne aspekty produkcji gry, na przykład lokalizację na inne języki, co może być kluczowe dla sukcesu gry na międzynarodowych rynkach. Przykładem może być wsparcie językowe dla gry wydawanej na rynek chiński, co znacznie zwiększa jej potencjalny zasięg.

Wydawcy często finansują również rozwój gier, licząc na późniejszy zysk ze sprzedaży. Nie zawsze oznacza to finansowanie całego projektu – może to być na przykład wsparcie tylko dla określonych elementów, takich jak marketing czy lokalizacja.

Zysk wydawcy najczęściej pochodzi z podziału przychodów ze sprzedaży gry z deweloperem, co motywuje obu partnerów do maksymalizacji sukcesu rynkowego gry.

21. Semantyczne wyszukiwanie informacji w sieci Web.

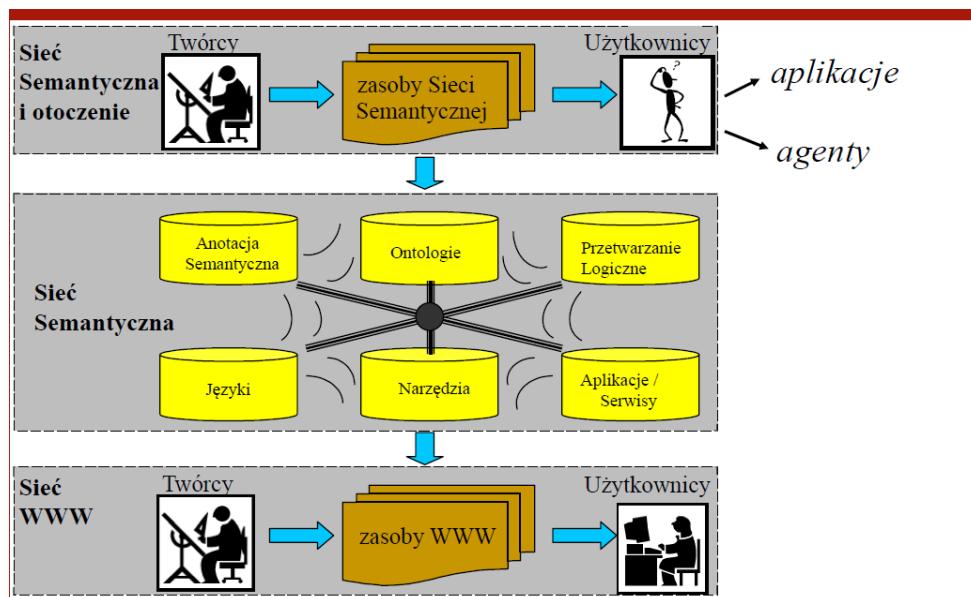
Z wykładu:

„Sieć Semantyczna do **wizja**: **idea**, aby dane istniejące w sieci WWW zdefiniować i powiązać w sposób możliwy do wykorzystania przez aplikacje, nie tylko jako wyświetlane dane, ale do automatycznego przetwarzania, integrowania, czy wykorzystywania w aplikacjach innych niż pierwotnie zaprojektowano”

„Sieć Semantyczna to **inicjatywa**, której celem jest rozszerzanie aktualnej sieci WWW i wyposażenie sieci WWW w powszechnie i automatycznie dostępne zasoby, stworzenie „Zaufanej Sieci” (ang. *Web of Trust*) dostarczającej powszechnie dostępnej platformy, która umożliwi współdzielenie i przetwarzanie danych zarówno przez narzędzia automatyczne jak i przez ludzi. „

Semantyczna Sieć jest rozszerzeniem obecnej sieci WWW, której celem jest umożliwienie komputerom zrozumienia i przetwarzania informacji w sposób bardziej zbliżony do ludzkiego pojmowania. Obejmuje ona metadane, ontologie oraz logikę deskrypcyjną, co pozwala na tworzenie powiązań i interpretację danych nie tylko na poziomie syntaktycznym, ale również semantycznym.

Semantyczne wyszukiwanie to podejście do przetwarzania zapytań i wyszukiwania informacji w sieci, które koncentruje się na zrozumieniu znaczenia i kontekstu zapytań użytkowników. W przeciwieństwie do tradycyjnego wyszukiwania opartego na dopasowywaniu słów kluczowych, semantyczne wyszukiwanie stara się „zrozumieć” intencję za zapytaniem oraz relacje między pojęciami.



