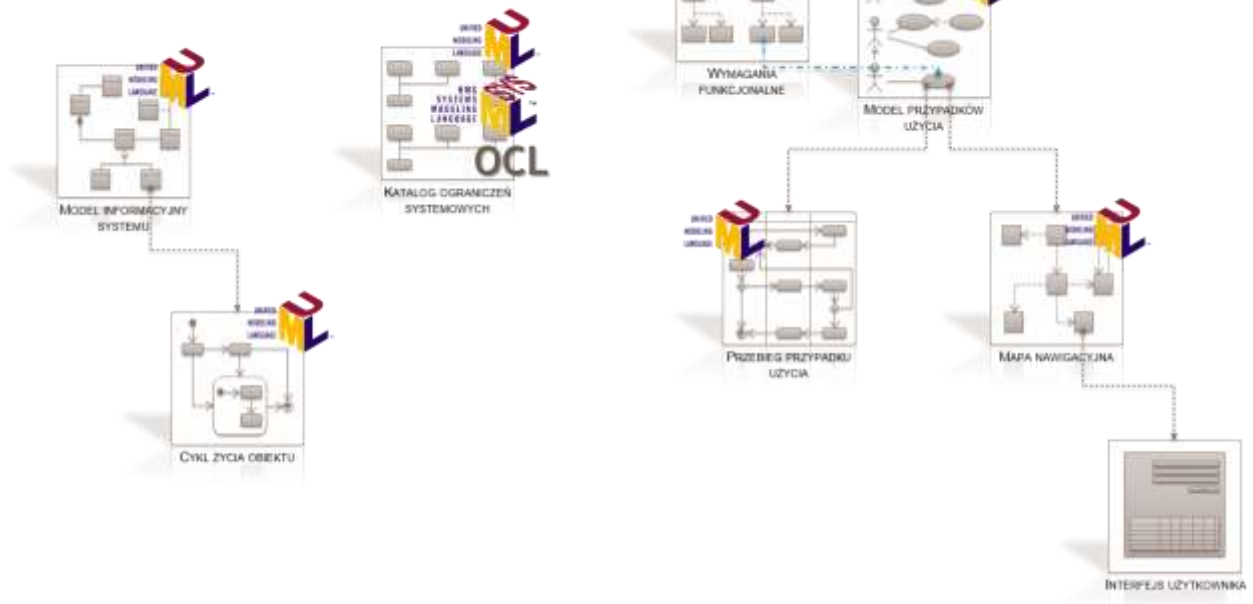
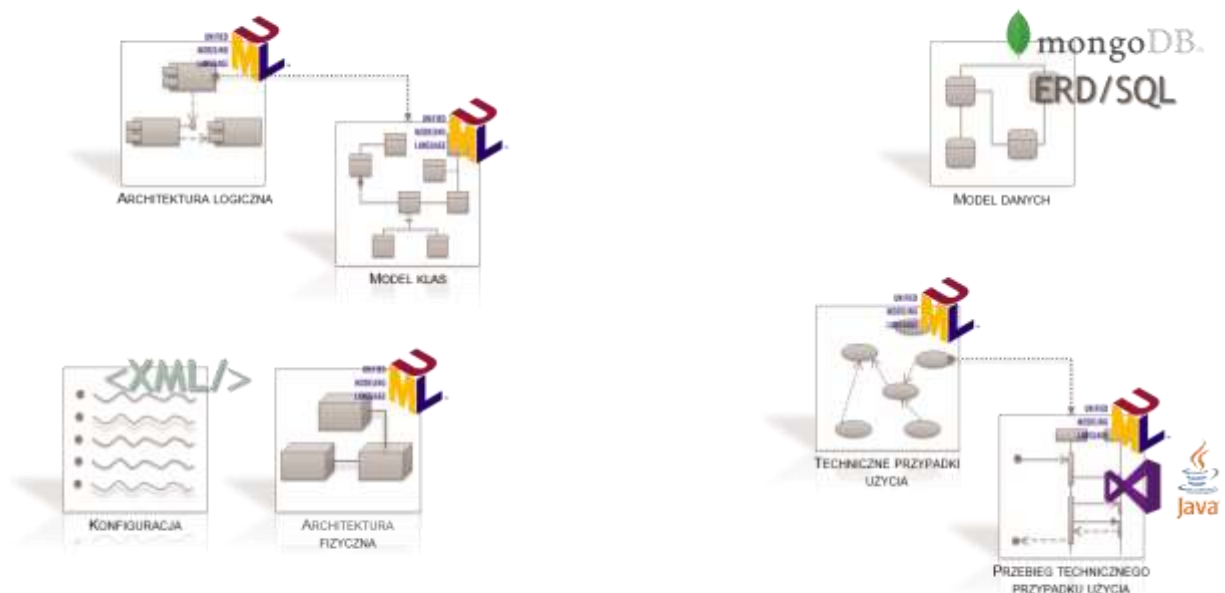


ANALIZA SYSTEMOWA



PROJEKT SYSTEMU



ZADANIE 1. PRZYGOTOWANIE NARZĘDZIA I INICJALIZACJA PROJEKTU

INSTRUKCJA LABORATORIUM

1. Należy utworzyć nowy projekt: opcja 'File → New Project → Create Blank Project'.
2. Wykorzystując opcje konfiguracji projektu (opcja 'Tools → Project option') należy:
 - a. zdefiniować typy danych zgodne ze standardem UML (opcja 'Data Type');
 - b. ustawić domyślne wartości (opcja 'Diagramming') dla:
 - klas (zakładka: Class → Presentation) – prezentacja liczności atrybutów ('Show attribute multiplicity'), prezentacja wartości początkowej atrybutów ('Show attribute initial value'), niezdefiniowana domyślna widoczność atrybutów ('Default Visibility → Attribute: Unspecified'), domyślna liczność atrybutów równa 1 ('Default Attribute Value → Multiplicity: 1'), atrybuty będące kolekcjami domyślnie nieuporządkowane i unikalne (odznaczone cechy uporządkowania – 'Ordered' i unikalności – 'Unique');
 - asocjacji (zakładka: Association) – widoczne uporządkowanie końców asocjacji ('Show multiplicity constraints: ordered'), prezentacja kierunku odczytu etykiety ('Show direction');
 - wymagań (zakładka: Requirement Diagram) – widoczność tylko atrybutów z przypisaną wartością ('Show Attributes: Show Non-empty Attributes'), możliwość łamania wartości przypisanych atrybutom ('Wrap member');
 - środowiska (zakładka: Environment) – uwzględnienie cechy 'Support HTML Tagged value';
 - c. ustawić domyślny format (kolory, czcionki) dla elementów diagramów (zakładki: Shape, Connector).
3. Należy przygotować strukturę projektu – zdefiniować składowe projektu odzwierciedlające poszczególne artefakty modelu analizy biznesowej i systemowej zgodnie z propozycją zawartą poniżej (poszczególne elementy należy zdefiniować bez nałożonych na nie stereotypów – te zostaną uwzględnione w ramach realizacji kolejnego zadania).

STRUKTURA PROJEKTU – PROPOZYCJA JEGO REALIZACJI W NARZĘDZIU VISUALPARADIGM

Biorąc pod uwagę zależności poszczególnych modeli analizy oraz ich artefaktów składowych definiowanych w ramach prezentowanego podejścia, projekt powinien być specyfikowany w następującej strukturze:

▢ Typy danych

▢ Typy danych : ClassDiagram [0..1]

▢ <nazwa_typu> : DataType [1..*]

➤ CIM

➤ Model informacyjny organizacji

- 📊 Słownik terminów : GlossaryGrid
 - ☐ <termin> [1..*]
- 📄 [CIM] Model pojęć : ClassDiagram [1..*]
 - ☐ <nazwa_pojęcia> : Class [1..*]
 - 📄 [CIM] <nazwa_pojęcia> : StateMachineDiagram [0..1]
 - <nazwa_stanu> : State [1..*]

➤ Model procesów biznesowych

- 📄 Architektura procesów biznesowych : BusinessProcessDiagram
 - <nazwa_procesu> : Sub-Process [1..*]
 - 📄 <nazwa_procesu> : BusinessProcessDiagram
 - <nazwa_czynności> : Task [1..*]

➤ Reguły biznesowe

- ☐ «BusinessRule» <nazwa_reguły_biznesowej> : Requirement [0..*]

➤ Model wymagań

- 📄 Wymagania funkcjonalne : RequirementDiagram
 - ☐ «FunctionalRequirement» <nazwa_wymagania> : Requirement [1..*]

➤ PIM

➤ Model informacyjny systemu

- 📄 [PIM] Model informacyjny systemu : ClassDiagram [1..*]
 - ☐ <nazwa_klasy> : Class [1..*]
 - 📄 [PIM] <nazwa_klasy> : StateMachineDiagram [0..1]
 - <nazwa_stanu> : State [1..*]
 - ☐ [PIM] Słowniki [0..1]
 - ☐ «Dictionary» <nazwa_słownika> : Class [1..*]
 - ☐ [PIM] Parametry systemowe [0..1]
 - ☐ «SystemParameter» <nazwa_parametru> : Class [1..*]

