Projektowanie architektury systemu, część I

Bogumiła Hnatkowska

Cele wykładu

- Wprowadzenie pojęć:
  - architektura,
  - perspektywa,
  - widok

  oraz zależności między nimi
- Definicja wybranych perspektyw architektonicznych

Projektowanie architektury systemu

- Cel:
  - Propozycja architektury systemu oprogramowania z uwzględnieniem **wymagań funkcjonalnych i niefunkcjonalnych**
  - Prezentacja architektury z różnych perspektyw

Architektura

- Architektura – **dekompozycja** systemu na składowe (elementy) wraz z **definicją sposobów interakcji** pomiędzy elementami.
- Architektura – struktura organizacyjna systemu; architektura może być rekursywnie **dekomponowana** na części, które wchodzą w interakcje na interfejsach; na architekturę składają się również **zależności łączące części** oraz **ograniczenia** nałożone na części (RUP)
- Architektura – zbiór strategicznych decyzji projektowych, które wpływają na większość lub wszystkie elementy systemu. Są to decyzje, które trudno zmienić (M. Fowler)

Architektura, c.d.

- “Software architecture encompasses the set of significant decisions about the organization of a software system including the selection of the structural elements and their interfaces by which the system is composed; behavior as specified in collaboration among those elements; composition of these structural and behavioral elements into larger subsystems; and an architectural style that guides this organization. Software architecture also involves functionality, usability, resilience, performance, reuse, comprehensibility, economic and technology constraints, tradeoffs and aesthetic concerns.” [Philippe Kruchten, Grady Booch, Kurt Bittner, and Rich Reitman]

- “The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them. [Bass, Clements, Kazman]

Rola architektury

- Komunikacja i zrozumienie
- Ponowne wykorzystanie
- Decyzje w zakresie konstrukcji i ewolucji systemu
- Analiza własności

Cykl biznesowy tworzenia architektury

- Zrozumienie wymagań
- Opracowanie lub wybór architektury (spośród architektur referencyjnych lub styli architektonicznych)
- Udokumentowanie i komunikowanie architektury
- Analizowanie i ocena architektury
- Implementacja systemu opartego o zdefiniowaną architekturę

Kluczowe zasady projektowania architektury

- Separacja pojęć
- Zasada pojedynczej odpowiedzialności (na poziomie modułów/komponentów)
- Zasada najmniejszej wiedzy (least knowledge principle)
- Używaj modeli i wizualizacji jako narzędzia komunikacji i współpracy
- Projektuj architekturę w sposób iteracyjny i przyrostowy . Minimalizuj prace projektowe

Definicja architektury – przykład 1

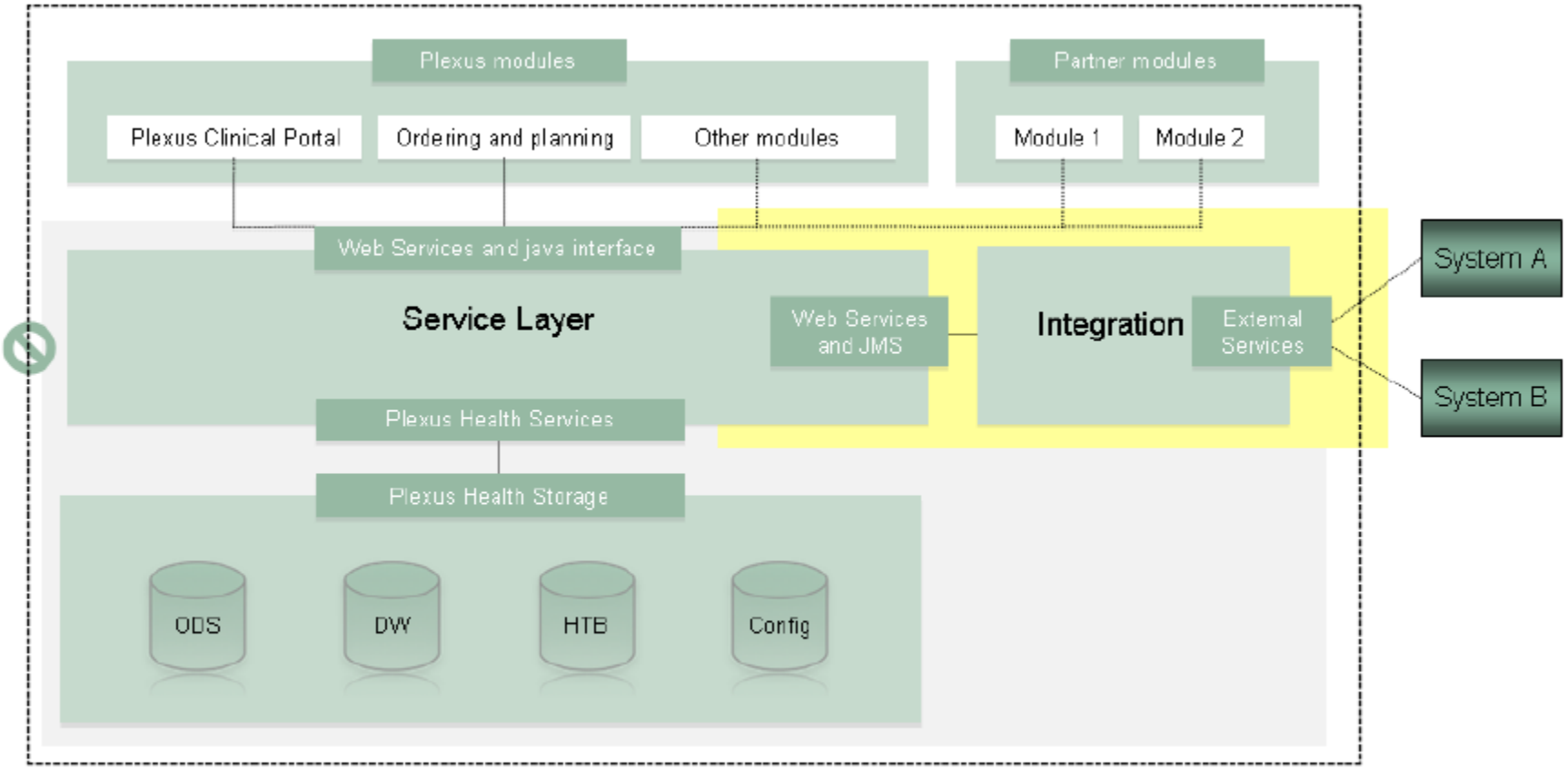


Fig. 1. Plexus architecture

Wszystkie przykłady pochodzą z [8]

Definicja architektury – przykład 2

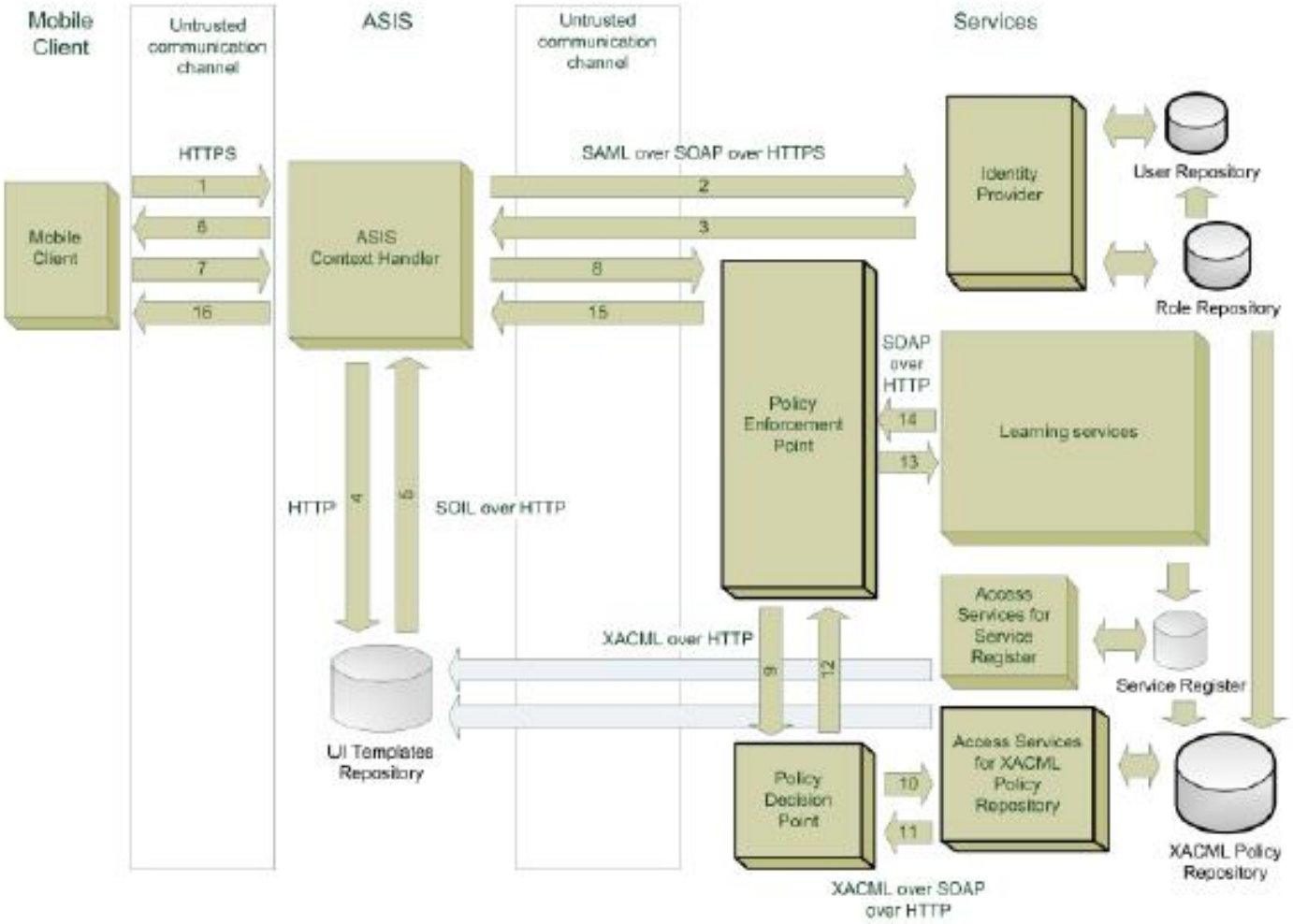


Fig. 1 Architecture and data flow of the extended secure MILES 2.0 e-learning system