Do opisu danych w kontekście Semantycznej Sieci stosowane są specjalistyczne **języki**, takie jak:

- RDF (Resource Description Framework): Umożliwia opisywanie zasobów w sposób, który może być przetwarzany przez maszyny. RDF wykorzystuje węzły (przedmioty, podmioty) i łuki (relacje) do tworzenia grafów danych.
- OWL (Web Ontology Language): Jest używany do tworzenia bardziej złożonych ontologii, które określają klasy, właściwości i relacje między pojęciami.
- SPARQL: Język zapytań przeznaczony do przeszukiwania danych RDF, pozwalający na wyszukiwanie informacji opartych na semantyce danych

Ontologie odgrywają kluczową rolę w semantycznym wyszukiwaniu, ponieważ umożliwiają definiowanie i interpretację znaczeń terminów oraz relacji między nimi. W kontekście wyszukiwania informacji, ontologie pozwalają na:

- **Jednolitą interpretację danych:** Umożliwiają komputerom i aplikacjom zrozumienie, co dokładnie reprezentują dane terminy i jak są one powiązane.
- **Automatyczne przetwarzanie zapytań:** Wykorzystanie ontologii pozwala systemom na zrozumienie zapytań użytkowników w kontekście semantycznym, co prowadzi do dokładniejszych wyników

Anotacje semantyczne to dodatkowe informacje (metadane), które są dodawane do zasobów internetowych, aby opisywać ich zawartość w sposób, który może być automatycznie przetwarzany przez komputery. Dzięki temu możliwe jest:

- **Lepsze indeksowanie treści:** Systemy mogą lepiej rozumieć i kategoryzować zawartość stron internetowych.
- **Dokładniejsze wyszukiwanie:** Użytkownicy mogą uzyskiwać wyniki, które są bardziej trafne i dostosowane do ich zapytań, dzięki zrozumieniu kontekstu i znaczenia treści

Przetwarzanie logiczne w kontekście Semantycznej Sieci odnosi się do zdolności systemów do wnioskowania na podstawie zdefiniowanych reguł i ontologii. Pozwala to na:

- **Wyciąganie wniosków:** Systemy mogą dedukować nowe informacje z już dostępnych danych.
- **Automatyczne reagowanie na zdarzenia:** Na podstawie zrozumienia semantycznego, systemy mogą podejmować decyzje lub działania w odpowiedzi na określone zdarzenia

Narzędzia przyjazne dla użytkownika są niezbędne do:

- opisywania/wprowadzania metadanych (ręcznej anotacji treści), czy automatycznej generacji metadanych,
- tworzenia i sprawdzania poprawności ontologii,
- pozyskiwania wiedzy (w postaci reguł),
- parsowania i przetwarzania języka naturalnego.

Z opracowania od Mateusza/Adama:

Najlepiej to przedstawić na przykładzie wyszukiwarki Googla. Na początku do wyszukiwania była brana fraza i słowa kluczowe. Później został wprowadzony algorytm Koliber, dzięki któremu trafność wyników wyszukiwania znacznie się poprawiła. Koliber miał bowiem na celu lepsze zrozumienie intencji użytkownika. W praktyce oznaczało to bardziej znaturalizowane wyszukiwanie i zamiast zestawu fraz, można było wprowadzać pełne zapytania np. „Gdzie w Poznaniu jest sklep z butami Adidas?”.

Algorytm był niejako dopełnieniem innej funkcjonalności, stanowiącej podwaliny pod współczesne semantyczne wyszukiwanie na masową skalę (istniały mniej znane wyszukiwarki semantyczne jak np. Wolfram Alfa, True Knowledge, Yebol, Google Squared). Chodzi o Graf Wiedzy.

Nowa funkcja wyszukiwarki Google sprawiła, że wyszukiwanie przestało koncentrować się wyłącznie na frazach. Zadaniem Grafu Wiedzy było znalezienie konkretnej informacji na zadane pytanie (a pytań może być wiele i to zadanych na różne sposoby). Zapewniał także użytkownikom możliwość poruszania się po tematach powiązanych.

Co odróżniało Graf Wiedzy od dotychczasowych wyszukiwań?