➤ Model funkcjonalności systemu

- 📄 Hierarchia aktorów : UseCaseDiagram [0..1]

```

✱ <nazwa_aktora> : Actor [1..*]
  ☒ <nazwa_aktora> : UseCaseDiagram [1]
    ○ <nazwa_funkcjonalności> : UseCase [1..*]
      📄 [UC] <nazwa_funkcjonalności> : ActivityDiagram
        ○ <nazwa_akcji> : Action [1..*]
      📄 [NM] <nazwa_funkcjonalności> : ClassDiagram
        ☐ «Transient» [UI] <nazwa_klasy> : Class [0..*]
    ○ <nazwa_funkcjonalności_reużywanej> : UseCase [0..*]
      //struktura zagnieżdżona jak dla zwykłej funkcjonalności
  ☐ Interfejs użytkownika
    ☐ [UI] <nazwa_elementuUI> : Class [1..*]
    ☒ [UI] <nazwa_elementuUI> : UserInterfaceDiagram

```


ZADANIE 2. STANDARYZACJA PROJEKTU – APLIKACJA PROFILU

INSTRUKCJA LABORATORIUM

1. Należy utworzyć nowy projekt o nazwie 'PO_Profile': opcja 'File → New Project → Create Blank Project'.
2. W zdefiniowanym projekcie należy utworzyć nowy profil (widok 'Model Eksplorator', opcja menu kontekstowego dla projektu 'New Profile') a w nim diagram profilu:
 <profil> → Sub Diagrams → New Profile Diagram
3. Na diagramie należy zdefiniować stereotypy, które zostały wykorzystane w strukturze projektu z zadania 1 (aplikowane do różnych elementów diagramów – «BusinessRule» (*Requirement*), «FunctionalRequirement» (*Requirement*), «SystemParameter» (*Class*) «Dictionary» (*Class*), «Transient» (*Class*). Dla stereotypów należy zdefiniować metaatrybuty różnych typów, w szczególności dla reguły biznesowej metaatrybut typu wyliczeniowego specyfikujący rodzaj reguły (zgodny z klasyfikacją reguł biznesowych).
4. Utworzony profil należy zaaplikować do projektu utworzonego w ramach zadania 1 – należy zamknąć projekt z profilem, otworzyć projekt utworzony w ramach zadania 1, powiązać przez referencję profil: opcja 'File → Manage Referenced Project... → Add → <projekt z profilem>'.
5. Na elementy diagramów utworzone w ramach zadania 1.3 należy nałożyć stereotypy z profilu, np.:
 <reguła> → Stereotypes → .../PO_Profile.vpp → BusinessRule.

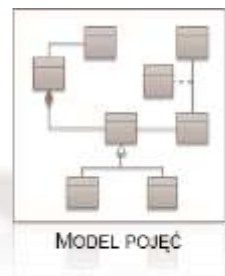
PROJEKT 2 I 3. OPRACOWANIE WIZJI SYSTEMU I SŁOWNIKA POJĘĆ BIZNESOWYCH. DEFINICJA MODELU BIZNESOWEGO I REGUŁ BIZNESOWYCH

Lab#	Temat	Uwagi	Rezultaty (punkty)	Narzędzia	Elementy podlegające sprawdzeniu
2	Wizja. Słownik.	Studenci indywidualnie budują <u>wizję</u> i słownik dla tematu wiodącego.	Wizja (max. 5 p). Słownik (max. 5 p).	Edytor tekstu – wizja, zgodnie z szablonem (użytkownicy, cechy, etc.) Visual Paradigm – słownik	Spójność (wewnętrzna, zewnętrzna), niesprzeczność
3	Model domenowy. Reguły biznesowe.	Studenci dla tematu wiodącego na podstawie zdefiniowanego słownika definiują diagram domenowy oraz min. 10 reguł biznesowych (różnych typów) w języku naturalnym. Diagram domenowy zawiera klasy i związki. Atrybuty umieszczamy tylko o ile występują w regułach biznesowych. Reguły biznesowe powinny być podzielone na kategorie. Zaleca się stosowanie szablonów RuleSpeak.	Model domenowy (max. 5p) Reguły biznesowe (różnych typów) (5 p.)	Visual Paradigm – diagram klas Visual Paradigm – diagram wymagań SysML, Edytor tekstu lub Excel - reguły biznesowe; w przypadku reguł strukturalnych zaleca się ich „dopięcie” do elementów diagramu klas (constrains)	Poprawność składniowa i semantyczna. Poprawność reguł biznesowych. Spójność reguł z modelem domenowym.

Etap procesu wytwórczego	Analiza biznesowa	
Model/poziom zgodnie z założeniami podejścia MDA	Computation independent model (CIM)	
Artefakt procesu wytwórczego	Model procesowy organizacji::Model informacyjny organizacji:: Słownik terminów	
Stosowane przedrostki diagramów	[CIM] (GlossaryGrid)	
Stosowane przedrostki elementów	N/A	
Opis		
<p>Identyfikacja terminów jakimi opisywana jest organizacja.</p> <p>Dla poszczególnych terminów:</p> <ul style="list-style-type: none">▪ opracowanie definicji;▪ wskazanie synonimów, skrótów oraz stosowanych form zapisu (na potrzeby powiązania z elementami składowymi modeli opracowywanych w ramach kolejnych etapów procesu wytwórczego).		

Narzędzie (wykorzystane diagramy i rozszerzenia)

Diagram *GlossaryGrid* (udostępniony przez VisualParadigm) skonfigurowany na potrzeby realizowanego procesu wytwórczego.

Etap procesu wytwórczego	Analiza biznesowa	
Model/poziom zgodnie z założeniami podejścia MDA	Computation independent model (CIM)	
Artefakt procesu wytwórczego	Model procesowy organizacji::Model informacyjny organizacji:: Model pojęć	
Stosowane przedrostki diagramów	[CIM] (ClassDiagram)	
Stosowane przedrostki elementów	N/A	
Opis		
<p>Identyfikacja bytów / encji biznesowych, ich cech strukturalnych oraz relacji pomiędzy bytami - danych, które stanowią dane / informacje przetwarzane w organizacji (w szczególności tych bytów, do których odwołania pojawiają się w regułach oraz wymaganiach biznesowych, obiektów / danych przetwarzanych w ramach procesów biznesowych) – reprezentacja terminów ze słownika na potrzeby strukturalizacji, poszukiwań encji (terminy mogą być odzwierciedlone przez klasy modelu, właściwości klas, zależności pomiędzy klasami).</p> <p>Definicja pojęć - tj. powiązanie elementów modelu z terminami ze słownika.</p>		
Narzędzie (wykorzystane diagramy i rozszerzenia)		
Diagram klas UML.		

Wszystkie zadania zdefiniowane w dalszej części instrukcji należy wykonać dla projektu utworzonego przy wykorzystaniu szablonu struktury projektu („Szablon_projektu.vpp”) i powiązanego z nim projektu z profilem („BAProfile.vpp”). Szablon jest zdefiniowany z uwzględnieniem wytycznych dotyczących konfiguracji projektu z zadania 1.2, zawiera strukturę projektu zgodną z wytycznymi z zadania 1.3.

W szablonie zostały zdefiniowane macierze służące sprawdzeniu poprawności (spójności i kompletności) niektórych artefaktów opracowywanych w ramach projektu (dodatkowy pakiet „Impact Analysis” w strukturze projektu).




INSTRUKCJA LABORATORIUM

1. Należy otworzyć projekt 'Szablon_projektu.vpp', przy otwieraniu projektu należy wskazać projekt z profilem 'BAProfile.vpp'.
2. Projekt należy zapisać pod nazwą identyfikującą autora/autorów projektu i w nim realizować kolejne zadania instrukcji.

ZADANIE 3. IDENTYFIKACJA ŹRÓDEŁ WIEDZY DZIEDZINOWEJ, KONFIGURACJA SŁOWNIKA TERMINÓW

CIM

Model informacyjny organizacji

-  Słownik terminów : GlossaryGrid
 -  `<termin> [1..*]`
-  [CIM] Model pojęć : ClassDiagram [1..*]
 -  `<nazwa_pojęcia> : Class [1..*]`





INSTRUKCJA LABORATORIUM

1. Należy skonfigurować słownik terminów (tj. CIM::Model informacyjny organizacji:: Słownik terminów) definiując etykiety odpowiadające wykorzystywanym źródłom wiedzy dziedzinowej – tj. za pomocą opcji 'Manage Label...' należy dodać etykiety reprezentujące materiały dziedzinowe stanowiące źródło wiedzy dziedzinowej w tym terminów opisu dziedziny (np. regulaminy, rozporządzenia, opisy procedur, przykładowe formularze, itd.).
2. Prezentację słownika należy rozszerzyć (opcja 'Configure Columns') dodając kolumnę Labels, jeżeli ta nie jest prezentowana.

ZADANIE 4. MODELOWANIE ZAWARTOŚCI INFORMACYJNEJ ORGANIZACJI

CIM

Model informacyjny organizacji

-  Słownik terminów : GlossaryGrid
 -  `<termin> [1..*]`
-  [CIM] Model pojęć : ClassDiagram [1..*]
 -  `<nazwa_pojęcia> : Class [1..*]`

INSTRUKCJA LABORATORIUM

1. Należy opracować słownik terminów (tj. uzupełnić CIM::Model informacyjny organizacji:: Słownik terminów). Dla każdego zidentyfikowanego terminu opisującego dziedzinę problemu należy zdefiniować w słowniku nowy termin (opcja 'New term'), uzupełnić jego definicję, wskazać kategorię terminu (uczestnik procesu biznesowego, zdarzenie, dokument, obiekt, itd.), źródło terminu (przynajmniej jedno ze zdefiniowanych w ramach zadania 3) oraz ewentualne synonimy, skróty i akronimy.

definicja: `<termin>` → Open Specification... (zakładka 'General') → Description

synonim, skrót, akronim: `<termin>` → Open Specification... (zakładka 'General') → Aliases → Add

kategoria: `<termin>` → Open Specification... (zakładka 'General') → Label → +

źródło: `<termin>` → Open Specification... (zakładka 'General') → Label → +

2. Terminy, które odwzorowują dane przetwarzane w organizacji (wiedzę organizacji) należy uwzględnić w modelu pojęć – za pomocą opcji 'Transit to Class' zdefiniować klasy modelu. Nazwy klas należy zapisać zgodnie z konwencjami UML. Utworzone klasy należy uwzględnić na diagramie modelu pojęć.