Definicja architektury – przykład 3

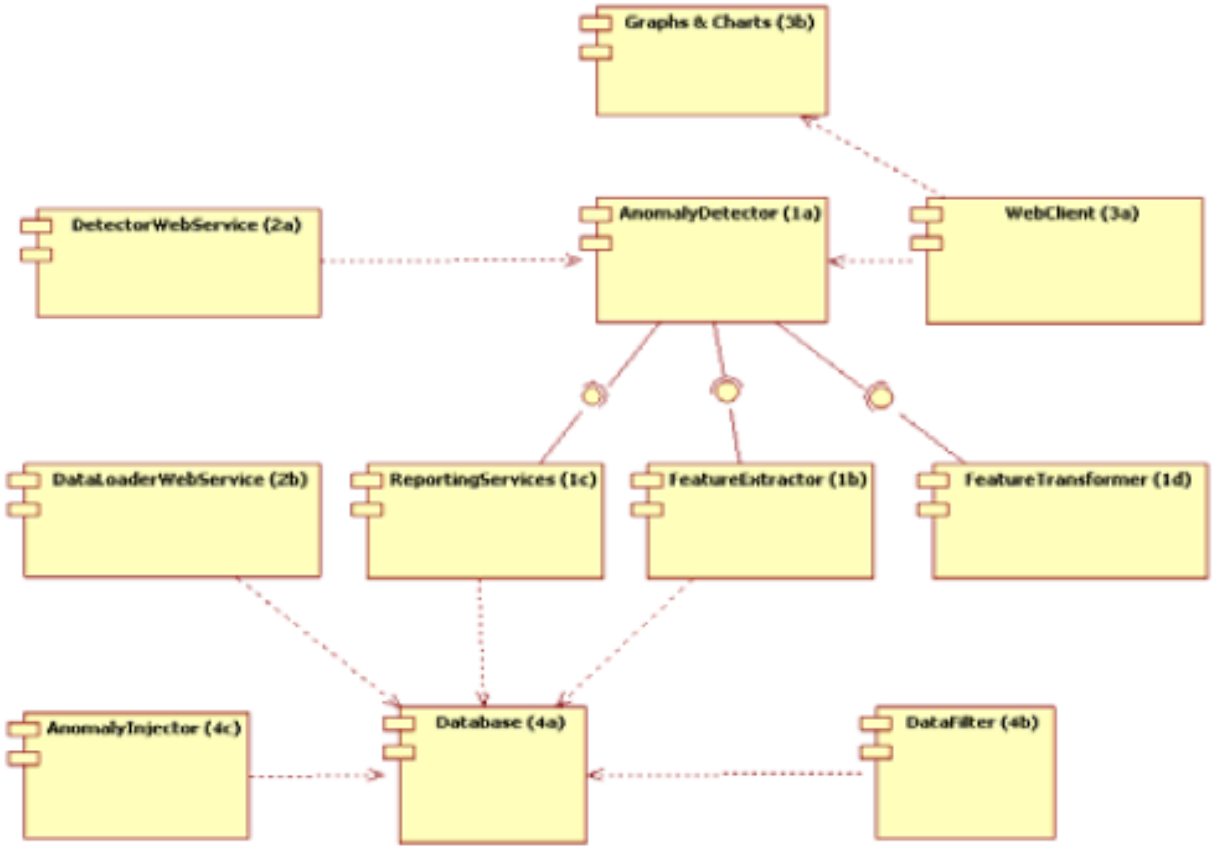


Fig. 4. Detailed Security Analyzer architecture

Definicja architektury – przykład 4

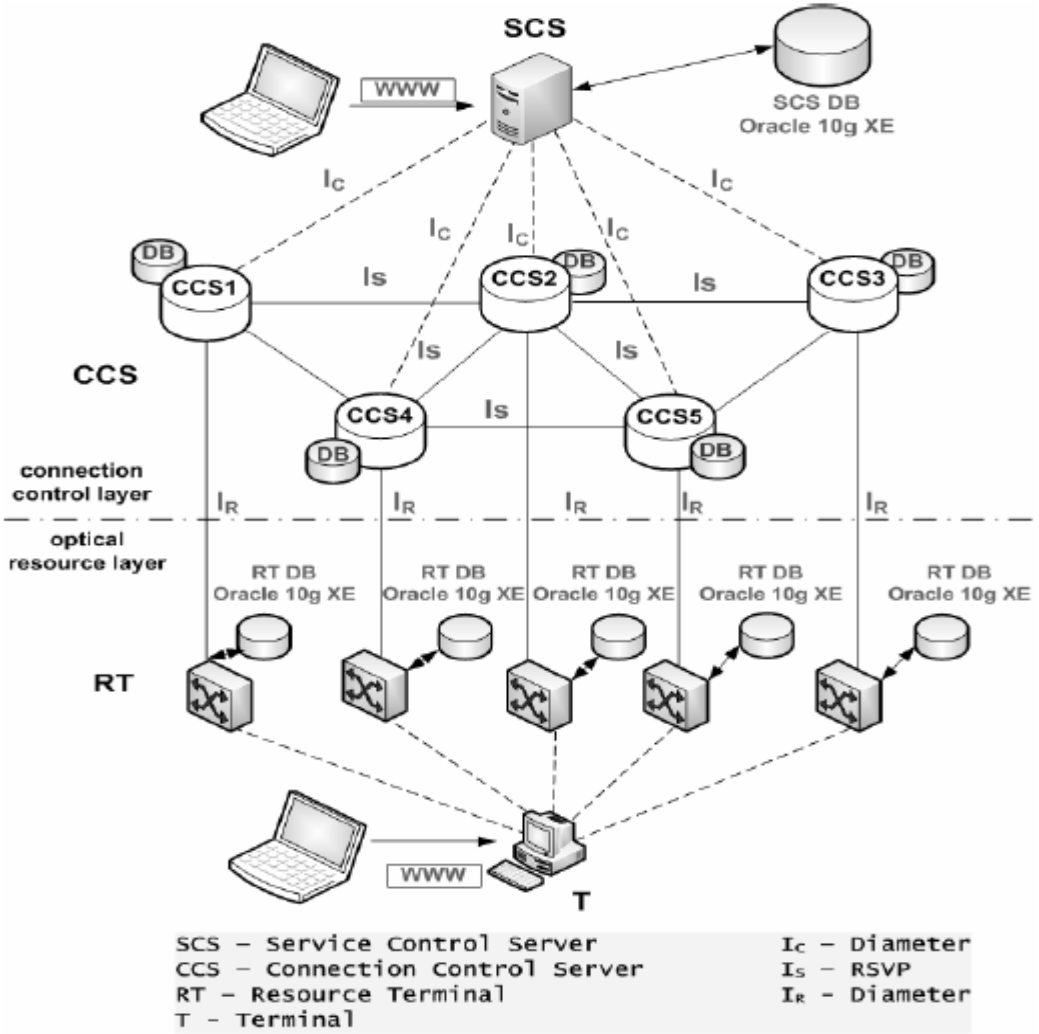


Fig. 1. The concept of ASON/GMPLS architecture testbed

Definicja architektury – przykład 6

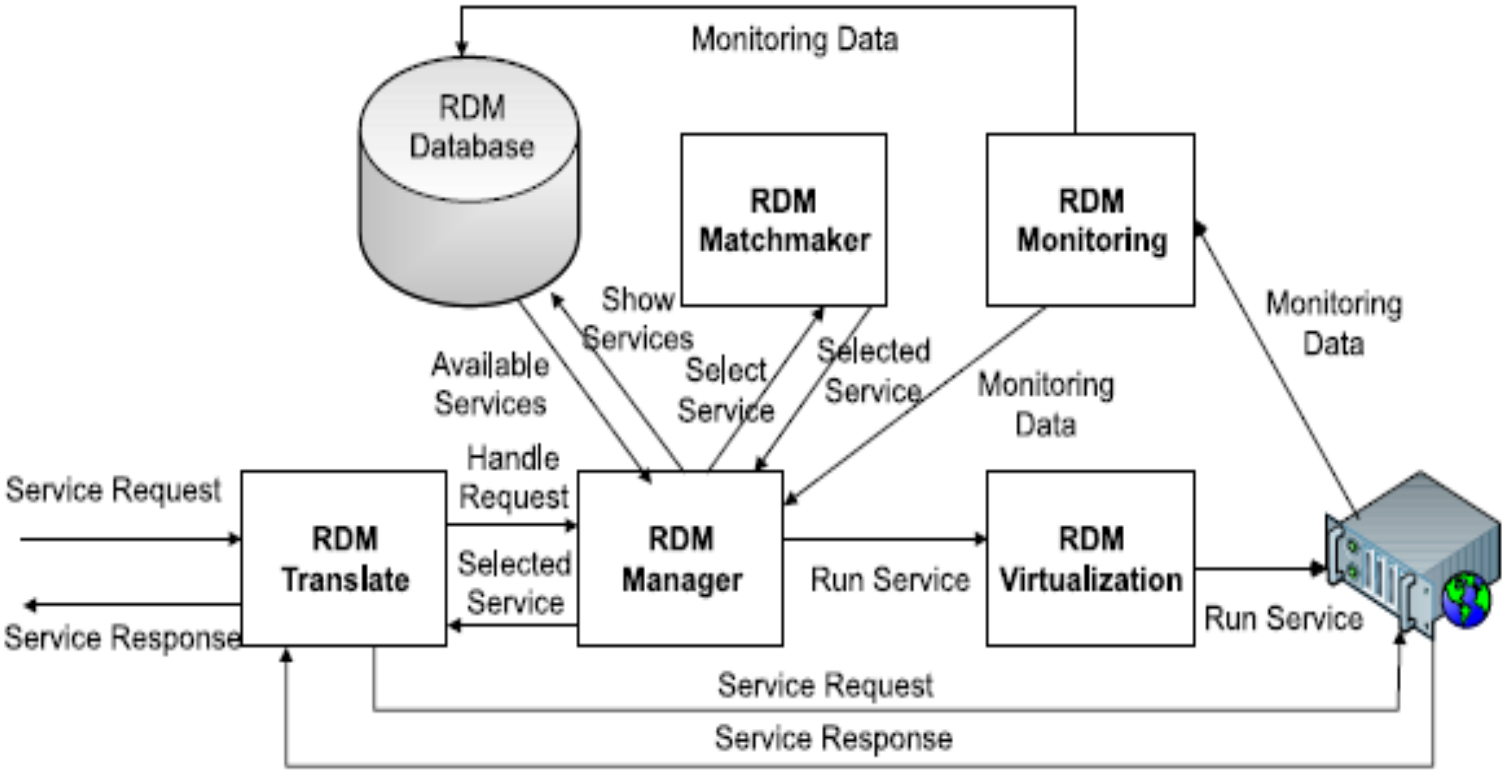


Fig. 1. The architecture of the RDM

Definicja architektury

- Nie jest możliwe opisanie własności funkcjonalnych i jakościowych złożonego systemu na jednym modelu, który będzie zrozumiały i wartościowy dla wszystkich udziałowców → potrzeba wprowadzenia perspektyw i widoków architektonicznych

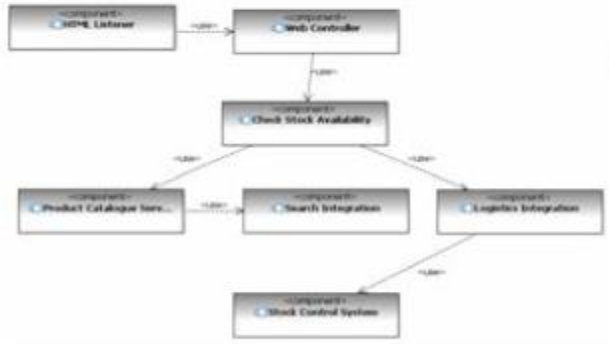
Perspektywy i widoki architektoniczne

- Perspektywa (Viewpoint) – **specyfikacja terminów**, w których będzie definiowany **widok**,
  - np. przypadki użycia i aktorzy
- Widok architektoniczny systemu (view) – **reprezentacja systemu** zgodna z wybraną **perspektywą**
  - np. aktor: Użytkownik, przypadek użycia: Zakładanie konta

Perspektywy i widoki architektoniczne

- *Will the system be available?*
- *Will the system let me create orders?*
- *Will the system be secure?*

Stakeholder —*has*→ Concerns ←*addresses*— Viewpoint



conforms to ↑

Filters and documents

View —→ Model —*abstraction of*→ System

Konsekwencje stosowania perspektyw i widoków

- Korzyści:
  - Separacja pojęć
  - Poprawa komunikacji między różnymi grupami udziałowców
  - Zarządzanie złożonością
- Wady:
  - Łatwo wprowadzić niespójności między widokami
  - Fragmentacja opisów (utrudnione zrozumienie całości)

Perspektywy i widoki architektoniczne, c.d.

- Klasyfikacja Rozanski, Woods:
  - Perspektywa kontekstowa
  - Perspektywa funkcjonalna
  - Perspektywa informacyjna
  - Perspektywa współbieżności
  - Perspektywa wytwarzania
  - Perspektywa rozmieszczenia (wdrożenia)
  - Perspektywa operacyjna

Klasyfikacja Rozanski. Perspektywa kontekstowa

- Definicja: opisuje zależności i interakcje między systemem i jego otoczeniem (ludzie, systemy, urządzenia)
- Co opisuje:
  - Zakres systemu i jego odpowiedzialności (funkcjonalne) opisane ogólnie
  - Identyfikacja zewnętrznych bytów i wymienianych danych (zawartość, zakres, znaczenie)
  - Charakterystyka zewnętrznych bytów i ich interfejsy (żądania, zdarzenia)
- Modele:
  - *Diagramy kontekstowe* (np. okrojona wersja diagramu przypadków użycia; diagramy kontekstowe SysML, diagramy DFD pierwszego poziomu)
  - Scenariusze interakcji (kolejność zdarzeń, komunikatów, ograniczenia czasowe) – opisy tekstowe, diagramy sekwencji, diagramy aktywności
- Udziałowcy:
  - Wszyscy, ale zwłaszcza: nabywcy, wytwórcy, administratorzy

Klasyfikacja Rozanski. Perspektywa kontekstowa. Diagram kontekstowy – przykład 1

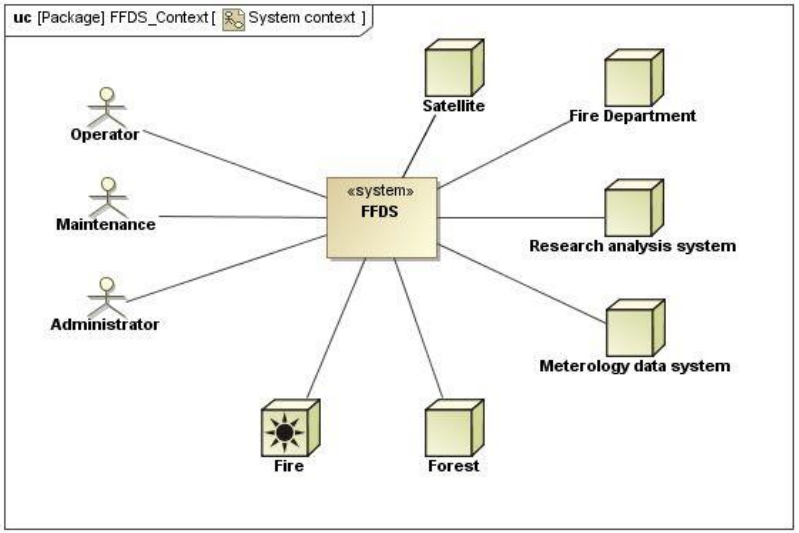
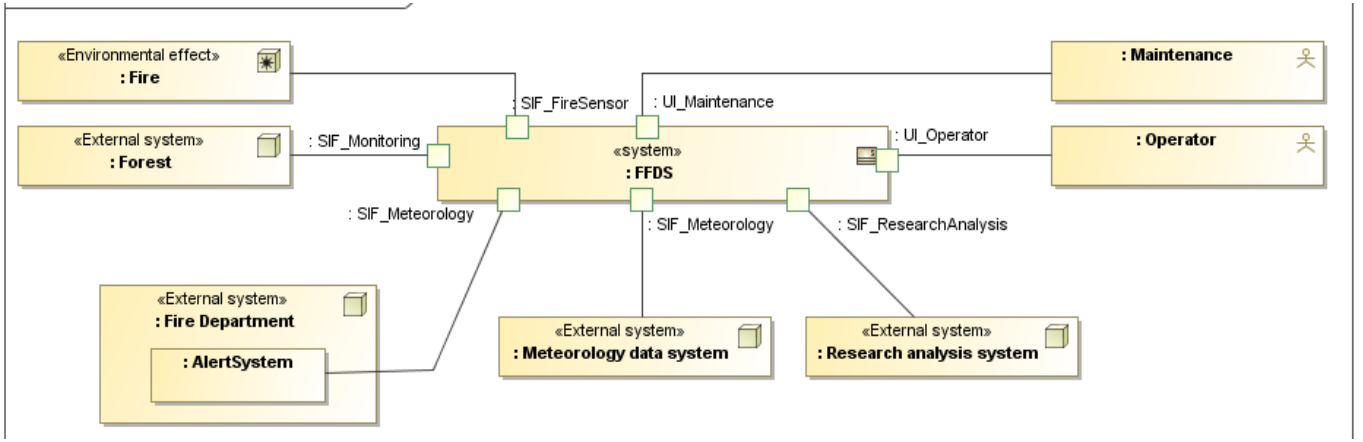


