Inżynieria oprogramowania II

Wykład 5:

"UPEDU: Rozpoznanie wymagań
(ang. *requirements discipline*)"

Marek Krętowski
e-mail: mkret@wi.pb.edu.pl
http://aragorn.pb.bialystok.pl/~mkret

Na podstawie podręcznika: „Software Engineering Process with the UPEDU" P. Robillard, P. Kruchten, P. d'Astous, Addison-Wesley, 2003

Wprowadzenie

- Dyscyplina wymagań składa się z grupy czynności wykonywanych w celu wychwycenia, walidacji i zarządzania wymaganiami wobec systemu
- Wymagania mogą być postrzegane jako swego rodzaju kontrakt (porozumienie) pomiędzy klientem i organizacją tworzącą system
- Stanowią punkt odniesienia dla osoby zarządzającej projektem i umożliwiają kontrolowaną ewolucję systemu
- Wykorzystywane są ponadto przez zespół projektowy, testerów oprogramowania, grupy zapewnienia jakości oraz osoby przygotowujące dokumentację
- Celem wymagań jest stworzenie bazy, który pozwoli na efektywną komunikację pomiędzy wszystkimi zaangażowanymi w proces wytwórczy
- Na wstępie podejmowane są decyzje typu: Co stanowi wymaganie? Jaki format dokumentów i ich formalność? Poziom szczegółowości opisu, relatywne priorytety i szacowane koszty poszczególnych życzeń, wstępny zakres systemu, ...

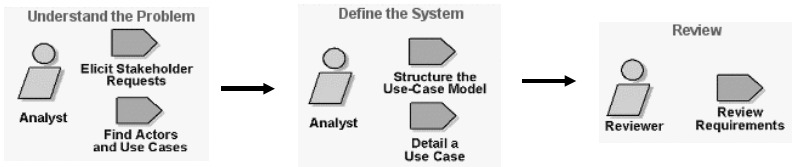
Zainteresowani i zaangażowani
(ang. *stakeholders*)

- Kluczem do opracowania efektywnych wymagań jest wychwycenie osób lub organizacje
  - **mające bezpośredni lub pośredni wpływ** na opracowywane elementy wymagań,
  - **na których działanie będzie miał wpływ** (może mieć) tworzony system,
  - również użytkownicy końcowi systemu,
  - osoby zarządzające procesem wytwórczym
  - osoby biorące udział w procesach organizacji, na które wpłynie tworzony system
  - inżynierowie odpowiedzialni za rozwój i utrzymanie systemu
  - klienci organizacji, którzy korzystają z systemu
  - zewnętrzne ciała jak organizacje kontrolne lub certyfikujące
- Widzą rozwiązywany problem z różnych perspektyw i mogą mieć odmienne oczekiwania w stosunku do systemu
- Wybierając osoby biorące udział w zdobywaniu informacji należy brać pod uwagę ich wiedzę, umiejętności komunikacji oraz dostępność (najlepiej sprawdza się niewielka grupa; od 2 do 5 osób)

Czynności procesu

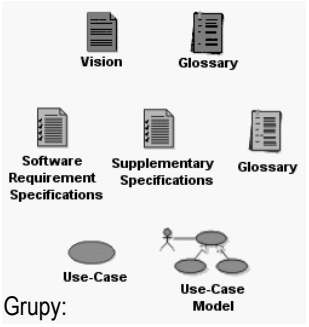
- Najpierw należy zrozumieć problem, który system będzie rozwiązywał a następnie zdefiniować system, który ma zostać stworzony
- Wyróżniono 3 grupy czynności (łącznie 5 czynności): 4 wykonywane przez rolę Analityka, jedna przez Recenzenta (ang. reviewer)

  wydobycie oczekiwań zainteresowanych; wyszukanie aktorów i przypadków użycia; zbudowanie modelu przypadków użycia; uszczegółowienie przypadków użycia; przegląd wymagań
- Niezbędne jest przetłumaczenie i zorganizowanie sposobu rozumienia systemu przez poszczególne osoby w sensowne opisy zamierzonego systemu



Artefakty wymagań

- Na ich bazie możliwe jest wypracowanie uzgodnień pomiędzy klientem a użytkownikami co do tego co system powinien robić
- Pozwalają zespołowi wykonawczemu zrozumieć wymagania w stosunku do systemu
- Ograniczają zakres systemu
- Nawet najbardziej starannie dokumentowane wymagania ulegają zwykle zmianom w kolejnych wersjach systemu
- Złożoność zarządzania zmianami wzrasta, gdy zmiany pewnych wymagań mają wpływ na inne
- Struktura w ramach której gromadzone są wymagania powinna być odpowiednio przygotowana na wprowadzanie zmian i zawierać możliwość definiowania łatwych do prześledzenia związków pomiędzy wymaganiami i innymi dokumentami procesu