`<termin>` → Transit to New Class (nazwę klasy zapisać zgodnie z konwencjami UML) → Show in existing diagram (wybrać [CIM] Model pojęć)

Jeżeli definicja klasy wymagała zmiany domyślnej nazwy (nazwy terminu), wykorzystaną formę zapisu należy zapisać jako inną formę przekształcanego terminu.

`<termin>` → Open Specification... (zakładka 'General') → Other forms → Add

Sprawdzenie:

- **macierz:** Impact Analysis::Kompletność definicji pojęć (Konfiguracja macierzy: zakres – Model/Package CIM::Model informacyjny organizacji, wiersze – Class, kolumny – Term, by – Transitor); analizy wymagają elementy pozostałe po wyborze opcji 'Non-matches only',
 - **słownik:** CIM::Model informacyjny organizacji:: Słownik terminów dodatkowa kolumna 'Transit to' powinna zawierać klasę utworzoną na podstawie terminu,
 - **diagram** – oznaczenie opcji podkreślającej terminy ze słownika: CIM::Model informacyjny organizacji::Model pojęć, opcja menu kontekstowego: Presentation Options → Highlight Glossary Terms, wszystkie klasy modelu pojęć powinny być powiązane z terminem ze słownika, tj. mieć podkreśloną nazwę.
3. Definicje pojęć należy uzupełnić – uwzględnić strukturę danych oraz zależności zachodzące pomiędzy pojęciami (zdefiniować atrybuty oraz asocjacje pojęć).

Etap procesu wytwórczego	Analiza biznesowa	
Model/poziom zgodnie z założeniami podejścia MDA	Computation independent model (CIM)	
Artefakt procesu wytwórczego	Model procesowy organizacji:: Katalog reguł biznesowych	
Stosowane przedrostki diagramów	[BR] (RequirementDiagram)	
Stosowane przedrostki elementów	N/A	
Opis		
Identyfikacji i specyfikacja reguł biznesowych.		
Narzędzie (wykorzystane diagramy i rozszerzenia)		
Diagram wymagań SysML, Profil ze standardem (stereotyp «BusinessRule» dla reguł biznesowych, stereotyp «SystemConstraint» dla ograniczeń zidentyfikowanych przy okazji specyfikacji wymagań wobec funkcjonalności systemu), RuleSpeak, DMN, Tabela decyzyjna, Diagram klas UML (powiązanie ograniczeń strukturalnych).		

ZADANIE 5. SPECYFIKACJA REGUŁ BIZNESOWYCH

➤ CIM

...

➤ Reguły biznesowe

☐ «*BusinessRule*» <*nazwa_reguły_biznesowej*> : Requirement [0..*]

➤ Model wymagań

INSTRUKCJA LABORATORIUM

1. Na podstawie dostępnych materiałów należy zdefiniować reguły biznesowe i zamodelować je za pomocą elementu *Requirement*, oznaczonego stereotypem «*BusinessRule*». Reguły należy umieścić na diagramie wymagań z regułami (tj. *CIM::Reguły biznesowe::Reguły biznesowe*).

W związku z nieprawidłowym działaniem narzędzia VP należy postępować w następującej kolejności:

- a) utworzyć wymaganie,

- b) zdefiniować jego nazwę i treść,
- c) usunąć stereotyp «requirement»,
- d) dodać stereotyp «BusinessRule».

<reguła> → Stereotypes → .../BAProfile.vpp → BusinessRule

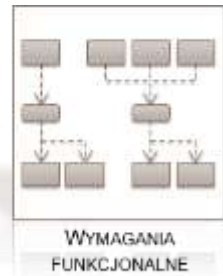
2. Zdefiniowane reguły należy odpowiednio zaklasyfikować – dla każdej z reguł wskazać jej typ:

<reguła> → Open Specification... (zakładka 'Attributes') → typ

3. Dla każdej ze zdefiniowanych reguł biznesowych należy zweryfikować jej precyzję i jednoznaczność oraz kompletność modelu pojęć względem reguły – uzupełnić model pojęć o brakujące atrybuty i asocjacje.

PROJEKT 4. SPECYFIKACJA WYMAGAŃ

Lab#	Temat	Uwagi	Rezultaty (punkty)	Narzędzia	Elementy podlegające sprawdzeniu
4	Specyfikacja wymagań.	Specyfikacja wymagań w postaci: diagramu wymagań SysML ze zbiorem historyjek	Specyfikacja wymagań (2.5 p.)	Visual Paradigm – diagram wymagań SysML lub, alternatywnie Word, Excel	Śladowalność do wizji

Etap procesu wytwórczego	Analiza biznesowa	
Model/poziom zgodnie z założeniami podejścia MDA	Computation independent model (CIM)	
Artefakt procesu wytwórczego	Model wymagań::Wymagania funkcjonalne	
Stosowane przedrostki diagramów	[FR] (RequirementDiagram)	
Stosowane przedrostki elementów	N/A	
Opis		
Specyfikacja wymagań biznesu wobec funkcjonalności systemu. Wskazanie roli użytkownika, dla której wymaganie jest przewidziane. Strukturalizacja wymagań.		
Powiązanie wymagań biznesu z regułami biznesowymi, z których wymagania wynikają.		
Narzędzie (wykorzystane diagramy i rozszerzenia)		
Diagram wymagań SysML, Profil ze standardem (stereotyp «FunctionalRequirement» dla wymagania funkcjonalnego).		

ZADANIE 6. MODELOWANIE WYMAGAŃ

➤ CIM

...

➤ Model wymagań

📁 Wymagania funkcjonalne : RequirementDiagram

☐ «FunctionalRequirement» <nazwa_wymagania> : Requirement [1..*]

INSTRUKCJA LABORATORIUM

1. Na podstawie dostępnych materiałów należy zdefiniować wymagania funkcjonalne – zamodelować je za pomocą elementu `Requirement`, oznaczonego stereotypem «`FunctionalRequirement`». Wymagania należy umieścić na diagramie wymagań z wymaganiami funkcjonalnymi (tj. `CIM::Model` wymagań::Wymagania funkcjonalne).

W związku z nieprawidłowym działaniem narzędzia VP należy postępować w następującej kolejności:

- a) na odpowiednim diagramie wymagań należy utworzyć wymaganie,
 - b) zdefiniować jego nazwę i treść (treść wymagania należy zapisać wykorzystując szablon user stories),
 - c) dodać stereotyp «`FunctionalRequirement`»:
`<wymaganie> → Stereotypes → ../BAPProfile.vpp → FunctionalRequirement`
 - d) usunąć stereotyp «`requirement`».
2. Należy zdefiniować zależności `containment` oraz «`derive`» i «`copy`» pomiędzy wymaganiami.
 3. Wymagania związane z weryfikacją spełnienia reguł biznesowych należy powiązać z regułami za pomocą relacji «`derive`».
 4. Reguły, które pojawiają się w związku z wdrożeniem projektowanego systemu, należy dodać do modelu jako `Requirement`, oznaczony stereotypem «`SystemConstraint`» i również powiązać z odpowiednimi wymaganiami:

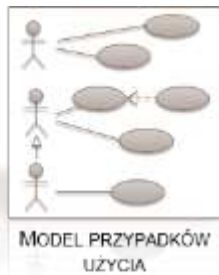
`<wymaganie> → Stereotypes → ../BAPProfile.vpp → SystemConstraint`

Sprawdzenie:

- **macierz:** `Impact Analysis::Uwzględnienie reguł biznesowych` (Konfiguracja macierzy: zakres – `Project`, wiersze – `Requirement «BusinessRule» || «SystemConstraint»`, kolumny – `Requirement «FunctionalRequirement»`, by – `Relationship: Derive`). Analizy wymagają reguły pozostałe po wyborze opcji `'Non-matches only'`.