- znalezienie najtrajniejszego podsumowania – umożliwia zebranie najważniejszych informacji w jednym miejscu. Zebrane dane są zaprezentowane w pewnej strukturze, np. w przypadku osoby mamy informacje kiedy się urodziła i zmarła, kim była, jakie są jej dzieła lub dokonania, itp.
- znalezienie konkretnej rzeczy – wiązało się to z wieloznacznością słów, w niektórych bowiem przypadkach pod jednym hasłem może kryć się kilka znaczeń, tak przykładowo po wpisaniu „Chopin” w wyszukiwarkę może pojawić nam się w wynikach zarówno kompozytor, jak i lotnisko w Warszawie imienia tego właśnie kompozytora.
- rozszerzenie oraz pogłębienie poszukiwań – Graf Wiedzy pozwolił na zgłębienie wiedzy na dany temat poprzez odkrycie nowych powiązań z innymi tematami, np. możemy dowiedzieć się, że w Polsce znajduje się Muzeum Chopina w Warszawie, że organizowany jest Festiwal „Chopin i jego Europa”, itp.

Wprowadzenie algorytmu Koliber oraz Grafu Wiedzy było oznaką: po pierwsze – tego, w którą stronę zmierza Google w zakresie wyszukiwarki, po drugie - tego, czego oczekują od wyszukiwarki odbiorcy. A oczekują uzyskania konkretnej odpowiedzi na naturalnie formułowane zapytania. To właśnie zapewnia wyszukiwarka semantyczna.

Jej zadaniem jest przeszukiwanie zawartości strony internetowej przy jednoczesnym ciągłym interpretowaniu ich treści poprzez semantyczną i gramatyczną analizę języka naturalnego tworzącego stronę. Ponadto wyszukiwarki semantyczne „uczą się” nowych powiązań, nowych relacji, dzięki czemu w efekcie mają dostarczać precyzyjniejsze wyniki.

Obecnie wyszukiwarka Google do tworzenia powiązań między informacjami wykorzystuje Wikipedię oraz otwarte lub wykupione bazy danych (jak Freebase). Docelowo użytkownicy mają

poprzez swoją aktywność (najczęściej odwiedzane strony, zainteresowania, czytane artykuły, poszukiwane wydarzenia, itp.) budować sieć powiązań – właściwą dla siebie.

26. Zarządzanie ryzykiem w projektach informatycznych

Ryzyko - prawdopodobieństwo wystąpienia niepożdanego zdarzenia oraz jego wpływ na cele projektu (Ryzyko prowadzi do powstania problemu).

Ryzyko to prawdopodobieństwo, że w danym punkcie cyklu życia oprogramowania, jego zaplanowane cele nie zostaną osiągnięte w ramach dostępnych zasobów.

Zarządzanie ryzykiem to unikanie lub neutralizacja skutków niepomyślnych dla projektu Wydarzeń. Zarządzanie ryzykiem to systematyczny proces identyfikowania, analizowania i reagowania na ryzyka w projekcie. Obejmuje maksymalizację prawdopodobieństwa wystąpienia i wpływu pozytywnych zdarzeń na cele projektu oraz minimalizację prawdopodobieństwa wystąpienia i wpływu negatywnych zdarzeń.

W czasie procesu zarządzania ryzyka są oceniane i systematycznie obniżane do akceptowanego poziomu.

Efektywny menedżer musi identyfikować i osłabiać ryzyka w czasie całego cyklu życia systemu. Nie można wyeliminować wszystkich ryzyk, trzeba skupiać się na najbardziej krytycznych ryzykach dla powodzenia przedsięwzięcia do momentu aż osiągną akceptowalny poziom.

Ryzyka:

- przekroczenie budżetu
- zmiana założeń
- przekroczenie ram czasowych
- niespełnianie potrzeb użytkownika

Na ryzyko wpływają głównie:

- wielkość projektu
- doświadczenie technologiczne
- uniwersalność systemu
- złożoność systemu

Cykl zarządzania ryzykiem

- Identyfikacja - poszukiwanie i określanie ryzyk
 - Analiza- przetwarzanie danych o ryzyku w informacje niezbędne do podjęcia decyzji
 - Planowanie - informacje o ryzyku są przekształcane w decyzje i działania, wdrożenie działań
 - Śledzenie - monitorowanie wskaźników ryzyka
 - Kontrola - korygowanie odchyleń od zaplanowanych działań postępowania z ryzykiem
 - Komunikacja - dostarczanie informacji i zbieranie informacji zwrotnych na temat działań
- postępowania z ryzykiem, bieżących i powstających ryzyk