Grupy:

- wejściowe (ang. input): wizja i słownik
- opisowe (ang. descriptive): specyfikacje wymagań
- związane z modelowaniem: przypadki użycia i ich model

Definiowanie wymagań

- Wymagania dla systemu są zawsze wyprowadzane z istniejącej wiedzy, którą należy odpowiednio zorganizować w celu przedstawienia systemu
- Wiedza ta jest jednak rozproszona pomiędzy wiele osób
- Informacje związane z planowanym systemem zwykle ograniczone do wyobrażeń zainteresowanych osób, natomiast pewna wiedza związana z funkcjonalnością systemu może być znaleziona w literaturze dziedziny
- Przyjmuje się, że wymagania wobec systemu powstają na bazie zrozumienia potrzeb wszystkich zainteresowanych podmiotów, informacji związanych z konkretnym, rozpatrywanym zastosowaniem oraz na podstawie dokumentu wizji
- Słownik, którego pozycje są częściowo wypełniane podczas rozpoznawania wymagań, definiuje terminy z dziedziny problemu niezbędne do zrozumienia wymagań

Wizja (ang. *vision*)

- Dokument tworzony przez nie-techników w celu przedstawienia zainteresowanym osobom opisu systemu na wysokim poziomie ogólności
- Czasami wizja jest wykorzystywana w umowach dotyczących realizacji projektu, gdyż zawiera zwykle większość, przedstawionych w formie narracyjnej, wymagań dla produktu
- UPEDU przyjmuje, że dokument wizji jest dostępny czyli jego budowa nie jest częścią procesu,
- Dokumentacja na poziomie systemu odpowiadająca na podstawowe pytania **Co?** i **Dlaczego?** dotyczące systemu

"whats + whys"

- Wykorzystywana jako odniesienie, z którym konfrontowane powinny być wszystkie przyszłe decyzje
- Skoncentrowana na: celu, potrzebach użytkowników, docelowym rynku, środowisko użytkowników i platformach programowo-sprzętowych, cechach produktu

Słownik (ang. *glossary*)

- Definiuje ważne, specyficzne terminy związane najczęściej z dziedziną (terminologią) problemu, które będą wykorzystywane podczas projektu
- Pozwala uniknąć nieporozumień a ponadto może być pomocny dla osób studiujących np. przypadki użycia do wyjaśnienia pojawiających się pojęć
- Istotne jest aby był tylko jeden słownik i aby terminy w nim zawarte były konsekwentnie wykorzystywane we wszystkich tekstowych opisach systemu

Specyfikacje wymagań systemu
(ang. Software Requirements Specifications)

- Przedstawiają wymagania funkcjonalne (funkcje, które system musi zapewniać oraz które mogą być testowane)
- Dokumenty zwykle zachowują strukturę standardu IEEE 830-1993 (1. Cel i zakres; 2. Opis czynników wpływających na planowany system i ich wymagań; 3. Wymagania programowe)
- Cechy dobrze skonstruowanego SRS (wg IEEE 830-1993):
  - poprawny (każde wymaganie przyczynia się do realizacji potrzeb)
  - kompletny (zawiera wszystkie istotne wymagania, odpowiedzi na wszystkie wejścia)
  - spójny (brak konfliktów pomiędzy wymaganiami)
  - niedwuznaczny (każde wymagania ma tylko jedną interpretację)
  - uporządkowany wg ważności i stabilności
  - weryfikowalny
  - modyfikowalny (łatwość, kompletność i spójność zmian)
  - ...

Dodatkowe wymagania
(ang. supplementary specifications)

- Przedstawiają wymagania niefunkcjonalne i zwykle narzucają ograniczenia na sam produkt (system) lub proces wytwórczy; mogą również dotyczyć sposobu utrzymania systemu (ang. maintenance)
- Jedna z klasyfikacji wymagań niefunkcjonalnych wyróżnia 5 dodatkowych atrybutów związanych z wymaganiami:
  - użyteczność (ang. usability) - czynniki ludzkie, estetykę, łatwość poznania i wykorzystania, spójność interfejsu użytkownika, dokumentację użytkownika, ...
  - niezawodność (ang. reliability) - obejmuje częstość i wagę błędnych wykonań
  - wydajność - performance - nakłada ograniczenia (np. czasowe) na wymagania funkcjonalne (np. poziom tranzakcyjności, czas odpowiedzi, czas odbudowy, wykorzystanie pamięci)
  - rozbudowywalność (ang. supportability) -możliwość naprawy, modyfikacji i rozwoju po przekazaniu do użytkowania
  - ograniczenia projektowe