PROJEKT 5. DIAGRAM PRZYPADKÓW UŻYCIA

Lab#	Temat	Uwagi	Rezultaty (punkty)	Narzędzia	Elementy podlegające sprawdzeniu
5	Diagram przypadków użycia.	Na podstawie specyfikacji wymagań studenci definiują diagram przypadków użycia. Do przypadków użycia przypisują realizowane przez przypadek historie użytkownika – śladują wymagania z diagramu wymagań.	Model PU + opisy streszczające (2.5 p.)	Visual Paradigm – diagram przypadków użycia + opisy streszczające (śladowanie do wymagań)	Śladowalność przypadków użycia do specyfikacji wymagań

Etap procesu wytwórczego	Analiza systemowa	
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)	
Artefakt procesu wytwórczego	Model funkcjonalności systemu:: Model przypadków użycia	
Stosowane przedrostki diagramów	N/A	
Stosowane przedrostki elementów	N/A	
Opis		
<p>Analiza wyspecyfikowanych wymagań biznesu wobec funkcjonalności systemu. Identyfikacja oczekiwanych funkcjonalności systemu (definicja przypadków użycia) oraz możliwości ich użycia przez poszczególnych użytkowników systemu (definicja aktorów, określenie zależności pomiędzy przypadkami użycia i aktorami).</p> <p>Definicja/rozważenie funkcjonalności administracyjnych (obsługa danych słownikowych, synchronizacji, importu/eksportu danych) oraz funkcjonalności związanych z realizacją operacji 'w tle' – uruchamianych upływem czasu (modelowanych jako przypadki użycia przypisane aktorowi umownie oznaczanemu jako Czas lub Zegar).</p>		
Określenie zależności pomiędzy przypadkami użycia oraz wymaganiami biznesowymi i regułami biznesowymi.		
Narzędzie (wykorzystane diagramy i rozszerzenia)		
Diagram przypadków użycia UML, opisy dla poszczególnych aktorów oraz przypadków użycia (sekcja documentation), Diagram wymagań SysML.		

ZADANIE 7. [ZADANIE PRZYGOTOWUJĄCE DO DALSZYCH PRAC ANALITYCZNYCH]

INSTRUKCJA LABORATORIUM






1. Należy zidentyfikować (wypisać /wyróżnić w treści wymagań oraz na przykładowych dokumentach i formularzach) kandydatów na:
 - a. aktorów projektowanego systemu;
 - b. przypadki użycia reprezentujące funkcjonalności proponowane aktorom, tj:
 - pogrupować wymagania funkcjonalne. Jako kryterium grupowania należy przyjąć: aktora, a w podgrupie – główny obiekt (klasę z modelu pojęć), którego dotyczy wymaganie (aktora oraz obiekt należy wskazać dla każdego wymagania);
 - dla uzyskanych grup (aktor::obiekt) należy zdefiniować przypadki użycia – w pierwszej iteracji będą to ‘grube’ przypadki użycia obejmujące wszystkie działania CRUD (typu ‘Zarządzanie ...’ / ‘Obsługa ...’), które w ramach dalszych prac analitycznych zostaną ewentualnie podzielone na kilka ‘mniejszych’ (wypisać proponowane nazwy przypadków użycia),
 - dla każdego przypadku należy wskazać aktora, któremu przypadek jest dedykowany,
 - dla każdego wymagania należy wskazać przypadek użycia, który je realizuje.

ZADANIE 8. MODELOWANIE FUNKCJONALNOŚCI SYSTEMU

PIM

...

Model funkcjonalności systemu

```
 Hierarchia aktorów : UseCaseDiagram [0..1]
    * <nazwa_aktora> : Actor [1..*]
         <nazwa_aktora> : UseCaseDiagram [1]
            ○ <nazwa_funkcjonalności> : UseCase [1..*]
                 [UC] <nazwa_funkcjonalności> : ActivityDiagram
                    ○ <nazwa_akcji> : Action [1..*]
                 [NM] <nazwa_funkcjonalności> : ClassDiagram
                     «Transient» [UI] <nazwa_klasy> : Class [0..*]
```

INSTRUKCJA LABORATORIUM

1. Na podstawie wymagań funkcjonalnych należy zdefiniować hierarchię aktorów (tj. uzupełnić diagram PIM::Model funkcjonalności system::Hierarchia aktorów).

2. Dla każdego zdefiniowanego aktora należy utworzyć nowy poddiagram przypadków użycia o nazwie 'Funkcjonalności <aktora>'.

<aktor> → Sub Diagrams → New Diagram... → Use Case Diagram

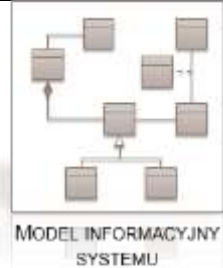
3. Dla poszczególnych aktorów należy zdefiniować przypadki użycia reprezentujące funkcjonalności systemu przewidziane dla nich. Zdefiniowane przypadki użycia należy umieścić na odpowiednim diagramie przypadków użycia danego aktora.
4. Należy rozważyć funkcjonalności, które nie wynikają wprost z wymagań funkcjonalnych: administracyjne (takie jak obsługa danych słownikowych, synchronizacji, importu/eksportu danych) oraz funkcjonalności związane z realizacją operacji 'w tle' – uruchamianych upływem czasu (modelowane jako przypadki użycia przypisane aktorowi umownie oznaczanemu jako 'Czas/Zegar'). Zdefiniowane przypadki użycia należy umieścić na odpowiednim diagramie przypadków użycia.
5. Zdefiniowane przypadki użycia (ich widoki) należy umieścić na diagramie/diagramach wymagań i powiązać relacją «satisfy» z wymaganiami funkcjonalnymi, które te przypadki realizują.
6. Należy ocenić kompletność modelu przypadków użycia względem wymagań:
 - a. Aktorzy modelu przypadków użycia powinni wynikać z ról-użytkowników zdefiniowanych w ramach wymagań funkcjonalnych.
 - b. Dla każdego wymagania, dla którego podjęto decyzję o realizacji w ramach danej wersji systemu należy wskazać przypadki użycia, które wymaganie realizują.
 - c. Dla każdego przypadku użycia należy wskazać wymagania biznesu, które realizuje dany przypadek użycia. Wyjątek stanowią przypadki użycia, które pośrednio wynikają z wymagań biznesu (administracyjne/wspierające).

Sprawdzenie:

- **macierz:** Impact Analysis::Realizacja wymagań funkcjonalnych (Konfiguracja macierzy: zakres – Project, wiersze – Requirement «FunctionalRequirement», kolumny – UseCase, by – Relationship:Satisfy. Analizy wymagają wszystkie wymagania, które zostają po zawężeniu za pomocą opcji 'Non-matches only'.

PROJEKT 6A. MODEL INFORMACYJNY

Lab#	Temat	Uwagi	Rezultaty (punkty)	Narzędzia	Elementy podlegające sprawdzeniu
6A.	Model informacyjny.	<p>Studenci budują model informacyjny jako uszczegółowienie modelu domenowego. Uszczegółowienie może ograniczać się do rozważanych przypadków użycia, ale model powinien pozostać całościowy.</p> <p>Uwaga: elementy rozważane (przypadki użycia, klasy) odpowiednio pokolorowane.</p> <p>Studenci mogą, opcjonalnie, przetłumaczyć reguły biznesowe do OCL.</p>	<p>Model informacyjny – 5 p.</p> <p>Reguły biznesowe w OCL (max. 5p.*)</p>	<p>Visual Paradigm – diagram klas</p> <p>Narzędzie wspierające OCL (opcja)</p>	Spójność wewnętrzna.

Etap procesu wytwórczego	Analiza systemowa	
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)	
Artefakt procesu wytwórczego	Model informacyjny systemu	
Stosowane przedrostki diagramów	[PIM] (ClassDiagram)	
Stosowane przedrostki elementów	N/A	
Opis		
Definicja typów danych - prostych, złożonych, wyliczeniowych (innych niż predefiniowane w standardzie UML).		
<p>Modelowanie zawartości informacyjnej systemu – identyfikacja klas dziedzinowych; określenie związków pomiędzy klasami; specyfikacja właściwości klas.</p> <p>Klasy modelu powinny mieć zdefiniowane:</p> <ul style="list-style-type: none">▪ referencję na definicję w słowniku terminów,▪ wszystkie ważne z perspektywy projektu atrybuty, z uwzględnieniem atrybutów statycznych oraz wnioskowanych, w szczególności należy rozważyć atrybuty pozwalające na rejestrację informacji o wykonanych akcjach (np. wysłaniu maila, rozpatrzeniu, itp.) – każdy z atrybutów powinien mieć ustalony typ danych oraz licznosc;▪ powiązania (asocjacje) z innymi klasami modelu – każde powiązanie powinno mieć zdefiniowaną nazwę z uwzględnieniem kierunku odczytu oraz dla każdego końca (roli reprezentowanej przez dany koniec) nazwę oraz licznosc.		

<ul style="list-style-type: none"> relacje generalizacji z innymi klasami, dla każdej hierarchii należy rozważyć ewentualną redefinicję właściwości.
<p>Definicja atrybutów odzwierciedlających maszyny stanów, a dla atrybutów – definicja typów wyliczeniowych uwzględniających możliwe stany maszyn.</p> <p>Wskazanie wartości odzwierciedlających stany początkowe maszyn jako wartości początkowych/ domyślnych dla atrybutów.</p>
Narzędzie (wykorzystane diagramy i rozszerzenia)
<p>Diagram pakietów UML, Diagram klas UML, Profil ze standardem (stereotyp «Dictionary» dla klas reprezentujących słowniki, stereotyp «SystemParameter» dla klas specyfikujących parametry systemowe).</p>
<p>Diagram klas UML, Diagram maszyny stanów UML (poddigram uzupełnianej klasy).</p>

ZADANIE 9. [ZADANIE POMOCNICZE PRZYGOTOWUJĄCE DO DALSZYCH PRAC ANALITYCZNYCH]

INSTRUKCJA LABORATORIUM

- Należy zidentyfikować (wypisać / wyróżnić w treści wymagań oraz na przykładowych dokumentach i formularzach) kandydatów na:
 - aktorów – użytkowników systemu,
 - klasy (klasy podejrzane o reprezentację tego samego elementu lub zależność dziedziczenia zapisać obok siebie),
 - typy wyliczeniowe i wartości tych typów,
 - właściwości klas, w tym atrybuty pozwalające na rejestrację informacji o wykonywanych akcjach oraz właściwości wnioskowane / wyliczane, a dla tych – reguły wyliczeń (przy klasie-właścicielu właściwości),
 - stany / statusy (przy klasie, której stan dotyczy),
 - asocjacje (w szczególności zależności, które nie wynikają wprost z właściwości) (przy jednej z klas uczestniczących w asocjacji).