Diagram przypadków użycia UML

Diagram kontekstowy SysML



http://model-based-systems-engineering.com/tag/sysmod/

Klasyfikacja Rozanski. Perspektywa kontekstowa. Diagram kontekstowy – przykład 2

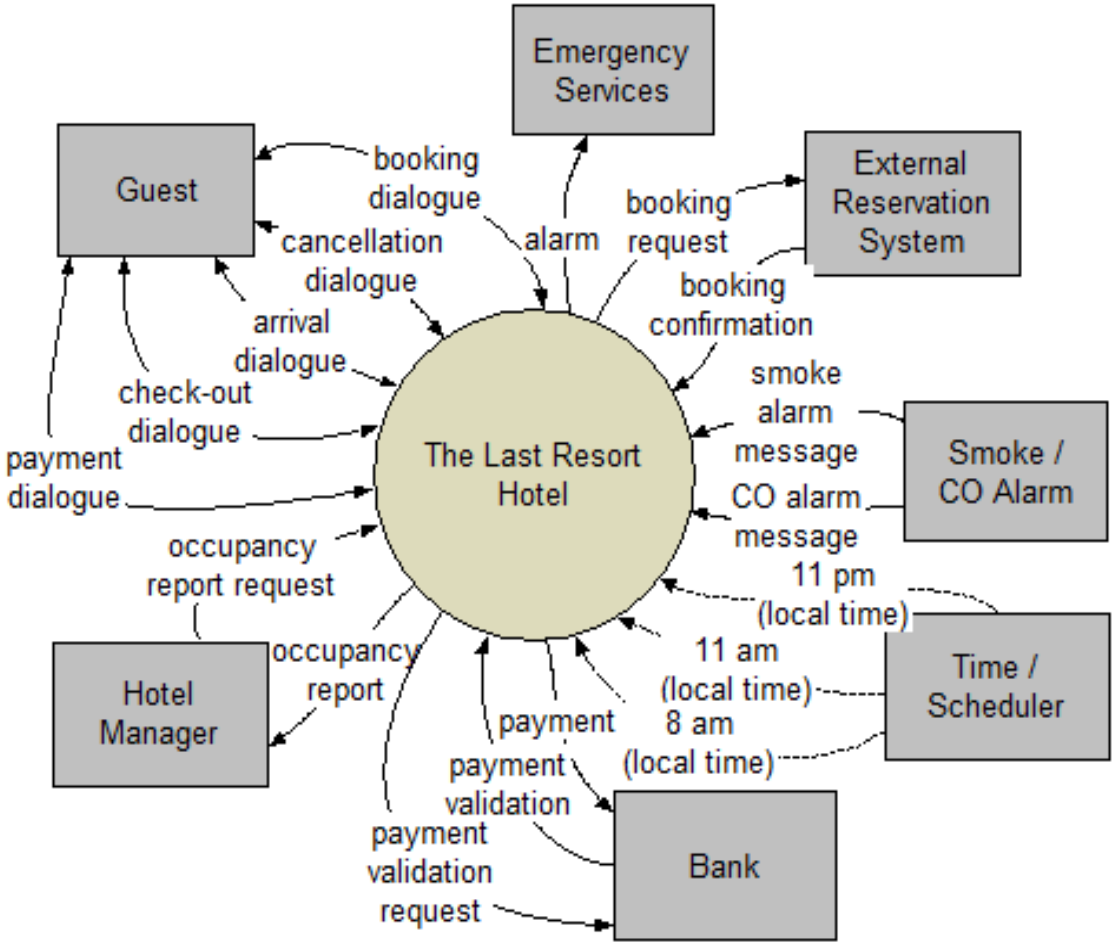


Diagram DFD, http://en.wikipedia.org/wiki/File:ContextDiagram_LastResortHotel.png

Klasyfikacja Rozanski. Perspektywa kontekstowa. Diagram kontekstowy – przykład 3

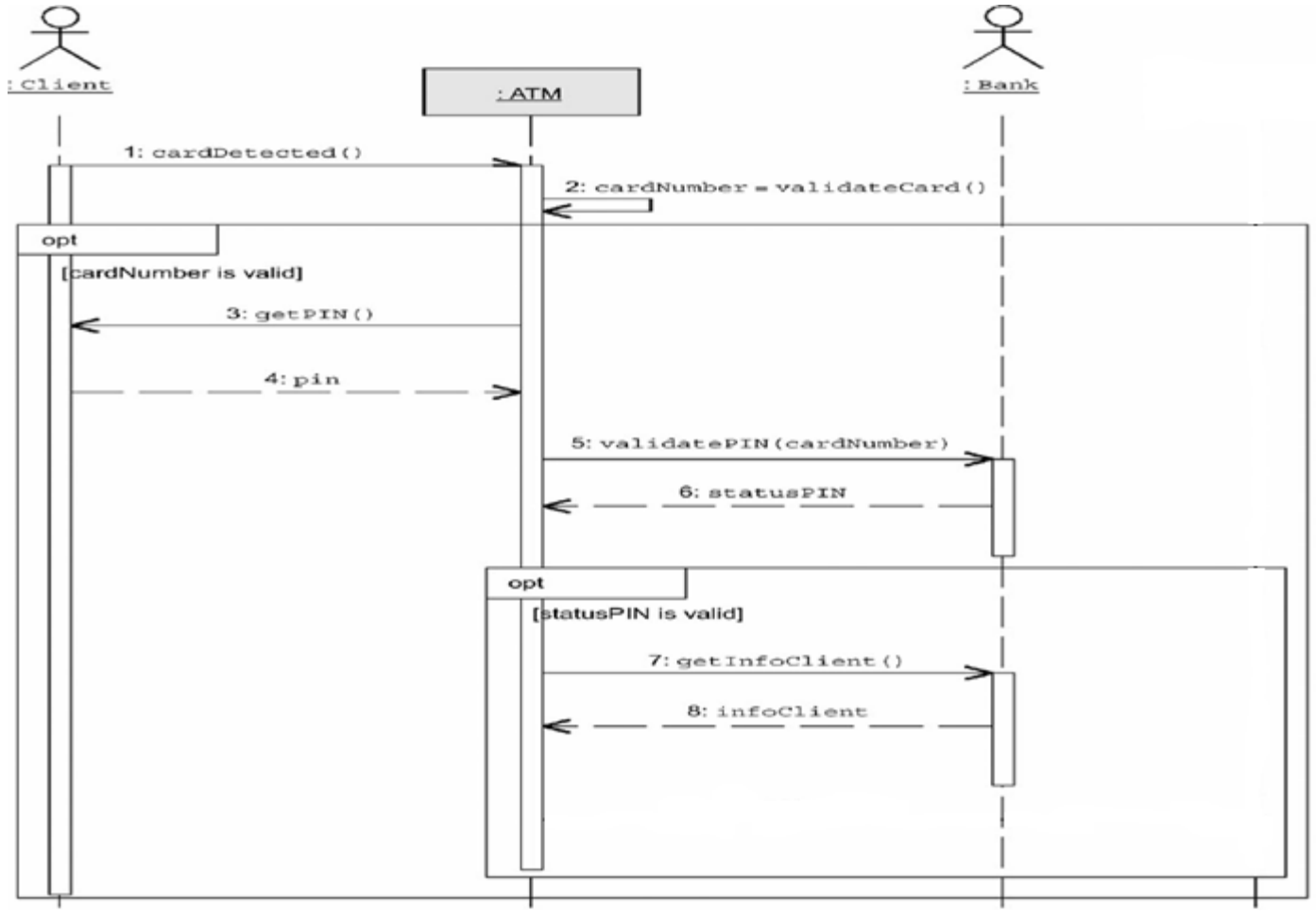


Diagram kontekstowy po usunięciu Transactions
https://www.researchgate.net/figure/220458560_fig2_FIGURE-2-A-sequence-diagram-of-the-Register-use-case

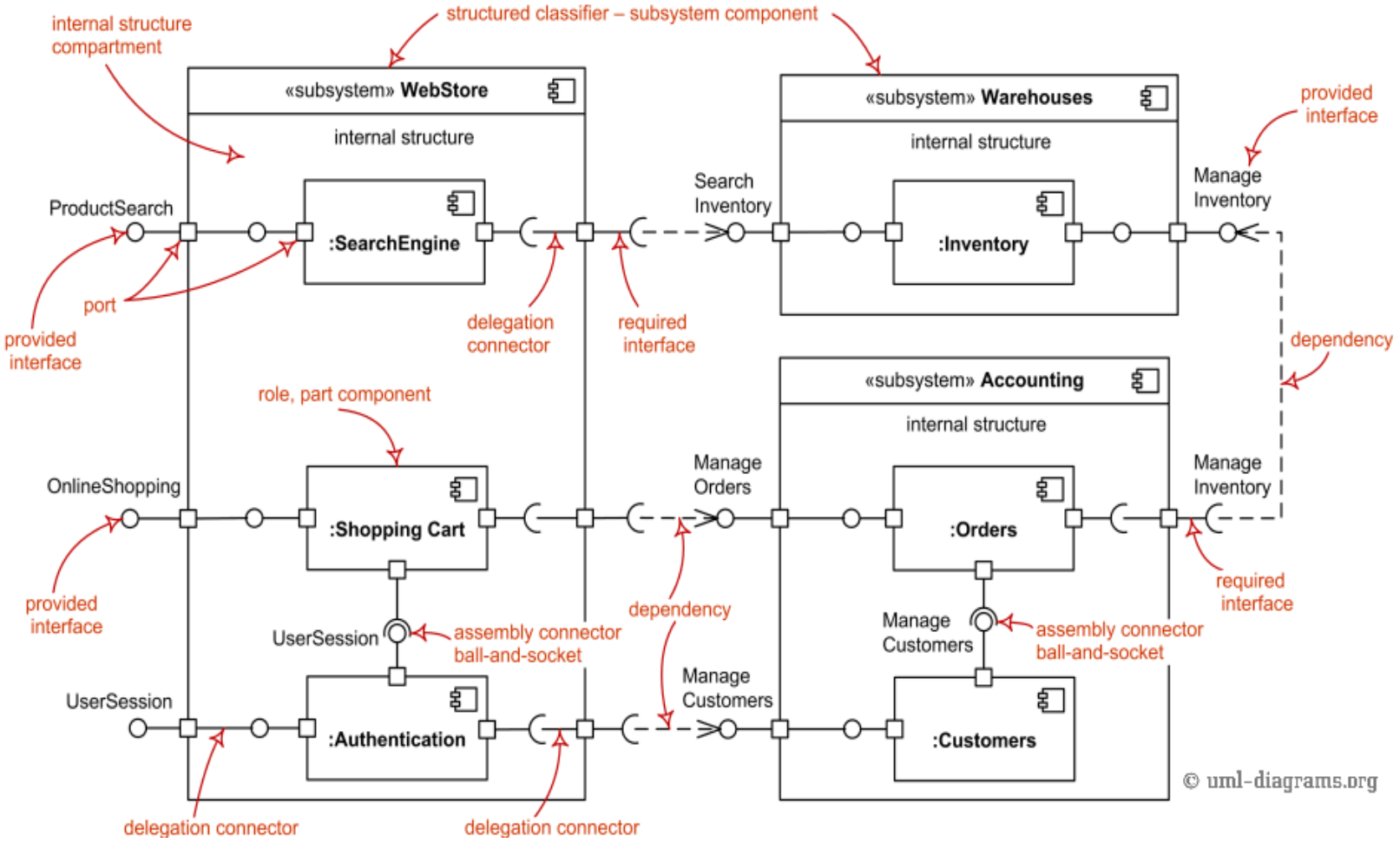
Klasyfikacja Rozanski. Perspektywa funkcjonalna

- Definicja: elementy **czasu wykonania (moduły funkcjonalne/ podsystemy)**, ich odpowiedzialności, interfejsy i podstawowe interakcje między nimi
- Moduł funkcjonalny – (złożony) byt programowy (np. podsystem) zwykle kompilowany /pakowany do jednego artefaktu, np. plku *.ear, *.war, *.dll, *.jar, *.exe, lub reprezentujący usługę sieciową
- Co opisuje
  - *Podział na elementy funkcjonalne wraz z definicją ich interfejsów*
  - Wewnętrzną strukturę elementów (white-box)
  - Sposób realizacji funkcjonalności przez współpracujące elementy
- Modele:
  - Model struktury funkcjonalnej (dekompozycja na moduły funkcjonalne) → diagramy komponentów,
  - Wewnętrzna struktura elementów → diagramy komponentów
  - Sposób realizacji funkcjonalności → diagramy sekwencji, diagramy aktywności
- Udziałowcy:
  - Wszyscy, a zwłaszcza wytwórcy

Klasyfikacja Rozanski. Perspektywa funkcjonalna, c.d.