Techniki identyfikacji ryzyka

- przegląd dokumentacji projektu
- gromadzenie informacji
 - burza mózgów
 - metoda delficka - przewidywanie na podstawie wiedzy, doświadczenia i opinii ekspertów z danej dziedziny
 - Wywiady
 - SWOT (Strengths, Weaknesses, Opportunities, Threats) (mocne strony, słabe strony, szanse, zagrożenia)
 - mocne i słabe strony - czynniki wewnętrzne
 - szanse i zagrożenia - czynniki zewnętrzne
- listy kontrolne - lista z możliwymi ryzykami
- analiza założeń - taksonomia?
- techniki oparte na diagramach
 - Przyczynowo-skutkowe
 - przepływu dla systemu i procesów
 - wpływu i oddziaływanie

Taksonomia ryzyka projektu informatycznego

- identyfikuje i wyjaśnia techniczne i menedżerskie niepewności i problemy
- 3 klasy
 - inżynieria produktu- techniczne aspekty pracy do wykonania
 - środowisko rozwoju- metody, procedury i narzędzia użyte do wytworzenia produktu
 - ograniczenia programu- czynniki kontraktowe, organizacyjne i operacyjne, w ramach których opracowywane jest oprogramowanie, lecz które zazwyczaj jest poza bezpośrednią kontrolą lokalnego zarządu
- klasy dzielone na elementy i atrybuty

Diagram procesu decyzyjnego (Process Decision Program Chart - PDPC)

- graficzna reprezentacja alternatywnych działań dla realizacji planu w warunkach ryzyka
- używane przy tworzeniu planów do identyfikacji potencjalnych zagrożeń
- po zidentyfikowaniu ryzyk używany do określenia i wyboru zbioru możliwych środków zaradczych
- używany do planowania sposobów unikania i eliminacji zidentyfikowanych ryzyk
- Plan->możliwy problem->możliwy środek zaradczy

Postępowanie z ryzykiem

- Unikanie ryzyka- podejmowanie alternatywnych działań, nie tych związanych z ryzykiem
- Obniżanie ryzyka - działania obniżające, ale nie eliminujące zagrożenia
- Plany awaryjne - nie obniżają prawdopodobieństwa wystąpienia ryzyka, dodatkowe plany działania jak wystąpi ryzyko - zmniejszenie kosztów i zakłóceń

Miara ryzyka - porównywanie ryzyk

Efektywność redukcji ryzyka - porównanie środków zaradczych

Analiza i ocena ryzyka

- oszacowanie wpływu ryzyka na przebieg projektu
- prawdopodobieństwo, że ryzyko wystąpi

$$\text{Miara ryzyka} = \text{prawdopodobieństwo wystąpienia} \times \text{strata spowodowana ryzykiem}$$

ang. RI – Risk Impact lub RE – Risk Exposure

$$Ryzyko = \sum_i^n \text{prawdop}(r_i) \cdot \text{strata}(r_i)$$

Metoda punktowa szacowania ryzyka

- każdy element określony za pomocą pojedynczej wartości liczbowej
- podział zadań na kategorie ryzyka

- wagî dla ryzyk
- akcje zapobiegające ryzyku

Najlepiej się posłużyć przykładem (światowym standardem) metody kontroli realizacji projektu:

Earned Value Method (Metoda wartości uzyskanej):

- Głównym celem EVM jest szacowanie całkowitych kosztów projektu na podstawie rozbieżności pomiędzy monitorowanymi aktualnymi kosztami i postępami prac projektowych a ustalonym harmonogramem i planem uruchamiania środków finansowych w projekcie.
- Opiera się na obliczaniu kilku podstawowych wskaźników oraz śledzeniu ich trendów i odchyлеń.
- Stosowanie metodyki Earned Value daje dobre rezultaty dla projektów trwających ponad rok.
- Zgodnie z przyjętymi kryteriami pierwsze szacowanie należy wykonać na etapie ok. 15% zaawansowania projektu, a kolejne najlepiej co miesiąc.
- metoda ta pozwala nie tylko na obiektywną ocenę postępów projektu, ale również na prognozowanie jego przyszłych parametrów,
- sposób mierzenia postępu realizacji prac projektu z uwzględnieniem wykonania zadań i kamieni milowych,
- obliczanie stopnia zaawansowania zadania na podstawie stosunku faktycznego nakładu pracy do nakładu planowanego.