ZADANIE 10. MODELOWANIE ZAWARTOŚCI INFORMACYJNEJ SYSTEMU

PIM

Model informacyjny systemu

 [PIM] Model informacyjny systemu : ClassDiagram[1..*]

 <nazwa_klasy> : Class [1..*]

```

[?] [PIM] <nazwa_klasy> : StateMachineDiagram [0..1]
    ○ <nazwa_stanu> : State [1..*]
    ▢ [PIM] Słowniki [0..1]
        ▢ «Dictionary» <nazwa_słownika> : Class [1..*]
    ▢ [PIM] Parametry systemowe [0..1]
        ▢ «SystemParameter» <nazwa_parametru> : Class [1..*]

```

INSTRUKCJA LABORATORIUM

1. Należy utworzyć kopię modelu pojęć opracowanego w ramach prac analizy biznesowej – za pomocą opcji 'Paste Model Element' wkleić elementy modelu pojęć na diagram PIM: : Model informacyjny systemu:: [PIM] Model informacyjny systemu.
2. Z modelu należy usunąć elementy spoza zakresu projektu, dodać nowe, zmodyfikować istniejące, tak aby odzwierciedlały dane przetwarzane w systemie. W modelu należy uwzględnić klasy odzwierciedlające aktorów/role użytkowników – tj. klasy reprezentujące konta użytkowników oraz atrybuty pozwalające na rejestrację informacji o wykonanych akcjach (np. wysłaniu maila, rozpatrzeniu dokumentu, itp.).
3. Dla poszczególnych klas modelu należy:
 - a. uzupełnić referencję klasy na pojęcie ze słownika terminów. Jeżeli termin nie jest zdefiniowany należy dodać go do słownika. Należy dodać nowe źródło dla terminów 'wymagania biznesowe'. Jeżeli termin jest zdefiniowany, ale zapis nie jest zgodny z konwencją UML, wykorzystaną formę zapisu należy wskazać jako inną formę terminu;

Sprawdzenie:

- Diagram: oznaczenie opcji podkreślającej terminy ze słownika: PIM::Model informacyjny systemu::[PIM]Model informacyjny systemu, opcja menu kontekstowego Presentation Options→Highlight Glossary Terms, wszystkie klasy modelu powinny mieć zdefiniowane powiązanie z terminem ze słownika, tj. podkreśloną nazwę.
 - b. uzupełnić sygnatury atrybutów – dla każdego atrybutu należy zdefiniować typ danych (czego konsekwencją może być zamiana atrybutu w koniec asocjacji), licznosc, określić zakres (tj. atrybut statyczny / obiektu), zaznaczyć ewentualną wnioskowalność atrybutu;
 - c. uzupełnić definicje asocjacji – dla każdej asocjacji należy zdefiniować nazwę, rozważyć zamianę asocjacji w agregację lub kompozycję, dla każdego końca asocjacji należy zdefiniować: nazwę roli, licznosc, zakres, ewentualną wnioskowalność.
4. Dla zdefiniowanych klas modelu należy rozważyć relację generalizacji pomiędzy klasami.
 5. Należy ocenić kompletność modelu informacyjnego systemu względem funkcjonalności reprezentowanych przez przypadki użycia – dla każdego przypadku użycia należy wskazać dane tworzone w ramach jego realizacji (np. odpowiednio pokolorować przypadki użycia i klasy – taki sam kolor linii oraz czcionki ustawić dla przypadku użycia oraz klasy, której obiekty są tworzone w ramach realizacji

przypadku użycia, atrybutu, jeżeli jego wartość jest ustalana w ramach realizacji przypadku użycia, asocjacji, jeżeli powiązanie jest tworzone w ramach realizacji przypadku użycia).

Etap procesu wytwórczego	Analiza systemowa
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)
Artefakt procesu wytwórczego	Model informacyjny systemu:: Ograniczenia systemowe (I) (specyfikacja ograniczeń nałożonych na elementy modelu informacyjnego systemu)
Stosowane przedrostki diagramów	N/A
Stosowane przedrostki elementów	N/A
Opis	
<p>Specyfikacja niezmienników – ograniczeń nałożonych na elementy modelu: klasy, atrybuty, związki:</p> <ul style="list-style-type: none"> ▪ wskazanie właściwości jednoznacznie identyfikujących obiekty klasy (ograniczenie <code>id</code>), ▪ uzupełnienie ograniczeń <code>readOnly</code>, ▪ dla właściwości kolekcyjnych rozważenie uporządkowania (ograniczenie <code>ordered</code>) oraz możliwości powtórzeń (ograniczenie <code>nonunique</code>), ▪ uzupełnienie ograniczeń dla zbioru generalizacji (<code>complete/incomplete</code>, <code>disjoint/overlapping</code>), ▪ rozważenie dynamicznej zmiany typu, ▪ rozważenie redefinicji właściwości. <p>Definicja wartości początkowych/domyślnych dla właściwości, w szczególności dla statusów.</p> <p>Definicja algorytmów wyliczania dla właściwości wnioskowanych.</p>	
Narzędzie (wykorzystane diagramy i rozszerzenia)	
<p>Diagram klas UML, Definicja ograniczeń za pomocą:</p> <ul style="list-style-type: none"> ▪ konstrukcji diagramu klas UML (liczności właściwości; predefiniowanych ograniczeń np. <code>id</code>, <code>readOnly</code>, <code>nonunique</code>, <code>ordered</code>, <code>subsets</code>, <code>union</code>, itp. oraz konstrukcji takich jak: zbiór generalizacji, redefinicja odziedziczonej właściwości, kwalifikator), ▪ wyrażeń języka OCL: <code>inv</code>, <code>init</code>, <code>derive</code> (definicja elementów <code>Constrain</code> nałożonych na klasy modelu informacyjnego). 	

ZADANIE 11. SPECYFIKACJA OGRANICZEŃ NAŁOŻONYCH NA ELEMENTY MODELU INFORMACYJNEGO SYSTEMU

INSTRUKCJA LABORATORIUM

1. Na elementy modelu należy nałożyć ograniczenia:

- a. wskazać właściwości jednoznacznie identyfikujące obiekty klas (ograniczenie `id`),
- b. dla atrybutów, których wartość nie może być zmieniona uzupełnić ograniczenie – `readOnly`,
- c. dla właściwości kolekcyjnych należy rozważyć uporządkowanie (ograniczenie `ordered`) oraz możliwość powtórzeń (ograniczenie `nonunique`),
- d. należy rozważyć wartości początkowe właściwości,
- e. dla właściwości wnioskowanych należy zapisać algorytm wyliczeń,
- f. dla zdefiniowanych w modelu relacji generalizacji należy rozważyć:
 - i. redefinicję odziedziczonych właściwości,
 - ii. dynamiczną zmianę typów,
 - iii. zbiory generalizacji. Dla każdego zdefiniowanego zbioru należy uzupełnić ograniczenia `complete/disjoint`.


Etap procesu wytwórczego	Analiza systemowa	
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)	
Artefakt procesu wytwórczego	Model informacyjny systemu:: Cykl życia obiektu	
Stosowane przedrostki diagramów	[PIM] (StateMachineDiagram)	
Stosowane przedrostki elementów	N/A	
Opis		
Dla wybranych / kluczowych klas dziedzinowych opracowanie modelu zachowania – tj. specyfikacja cyklu życia elementu modelu informacyjnego systemu – identyfikacja stanów / statusów, w których opisywany obiekt może się znaleźć oraz możliwych przejść pomiędzy tymi stanami.		
Powiązanie przejść pomiędzy stanami cyklu życia elementu modelu informacyjnego a funkcjonalnościami systemu (wskazanie przypadków użycia w ramach których zmiana stanu jest realizowana).		
Narzędzie (wykorzystane diagramy i rozszerzenia)		

Diagram maszyny stanów UML (poddigram dla opisywanej klasy dziedzicznej), Diagram przypadków użycia UML, Profil ze standardem (definicja wyzwalaczy przejść – atrybut useCase stereotypu «CallUseCase»).

ZADANIE 12. MODELOWANIE CYKLU ŻYCIA OBIEKTÓW MODELU INFORMACYJNEGO SYSTEMU

📁 PIM

📁 Model informacyjny systemu

```

[PIM] Model informacyjny systemu : ClassDiagram [1..*]
    <nazwa_klasy> : Class [1..*]
        [PIM] <nazwa_klasy> : StateMachineDiagram [0..1]
            <nazwa_stanu> : State [1..*]
        [PIM] Słowniki [0..1]
    
```

INSTRUKCJA LABORATORIUM

1. Należy opracować maszyny stanów opisujące cykle życia klas, dla których zidentyfikowano więcej niż dwa stany/statusy. Dla klas należy zdefiniować poddiagramy maszyn stanów, nazwy diagramów należy poprzedzić przedrostkiem [PIM].

