Łódź, dnia 04.08.2023.

dr hab. inż. Aneta Poniszewska-Marańda, prof. uczelni
Instytut Informatyki
Politechnika Łódzka

RECENZJA ROZPRAWY DOKTORSKIEJ DLA RADY NAUKOWEJ DYSCYPLINY „INFORMATYKA TECHNICZNA I TELEKOMUNIKACJA" POLITECHNIKI WROCŁAWSKIEJ

Tytuł rozprawy: Using Agile Experimentation in Industrial Software Development Environment *to study Continuous Test-Driven Development* (Zwinne eksperymenty w przemysłowym środowisku wytwarzania oprogramowania *na przykładzie praktyki programowania sterowanego ciągłymi testami*)
Autor rozprawy: mgr inż. Marcin Kawalerowicz
Promotor rozprawy: dr hab. inż. Lech Madeyski, prof. uczelni

Recenzja została sporządzona na podstawie pisma Przewodniczącego Rady Naukowej Dyscypliny Informatyka Techniczna i Telekomunikacja, Politechniki Wrocławskiej, prof. dr hab. inż. Michała Woźniaka, z dnia 05.07.2023 r.

Ocena układu i zawartości rozprawy doktorskiej

Rozprawa zawiera osiem rozdziałów, załączniki oraz bibliografię. Posiada 143 strony, w tym 17 stron załączników oraz 16 stron bibliografii. W rozprawie umieszczono 190 pozycji bibliograficznych, dobrze dobranych i aktualnych.

Rozdział pierwszy jest wprowadzeniem do pracy. Przedstawia motywację, jaka przyświecała Autorowi podczas realizacji pracy doktorskiej i pisania rozprawy oraz kontekst badań. Prezentuje cel i zakres rozprawy, pytania badawcze, metryki zastosowane dla zdefiniowanych pytań badawczych oraz pokrótce opis struktury pracy.
Praca dotyczy zagadnień związanych z procesem budowy i rozwoju oprogramowania, w aspekcie przeprowadzania eksperymentów w projektach informatycznych w środowisku przemysłowym. Wykonywanie tego typu eksperymentów jest realizowane poprzez zastosowanie proponowanej metody zwinnego eksperymentowania w przemysłowym środowisku wytwarzania oprogramowania, celem przeprowadzenia efektywnej pracy badawczej w trakcie trwania projektu.
Kontekst badań jest osadzony we współczesnej literaturze, dotyczącej poruszanej tematyki. Autor w ogólny sposób odnosi się do wybranych metod i koncepcji wytwarzania oprogramowania, w szczególności zwinnego wytwarzania oprogramowania oraz do wybranych pozycji literatury, jednak w wystarczający sposób jak na Wprowadzenie do pracy, prezentując ich główne założenia, zalety i wady.
Cel pracy został zdefiniowany następująco:
„*(...) opracowanie ram metody zwanej Agile Experimentation dla minimalnie inwazyjnego eksperymentowania w środowisku przemysłowego wytwarzania oprogramowania, która może być zastosowana w komercyjnych projektach wytwarzania oprogramowania (...)*".

Z celem pracy powiązano dwa pytania badawcze, dotyczące zastosowanych w proponowanym podejściu praktyk CTDD (*Continuous Test-Driven Development*, Programowanie Sterowane Ciągłymi Testami) i CBOP (*Continuous Build Outcome Prediction*, Ciągła Predykcja Wyników Budowy Oprogramowania). Dla obu pytań sformułowano odpowiednie metryki: czas programowania oraz zdolność redukcji kończących się porażką procesów budowy oprogramowania.
Niestety w pracy nie zdefiniowano tezy pracy. Można domniemywać, że teza pracy mieści się w sformułowanych pytaniach badawczych.

Rozdział drugi pracy prezentuje koncepcję „Agile Experimentation", składającą się ze zbioru pięciu postulatów określonych w formie manifestu. Opis koncepcji poprzedzony jest analizą stanu wiedzy w obszarze zwinnego eksperymentowania oraz określeniem jego motywacji i celów. Następnie prezentowana jest charakterystyka koncepcji zwinnego eksperymentowania, opisana poprzez jego manifest, składający się z 5 postulatów, dotyczących: (1) stosowania eksperymentów, składających się z małych i pojedynczych przypadków zamiast eksperymentów na dużą skalę, (2) dbania o poprawność metodologiczną eksperymentów, (3) stosowania odpowiedniego projektu, (4) stosowania bezproblemowych narzędzi do gromadzenia danych, aby nie zakłócać rzeczywistego środowiska programistycznego, (5) stosowania danych ilościowych dostarczanych na bieżąco, celem podejmowania świadomych decyzji na wczesnym etapie w oparciu o te dane. Na koniec opisano metodę Rapid Review jako sposób dokonywania przeglądu literatury, wraz z uzyskanymi wynikami przeglądu przeprowadzonego dla tematu zwinnego eksperymentowania, w tym dla zdefiniowanego pytania „RRQ1: Czy istnieją jakieś struktury, które komercyjna firma zajmująca się tworzeniem oprogramowania może wykorzystać, aby umożliwić oszczędne i zwinne eksperymentowanie w swojej codziennej pracy?".