- Aktywności:
  - Identyfikacja elementów funkcjonalnych
  - Przypisanie odpowiedzialności elementom – opis tekstowy lub śladowanie do wymagań w narzędziu
  - Projektowanie interfejsów
    - W UML
    - W językach programowania
    - W języku definicji interfejsów (w zależności od stosowanej technologii), np. IDL (np. COM), WSDL dla usług
    - Opis wymienianych komunikatów, np. w JSON, XML czy XML Schema
    - Projektowanie połączeń – typy, charakterystyki jakościowe
  - Definicja i analiza kluczowych scenariuszy

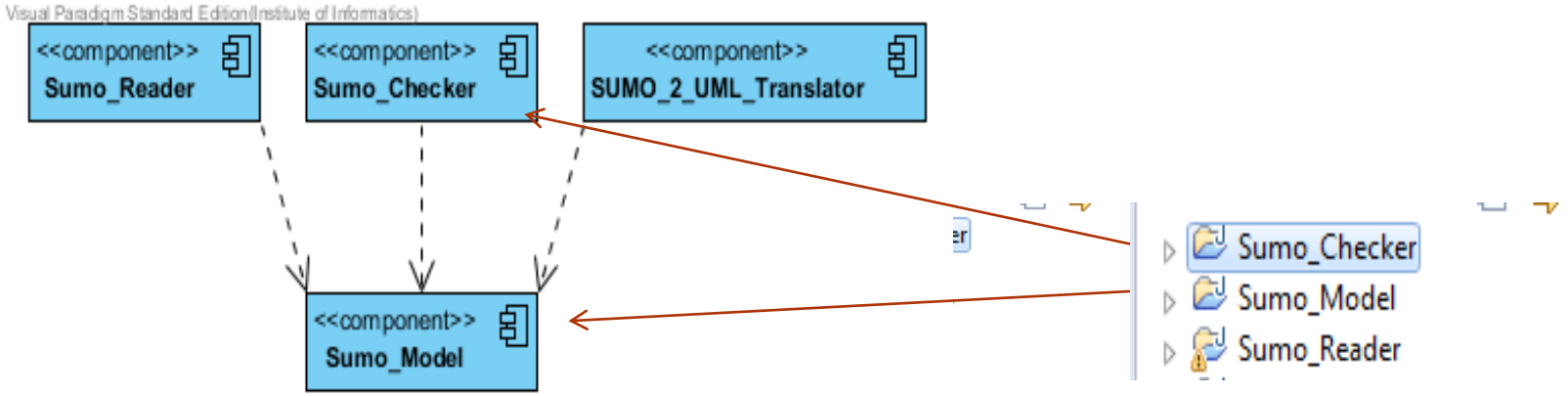
Klasyfikacja Rozanski. Perspektywa funkcjonalna. Model struktury funkcjonalnej – przykład 1



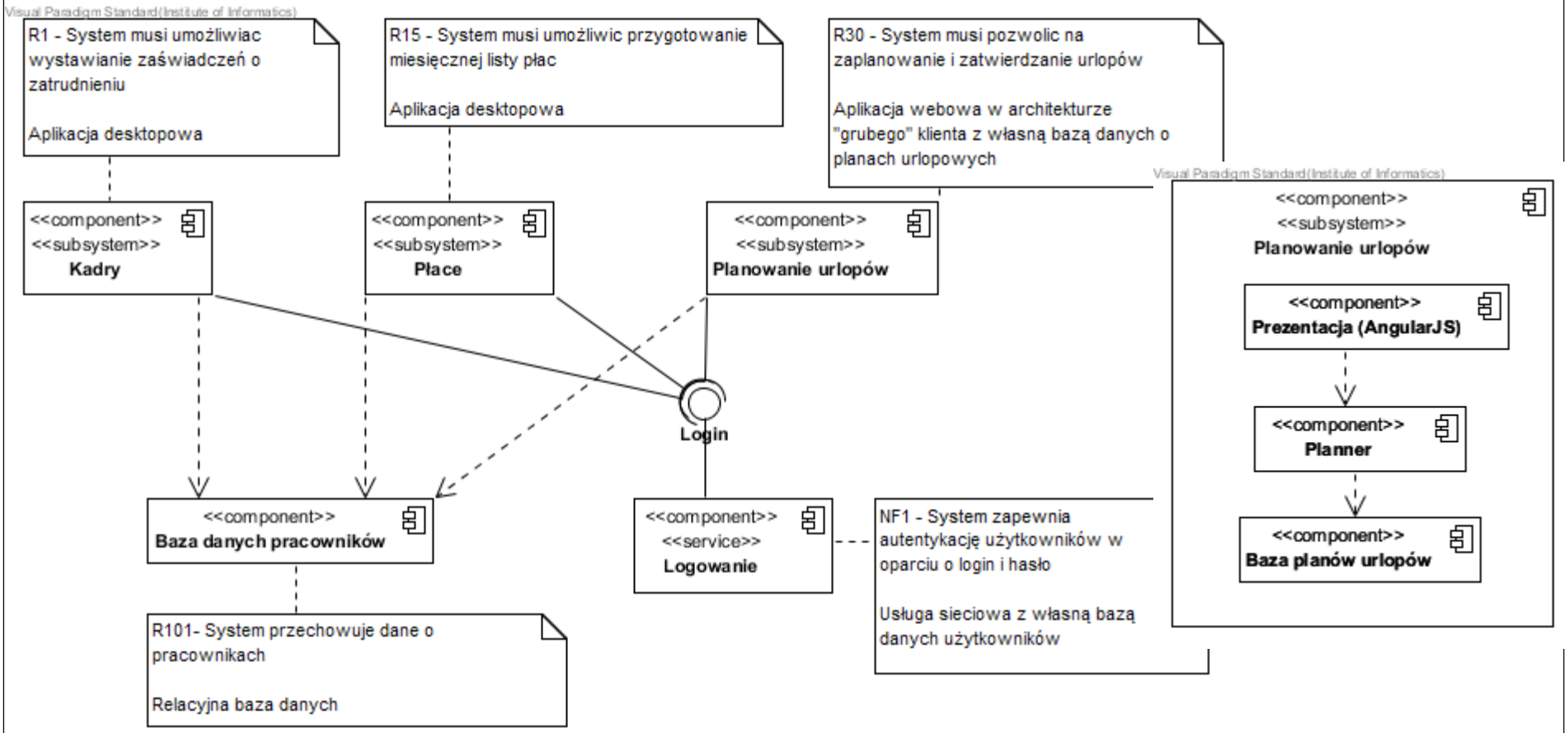
http://www.uml-diagrams.org/component-diagrams.html

Klasyfikacja Rozanski. Perspektywa funkcjonalna. Model struktury funkcjonalnej – przykład 2

- Translator: Sumo → UML:
  - Sumo_Model – meta-model SUMO
  - Sumo_Reader – wczytanie plików Sumo; parser leksykalny; zbudowanie instancji meta-modelu
  - Sumo_Chcecker – sprawdzenie poprawności semantycznej (statycznej)
  - SUMO_2_UML_Translator – tłumaczenie wybranych elementów Sumo do UML



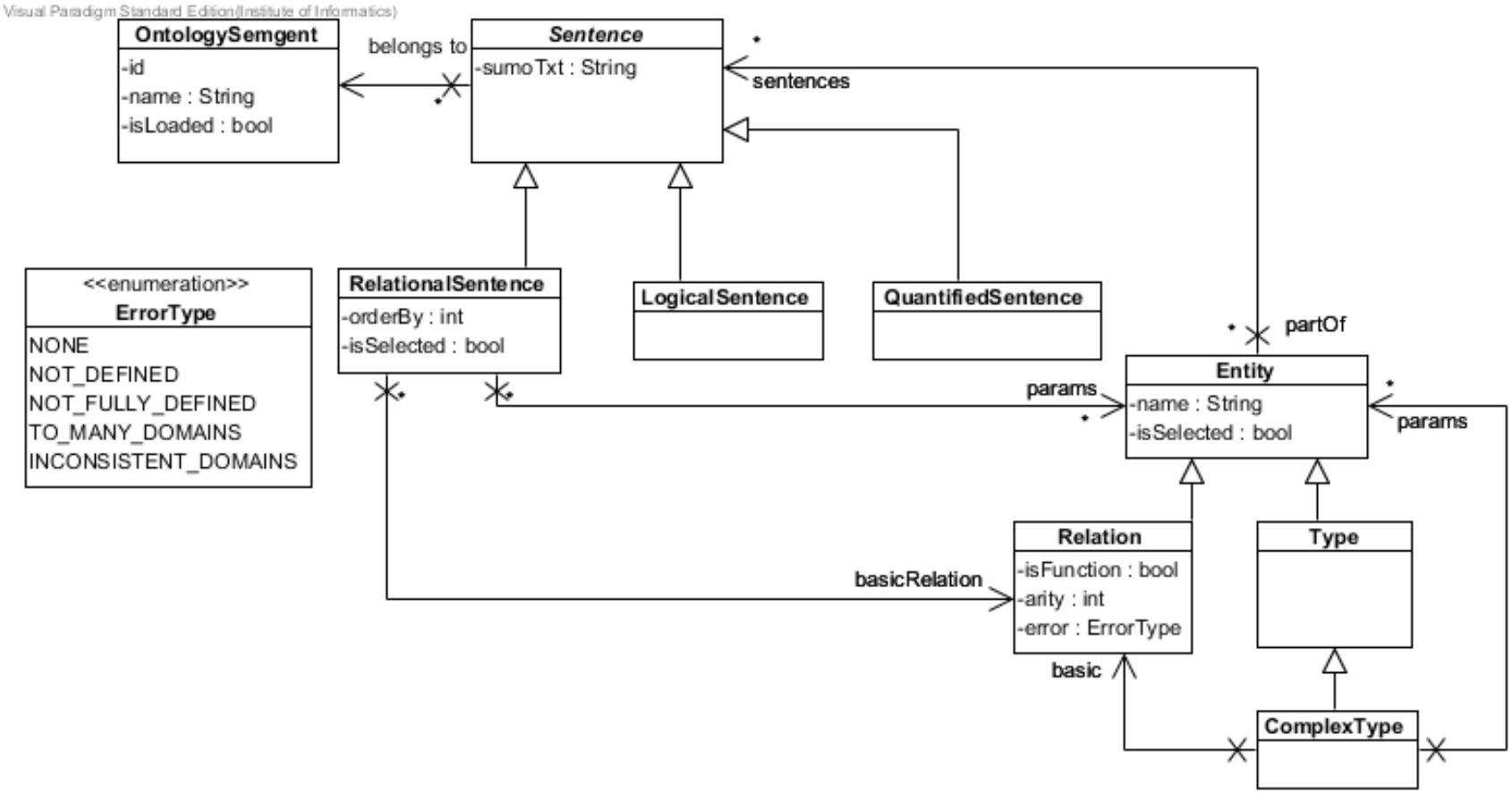
Klasyfikacja Rozanski. Perspektywa funkcjonalna. Model struktury funkcjonalnej - przykład 3



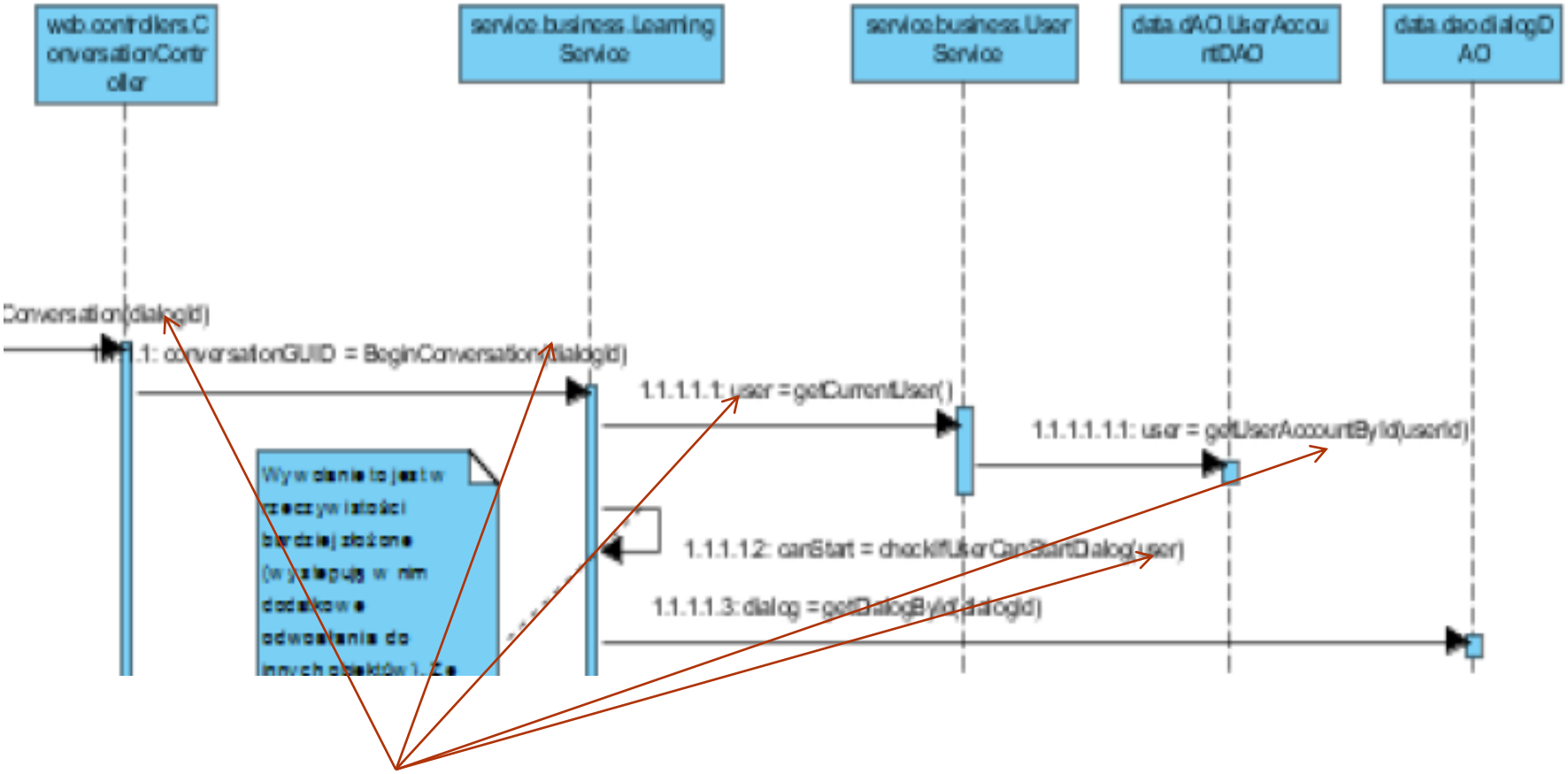
Klasyfikacja Rozanski. Perspektywa informacyjna

- Definicja: Opisuje sposób w jaki system zapamiętuje, przetwarza i przesyła informacje
- Co opisuje:
  - *Struktura informacji*
  - Własności informacji
  - *Modele przechowywania, np. baza relacyjna, baza NoSQL*
  - Cykl życia i przepływ informacji, w tym mechanizmy transakcji
- Modele:
  - Statyczne modele struktury informacji – diagramy klas, ERD
  - Modele przepływu informacji – diagramy sekwencji, DFD
  - Modele cyklu życia informacji – diagramy stanów
  - Modele własności informacji – opisy tekstowe
- Udziałowcy:
  - Głównie użytkownicy końcowi, zamawiający, wytwórcy, testerzy

Klasyfikacja Rozanski. Perspektywa informacyjna. Statyczne modele struktury informacji - przykład