Podstawą stosowania metody EarnedValue jest wyznaczenie Wartości Uzyskanej (BCWP - BudgetedCostof WorkPerformed, WU), obliczanej ze wzoru:

$$WU = \text{budżet zadania} * \% \text{ wykonania zadania}$$

Znacznie trudniejszą rzeczą jest określenie na daną chwilę procentu wykonania zadania, szczególnie dla zadań długotrwałych. Istnieje wiele technik, które umożliwiają oszacowanie tej wartości:

- Metoda 0/100

Najprostsza z metod. Polega na przydzielaniu wartości uzyskanej w wysokości 100% planowanych kosztów dopiero po całkowitym wykonaniu zadania. Jest ona dobra dla małych krótkotrwałych zadań.

- Metoda 50/50
- Jest podobna do poprzedniej, jednak po rozpoczęciu zadania uzyskuje ono 50% planowanego budżetu. Kolejne 50% przydzielane jest dopiero po ukończeniu zadania. Gdy mamy dużo takich zadań, statystyczny rozkład ich rozpoczęć i zakończeń powoduje, że krzywa wartości uzyskanych dobrze odpowiada rzeczywistości.

- Itd..

efektywność redukcji ryzyka = oszczędność z redukcji / koszt redukcji
ang. **RRL – Risk Reduction Leverage**

$$RRL = \sum_i^n \frac{strata_{przed}(r_i) - strata_{po}(r_i)}{koszt_{redukacji}(r_i)}$$

Zalety EVM:

1. Kompleksowa kontrola realizacji projektu w aspekcie rzeczowym, czasowym i kosztowym
2. Obiektywna miara postępu projektu, wyrażona w jednej jednostce niezależnie od różnych jednostek obmiaru poszczególnych prac
3. Identyfikacja rodzaju i wielkości odchyлеń budżetu
4. Analiza trendu wydajności realizacji projektu
5. System wczesnego ostrzegania (wczesna prognoza oszacowania kosztu końcowego)
6. Możliwość analizy na różnych poziomach struktury podziału prac (WBS) i struktury organizacyjnej (OBS)
7. Prostota - kilka, intuicyjnie zrozumiałych wskaźników
8. Standard wymiany informacji o projekcie na poziomie wyższego kierownictwa
9. Dobrze udokumentowany, powszechnie znany i stosowany standard kontroli realizacji projektów

27. Zarządzanie zespołami ludzkimi w projektach informatycznych

Większość profesjonalnego oprogramowania tworzona jest przez zespoły składające się z dwustu do kilkuset osób. Nie ma możliwości, żeby wszyscy członkowie tak dużego zespołu pracowały razem nad jednym problemem. Dlatego zespoły dzieli się na kilka mniejszych grup, z których każda pracuje nad inną częścią danego systemu. Przyjmuje się zasadę, że grupy powinny liczyć do 8 osób. Utworzenie wydajnie pracującej grupy to zadanie menedżera. W grupie powinna panować równowaga umiejętności technicznych, doświadczenia i osobowości.

KIEROWNIK PROJEKTU

- cel: dostarczenie produktu w wymaganym czasie, w ramach określonego budżetu, posiadającego odpowiednią jakość
- Funkcje
 - Planowanie
 - Organizowanie
 - Motywowanie
 - Kontrolowanie
- odpowiedzialność za planowanie i prognozowanie
 - odpowiedzialność interpersonalna (prowadzenie zespołu, komunikacja z klientem, zarządem)
 - odpowiedzialność za stan informacji (monitorowanie wydajności, postępu prac, informowanie o zadaniach, stanie projektu)
 - odpowiedzialność decyzyjna (zarządzanie zasobami, negocjacje kontraktów)
- zaspokaja potrzeby jednostki, zespołu, zadania
- rozwiązuje konflikty (kompromis, łagodzenie, narzucenie/przymus, unikanie)

MOTYWOWANIE

Zadaniem kierownika jest motywowanie zespołu jako całości, jak i każdej osoby z osobna.