Specyfikacje wymagań

Software Requirements Specification	The Software Requirements Specification (SRS) captures the complete software requirements for the system, or a portion of the system. When using use-case modeling, this artifact consists of a package containing use cases of the use-case model and applicable Supplementary Specifications.
Role:	Analyst
More Information:	➜ Checkpoints:Software Requirements Specification ➜ Concepts: Requirements ➜ Guidelines: SRS/Non-Functional Requirements

Supplementary Specifications	The Supplementary Specifications capture the system requirements that are not readily captured in the use cases of the use-case model. Such requirements include: ➜ Legal and regulatory requirements, and application standards ➜ Quality attributes of the system to be built, including usability, reliability, performance, and supportability requirements ➜ Other requirements such as operating systems and environments, compatibility requirements, and design constraints
Role:	Analyst
Enclosed in:	Software Requirements Specification
More Information:	➜ Checkpoints:Supplementary Specifications ➜ Concepts: Requirements ➜ Guidelines: SRS/Non-Functional Requirements

Elementy modeli

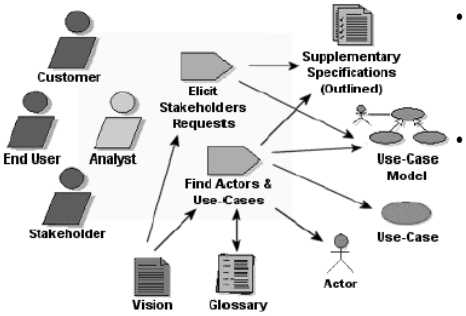
- Użytkownikom oprogramowania zwykle łatwiej jest przedstawić przykładowe działanie planowanego systemu niż w sposób abstrakcyjny zdefiniować jego wymaganą funkcjonalność
- Aby to ułatwić a jednocześnie uporządkować wprowadzona została notacja przypadków użycia (opisują one interakcję pomiędzy użytkownikiem a systemem koncentrując się na tym co system “robi” dla użytkownika)
- Notacja przypadków użycia może być w łatwy sposób wytłumaczona dla potencjalnego użytkownika (prosta notacja graficzna, ustrukturalizowany opis w języku naturalnym); dzięki czemu możliwa jest praca bezpośrednio z użytkownikami w celu opisania (lub zdefiniowania) zachowania systemu
- Specyficzna (szczególna) ścieżka zdarzeń obserwowalna w ramach przypadków użycia nazywana jest scenariuszem; często łatwiej jest rozpoczynać od konkretnych scenariuszy, które następnie są sukcesywnie rozszerzane aż do osiągnięcia pełnych przypadków użycia

Artefakty modelowe

Use Case	A use case defines a set of use-case instances, where each instance is a sequence of actions a system performs that yields an observable result of value to a particular actor.
UML representation:	Use Case
Role:	Analyst
Enclosed in:	Software Requirements Specification
More information:	→ Guidelines:Use Case → Checkpoints:Use Case

Use-Case Model	The use-case model is a model of the system's intended functions and its environment, and serves as a contract between the customer and the developers. The use-case model is used as an essential input to activities in analysis, design, and test.
UML representation:	Model, stereotyped as «use-case model».
Role:	Analyst
More information:	→ Guidelines:Use-Case Model → Guidelines:Use-Case Diagram → Checkpoints:Use-Case Model

Zrozumienie problemu



- Celem jest przede wszystkim zebranie i wydobycie informacji od zaangażowanych osób aby zrozumieć ich rzeczywiste potrzeby
- Uzyskane postulaty (wnioski) mogą być traktowane jako tzw. lista życzeń (ang. wish-list) i podstawowe źródło wykorzystane do zdefiniowania generalnych wymagań (poziom Wizji)

Wykorzystywane techniki:

- rozmowy (wywiady)
- warsztaty wymagań oraz redukcja pomysłów
- burze mózgów
- przeglądy istniejących wymagań

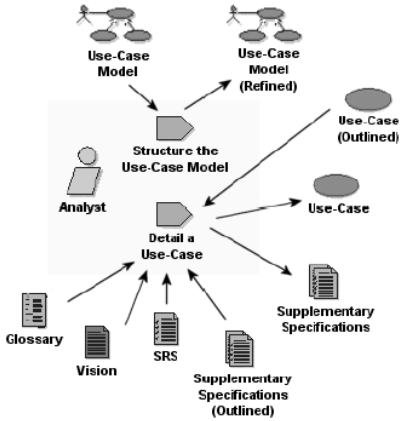
Wydobywanie oczekiwań

Purpose	
→ To understand who are the stakeholders of the project. → To collect requests on what needs the system should fulfill. → To prioritize stakeholder requests.	
Steps	
→ Determine Sources for Requirements → Gather Information → Conduct Requirements Workshops → Evaluate Your Results	
Input Artifacts: → Vision	Resulting Artifacts: → Use-Case Model → Supplementary Specifications
Role: Analyst	

