Uniwersytet Zielonogórski

Wydział Informatyki, Elektrotechniki i Automatyki

Kierunek: *Informatyka*

Specjalność: *Sieciowe systemy informatyczne*

Student (Grupa): *Piotr Joński (431IDZ)*

Promotor: *dr inż. Andrzej Marciniak*

Konsultant:

Temat Pracy:

- wersja polska — ***System zarządzania projektami dedykowany metodyce Scrum***
- wersja angielska — ***Project management system dedicated to Scrum methodology***

Cel Pracy:
PRZEPISAĆ Z KARTY DYPLOMOWEJ

Zakres Pracy:

1. Zakres pracy 1
2. Zakres pracy 2
3. Zakres pracy 3

Data podjęcia tematu: 01.01.2000

Podpisy:

Promotor	Student

Data przyjęcia pracy	Podpis promotora

Wyrażam zgodę na wykorzystanie do celów dydaktycznych i badawczych wyników zawartych w pracy dyplomowej i powstałych programów komputerowych.

Data	Podpis studenta

Uniwersytet Zielonogórski

Wydział Informatyki, Elektrotechniki i Automatyki

Praca dyplomowa

Kierunek: Informatyka

System zarządzania projektami dedykowany metodyce Scrum

Piotr Joński

Promotor:
dr inż. Andrzej Marciniak

Zielona Góra, 02 2016

Piotr Joński
431IDZ
Wydział Informatyki,
Elektrotechniki i Automatyki UZ

Zielona Góra 10/02/2016

OŚWIADCZENIE

Świadomy odpowiedzialności karnej oświadczam, że przedkładana praca dyplomowa pt.

System zarządzania projektami dedykowany metodyce Scrum

została napisana przeze mnie samodzielnie i nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadaniem dyplomu wyższej uczelni lub tytułów zawodowych. Jednocześnie oświadczam, że w/w praca nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994r. o prawie autorskim i prawach pokrewnych innych osób (DZ. U. z roku 2000 Nr 80 poz. 904) oraz dóbr osobistych chronionych prawem cywilnym.

Oświadczam również, że egzemplarz pracy dyplomowej w formie wydruku komputerowego jest zgodny z egzemplarzem pracy dyplomowej w formie elektronicznej.

.....

podpis

Streszczenie

CO TU DAĆ?

słowa kluczowe: praca dyplomowa, skład komputerowy, formatowanie dokumentu.

Spis treści

Spis rysunków

Spis tabel

Rozdział 1

Wstęp

1.1. Wprowadzenie i przegląd literatury

Co tu dać?

1.2. Cel i zakres pracy

Celem pracy było wytworzenie systemu do zarządzania projektami dedykowanego metody Scrum. Obecnie na rynku jest wiele zarówno darmowych jak i płatnych narzędzi, które oferują możliwość prowadzenia projektów rożnymi metodykami. W zakres pracy wchodzą:

1. zapoznanie się z literaturą tematu,
2. opracowanie założeń projektu,
3. spis wymagań funkcjonalnych i niefunkcjonalnych systemu,
4. implementacja wszystkich funkcjonalności oraz usunięcie powstałych błędów,
5. przetestowanie systemu ,
6. wytworzenie części opisowej pracy.

1.3. Struktura pracy

Opis rozdziałów zostanie uzupełniony na końcu pracyitemize

Rozdział 2

Wprowadzenie

2.1. Szkic problemu

Szybki rozwój w dziedzinie informatyki spowodował wzrost zapotrzebowania na systemy, które pomogłyby rozwiązywać problemy kwestii organizacyjnej projektów. W dzisiejszych czasach zarządzanie projektami jest szerokim zagadnieniem, a na jego temat powstało wiele publikacji. Autorzy proponują takie rozwiązania dot. zarządzania projektami jak:

1. scrum,
2. lean Softrware Development,
3. feature Driven Development,
4. test Driven Develipment.

Nie jest to wyczerpująca lista metodyk - istnieje wiele innych metod tworzenia oprogramowania, jednak opis wszystkich wykracza po za zakres tej pracy. W mojej pracy dyplomowej poruszam temat systemu do zarządzania projektami, który jest dedykowany metodyce Scrum. Nie bez powodu wybrana została właśnie ta zwinna metodyka - jest to na chwilę obecną najbardziej popularna metoda wytwarzania oprogramowania, a umiejętności pracy wraz z nią są pożądane przez większość pracodawców na całym świecie.