<klasa> → Sub Diagrams → New Diagram... → State Machine Diagram

2. Dla opracowanych maszyn należy:

- a. wskazać potencjalne stany obiektów,
- b. zdefiniować możliwe przejścia pomiędzy stanami,
- c. dla każdego przejścia pomiędzy stanami należy wskazać wyzwalacz przejścia – przejście należy opisać stereotypem «CallUseCase», wskazać przypadek użycia, w ramach którego realizowana jest modelowana zmiana stanów (wyjątek stanowią przejścia wnioskowane/warunkowe):

<przejście> → Stereotypes → .../BAPProfile.vpp → CallUseCase

<przejście> → Open Specification... (zakładka Tagged Values) → usecase.Value → <przypadek użycia>

W szczególności musi być zdefiniowany przynajmniej jeden przypadek użycia, w ramach którego obiekt opisywany maszyną stanów jest tworzony – przypadek wskazany dla przejścia wychodzącego z pseudostanu początkowego maszyny stanów.

Sprawdzenie:

- diagram: PIM::Model informacyjny systemu::<klasa>::[PIM]<nazwa klasy>, opcja menu kontekstowego Presentation Options → Show Tagged Values → Show Non-Empty,

wszystkie przejścia wychodzące ze stanów oraz z pseudostanu początkowego powinny mieć zdefiniowane wyzwalacze.

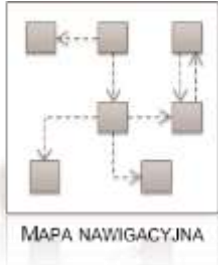
- d. zdefiniować ewentualne warunki przejść oraz akcje powiązane z przejściami (tj. akcje wykonywane przy okazji przejść).
3. Dla każdej maszyny stanów należy zdefiniować typ wyliczeniowy, dla każdego ze stanów maszyny literał utworzonego typu wyliczeniowego. W klasie, dla której zdefiniowano maszynę stanów należy zdefiniować atrybut 'stan/status' zdefiniowanego typu wyliczeniowego. Na podstawie stanu początkowego maszyny stanów należy wskazać wartość początkową atrybutu.


Sprawdzenie:

- enumerator reprezentujący stany maszyny, graficznie przedstawiony również na diagramie opisywanej maszyny, dla każdego stanu zdefiniowany literał enumeratora równy nazwie stanu.

PROJEKT 7. OPRACOWANIE PROTOTYPU INTERFEJSU UŻYTKOWNIKA

Lab#	Temat	Uwagi	Rezultaty (punkty)	Narzędzia	Elementy podlegające sprawdzeniu
7	Prototyp interfejsu.	Dla dwóch ustalonych na L6 przypadków użycia studenci budują prototyp interfejsu (przynajmniej dla wątków głównych); mogą to być rysunki odręczne lub wykonane w dowolnym narzędziu, np. w NetBeans, Visio etc. Za zdefiniowanie prototypu dla wątków alternatywnych ze wskazaniem reguł przewodnika stylu – dodatkowe punkty.	Prototyp interfejsu dla wątków głównych po 3p. za PU; prototypy dla wątków alternatywnych dodatkowo po 2p. za PU; za wskazanie reguł przewodnika stylu dodatkowo 2.5p.*	Dowolne dostępne narzędzie	Możliwość realizacji PU z wykorzystaniem zaproponowanego interfejsu.

Etap procesu wytwórczego	Analiza systemowa	
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)	
Artefakt procesu wytwórczego	Model funkcjonalności systemu::Mapa nawigacyjna przypadku użycia	
Stosowane przedrostki diagramów	[NM] (ClassDiagram)	
Stosowane przedrostki elementów	[UI] (Class)	
Opis		
<p>Dla każdego przypadku użycia opracowanie mapy nawigacyjnej – definicja klas reprezentujących elementy interfejsu użytkownika (strony, formularze, panele) oraz możliwych schematów nawigacji pomiędzy tymi elementami (powiązanie elementów zgodnie z ustalonymi możliwościami przejść pomiędzy nimi w ramach realizacji przypadku użycia).</p>		
<p>Uzupełnienie mapy o elementy związane z uruchomieniem przypadku użycia – klasy reprezentujące różne miejsca wywołania przypadku oraz odpowiednie ścieżki (etykietowane zdefiniowanym w profilu stereotypem).</p>		
<p>Wskazanie prezentowanych/przetwarzanych danych dla poszczególnych elementów mapy nawigacyjnej – definicja atrybutów klas mapy nawigacyjnej, powiązanie ich z klasami modelu informacyjnego systemu.</p>		
Narzędzie (wykorzystane diagramy i rozszerzenia)		
<p>Diagram klas UML specyfikujący mapę nawigacyjną (poddigram dla specyfikowanego przypadku użycia), Profil ze standardem.</p>		
<p>Diagram klas UML specyfikujący model informacyjny systemu.</p>		

Etap procesu wytwórczego	Analiza systemowa	 INTERFEJS UŻYTKOWNIKA
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)	
Artefakt procesu wytwórczego	Model funkcjonalności systemu:: Interfejs użytkownika	
Stosowane przedrostki diagramów	[UI] (UserInterfaceDiagram)	
Stosowane przedrostki elementów	N/A	
Opis		
Dla każdej klasy mapy nawigacyjnej opracowanie prototypu interfejsu – projektu ekranu / strony, panelu.		
Narzędzie (wykorzystane diagramy i rozszerzenia)		
Diagram interfejsu użytkownika (poddigram dla klasy z mapy nawigacyjnej modelującej definiowany element interfejsu).		

ZADANIE 13. PROJEKT GRAFICZNY INTERFEJSU UŻYTKOWNIKA

➤ Model funkcjonalności systemu

...

➤ Interfejs użytkownika

☐ [UI] <nazwa_elementuUI> : Class [1..*]

▣ [UI] <nazwa_elementuUI> : UserInterfaceDiagram

INSTRUKCJA LABORATORIUM

1. Dla przypadku użycia należy utworzyć pomocniczy poddiagram z wymaganiami realizowanymi przez przypadek użycia. Diagram należy nazwać jak przypadek użycia, nazwę poprzedzić przedrostkiem [TEMP_RD]:

<przypadek użycia> → Sub Diagrams → New Diagram... → Requirement Diagram

Przypadek użycia (jego widok) należy umieścić na diagramie wymagań. Za pomocą opcji Related elements → Visualize Related Model Element... z menu kontekstowego dla przypadku, należy umieścić na diagramie wymagania, które ten przypadek realizują (powiązane z przypadkiem relacją «satisfy»).

Użyteczne może być przełączenie na widok tabelaryczny – opcja Switch to Tabular View

2. Za pomocą diagramu/ów User Interface Diagram należy przygotować prototyp interfejsu użytkownika – na osobnych diagramach należy zamodelować poszczególne strony, formularze, panele (inne niż standardowe okna dialogowe) wykorzystywane w ramach realizacji przypadku użycia.

Prototyp interfejsu użytkownika należy uzupełnić – zdefiniować jego elementy składowe – zamodelować kontrolki UI lub wkleić jako obraz skreeny prototypów przygotowanych w innym narzędziu.

ZADANIE 14. MODELOWANIE MAPY NAWIGACYJNEJ

➤ Model funkcjonalności systemu

```
...
○ <nazwa_funkcjonalności> : UseCase [1..*]
  [UC] <nazwa_funkcjonalności> : ActivityDiagram
    ○ <nazwa_akcji> : Action [1..*]
  [NM] <nazwa_funkcjonalności> : ClassDiagram
    [UI] <nazwa_klasy> : Class [0..*]
```

➤ Interfejs użytkownika

```
[UI] <nazwa_elementuUI> : Class [1..*]
  [UI] <nazwa_elementuUI> : UserInterfaceDiagram
```

INSTRUKCJA LABORATORIUM

1. Dla przypadku użycia należy utworzyć poddiagram klas z mapą nawigacyjną. Diagram należy nazwać jak przypadek użycia, nazwę poprzedzić przedrostkiem [NM]:

<przypadek użycia> → Sub Diagrams → New Diagram... → Class Diagram

2. Na diagramie należy utworzyć klasy reprezentujące strony, formularze, panele, które umożliwią realizację przypadku użycia (klasy reprezentujące diagramy utworzone jako rozwiązanie pkt. 2). Nazwy klas powinny być takie same jak diagramów, które reprezentują, należy je poprzedzić przedrostkiem [UI]. Na klasy nałożyć odpowiednie stereotypy np. «ServerPage», «ModalPopup»:

<klasa> → Stereotypes → .../BAPProfile.vpp → ServerPage|ModalPopup...

Utworzone klasy należy przenieść do folderu PIM::Model funkcjonalności::Interfejs użytkownika.

Jeżeli w ramach realizacji przypadku użycia wykorzystana będzie strona, formularz, panel współdzielony z innym przypadkiem użycia i klasa reprezentująca element została już utworzona w projekcie, na diagramie z mapą nawigacyjną należy umieścić jej widok.

3. Dla każdej z utworzonych klas mapy nawigacyjnej należy zdefiniować poddiagram interfejsu użytkownika stanowiący projekt elementu reprezentowanego przez klasę (rozwiązanie pkt. 2).

<klasa mapy nawigacyjnej> → Sub Diagrams → Existing Diagram... → <utworzony diagram interfejsu>

4. Za pomocą relacji Dependency należy zamodelować przejścia pomiędzy stronami, formularzami, panelami możliwe w ramach przypadku użycia. Na relację nałożyć odpowiadające stereotypy np. «Navigates», «OpenPopup»:

<przejście> → Stereotypes → .../BAProfile.vpp → Navigates|OpenPopup ...