Rozdział trzeci przedstawia opis praktyki zwinnego tworzenia oprogramowania, sterowanej ciągłymi testami, nazwanej *Continuous Test-Driven Development (CTDD)*, której Autor rozprawy jest współautorem. CTDD łączy dwie techniki: programowanie sterowane testami (TDD) i ciągłe testowanie (CT) w celu wyeliminowania czasochłonnych prac ręcznych, które musi wykonać twórca oprogramowania. Na opis praktyki składa się analiza stanu wiedzy na temat TDD w przemyśle i w edukacji w ramach inżynierii oprogramowania, określenie jej motywacji i celów oraz wyniki przeprowadzonego Rapid Review wraz ze zdefiniowanym pytaniem: „RRQ2: Jaki jest stan badań w obszarze TDD wspieranego przez CT?".

Rozdział czwarty pracy prezentuje opis zwinnej praktyki tworzenia oprogramowania, opartej na ciągłej predykcji defektów, nazwanej *Continuous Defect Prediction (CDP)*, której Autor rozprawy również jest współautorem. Podobnie, jak w rozdziale trzecim, na opis praktyki składa się analiza stanu wiedzy na temat SDP (*Software Defect Prediction*) w ramach inżynierii oprogramowania, określenie jej motywacji i celów oraz wyniki przeprowadzonego Rapid Review wraz ze zdefiniowanym pytaniem: „RRQ3: Jaki jest stan badań nad przewidywaniem wyników kompilacji just-in-time na serwerze ciągłej integracji?". W ramach tego rozdziału Autor opisał również pokrótce autorską propozycję lekkiej implementacji CDP, nazwanej *Continuous Build Outcome Prediction (CBOP)*.

Rozdział piąty opisuje narzędzia stworzone w celu praktycznej realizacji zaproponowanych rozwiązań dla rozwiązania rzeczywistego problemu w firmie programistycznej oraz umożliwienia wykonania eksperymentów z ich udziałem. Ponadto, dokonano wstępnej oceny ich akceptacji.

W pierwszej części rozdziału opisano narzędzia NActivitySensor i RSActivitySensor, których celem jest gromadzenie w czasie rzeczywistym danych ilościowych o działaniach twórcy oprogramowania w jego zintegrowanym środowisku programistycznym (IDE). NActivitySensor stanowi rozszerzenie Microsoft Visual Studio. RSActivitySensor gromadzi dane o działaniach programisty wewnątrz rozszerzenia Visual Studio o nazwie Resharper. Obydwa narzędzia wykorzystano do przeprowadzenia eksperymentów CTDD.
W kolejnej części rozdziału opisano narzędzie AutoTest.Net4CTDD, które jest rozszerzeniem narzędzia AutoTest.NET do przeprowadzenia badań nad CTDD. Narzędzie AutoTest.NET4CTDD zostało również wstępnie ocenione w formie ankiety przeprowadzonej wśród programistów firmy CODEFUSION Sp. z o.o. Ankieta została oparta na modelu akceptacji technologii (TAM).
Ostatnim opisanym narzędziem jest narzędzie o nazwie Jaskier. Zostało ono zaprojektowane, aby umożliwić przeprowadzenie badań nad CDP, w szczególności w celu implementacji lekkiej wersji CDP, nazwanej CBOP. Jaskier to zestaw narzędzi, który poszerza zakres wszystkich wcześniej opisanych narzędzi. Jaskier ma dwóch aktorów: (1) twórcę oprogramowania (konsumenta modelu predykcyjnego) oraz (2) Serwer CI – umożliwia tworzenie/trenowanie modelu predykcyjnego (trenowanie/przetrenowanie może być również uruchomione ręcznie przez osobę np. menadżera lub badacza). Posiada dwa główne przypadki użycia: (1) uzyskaj predykcję, (2) trenuj/przetrenuj model predykcyjny. Jaskier został również wstępnie oceniony w formie ankiety, opartej na TAM. Tym razem badanie przeprowadzono na szerszej grupie odbiorców: byli to uczestnicy eksperymentu CBOP oraz doświadczeni użytkownicy korporacyjni.
W ostatniej części rozdziału zaprezentowano opis algorytmu dla modelu CBOP w formie diagramu czynności UML oraz listingów kodu w języku R, wraz z ich opisem.