Scrum jest zwinną metodyką wytwarzania oprogramowania, której początki sięgają połowy lat 80. Aby prawidłowo przedstawić omawiany temat należy zapoznać się z pojęciami, które są nieodłącznym elementem tej metody:

1. **product owner** - osoba odpowiedzialna za projekt,

2. **scrum master** - osoba, która pomaga zespołowi w organizacji czasu pracy a także rozwiązuje powstałe problemu,

3. **zespół developerski** - zespół programistów wspólnie pracujący nad wytworzeniem produktu,

4. **backlog produktu** - jest to zbiór zadań / funkcjonalności wchodzące w skład wytwarzanego projektu,

5. **sprint** - interwały czasowe, w których realizowane są zadania. Bezpośrednio przed każdym sprintem występuje planowanie, czyli wybieranie funkcjonalności do zrealizowania w danym sprincie. Bezpośrednio po sprincie powinna wystąpić retrospektywa oraz demo produktu, które jest często pomijane przez wiele zespołów,

6. **story** - najmniejsza jednostka podlegająca ocenie (wg. wybranej przez zespół skali).

7. **task** - zadania, które mogą być powiązane bezpośrednio ze story lub projektem.

8. **retrospektywa** - wydarzenie, które ma miejsce na koniec sprintu. W tym momencie zespół developerski spisuje wszystkie wady i zalety minionego sprintu oraz wspólnie z scrum masterem starają się rozwiązać zaistniałe problemy.

2.2. Przegląd dostępnych narzędzi

Wytwarzając ten system skupiłem się, aby wpasował się on w wybraną przeze mnie metodykę oraz rozwiązywał szereg mankamentów, które napotkałem podczas korzystania z obecnych już rozwiązań. W celu lepszego zrozumienia problemów postanowiłem opisać kilka z nich.

Pierwszym i niewątpliwie największym problemem tego typu systemów jest to, że są one płatne, przez co nie wszystkich na nie stać. Niektóre są darmowe, lecz tylko dla projektów typu open source, co nie sprawdza się podczas pracy nad wspólnymi projektami w firmie lub np. pracą dyplomową.

Kolejną wadą jest to, że nie są one proste w instalacji. Często występują jako potężne programy, przez co nie można ich bezpłatnie hostować w chmurze gdyż potrzebują dużo płatnych zasobów.

Ostatnim, jednak nie mniej ważnym problemem jest ich czytelność. Oferują one szereg zbędnej zwykłemu użytkownikowi - programiście - funkcjonalności, która jest przydatna

tylko w określonych warunkach. Taki ogrom zakładek i okienek powoduje często zamęt i niezrozumienie wśród programistów.

Celem pracy było wytworzenie systemu do zarządzania projektami dedykowanego metody Scrum. Obecnie na rynku jest wiele zarówno darmowych jak i płatnych narzędzi, które oferują możliwość prowadzenia projektów rożnymi metodykami. Mają one jednak pewne wady:

1. Darmowe narzędzia nie są wspierane, przez co ich rozwój stanął w miejscu wiele lat temu
2. Część z nich jest darmowa tylko dla 10 użytkowników, co powoduje, że nie są przydatne w większych firmach
3. Większość jest darmowa tylko dla open source, co wyklucza możliwość użycia ich w firmie
4. Posiadają przerost formy nad treścią - docelowy użytkownik - programista musi przedzierać się przez gąszcz funkcjonalności oraz poświęcać czas na system, zamiast na swoją pracę

Wytworzony przeze mnie system ma na celu sprostać wadom obecnego oprogramowania oraz spełniać następujące wymagania:

1. Łatwość instalacji - do tego celu zostało wykorzystane oprogramowanie docker, które wspiera szybkie wytwarzanie oprogramowania
2. Przejrzysty interfejs użytkownika - został osiągnięty dzięki open sourcowej bibliotece PrimeFaces
3. Bez zbędnej funkcjonalności - zostały zaimplementowane tylko obligatoryjne funkcje tego typu systemu oraz niewielka ilość udogodnień

2.3. Historyjki użytkownika

Jednym ze sposobów na prowadzenie dokumentacji projektu, jak i zbioru wymagań jest utrzymywanie rejestru historyjek użytkownika. Historyjki użytkownika są częścią zwinnych metody prowadzenia projektu. Jako że wytwarzanemu systemowi towarzyszy metodyka Scrum, nie mogło tutaj zabraknąć tego elementu.

Historyjka jest jednostką funkcjonalności w projektach XP. Pokazujemy postęp prac, dostarczając przetestowany i zintegrowany kod, który składa się na implementację danej historyjki. Historyjka powinna być zrozumiała i wartościowa dla klientów, testowalna przez programistów i na tyle mała, żeby programiści mogli zaimplementować sześć historyjek w takcie jednej iteracji[1].

Historyjki mogą mieć wiele wzorców. W tej pracy są stosowane dwa z nich:

- Jako *<typ użytkownika >* mogę *<nazwa zadania>*.
- Jako *<typ użytkownika >* mogę *<nazwa zadania>* w celu *<cel >*[6]

W projektowanym systemie występują cztery rodzaje użytkowników (administrator, product owner, scrum master oraz developer) z różnymi uprawnieniami. Teraz szczegółowo zostaną opisane uprawnienia każdego z nich w postaci *user story*. Warto pamiętać, że w systemie występuje uproszczony model użytkowników. Dodatkowo każdy z nich jest developerem, a co za tym idzie, może wykonywać akcje użytkownika developer oraz własne.

Administrator

1. Jako administrator mogę tworzyć nowych użytkowników w celu dodania ich do systemu.
2. Jako administrator mogę usuwać użytkowników z systemu.
3. Jako administrator mogę dodawać użytkowników do grup oraz usuwać z nich w celu modyfikacji zespołu.
4. Jako administrator mogę nadawać dowolne uprawnienia wszystkim użytkownikom.
5. Jako administrator mogę utworzyć projekt w celu dodania go do systemu.
6. Jako administrator mogę usunąć projekt w celu usunięcia z systemu oraz powiązanych z nim elementów t.j. sprint, story, backlog oraz inne. Operacja usuwania odbywa się kaskadowo.
7. Jako administrator mogę modyfikować projekt.
8. Jako administrator mogę tworzyć nowe grupy w celu dodania ich do systemu.

[1] K. Beck, M. Fowler, *Planning Extreme Programming*, Addison-Wesley, Boston 2000, s.42

9. Jako administrator mogę dodawać i usuwać grupy z projektów w celu przydzielenia uprawnień.
10. Jako administrator mogę tworzyć, usuwać oraz edytować statusy zadań.
11. Jako administrator mogę tworzyć, usuwać oraz edytować priorytety zadań.

Product owner

1. Jako product owner mogę tworzyć nowe zadania w projekcie, w celu dodania ich do backlogu.
2. Jako product owner mogę nadawać priorytety poszczególnym zadaniom w projekcie.
3. Jako product owner mam wgląd w cały projekt, do którego jestem przypisany.

Scrum master

1. Jako scrum master mogę tworzyć nowe sprinty w projekcie.
2. Jako scrum master mogę tworzyć nowe story w sprintach.
3. Jako scrum master mogę dodawać i usuwać zadania ze story.
4. Jako scrum master mogę tworzyć retrospektywy.
5. Jako scrum master mam wgląd we wszystkie projekty zespołów, w których jestem scrum masterem.

Developer

1. Jako developer mogę tworzyć i edytować zadania.
2. Jako developer mogę dodawać komentarze do zadań oraz retrospektyw.
3. Jako developer mogę usuwać swoje komentarze do zadań oraz retrospektyw.
4. Jako developer mogę edytować swój profil.

Bibliografia

[1] JavaServer Faces i Eclipse Galileo. Tworzenie aplikacji WWW *Andrzej Marciniak* Helion 2010

[2] Core JavaServer Faces. Wydanie II *David Geary, Cay S. Horstmann* Helion 2008

[3] Enterprise JavaBeans 3.0 *Bill Burke & Richard Monson-Haefel* Helion 2007

[4] JBoss AS 7. Tworzenie aplikacji *Francesco Marchioni* Helion 2014

[5] Core J2EE. Wzorce projektowe *Deepak Alur, John Crupi, Dan Malks* Helion 2004

[6] Scrum. O zwinnym zarządzaniu projektami. Wydanie II rozszerzone *Mariusz Chrapko* Helion 2015

[7] Java. Kompedium programisty. Wydanie VIII *Herbert Schildt* Helion 2012

[8] Zwinne wytwarzanie oprogramowania. Najlepsze zasady, wzorce i praktyki *Robert C. Martin* Helion 2015

[9] Git. Rozproszony system kontroli wersji *Włodzimierz Gajda* Helion 2013

[10] Mistrz czystego kodu. Kodeks postępowania profesjonalnych programistów *Robert C. Martin* Helion 2013

[11] Czysty kod. Podręcznik dobrego programisty *Robert C. Martin* Helion 2010

[12] Rusz głową! Wzorce projektowe *Eric Freeman, Elisabeth Freeman, Bert Bates, Kathy Sierra* Helion 2011

[13] Refaktoryzacja do wzorców projektowych *Joshua Kerievsky* Helion 2005