Dla każdej relacji należy uzupełnić informację o elementach wyzwalających przejście:

<przejście> → Open Specification... (zakładka Tagged Values) → event.Value → <kontrolka UI>

Jeżeli projekt interfejsu użytkownika został wykonany w innym narzędziu, a do projektu wklejony obraz należy dodać (nałożyć na obraz) kontrolki powodujące przejścia pomiędzy elementami interfejsu i wskazać je na mapie nawigacyjnej jako wyzwalacze przejść.

5. Na diagramie z mapą nawigacyjną należy uwzględnić kontekst wywołania przypadku użycia. Na diagramie należy dodać nowe lub widok już istniejących klas reprezentujących elementy interfejsu użytkownika (strony, formularze, panele, menu), z których jest możliwe wywołanie przypadku użycia.

Za pomocą relacji Dependency należy zamodelować przejścia – wywołania przypadku użycia. Na relację nałożyć stereotyp «StartUseCase»:

<przejście> → Stereotypes → .../BAProfile.vpp → StartUseCase

Dla każdej relacji należy uzupełnić informację o elementach wyzwalających przejścia:

<przejście> → Open Specification... (zakładka Tagged Values) → event.Value → <kontrolka UI>

6. Na diagramie z mapą nawigacyjną należy uwzględnić kontekst zakończenia przypadku użycia. Analogicznie jak dla wywołania przypadku użycia należy dodać przejścia, w tym przypadku opisane stereotypem «EndUseCase».

Etap procesu wytwórczego	Analiza systemowa
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)
Artefakt procesu wytwórczego	Model funkcjonalności systemu::Interfejs użytkownika:: Ograniczenia systemowe (II) (specyfikacja ograniczeń nałożonych na elementy interfejsu użytkownika)
Stosowane przedrostki diagramów	N/A
Stosowane przedrostki elementów	N/A

Opis
Definicja wymagań nałożonych na elementy interfejsu użytkownika.
Powiązanie składowych interfejsu użytkownika z danymi dostępnymi w systemie.
Narzędzie (wykorzystane diagramy i rozszerzenia)
Diagram interfejsu użytkownika, Definicja wymagań SysML (odpowiednio dostosowanych do potrzeb specyfikacji elementów interfejsu) na osobnej warstwie diagramu, Profil ze standardem, Wyrażenia języka OCL.
Diagram klas specyfikujący mapę nawigacyjną przypadku użycia, Diagram klas specyfikujący model informacyjny systemu, Diagram interfejsu użytkownika, Wymagania SysML, Profil ze standardem

ZADANIE 15. SZCZEGÓŁOWA SPECYFIKACJA INTERFEJSU UŻYTKOWNIKA – KONFIGURACJA KONTROLEK, DEFINICJA WARUNKÓW POPRAWNOŚCI

INSTRUKCJA LABORATORIUM

1. Prototyp interfejsu użytkownika należy uzupełnić – za pomocą stereotypów zdefiniowanych w profilu BAPProfile należy skonfigurować kontrolki UI, tj. wyspecyfikować warunkową widoczność, dostępność poszczególnych kontrolek, wartości domyślne, treści chmurzek z podpowiedziami, dla list rozwijanych wskazać możliwe wartości do wyboru, wartość wybraną domyślnie, itd.
2. Dla kontrolek interfejsu należy zdefiniować warunki poprawności – za pomocą stereotypów zdefiniowanych w profilu BAPProfile, które odpowiadają różnym walidacjom należy zdefiniować reguły poprawności wprowadzonych/edytowanych danych. Dla poszczególnych ograniczeń należy zdefiniować treści komunikatów informujących o naruszeniu warunków poprawności. Treść komunikatu należy zdefiniować jako literal enumeratora z wszystkimi komunikatami systemu `PIM::Model informacyjny systemu::Słowniki::Komunikaty` i wskazać przez referencję.

ZADANIE 16. SZCZEGÓŁOWA SPECYFIKACJA INTERFEJSU UŻYTKOWNIKA – MAPOWANIE PREZENTOWANYCH DANYCH

INSTRUKCJA LABORATORIUM

1. Mapę nawigacyjną należy uzupełnić o informacje na temat danych prezentowanych/wprowadzanych w ramach poszczególnych elementów interfejsu (stron, formularzy, paneli) reprezentowanych przez klasy mapy – tj. dla każdej klasy mapy nawigacyjnej, dla każdego obiektu i/lub listy obiektów prezentowanej/wprowadzanej na stronie/formularzu/panelu reprezentowanym przez klasę należy zdefiniować atrybut klasy.


Jako typ atrybutu klasy mapy nawigacyjnej należy wskazać klasę z modelu informacyjnego systemu. W szczególności należy uwzględnić dane nieutralane wykorzystywane w ramach opisywanych elementów (np. kryteria wyszukiwania, struktury raportów, struktury plików importu/eksportu danych). Klasy reprezentujące dane nieutralane należy dodać do mapy nawigacyjnej przypadku użycia, opisać je stereotypem «Transient»:

`<klasa> → Stereotypes → .../BAProfile.vpp → Transient`

2. Na prototypie interfejsu użytkownika należy zamodelować mapowanie danych 'wprowadzonych' na dane 'zapisywane' w systemie.

PROJEKT 6B. SPECYFIKACJA PRZYPADKÓW UŻYCIA

Lab#	Temat	Uwagi	Rezultaty (punkty)	Narzędzia	Elementy podlegające sprawdzeniu
6B	Specyfikacja przypadków użycia.	<p>Dla dwóch ustalonych z prowadzącym przypadków użycia studenci piszą w narzędziu (alternatywnie):</p> <p>1) specyfikację wejścia/wyjścia dla każdej z historii powiązanych z przypadkiem użycia + powiązane reguły biznesowe; wątki alternatywne na poziomie tytułów</p> <p>2) scenariusze przypadków (główny i alternatywne) w wersji tekstowej</p> <p>*) dodatkowo studenci mogą wizualizować specyfikację przypadków użycia za pomocą diagramów aktywności</p>	<p>Specyfikacja PU dla dwóch PU – po 5p. za PU;</p> <p>alternatywnie diagramy aktywności (po 7.5p* za diagram dla PU)</p>	<p>Visual Paradigm – specyfikacja PU w postaci scenariuszy lub specyfikacja historyjek w ramach PU (opis we/wy, powiązanych reguł biznesowych)</p> <p>Visual Paradigm – diagramy aktywności</p>	Spójność wewnętrzna.

Etap procesu wytwórczego	Analiza systemowa	
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)	
Artefakt procesu wytwórczego	Model funkcjonalności systemu:: Przebieg przypadku użycia	
Stosowane przedrostki diagramów	[UC] (ActivityDiagram)	
Stosowane przedrostki elementów	N/A	
Opis		
<p>Dla każdego przypadku użycia specyfikacja przebiegu – opracowanie scenariusza interakcji użytkownik-system, stanowiącego propozycję realizacji wymagań biznesu wskazanych jako realizowanych przez przypadek.</p> <p>Uwzględnienie przepływu obiektów w przebiegu przypadku użycia – powiązanie opisu sposobu korzystania z systemu (akcji przebiegu) z informacjami/danymi, na jakich pracuje system (z klasami modelu informacyjnego systemu).</p> <p>Przypisanie odpowiedzialności za realizację akcji z warstwy prezentacji przebiegu przypadku użycia poszczególnym elementom mapy nawigacyjnej.</p>		

Powiązanie elementów przebiegu przypadku użycia (tj. zdarzeń, przejść) z elementami prototypu interfejsu użytkownika (tj. z zaprojektowanymi kontrolkami).

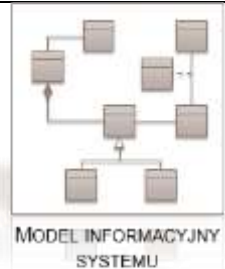
Narzędzie (wykorzystane diagramy i rozszerzenia)

Diagram aktywności UML (poddiagram dla specyfikowanego przypadku użycia), Profil ze standardem.

Diagram klas UML - dla obiektów/pinów akcji wskazanie odpowiednich typów – tj. klas dziedzicznych z modelu informacyjnego systemu.

Diagram klas UML specyfikujący mapę nawigacyjną przypadku użycia, Profil ze standardem (wskazanie elementów mapy nawigacyjnej - atrybut `elementUI` stereotypu «Presentation»).

Diagram aktywności UML, Diagram interfejsu użytkownika, Profil ze standardem (wskazanie elementów interfejsu użytkownika stanowiących wyzwalacz przejść, zdarzeń - atrybut `event` stereotypu «EventDriven»).

Etap procesu wytwórczego	Analiza systemowa	
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)	
Artefakt procesu wytwórczego	Model informacyjny systemu	
Stosowane przedrostki diagramów	[PIM] (ClassDiagram)	
Stosowane przedrostki elementów	N/A	
Opis		
Identyfikacja klas reprezentujących dane tymczasowe (nieutrwalane) przetwarzane w ramach realizacji funkcjonalności – poszczególnych przypadków użycia.		
Narzędzie (wykorzystane diagramy i rozszerzenia)		
Diagram pakietów UML, Diagram klas UML (z mapą nawigacyjną przypadku użycia w ramach, którego przetwarzany jest obiekt nieutrwalany), Profil ze standardem (stereotyp «Transient» dla klas specyfikujących dane tymczasowe / nieutrwalane).		