Motywacja zespołu ma źródło w osobistym zaangażowaniu kierownika, w sposobie przydziału i podziału pracy, jasnej wizji celu i sposobu jego osiągnięcia. Kierownik daje przykład przez własne zaangażowanie i zachowanie.

Motywację osobistą osiąga się przez stosunek między tymi osobami i "niepisaną umowę", czego dana osoba i kierownik od siebie oczekują.

Kluczowym elementem motywacji jest projektowanie pracy poszczególnych osób. Składa się na nią: odpowiednia ilość wyzwań, różnorodność zadań, perspektywa osiągnięcia celu (wyniku).

Ludzi motywuje się przez spełnienie ich potrzeb

- Potrzeby samorealizacji
- Potrzeby szacunku
- Potrzeby społeczne
- Potrzeby bezpieczeństwa
- Potrzeby fizjologiczne

3 ostatnie są najważniejsze z punktu widzenia kierownika.



ERGONOMIA PRACY

- mniejsze pomieszczenia (np. zespół w 1 pomieszczeniu)
- personalizacja stanowisk pracy
- pokój zebrań formalnych i nieformalnych
- praca na nowoczesnym sprzęcie
- atmosfera, równomierny rozkład pracy, brak przenoszenia odpowiedzialności.
- komfort psychiczny

ETAPY ROZWOJU ZESPOŁU

- Formowanie
- Burza
 - opór przed zadaniami, uzgadnianie
 - lider: rozwiązuje nieporozumienia
- Normowanie
 - proces burzy daje uzgodnione podejście do podejmowania decyzji, członkowie zespołu znają i akceptują role, zadania, ...
 - lider: rozwiązuje ewentualne konflikty, bardziej obserwuje niż interweniuje
- Wykonywanie
 - zespół pracuje najbardziej efektywnie
 - lider: zarządza wykonywaniem, doradza, zapewnia szkolenia, informacje zwrotne
- Przechodzenie
 - koniec projektu, pracownicy przechodzą do innych zadań
 - lider: na zebraniach podsumowuje pracę, motywuje do dalszej

AGILE

W zarządzaniu zespołami obecnie odchodzi się od podejścia opartego na "władzy" i ścisłej kontroli. Zamiast tego wspiera się samowystarczalność i samodzielność zespołów, stwarza się im warunki do wykonania powierzonej im pracy, zachęca do rozwoju (samodzielnego lub przez szkolenia). Coraz większą popularność zyskują metodyki zwinne (agile), nastawione na ludzi i ich komunikację. "Ludzie najważniejszym czynnikiem decydującym o sukcesie lub porażce projektu".

XP

- główny cel: sprawny rozwój oprogramowania:
 - niższe koszty,
 - mniejsza ilość błędów,
 - większa produktywność
- zmiany szansą na rozwój i doskonalenie
- nacisk na komunikację ustną
- praca stałym rytmem, bez nadgodzin
- Wartości
 - naciska na komunikację
 - informacje zwrotne od klientów, użytkowników
- Zasady:
 - ludzie tworzą oprogramowanie - praktyki są dla nich
 - ciągłe doskonalenie
 - jakość podstawą kontrolowania projektu
 - małe kroki
- Praktyki
 - niewielki, samowystarczalny zespół (10 osób)
 - praca w 1 pomieszczeniu
 - programowanie w parach
 - cykl tygodniowy, kwartalny
 - Scenariusze
 - programowanie poprzedzone testami - TDD
 - ciągła integracja - CI
 - Refaktoryzacja
 - projektowanie przyrostowe
 - komplikacje 10-minutowe
 - wspólny kod
 - klient częścią zespołu

SCRUM

- środowisko do organizacji i zarządzania pracą
- szkielet prowadzenia projektu, który można łączyć z innymi podejściami, technikami
- skupia się na usprawnianiu zarządzania projektem, procesu dostarczania i rozwoju oprogramowania
- 3 filary
 - przejrzystość -- jednoznaczny odbór, wszystko udostępnione dla tego kto tego

potrzebuje

- inspekcja -- częst kontrola do wykrywania nieprawidłowości
- adaptacja -- ograniczenie skutków nieprawidłowości, uniknięcie dalszych problemów