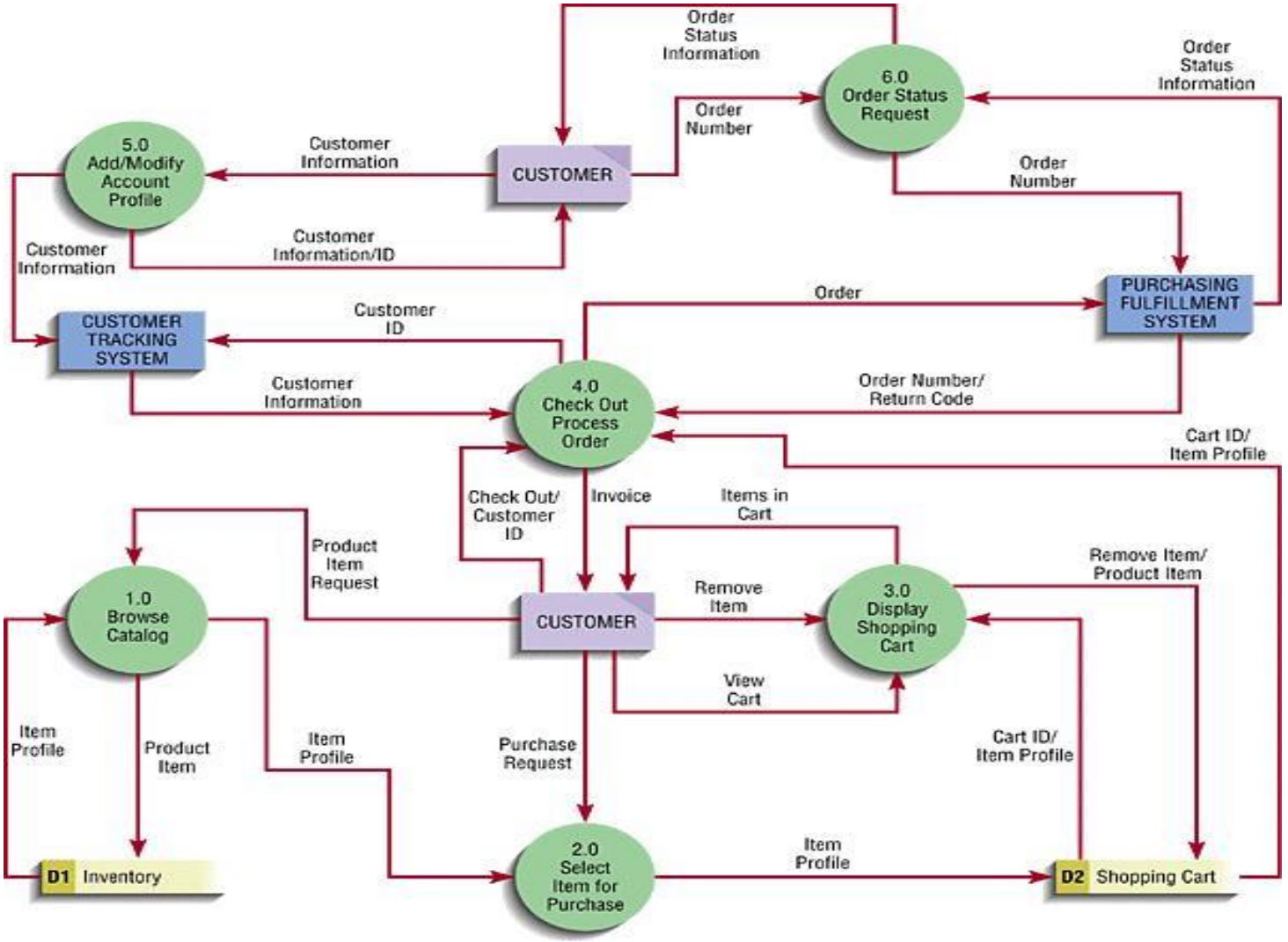


Klasyfikacja Rozanski. Perspektywa informacyjna. Modele przepływu informacji – UML przykład diagramu sekwencji

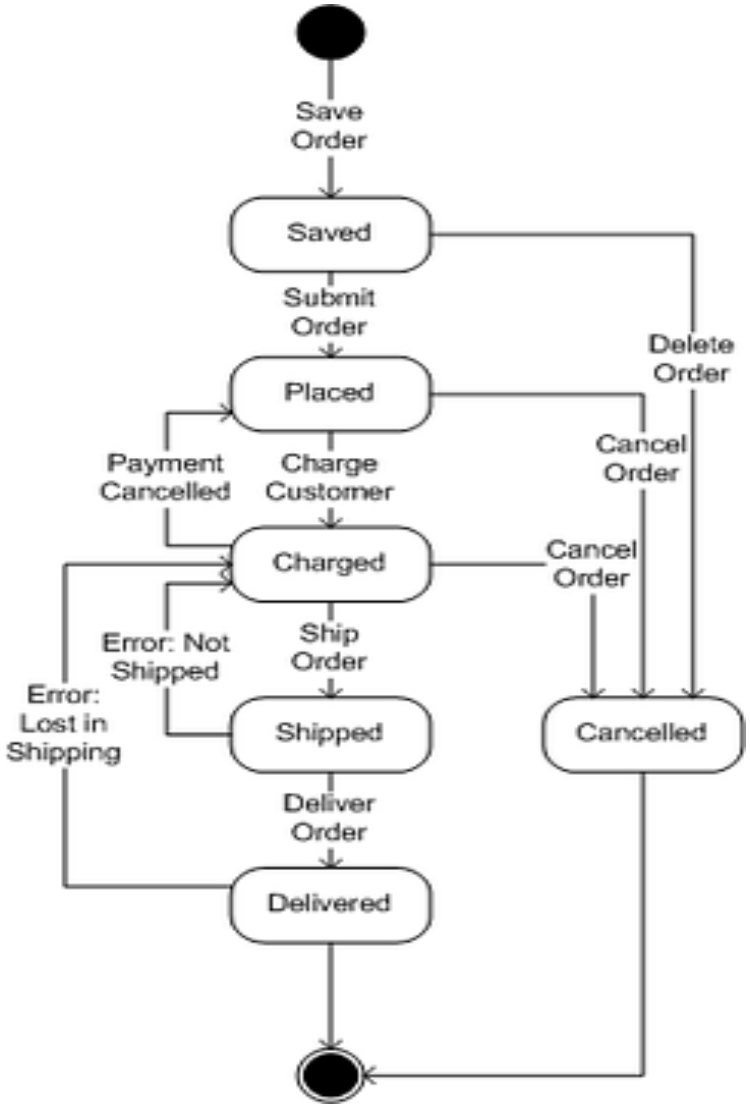


Elementy związane z przepływem danych

Klasyfikacja Rozanski. Perspektywa informacyjna. Modele przepływu informacji – DFD, przykład

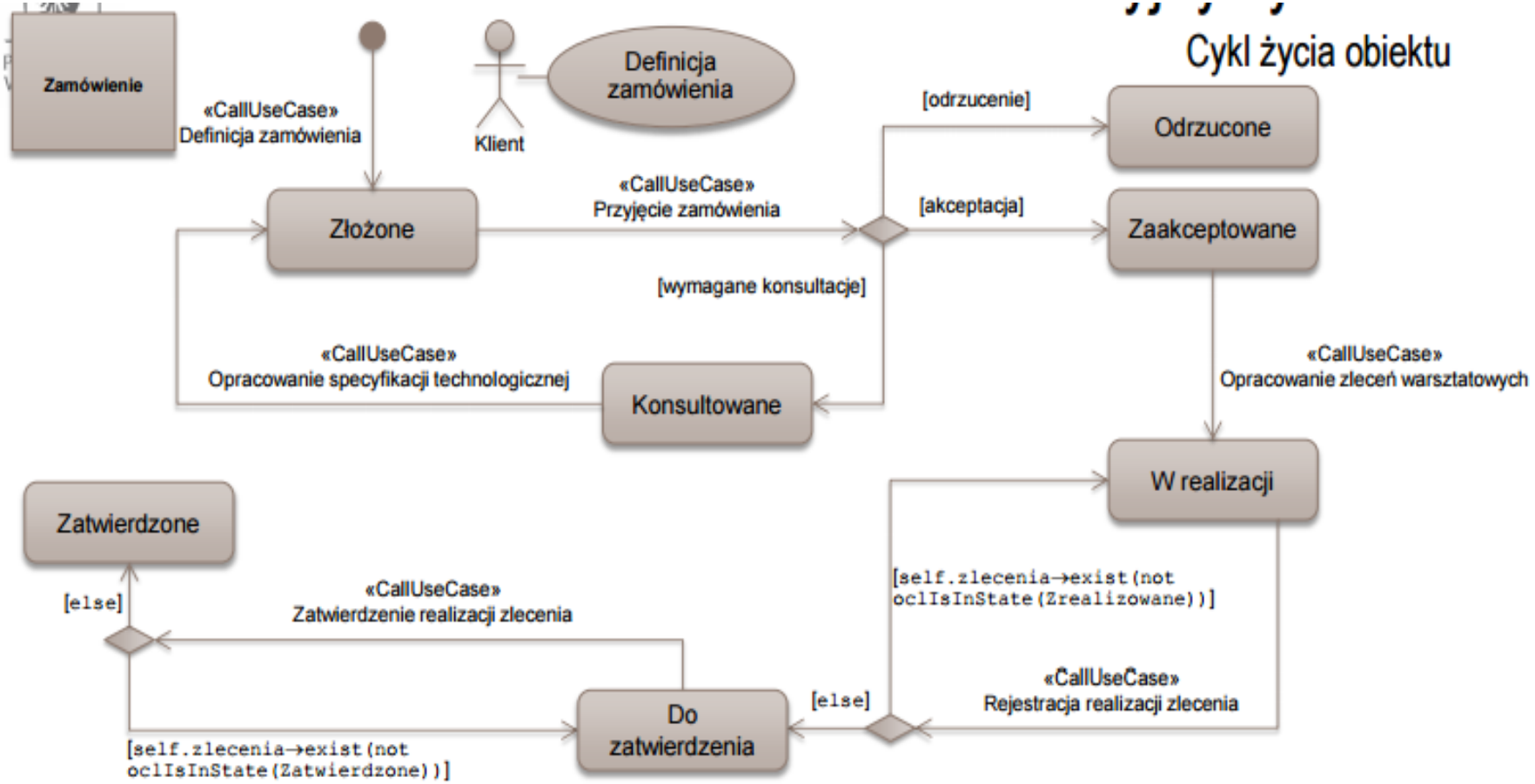


Klasyfikacja Rozanski. Perspektywa informacyjna. Modele cyklu życia informacji – diagramy stanów UML



http://tynerblain.com/blog/2007/03/29/uml-statechart-substates/

Klasyfikacja Rozanski. Perspektywa informacyjna. Modele cyklu życia informacji – diagramy stanów UML



Materiały z PO, A. Walkowiak

Klasyfikacja Rozanski. Perspektywa informacyjna. Modele własnosci

- Role: Owner (ostateczna wartość elementu), Creator, Updater, Deleter, Reader, Copy, Validater, kombinacja powyższych

(Sub)System\ Item	Customer	Product	Order
Catalog	None	Owner	None
Purchasing	Reader	Updater	Owner
Delivery	Copy	Reader	Reader

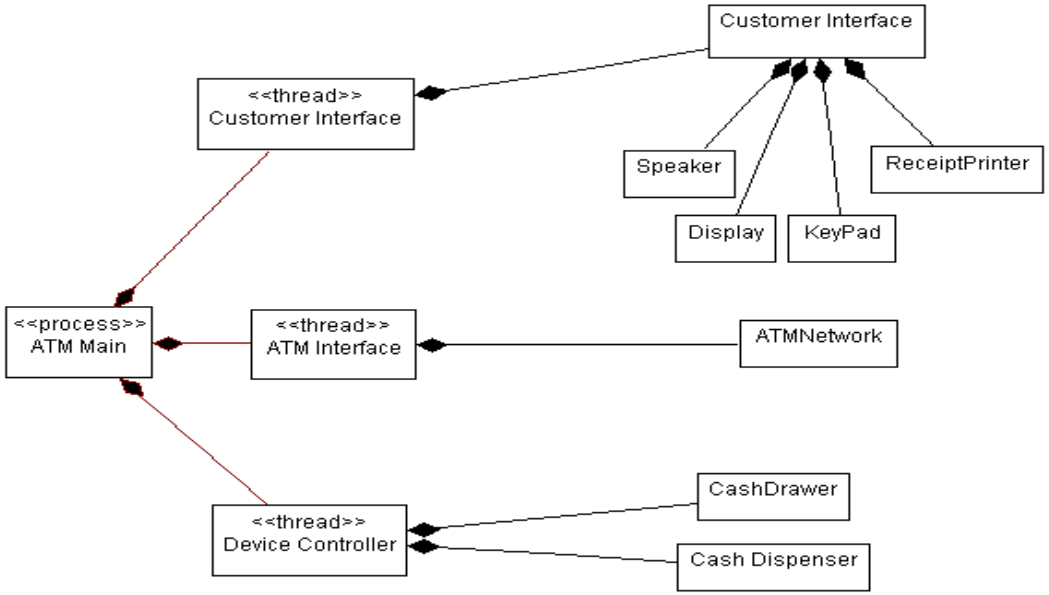
Klasyfikacja Rozanski. Perspektywa współbieżności

- Definicja: Opisuje strukturę współbieżności w systemie, odwzorowanie elementów funkcjonalnych do jednostek współbieżności oraz komunikację elementów współbieżnych (tworzenie, usuwanie, synchronizacja). Opisuje **elementy czasu wykonania** (np. klasy aktywne) **działające współbieżnie**; ich współpracę i synchronizację
- Co opisuje:
  - Strukturę zadań (lub innych elementów współbieżnych jak wątki, procesy)
  - Odwzorowanie elementów funkcjonalnych do zadań
  - Komunikację międzyprocesową
  - Mechanizmy synchronizacji
  - Mechanizmy tworzenia i usuwania elementów współbieżnych
- Modele:
  - Modele współbieżności
- Udziałowcy:
  - Wytwórcy, testerzy, wybrani administratorzy

Klasyfikacja Rozanski. Perspektywa współbieżności, c.d.