Etap procesu wytwórczego	Analiza systemowa
Model/poziom zgodnie z założeniami podejścia MDA	Platform independent model (PIM)

Artefakt procesu wytwórczego	Model funkcjonalności systemu::Przebieg przypadku użycia:: Ograniczenia systemowe (III) (specyfikacja akcji logiki biznesowej systemu)
Stosowane przedrostki diagramów	N/A
Stosowane przedrostki elementów	N/A
Opis	
Specyfikacja akcji logiki biznesowej systemu – definicja wymagań odnoszących się do sposobu realizacji akcji systemowych.	
Narzędzie (wykorzystane diagramy i rozszerzenia)	
Diagram aktywności UML, Definicje ograniczeń (elementów Constraint) nałożonych na akcje warstwy logiki biznesowej – wyrażen języka OCL (<i>pre</i> , <i>post</i> , <i>body</i>).	

ZADANIE 16. MODELOWANIE SCENARIUSZA PRZYPADKU UŻYCIA

INSTRUKCJA LABORATORIUM

1. Dla przypadku użycia należy utworzyć poddiagram aktywności. Diagram należy nazwać jak przypadek użycia, nazwę poprzedzić przedrostkiem [UC]:

<przypadek użycia> → Sub Diagrams → New Diagram... → Activity Diagram

2. Na diagramie należy zdefiniować partycje (*ActivityPartition*) reprezentujące uczestników opisywanej interakcji i grupujące kroki interakcji zgodnie z odpowiedzialnością za ich wykonanie: Aktora, Warstwę prezentacji, Warstwę logiki systemu.

Dla partycji reprezentującej aktora należy wskazać odpowiedniego aktora z modelu funkcjonalności systemu:

<partycja> → Open Specification... (zakładka General) → Represents: *<aktor>*

3. Jako akcje (*Action*) diagramu aktywności należy zdefiniować poszczególne kroki scenariusza:
 - Działania, za realizację których odpowiedzialny jest aktor (m. in. akcje wprowadzania / modyfikacji danych, wyboru obiektu do dalszego przetwarzania, wyboru opcji czy potwierdzenia decyzji).
 - Działania uwzględnione w partycji warstwy prezentacji – akcje związane z wizualizacją elementów interfejsu użytkownika prezentujące dane i informacje, umożliwiające edycję danych, wybór obiektów czy opcji. Jeżeli akcja jest realizowana za pomocą zaprojektowanego elementu interfejsu użytkownika na akcje należy nałożyć stereotyp «Presentation». Element interfejsu należy wskazać za pomocą atrybutu:

<akcja> → Open Specification... (zakładka Tagged Values) → elementUI.Value → <klasa mapy nawigacyjnej>.

Jeżeli akcja jest realizowana za pomocą standardowego okna dialogowego na akcje należy nałożyć stereotyp «PresentationMessage». Typ okna oraz komunikat okna dialogowego należy wskazać za pomocą atrybutów stereotypu. Treść komunikatu należy zdefiniować jako literał enumeratora z wszystkimi komunikatami systemu PIM::Model informacyjny systemu::Słowniki::Komunikaty i wskazać przez referencję.

typ okna: <akcja> → Open Specification... (zakładka Tagged Values) → dialogType.Value → <typ>,

komunikat: <akcja> → Open Specification... (zakładka Tagged Values) → message.Value → <komunikat>.

- Zgrupowane w warstwie logiki działania związane z realizacją logiki biznesowej systemu (m in. akcje odczytu i zapisu danych trwałych, weryfikacji poprawności przetwarzanych danych, transformacji danych czy wywoływania usług systemów zewnętrznych).
4. W definicji poszczególnych akcji należy uwzględnić dane przetwarzane przez akcję – należy zdefiniować piny wejściowe i wyjściowe akcji. Jako typ każdego z pinów należy wskazać klasę z modelu informacyjnego systemu, jeżeli przetwarzane dane nie są danymi trwałymi należy zdefiniować i odpowiednio wyróżnić klasę reprezentującą dane tymczasowe.

PROJEKT 15. OCENA PROJEKTU

Poniżej zamieszczono listę pytań weryfikujących częściowo poprawność kilku artefaktów wytworzonych w ramach realizowanego procesu wytwórczego.

1. Struktura projektu (podział na modele, pakiety oraz zagnieżdżenie w nich elementów) jest prawidłowa: TAK / NIE / CZĘŚCIOWO

Przykłady nieprawidłowości:

2. Nie ma w projekcie 'śmieci' (elementów, które nie zostały usunięte z repozytorium): TAK / NIE / CZĘŚCIOWO

Przykłady nieprawidłowości:

SŁOWNIK TERMINÓW

3. Zostały zdefiniowane wszystkie ważne z perspektywy projektu (wynikające z definicji procesów biznesowych oraz reguł biznesowych) terminy: TAK / NIE / CZĘŚCIOWO

Przykłady braku:

4. Dla wszystkich terminów zostały przygotowane definicje: TAK / NIE / CZĘŚCIOWO

Przykłady braku:

5. Definicje terminów są jednoznaczne i zgodne z dziedziną problemu: TAK / NIE / CZĘŚCIOWO

Przykłady nieprawidłowości:

6. W słowniku zostały uwzględnione wszystkie synonimy terminów, które pojawiają się w dokumentach dziedziny problemu: TAK / NIE / CZĘŚCIOWO

Przykłady braku:

7. W słowniku zostały uwzględnione wszystkie 'inne formy zapisu' użyte na potrzeby definicji klas modelu pojęć: TAK / NIE / CZĘŚCIOWO

Przykłady braku:

MODEL POJĘĆ

8. Zostały zdefiniowane wszystkie ważne z perspektywy projektu pojęcia: TAK / NIE / CZĘŚCIOWO

Przykłady braku:

9. Dla wszystkich klas został zdefiniowany opis słowny pojęcia reprezentowanego przez klasę (tj. referencja na definicję ze słownika terminów): TAK / NIE / CZĘŚCIOWO

Przykłady niekompletności:

10. Model jest spójny z definicjami terminów ze słownika: TAK / NIE / CZĘŚCIOWO

Przykłady niespójności:

11. Zostały zdefiniowane wszystkie ważne z perspektywy projektu (wynikające z definicji procesów biznesowych oraz reguł biznesowych) atrybuty oraz asocjacje, z uwzględnieniem atrybutów statycznych oraz wnioskowanych: TAK / NIE / CZĘŚCIOWO

Przykład niekompletności:

12. Model jest spójny z regułami biznesowymi: TAK / NIE / CZĘŚCIOWO

Przykłady niespójności:

13. Brak błędów składniowych diagramu klas: TAK / NIE / CZĘŚCIOWO

Przykłady błędów:

14. Dla wszystkich atrybutów został ustalony prawidłowy typ danych (predefiniowany typ UML lub zdefiniowany w projekcie typ danych: prosty, złożony, wyliczeniowy): TAK / NIE / CZĘŚCIOWO

Przykłady braku typu:

Przykłady nieprawidłowości definicji:

15. Nazwy wszystkich właściwości klas (atrybutów oraz ról – z uwzględnieniem domyślnych) są unikalne: TAK / NIE / CZĘŚCIOWO

Przykłady nieprawidłowości:

16. Dla wszystkich asocjacji zostały zdefiniowane nazwy oraz ich kierunek odczytu albo zaznaczono, że jest to zależność całość - część (agregacja lub kompozycja). Nie nadużyto nazw: 'posiada', 'dotyczy', 'ma': TAK / NIE / CZĘŚCIOWO

Przykłady nieprawidłowości:

17. Dla wszystkich ról (końców asocjacji) zostały zdefiniowane nazwy (wyjątek stanowią role, dla których nazwa domyślna jest prawidłowa): TAK / NIE / CZĘŚCIOWO

Przykłady braków:

18. Dla wszystkich właściwości (atrybutów oraz ról) została zdefiniowana prawidłowa (zgodna z dziedziną problemu) liczność: TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności:

19. Dla kluczowych klas modelu zostały wskazane prawidłowe (zgodne z dziedziną problemu) właściwości identyfikujące obiekty danej klasy (wykorzystanie ograniczenia `id` i/lub kwalifikatora): TAK / NIE / CZĘŚCIOWO

Przykłady braku:

Przykłady niezgodności:

20. Dla wszystkich właściwości – kolekcji został prawidłowo ustalony ich rodzaj (wykorzystanie ograniczeń `ordered`, `nonunique`, `sequence`): TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności:

21. Dla każdej hierarchii dziedziczenia zostały określone prawidłowo (zgodnie z dziedziną problemu) cechy: `complete/incomplete`, `disjoint/overlapping`: TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności:

22. Nazwy wszystkich klas zostały zapisane zgodnie z konwencjami UML: TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności:

23. Nazwy wszystkich atrybutów zostały zapisane zgodnie z konwencjami UML: TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności:

24. Nazwy wszystkich asocjacji zostały zapisane zgodnie z konwencjami UML: TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności:

25. Nazwy wszystkich ról zostały zapisane zgodnie z konwencjami UML: TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności:

REGUŁY BIZNESOWE

26. Nazwy reguł odzwierciedlają ich zawartość: TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności:

27. Wszystkie reguły zostały prawidłowo zaklasyfikowane: TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności:

28. Treści wszystkich reguł zostały prawidłowo zapisane (np. w RuleSpeak zastosowano właściwy szablon, podmiot został zapisany w liczbie pojedynczej): TAK / NIE / CZĘŚCIOWO

Przykłady niezgodności: