Kod szkolenia: **J/ARCH/ANA**

Tytuł szkolenia: **Architektura systemów dla analityków**

Dni: 3

Opis:

Adresaci szkolenia:

Szkolenie przeznaczone jest dla analityków biznesowych chcących zapoznać się z architekturą systemów, w celu zrozumienia stosowanych rozwiązań oraz usprawnienia komunikacji z architektami. Co pozwoli lepiej definiować wymagania niefunkcjonalne oraz wykrywać aspekty istotne dla architektury (ograniczenia, ryzyka, potencjalne rozwiązania).

Cel szkolenia:

Celem szkolenia jest nabycie wiedzy koniecznej do wykrywania zagrożeń architektonicznych, dzięki czemu analityk biznesowy już w trakcie wywiadów z klientem będzie w stanie dokonywać pierwszych założeń architektonicznych, co z kolei pozwoli na zgłębienie kluczowych aspektów. Aby to osiągnąć konieczne jest zrozumienie nie tylko rozwiązań architektonicznych, ale również celów do osiągnięcia. Jest to wiedza przydatna nie tylko podczas tworzenia nowych systemów, gdyż zrozumienie aktualnej budowy systemu i osiąganych dzięki temu parametrów systemowych ułatwi ustalanie szczegółów podczas rozwoju istniejących systemów.

Na szkoleniu duży nacisk kładziony jest na osiągnięcie wysokiej świadomości konsekwencji doboru poszczególnych rozwiązań, technologii, wzorców czy innych decyzji architektonicznych. W oparciu o nią podczas ćwiczeń budowana jest umiejętność podejmowania decyzji architektonicznych oraz ich weryfikacji w warunkach nieklarownych wizji systemu czy dużej ilości założeń. Rozpatrujemy potencjalne rozwiązania z poziomu ich konsekwencji (wady i zalety) dla poszczególnych parametrów systemu.

Na szkoleniu wspominane są technologie wspierające nas w złożonych architekturach z podziałem na poszczególne warstwy, w tym specyfikacje z JEE. Dzięki czemu uczestnicy poznają główne cechy wybranych technologii oraz klasy rozwiązań, co ułatwia dalsze poszukiwania, gdyby ograniczenia obecnej architektury wymagały innego środowiska. Omawiamy także diagramy architektoniczne w UML na poziomie wystarczającym do zrozumienia budowy systemu. Ponieważ adresatami szkolenia są analitycy, a nie architekci, jedynie zapoznajemy się z modelami architektonicznymi i nie rozwijamy umiejętności ich tworzenia (zajmujemy się tym w wersji 5cio dniowej dla architektów).

Wymagania:

Wprowadzamy do zagadnień architektury od podstaw, w związku z czym nie brak wymagań wstępnych.

Parametry szkolenia:

3*8 godzin wykładów i warsztatów.

Program szkolenia:

1. Podstawy Architektury
    I. Czym jest architektura
        i. Architektura a projekt
        ii. Cele tworzenia architektury
    II. Kim jest architekt i jaką pełni rolę
        i. Kim jest architekt - różne poziomy
            A. Technolog
            B. Strateg
            C. Polityk
        ii. Co robi architekt
        iii. Potrzeba istnienia architekta a skala projektu
    III. Proces architektoniczny
    IV. Dokumentacja architektoniczna
    V. Zarządzanie ryzykiem
2. Parametry systemowe
    I. Czym są parametry systemowe
    II. Jak poprawnie definiować wymagania niefunkcjonalne
    III. Parametry systemowe
        i. Wygoda użytkownika (Usability)
        ii. Bezpieczeństwo (Security)
        iii. Wydajność (Performance)
            A. Przepustowość (Throughput)
            B. Czas odpowiedzi (Response Time)
            C. Czas reakcji (Responsivness)
        iv. Dostępność (Availability)
        v. Niezawodność (Reliability)
        vi. Skalowalność (Scalability)
        vii. Różne wymiary elastyczności systemu
            A. Rozszerzalność (Extensibility)
            B. Reużywalność (Reusability)
            C. Przenaszalność (Portablity)
            D. Elastyczność (Flexibility)
        viii. Realizowalność (Realizability)
        ix. Planowalność (Planability)
        x. Testowalność (Testability)
        xi. Utrzymanie (Maintainability)