- User stories - historyjki użytkownika? - lista wymagań, krótkie, na małej kartce + kryteria akceptacji
- Product backlog - Rejestr produktu? - user stories + priorytety
- Sprint

- iteracja - od tygodnia do miesiąca (wszystkie tyle samo)
- planowanie - wybór najważniejszych elementów z product backloga i stworzenie sprint backloga- rejestr sprintu?
- zespół sam wybiera kolejność zadań do realizacji
- codzienny stand-up do 15 minut
- wykresy spalania, wypalania
- na koniec przegląd sprintu (z interesariuszami) + retroperspektywa (analiza zespołu)

- Zespół Scrumowy

- 5-9 osób
- samowystarczalny, interdyscyplinarny
- Właściciel Produktu (ang. Product owner)
- Scrum Master
- Zespół developerski

Scrum vs agile:

<https://stackoverflow.com/questions/11469358/what-is-the-difference-between-scrum-and-agile-Development>

CRYSTAL

- rodzina metodyk, zbiór przykładów
- bazuje na podziale projektów pod względem krytyczności (błąd powoduje utratę czegoś) i liczby osób (nawiązuje do kolorów i twardości kryształów)
- programowanie jako gra zespołowa pomysłowości i komunikacji, gra skończona
- 4 punkty krytyczne
 - nacisk na komunikację
 - obecność ekspertów w miejscu pracy zespołu
 - przyrostowe tworzenie oprogramowania (co miesiąc)
 - zautomatyzowane testy
- zespół może wybrać dowolne praktyki jakie zna
 - wymagane:
 - częste dostarczanie
 - komunikacja osmotyczna
 - analiza + poprawianie met

28. Zastosowanie inteligencji biznesowej do weryfikacji hipotez i zwiększenia KPI (ang. Key Performance Indicators)

KPI (key performance indicators), czyli kluczowe wskaźniki efektywności to mierzalne wartości, które pokazują, jak skutecznie osoby zaangażowane w biznesie osiągają najważniejsze dla nich cele biznesowe. Wykorzystywane są w wielu różnych dziedzinach i pozwalają ocenić skuteczność działań biznesowych. Pozwalają ograniczyć dużą ilość informacji do najbardziej potrzebnych danych, które dają najlepszy obraz realizacji wyznaczonych celów. Wskaźniki KPI są charakterystyczne dla danej firmy, konkretnego stanowiska i będą się różnić w zależności od charakterystyki przedsiębiorstwa, branży, wielkości firmy bądź strategii działania.

Finanse

- EBITDA
- ROE (rentowność kapitałów własnych)
- Przepływy pieniężne z działalności operacyjnej
- Przychody ze sprzedaży
- Zadłużenie netto / EBITDA, gdzie Zadłużenie netto = zobowiązania – środki pieniężne

SaaS

- MRR (Monthly Recurring Revenue)
- CAC (cost of customer acquisition)
- LTV lifetime value
- RCS (Recurring Costs of Service)
- Customer Churn Rate

Sprzedaż

- Sprzedaż netto
- Średnia wartość transakcji
- Wartość utraconych szans sprzedaży
- Wartość sfinalizowanych transakcji w stosunku do ilości zamówień
- Średni przychód przypadający na jednego handlowca

Marketing

- Liczba nowych leadów
- Liczba subskrypcji newslettera
- Koszt pozyskania leadów
- Liczba zapytań ofertowych ze strony internetowej
- Zaangażowanie na kanałach marki w mediach społecznościowych

Obsługa klienta

- Liczba wykonanych telefonów
- Średnia ocena jakości obsługi
- Średnia liczba rozmów obsługiwanych przez jednego pracownika
- Liczba zgłoszonych reklamacji

- Średni czas rozmowy i rozwiązania problemu

E-commerce

- Zwiększenie ruchu na stronie
- Ruch na stronie wygenerowany przez poszczególne kanały
- Współczynnik konwersji
- Liczba sesji z dodaniem produktów do koszyka
- Czas spędzony na stronie

Business Intelligence applications are one way of monitoring and assessing the level of Key Performance Indicators being met. Business Intelligence systems compile data relating to profit, productivity, customer return, and marketing trends. With this information readily available managers and executives can review the data and put in place indicators as needed, then using the software on a regular basis they are able to assess the success rate of the plans they initiate.