Rozdział szósty pracy prezentuje ocenę podejścia CTDD w warunkach przemysłowych. Opisano dwa eksperymenty CTDD przeprowadzone w celu odpowiedzi na RQ1: „Czy praktyka CTDD zwiększa efektywność profesjonalnego programisty?". Porównano wydajność tworzenia oprogramowania przy użyciu CTDD z tym samym rozwojem przy użyciu TDD. W celu precyzyjnego określenia sposobu oceny wydajności wprowadzono pomiar czasu od stanu czerwonego do stanu zielonego (RTG) oprogramowania. Postawiono następującą hipotezę badawczą H1: „Zastosowanie CTDD w rzeczywistym projekcie oprogramowania powoduje, że programista jest szybszy w porównaniu z TDD pod względem czasu RTG".
Przedstawiono metodę badawczą oraz wyniki, które przeanalizowano i przedyskutowano. Wyniki pierwszego eksperymentu CTDD były raczej niejednoznaczne. W eksperymencie wziął udział tylko jeden uczestnik (studium pojedynczego przypadku). Spowodowało to konieczność przeprowadzenia drugiego eksperymentu. Badanie miało dokładnie taki sam plan, ale w eksperymencie wzięło udział dwóch uczestników (badanie małego n). Wyniki drugiego eksperymentu potwierdzają wyniki pierwszego eksperymentu. Udowodniono hipotezę, że „wykorzystanie CTDD w prawdziwym projekcie oprogramowania przyspiesza programistę w porównaniu z TDD pod względem czasu RTG". Zaobserwowano spadek czasu RTG przy stosowaniu CTDD w porównaniu z TDD.

Rozdział siódmy pracy został zbudowany według takiego samego schematu, jak rozdział szósty, prezentując ocenę podejścia CBOP również w warunkach przemysłowych, a tym samym odpowiadając na pytanie RQ2: „Czy praktyka CBOP zmniejsza liczbę nieudanych kompilacji na serwerze ciągłej integracji (CI)?". CBOP to lekka wersja CDP, która wykorzystuje klasyfikację do etykietowania możliwych wyników kompilacji (sukcesu lub niepowodzenia) w oparciu o historyczne dane CI i metryki (funkcje) pochodzące z repozytorium oprogramowania.

Podobnie, przedstawiono metodę badawczą oraz wyniki, które przeanalizowano i przedyskutowano. Eksperyment CBOP miał charakter on-line (nielaboratoryjny) i angażował profesjonalistów zajmujących się rzeczywistym i konkretnym problemem. Był to ciągły, napędzany biznesem, prawdziwy projekt inżynierii oprogramowania. Jego celem było zmierzenie wskaźnika nieudanych kompilacji (FBR) przy jednoczesnej kontroli stosowania praktyki CBOP. Zdefiniowano następującą hipotezę badawczą H2: „Stosowanie CBOP zmniejsza FBR".
Do przeprowadzenia eksperymentu wykorzystano zestaw narzędzi Jaskier. Model zbudowano wykorzystując zestaw 8 metryk, m.in. liczba wersji, liczba odrębnych osób zatwierdzających, liczba zmodyfikowanych linii. W eksperymencie brało udział 9 programistów. W wyniku eksperymentu stwierdzono, że „stosowanie praktyki CBOP nie wpływa na współczynnik nieudanych kompilacji (FBR)". Współczynnik nieudanych kompilacji nie zmniejszył się, gdy zastosowano CBOP. Ponadto, podczas korzystania z CBOP programiści wygenerowali nieco więcej nieudanych kompilacji niż bez CBOP.

Ósmy rozdział rozprawy zawiera podsumowanie pracy, opis osiągnięć pracy, odpowiedzi na pytania badawcze oraz perspektywy rozwoju tematu pracy w niedalekiej przyszłości.

Następnie umieszczono załączniki do głównej części pracy, które prezentują odpowiednio: wykaz prac Autora dotyczących bezpośrednio prezentowanej rozprawy i wykaz innych prac Autora oraz dane eksperymentalne zebrane podczas przeprowadzenia eksperymentów opisanych w rozprawie (dane eksperymentalne CTDD i dane eksperymentalne CBOP).

Zastosowana bibliografia znalazła się na końcu pracy, ale nie jest ujęta w spisie treści. Bibliografia zwiera 190 pozycji bibliograficznych.