xii. Serwisowalność (Serviceability)
xiii. Zarządzalność (Manageability)
IV. Wymiary systemu
i. Wymiary związane z infrastrukturą
A. Pojemność (Capacity)
B. Redundantność/Replikacja (Redundancy)
C. Modułowość (Modularity)
ii. Wpływ wymiarów na parametry systemu
iii. Inne wymiary systemu
A. Tolerancja (Tolerance)
B. Obciążenie (Workload)
C. Niejednorodność/Jednorodność (Homo/Heterogenity)
V. Priorytety parametrów systemu
i. Skąd wynikają priorytety?
ii. Problemy priorytetowania
3. Wzorce architektoniczne
I. Wprowadzenie do wzorców
i. Definicja wzorca
ii. Cechy i zalety wzorców
iii. Rodzaje wzorców
II. Wzorce architektoniczne
i. Problemy architektury komponentowej
ii. Wzorce podziału odpowiedzialności
A. MVC (Model View Controll)
B. Web-centric
C. Application-centric
D. Enterprise
a. Wymagania systemów Enterprise
b. Architektura Enterprise w JEE
E. Architektura wielowarstwowa (Layers Pattern)
a. Architektura wielowarstwowa w JEE
iii. Wzorce EAI (Enterprise Application Integration)
A. SOA (Service Oriented Architecture)
B. ESB (Enterprise Service Bus; Szyna Danych; Broker Integracyjny)
C. MOM (Message Oriented Middleware)
D. Microservices
a. Założenia
b. Monolit a Microservices
c. DB w monolicie a w Microservices
d. Dlaczego Microservices?
e. Problemy z microsercives
iv. Wzorce infrastruktury
A. Redundancja Ścieżek
B. Skalowanie pionowe
C. Skalowanie poziome (Replikacja)

D. Równoważenie obciążenia (Load Balancing)
E. Klastry (Clustering)
    a. HA (High Availability)
    b. Failover
F. Forward Proxy Cache

v. Wzorce blokowania zasobów
A. Blokowanie optymistyczne (Optimistic Lock)
B. Blokowanie pesymistyczne (Pessimistic Lock)
    a. Blokada domyślna (Implicit Lock)
    b. Blokowanie gruboziarniste (Coarse-Grained Lock)

vi. Inne wzorce architektoniczne
A. Dependency Injection
B. Dependency Inversion
C. Stable Dependency Principle
D. CQRS
E. Event Sourcing
F. Szablon wzorców POSA
G. Szablon wzorców PEAA
H. Szablon wzorców Core J2ee
I. Szablon wzorców EAI (wzorce JMS)

4. Architektura warstwy klienta i prezentacji
I. Podział klientów
    i. Klient gruby
    ii. Klient Cienki
        A. RIA
II. Przechowywanie sesji
III. Technologie klienta grubego
    i. Swing
    ii. SWT
    iii. RCP
IV. Klient gruby zanurzony w kliencie cienkim
    i. Applet
    ii. Java Web Start
    iii. Java FX
V. Technologie klienta cienkiego
    i. HTML Statyczny
    ii. HTML Dynamiczny
        A. Servlety + JSP + JSTL
        B. Portlety
        C. JSF (Java Server Faces)
        D. Ajax
            a. Java Script
            b. Prototype
            c. Ajax4JSF
            d. PrimeFaces
            e. GWT

E. Wsparcie JavaScript
    a. JSON
    b. jQuery
F. SPA/SPI
    a. Przykładowe frmeworki: Angular, Ember JS
    b. Przykładowe biblioteki JavaScript: React, Vue
G. WebSocket
    a. Protokół full-duplex
    b. Użycie w przeglądarce
    c. Użycie w JEE (@ServerEndpoint)
    d. Kiedy stosować WebSocket

5. Architektura warstwy biznesowej
    I. Przetwarzanie rozproszone
    II. Komunikacja zdalna a lokalna
    III. Optymalizacja komunikacji sieciowej
    IV. Protokoły komunikacyjne
        i. CORA i IIOP
        ii. Web Services
            A. SOAP i REST
            B. WSDL
            C. Repozytoria usług: UDDI, ebXML
            D. Specyfikacje JEE
                a. JAX-P (Java API for XML Processing)
                b. SAAJ (SOAP with Attachments API for Java)
                c. JAX-B (Java API for XML Binding)
                d. JAX-R (Java API for XML Registries)
                e. JAX-WS (Java API for XML Web Services)
            E. Web Service Orchestration (wsparcie procesów biznesowych)
            F. Transakcje długoterminowe w Web Service
        iii. Web Services REST
            A. REST i wsparcie JAX-RS (od JEE6)
            B. WADL i alternatywy (API Blueprint, Swagger)
            C. HATEOAS
        iv. GraphQL
            A. Czym jest GraphQL
            B. GraphQL vs REST
        v. Sockets (własny protokół)
        vi. RMI (Remote Method Invocation)
        vii. EJB i RMI-IIOP
            A. Rodzaje komponentów EJB
                a. Sesyjne
                    1. Statefull
                    2. Stateless
                    3. Singleton (od EJB3.1 – JEE6)
            B. MDB
            C. Encyjne (do EJB2.x)

viii. JNDI jako repozytorium EJB

6. Architektura warstwy integracji i zasobów
   I. Technologie utrwalania danych
      i. Bazy relacyjne
         A. JDBC
         B. Entity EJB
         C. JDO (także bazy obiektowe i inne)
         D. JPA
      ii. LDAP (bazy hierarchiczne)
      iii. NoQSL
         A. Charakterystyka rozwiązań NoSQL
         B. ACID a BASE
         C. CAP Theorem
      iv. JCR – Java Content Repository (systemy CMS)
      v. JCA – Java Connector Architecture (systemy EIS)
   II. Komunikacja asynchroniczna
      i. JMS (Java Message Service)
         A. Topic
         B. Queue
      ii. MDB EJB (Message Driven Bean)
   III. Systemy „Legacy”
   IV. Screen Scrapping
7. Modelowanie architektury w UML
   I. Diagram komponentów (component diagram)
      i. Komponent (component)
      ii. Komponenty zagnieżdżone
      iii. Interfejs (interface)
         A. Interfejs wymagany (required interface)
         B. Interfejs dostarczany (provided interface)
      iv. Złączenie (assembly)
   II. Diagram wdrożenia (deployment diagrram)
      i. Węzeł (node)
      ii. Łącze (communication path)
         A. Łącze kierunkowe
         B. Liczność łącza
      iii. Porty
      iv. Konektory
      v. Realizacja komponentu
      vi. Instancyjne diagramy wdrożenia
      vii. Niskopoziomowe diagramy wdrożenia
      viii. Szablony architektoniczne
      ix. Model wdrożenia na diagramach wdrożenia
         A. Po do model wdrożenia
         B. Artefakt
         C. Stereotypy artefaktów
            a. file

b. document
c. library
d. executable
e. script
f. source

D. Specyfikacja konfiguracji (deployment specification)
E. Relacje między artefaktami
a. Kompozycji (composition)
b. Zależności (dependency)
F. Instalacja artefaktów (deployment) deploy
G. Manifestacja (manifestation) manifest

III. Diagram pakietów (package diagram)
i. Pakiet
ii. Zagnieżdżanie (nest, nesting)
iii. Przestrzeń nazw
iv. Importowanie (package import)
A. import
B. access
v. Łączenie (merge)
vi. Diagramy pakietów i modelowanie warstw architektury
A. Przecięcie warstw i poziomów