Znajdowanie aktorów i przypadków użycia

Purpose	
→ To outline the functionality of the system. → To define what will be handled by the system and what will be handled outside the system. → To define who and what will interact with the system. → Divide the model into packages with actors and use cases. → Create diagrams of the use-case model. → Develop a survey of the use-case model.	
Steps	
→ Find Actors → Find Use Cases → Describe How Actors and Use Cases Interact → Present the Use-Case Model in Use-Case Diagrams → Develop a Survey of the Use-Case Model → Evaluate Your Results	
Input Artifacts: → Vision → Glossary	Resulting Artifacts: → Actor → Use Case → Use-Case Model → Supplementary Specifications → Glossary
Role: Analyst	
Guidelines: → Use-Case Workshop	

Zdefiniowanie systemu



- Wstępne definicje kluczowych elementów wyróżnione podczas czynności związanych ze zrozumieniem problemu podlegają uszczegółowieniu i ustrukturalizowaniu (aktorzy, przypadki użycia)
- Wymagania niefunkcjonalne (globalne) zawarte w Dodatkowych specyfikacjach podlegają rozszerzeniu

- Align the project team in their understanding of the system.
- Perform a high-level analysis on the results of collecting stakeholder requests.
- Refine the Vision to include the features to include in the system, along with appropriate attributes.
- Refine the use-case model, to include outlined use cases.
- More formally document the results in models and documents.

Strukturalizacja modelu przypadków użycia

Purpose

- To extract behavior in use cases that need to be considered as abstract use cases. Examples of such behavior are common behavior, optional behavior, exceptional behavior, and behavior that is to be developed in later iterations.
- To find new abstract actors that define roles that are shared by several actors.

Steps

- Establish Include-Relationships Between Use Cases
- Establish Extend-Relationships Between Use Cases
- Establish Generalizations Between Use Cases
- Establish Generalizations Between Actors
- Evaluate Your Results

Input Artifacts:	Resulting Artifacts:
Use-Case Model	Use-Case Model

Role: Analyst

Uszczegółowienie przypadków użycia

Purpose

- To describe the use case's flow of events in detail.
- To describe the use case's flow of events so that the customer and the users can understand it.

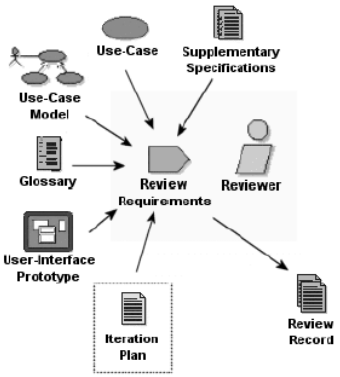
Steps

- Detail the Flow of Events of the Use Case
- Structure the Flow of Events of the Use Case
- Illustrate Relationships with Actors and Other Use Cases
- Describe the Special Requirements of the Use Case
- Describe Communication Protocols
- Describe Preconditions of the Use Case, <optional>
- Describe Postconditions of the Use Case, <optional>
- Describe Extension Points, <optional>
- Evaluate Your Results

Input Artifacts:	Resulting Artifacts:
Glossary Software Requirements Specifications Supplementary Specifications Use Case Vision	Use Case Supplementary Specifications

Role: Analyst

Przegląd wymagań



- Formalna weryfikacja opracowanych wymagań i odniesienie ich do oczekiwań klienta
- Istotną kwestią jest śledzenie zmian w wymaganiach
- Regularne przeglądy powinny być wykonywane każdorazowo gdy następują zmiany (nie tylko podczas rozpoznawania wymagań)

Przeglądanie wymagań

Purpose

- To formally verify that the results of Requirements conform to the customer's view of the system.

Steps

- Conduct Review Meetings

Checkpoints

- Vision
- Use-Case Model
- Actor
- Use Case
- Supplementary Specifications
- Software Requirements Specification
- Glossary

Input Artifacts:	Resulting Artifacts:
Glossary Iteration Plan Supplementary Specifications Use Case Use-Case Model User-Interface Prototype Vision	Review Record

Role: Reviewer

Work Guidelines:

- Reviews