- Klasa pasywna (UML): klasa przechowująca dane i świadcząca usługi innym klasom
- Klasa aktywna (UML): inicjuje i nadzoruje przepływ aktywności:
  - Wątek (thread): „lekki” proces
  - Proces: jednostka współbieżności i wykonywania operacyjnym

Figure 13.14 - Active class



Klasyfikacja Rozanski. Perspektywa współbieżności. Wykorzystywane notacje, c.d.

- Wykorzystywane notacje, UML

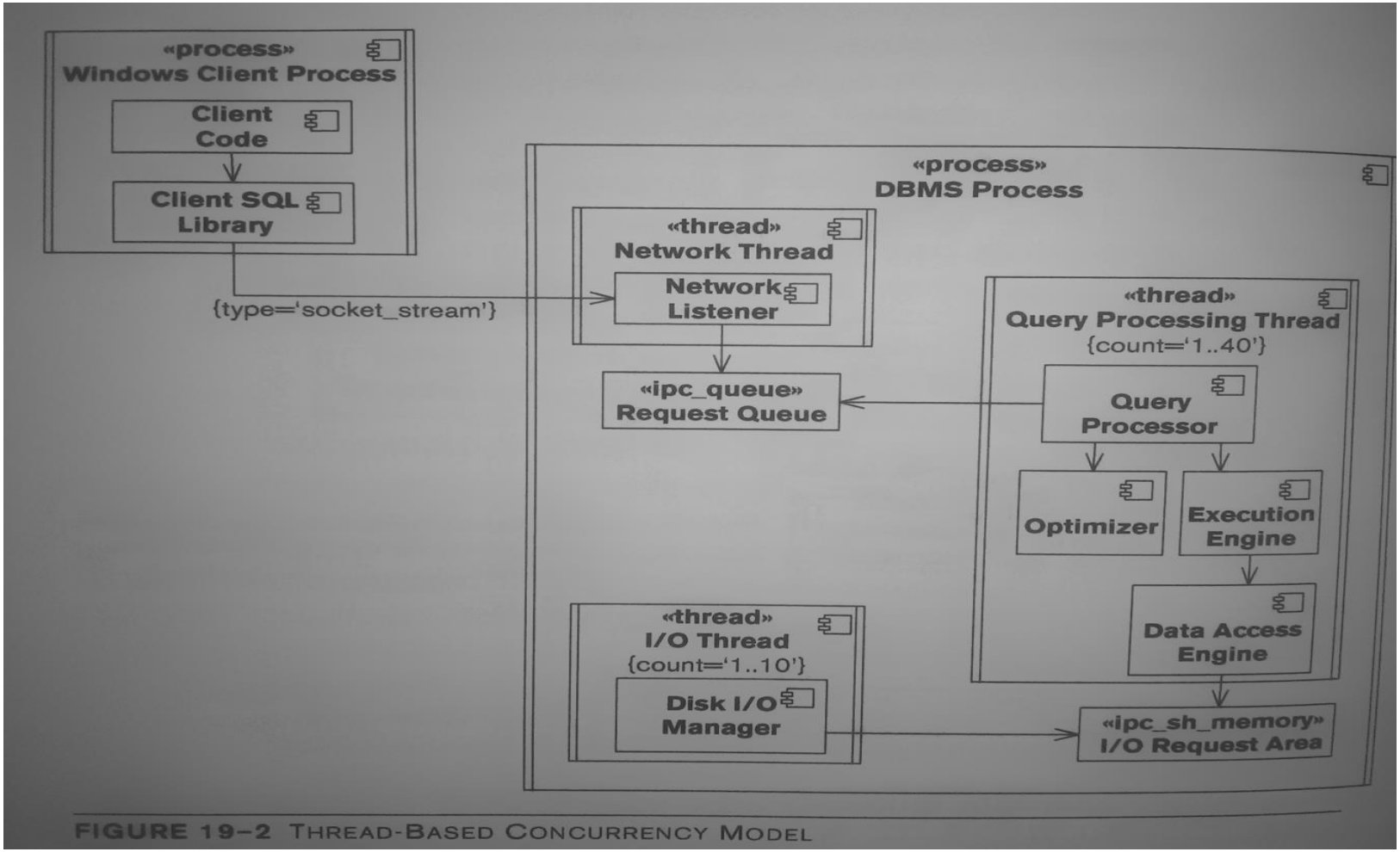
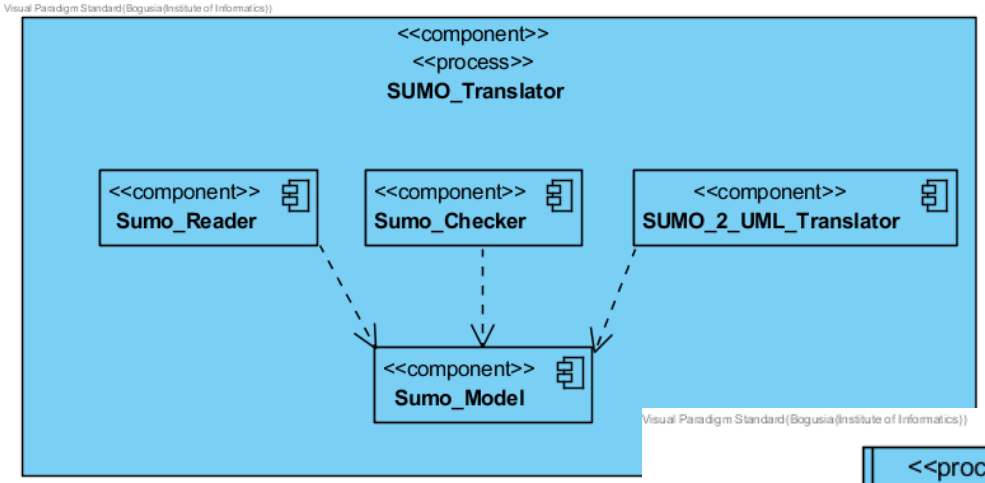
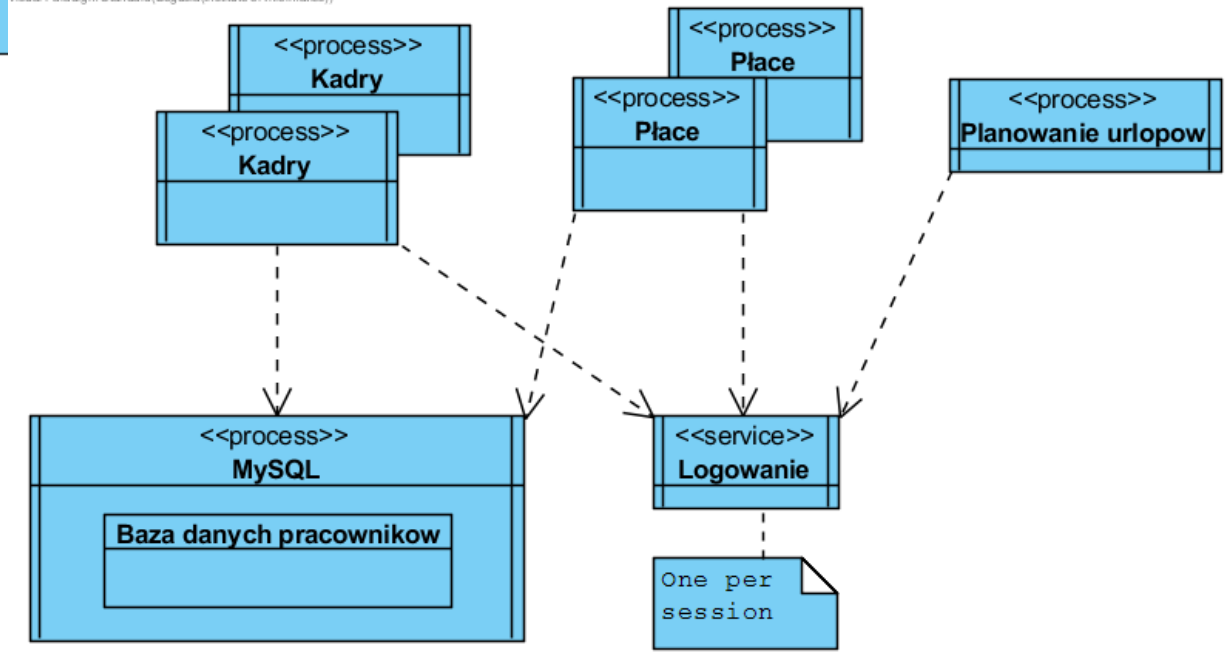


FIGURE 19–2 THREAD-BASED CONCURRENCY MODEL

Klasyfikacja Rozanski. Perspektywa współbieżności. Przykład

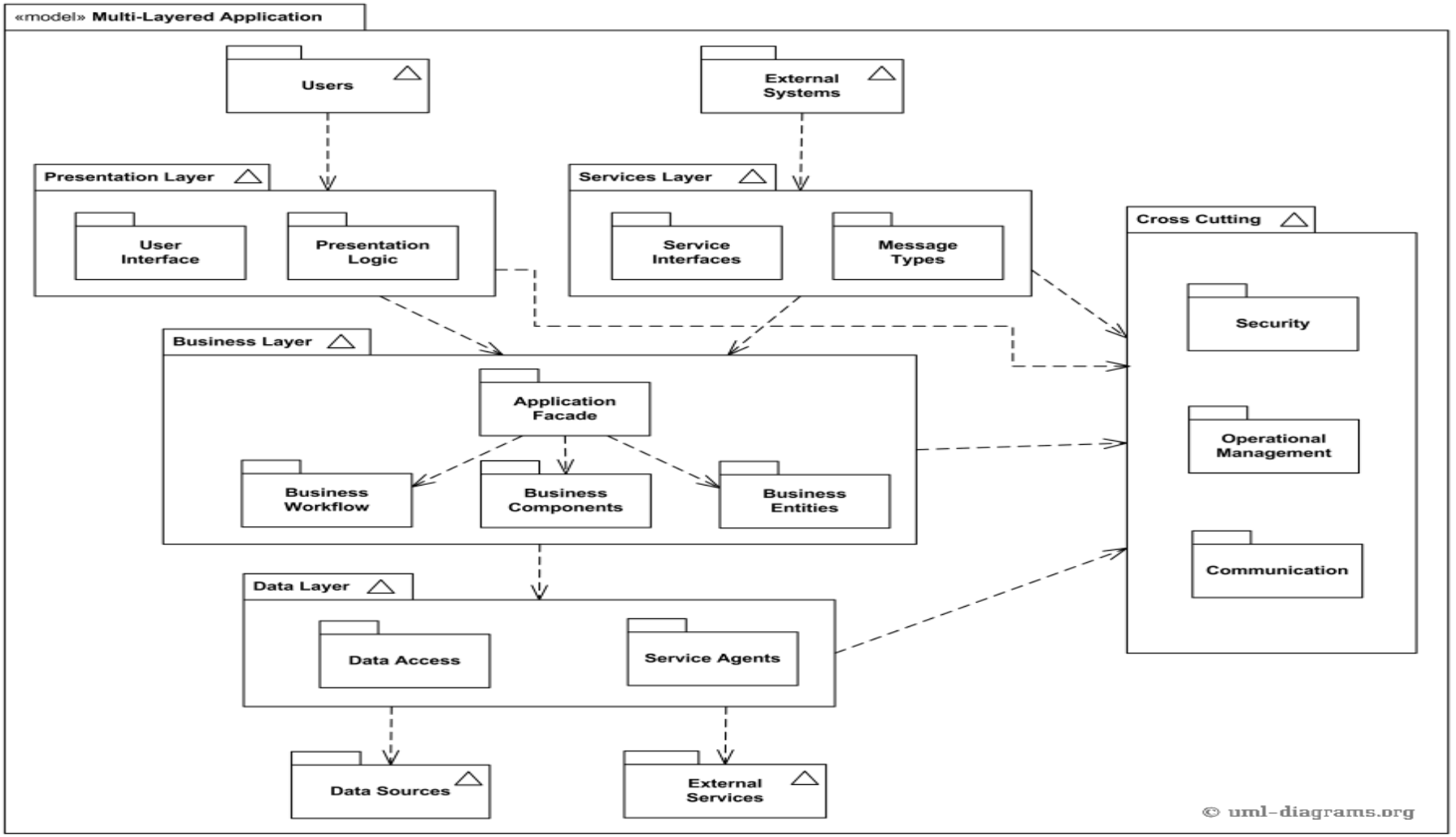




Klasyfikacja Rozanski. Perspektywa wytwarzania

- Definicja: Struktura kodu i jego zależności. Wsparcie dla procesu produkcji oprogramowania (w tym zarządzania konfiguracją).
- Co opisuje:
  - Organizację modułów/pakietów
  - Zasady „branchowania” w repozytorium
- Modele:
  - Model struktury modułów - diagram pakietów
  - Model rozwoju produktu
- Udziałowcy:
  - Wytwórcy i testerzy

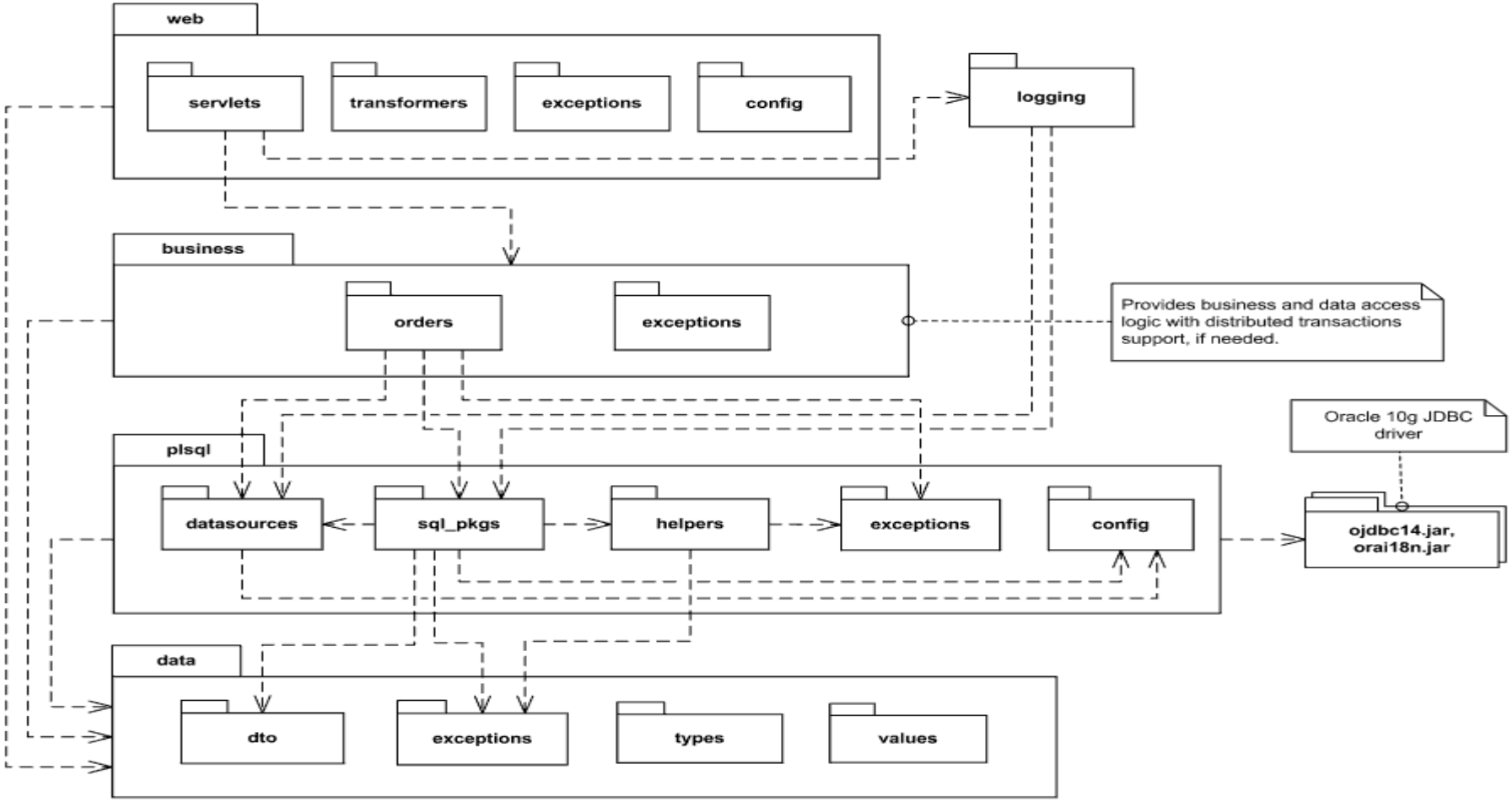
Klasyfikacja Rozanski. Perspektywa wytwarzania. Diagram pakietów - przykład



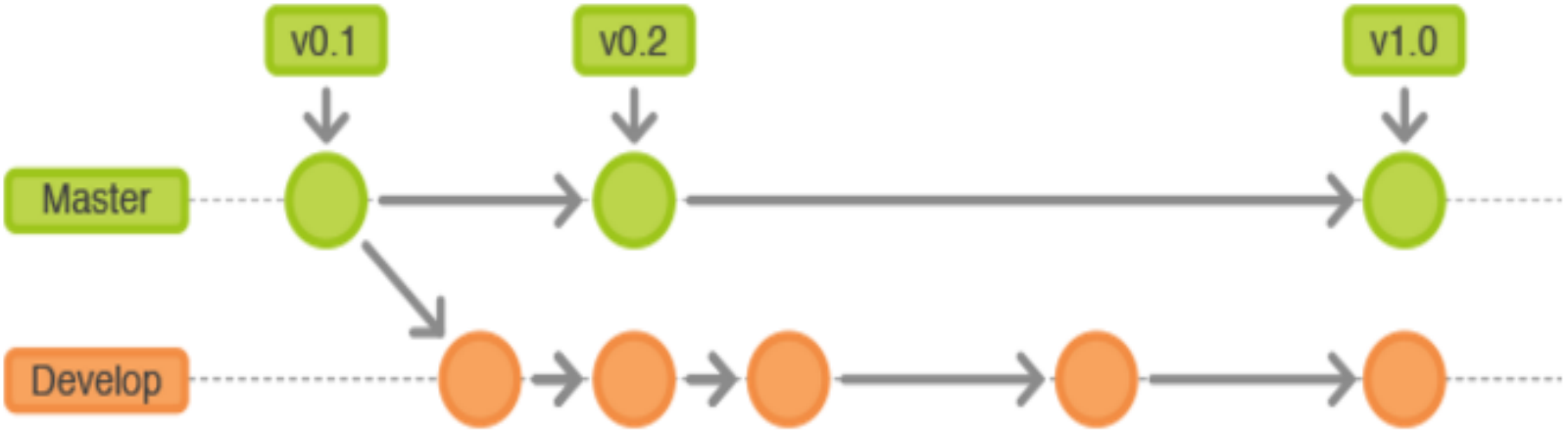
http://www.uml-diagrams.org/package-diagrams-overview.html

Klasyfikacja Rozanski. Perspektywa wytwarzania

- *Package diagram of some multi-layered web architecture from* http://www.uml-diagrams.org/package-diagrams-examples.html



Klasyfikacja Rozanski. Perspektywa wytwarzania. Model rozwoju produktu - przykład



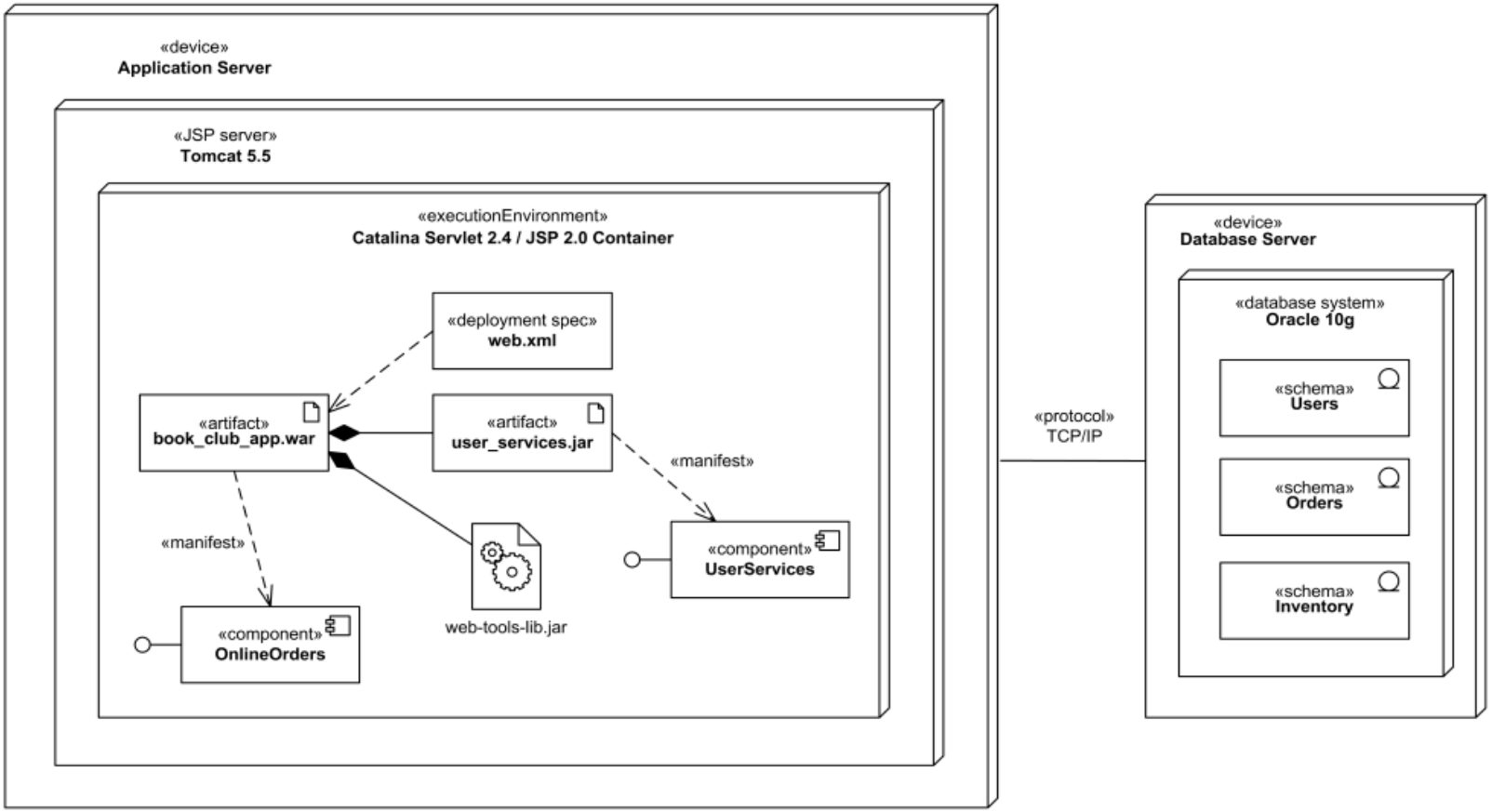
https://www.atlassian.com/pt/git/workflows#!workflow-gitflow

Klasyfikacja Rozanski. Perspektywa rozmieszczenia

- Definicja: Opisuje środowisko, w którym system będzie rozmieszczony, włączając w to zależności systemu od środowiska uruchomieniowego
- Opisuje:
  - Wymagane platformy uruchomieniowe
  - Specyfikację i liczbę elementów sprzętowych lub hostów
  - Wymagania na „obce” oprogramowanie
  - Wymagania sieciowe
  - Ograniczenia fizyczne (wymagane pojemności na dysku, zużywana moc)
- Modele (różne postacie diagramów rozmieszczenia):
  - Modele platform uruchomieniowych
  - Modele sieci (charakterystyka połączeń)
  - Uwaga: dwa powyższe modele można przedstawić na 1 diagramie
  - Modele zależności od technologii
- Udziałowcy:
  - Administratorzy, wytwórcy, testerzy, oceniający

Klasyfikacja Rozanski. Perspektywa rozmieszczenia

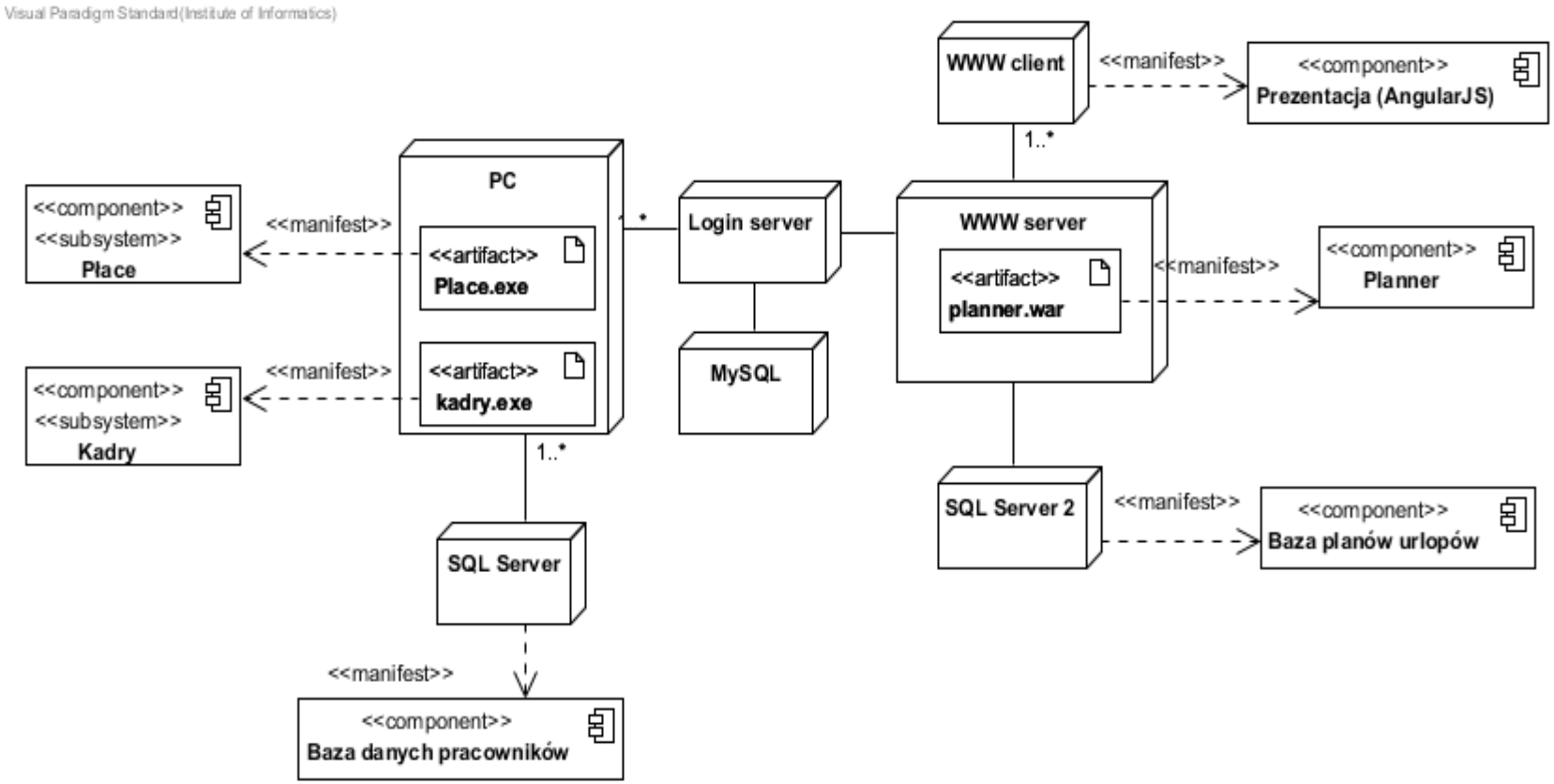
- Diagram rozmieszczenia na poziomie specyfikacji (typów)



Klasyfikacja Rozanski. Perspektywa rozmieszczenia

- Stereotypy dla węzłów:
  - <<device>> - fizyczny zasób obliczeniowy z procesorem, np. laptop, serwer
  - <<executionEnvironment>> - węzeł, który oferuje środowisko uruchomieniowe dla pewnych typów artefaktów, np. system operacyjny, kontener (serwletów), silnik przepływów
  - Inne są możliwe

Klasyfikacja Rozanski. Perspektywa rozmieszczenia, przykład



Klasyfikacja Rozanski. Perspektywa rozmieszczenia, c.d.

- Model platformy uruchomieniowej, przykład

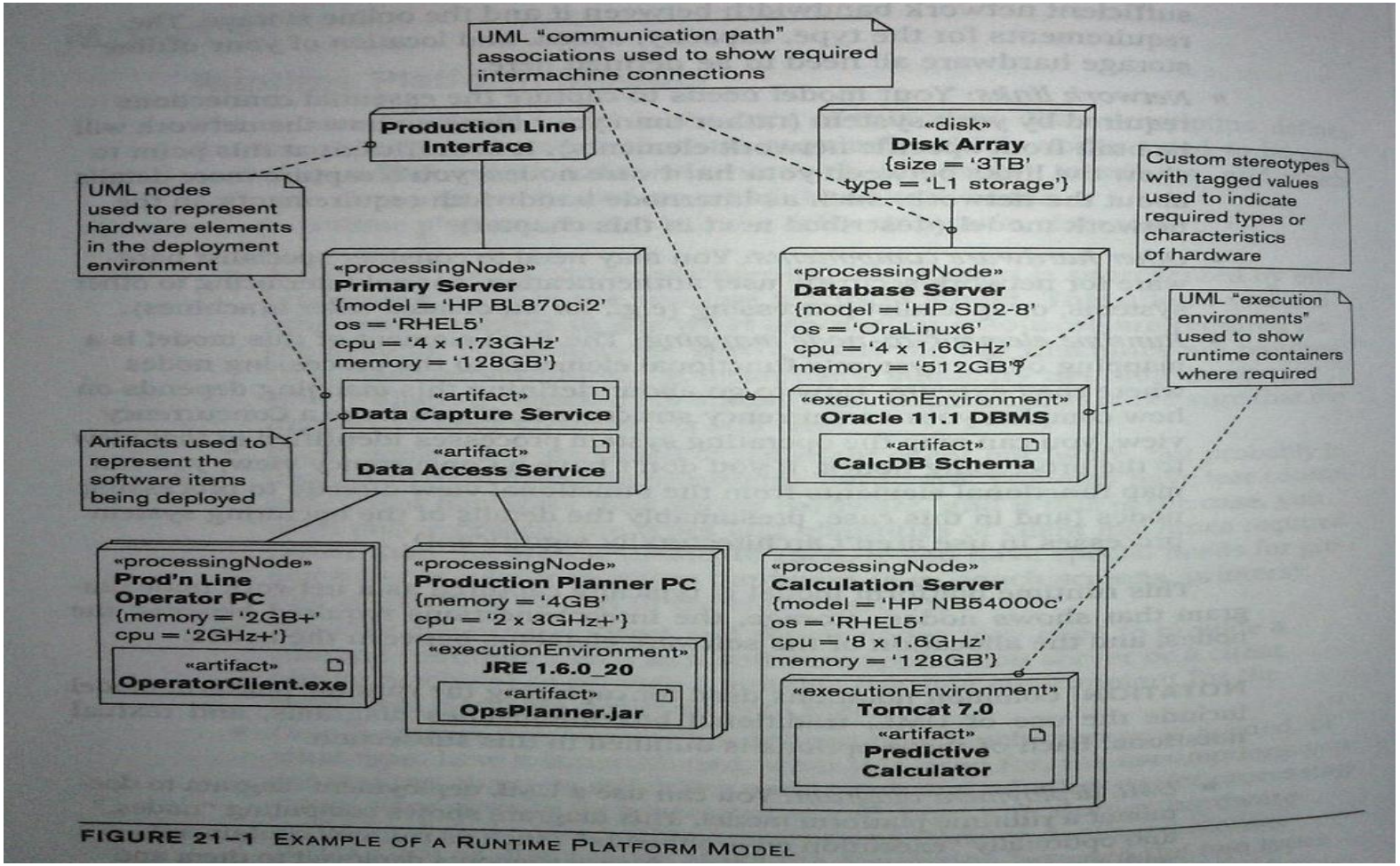


FIGURE 21–1 EXAMPLE OF A RUNTIME PLATFORM MODEL

Klasyfikacja Rozanski. Perspektywa rozmieszczenia, c.d.

- Model sieci, przykład

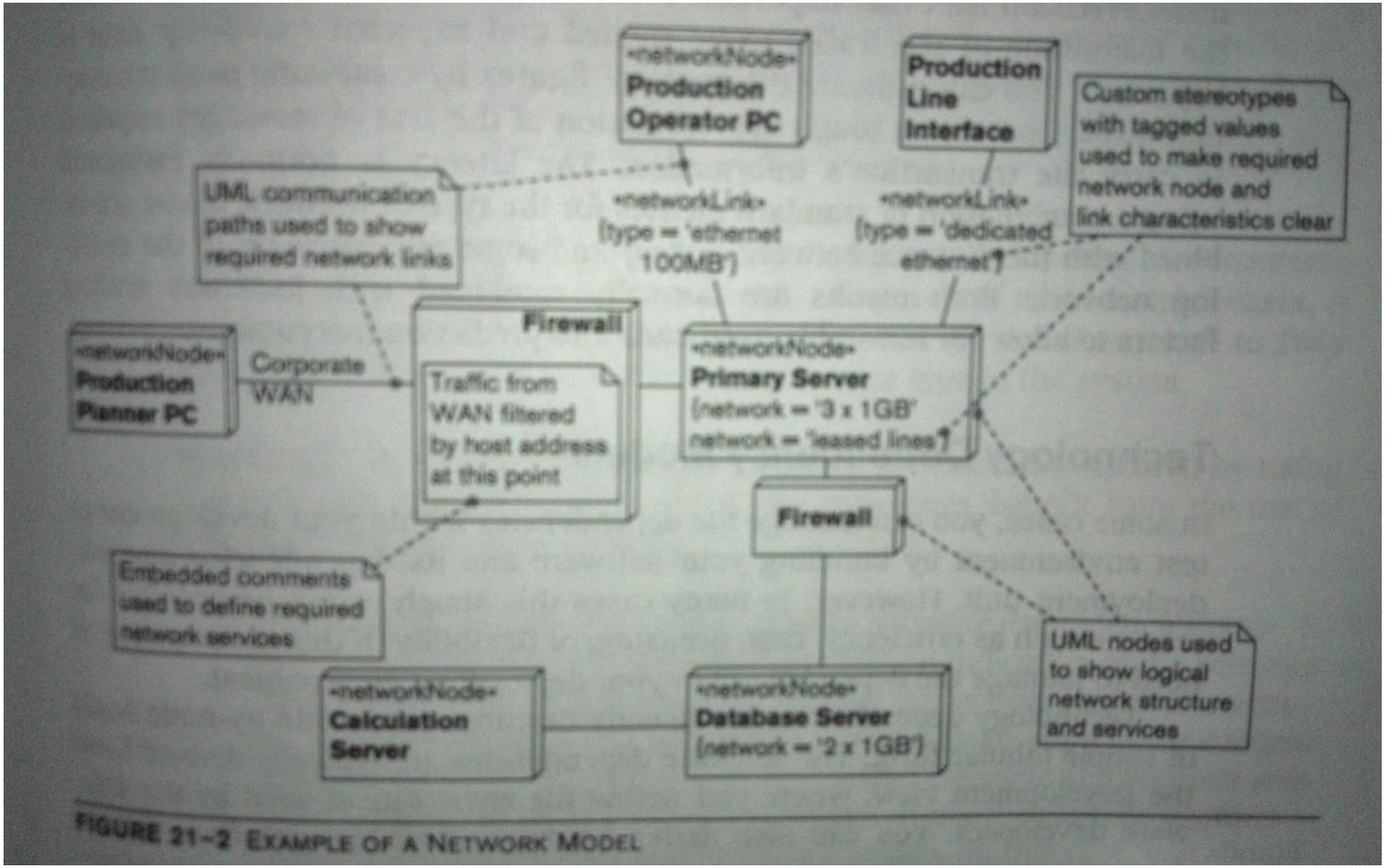


FIGURE 21–2 EXAMPLE OF A NETWORK MODEL

Klasyfikacja Rozanski. Perspektywa rozmieszczenia, c.d.

- Model zależności od technologii, przykład

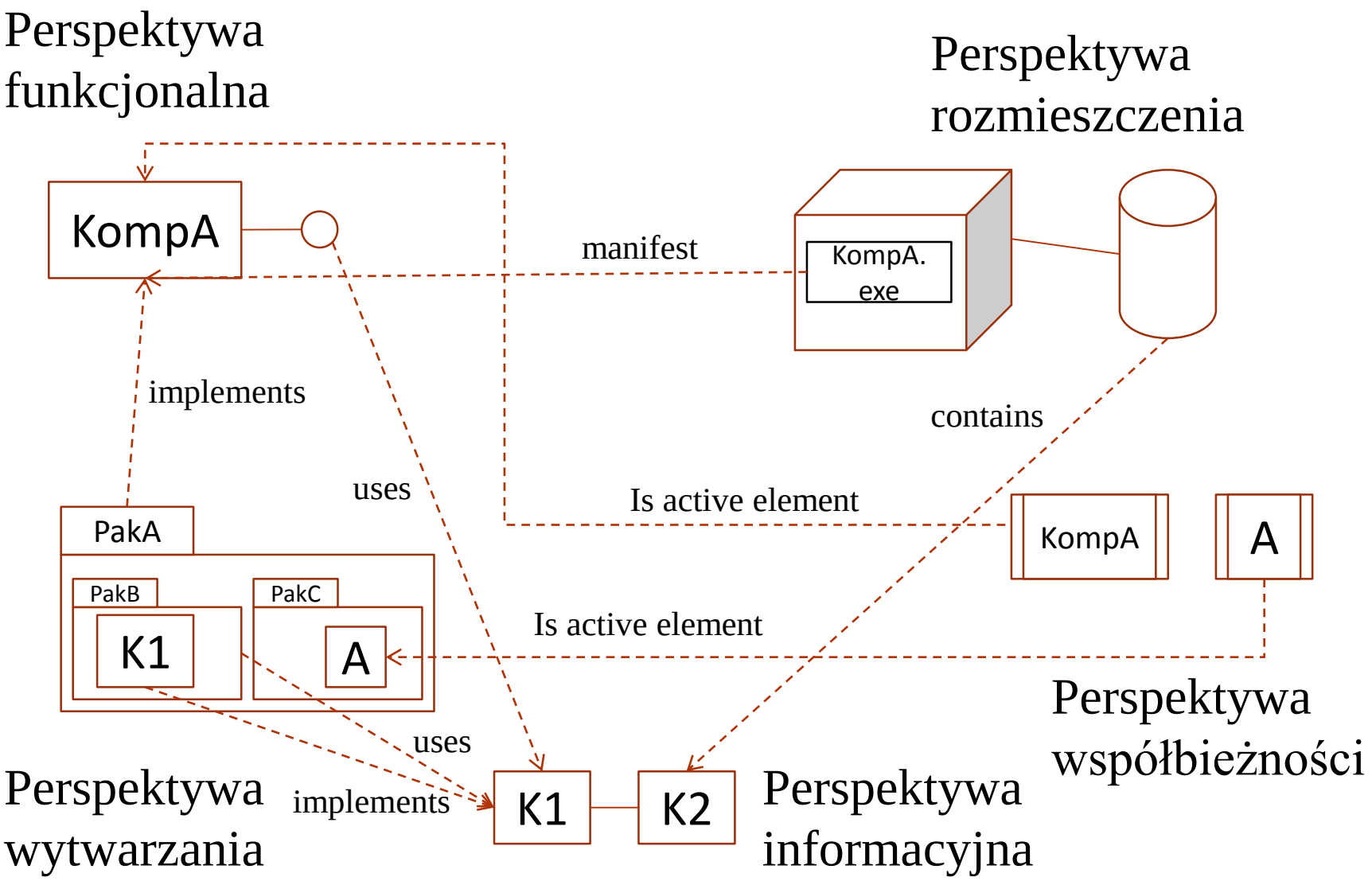
TABLE 21-2 SOFTWARE DEPENDENCIES FOR THE PRIMARY SERVER NODE

Component	Requires
Data Access Service	HP-UX 64-bit 11.23 + patch bundle B.11.23.0703
	HP aCC C++ runtime A.03.73
Data Capture Service	HP-UX 64-bit 11.23 + patch bundle B.11.23.0703
	HP aCC C++ runtime A.03.73
	Oracle OCI libraries 11.1.0.7
HP aCC C++ Compiler & Runtime	HP patch PHSS_35102
	HP patch PHSS_35103
Oracle OCI 11.1.0.7	HP-UX optional package X11MotifDevKit.MOTIF21
	HP-UX patch PHSS_37958

Klasyfikacja Rozanski. Perspektywa operacyjna

- Definicja – opisuje jak system będzie obsługiwany, administrowany i wspierany w jego środowisku produkcyjnym (poza zakresem prezentacji)
- Co opisuje:
  - Proces instalacji i upgradu
  - Migracje funkcji i danych (big bang, parallel run, staged migration)
  - Monitorowanie (w tym wydajności) i sterowanie
  - Wsparcie użytkowników
  - Wykonywanie kopii zapasowych i odtwarzanie po awarii
- Modele:
  - Modele instalacji
  - Modele migracji
  - Modele wsparcia
- Udziałowcy: administratorzy, inżynierowie produkcji, wytwórcy, testerzy, oceniający
- Zastosowanie: systemy pracujące w złożonym lub krytycznym środowisku operacyjnym

Zależności między perspektywami



Pytania kontrolne

- Co to jest architektura?
- Czym jest perspektywa i widok architektoniczny?
- Jakie wymagania powinna spełniać architektura?
- Podaj przykładowe perspektywy i omów zależności między nimi