Ocena celu pracy

Autor sformułował cel i zakres pracy w rozdziale 1. rozprawy. W rozdziale tym zarysowano problemy, związane z procesem wytwarzania oprogramowania w aspekcie przeprowadzania eksperymentów w projektach informatycznych w środowisku przemysłowym.

Z celem pracy powiązano dwa pytania badawcze: RQ1 „Czy praktyka CTDD zmniejsza czas tworzenia kodu produkcyjnego?" oraz RQ2 „Czy praktyka CBOP zmniejsza liczbę nieudanych procesów budowy oprogramowania na serwerze Ciągłej Integracji (CI)?". Dla tych pytań zdefiniowano odpowiednio dwie metryki: (1) czas programowania RTG i (2) nieudany współczynnik kompilacji (FBR, *Failed Build Ratio*), który stanowi stosunek „liczby procesów budowy zakończonych porażką do wszystkich wyników budowy oprogramowania".

Niemniej jednak teza pracy nie została jasno i precyzyjnie sformułowana i zaprezentowana. Jedynie można domniemywać, że teza pracy jest związana bezpośrednio ze zdefiniowanymi pytaniami badawczymi.

Ocena zastosowanych metod badawczych

Autor w swojej pracy stosuje następujące metody badawcze: analiza dostępnej literatury w temacie rozprawy, analiza przedmiotu badań, czyli zwinnych eksperymentów w środowisku przemysłowym dla sterowanego ciągłymi testami procesu wytwarzania oprogramowania, obserwacja prowadzonych projektów informatycznych, prace eksperymentalne, metody statystyczne i ilościowe w analizach danych uzyskanych z przeprowadzonych eksperymentów.

Uważam, że wszystkie zastosowane metody badawcze są poprawnie wybrane i zastosowane. Stanowią kompletny warsztat badawczy, niezbędny do przeprowadzenia badań w ramach tematu rozprawy.

Autor opiera się również na własnym praktycznym doświadczeniu, wyniesionym z pracy nad projektami komercyjnymi, realizowanymi między innymi w firmie informatycznej CODEFUSION, Sp. z o.o., którą zarządza. Takie doświadczenie stanowi niewątpliwie wartość dodaną w pracy nad poruszanym tematem oraz nad przygotowywaniem rozprawy, w szczególności w obszarze inżynierii oprogramowania, która w bardzo wielu aspektach bazuje na podejściu empirycznym.

Ocena zastosowanego piśmiennictwa

Rozprawa zawiera 190 pozycji bibliograficznych, w tym 4 pozycje internetowe, które głównie stanowią manifesty wybranych podejść w inżynierii oprogramowania. Zdecydowana większość pozycji to artykuły naukowe, opublikowane w czasopismach naukowych lub w materiałach konferencyjnych konferencji naukowych w ostatnich kilku-kilkunastu latach, a także raporty techniczne. Wśród pozycji są również prace opublikowane przez Autora – w sumie 10 prac, których jest współautorem.

Autor przedstawił analizę źródeł literaturowych częściowo w rozdziale 2 rozprawy – na temat zwinnego eksperymentowania, częściowo w rozdziale 3 – na temat TDD i CTDD oraz częściowo w rozdziale 4 – na temat CDP i CBOP. Wszystkie te trzy przeglądy literatury zostały zaprezentowane na zasadzie Rapid Review, które jest wskazywane jako lekki przegląd literatury, mający na celu dostarczenie dowodów dotyczących określonego problemu w odpowiednim czasie. Umożliwiło to sprawdzenie stanu badań w ramach tematu niniejszej rozprawy.

Przeprowadzona analiza obecnego stanu wiedzy w poruszanych obszarach inżynierii oprogramowania oraz istniejących rozwiązań wskazuje, że Autor poprawnie rozumie problemy, związane z tematem pracy. Posiada wiedzę na temat rozwiązań krajowych i światowych w poruszanej tematyce. Co prawda część istotnych zagadnień oraz problemów została jedynie zasygnalizowana, jednakże taka forma pozwala mieć nadzieję, że Autor posiada również wiedzę szczegółową na ich temat.

Ocena części rozprawy dotyczącej omówienia wyników badań

Tematyka badań Autora obejmuje proces przeprowadzania zwinnych eksperymentów w projektach informatycznych w środowisku przemysłowym. Badania przeprowadzone w ramach realizacji pracy obejmują dwa powiązane ze sobą obszary: (1) praktyka programowania sterowana ciągłymi testami i (2) praktyka ciągłej predykcji defektów, a w szczególności praktyka ciągłej predykcji wyników budowy oprogramowania.

Wyniki badań zostały zaprezentowane przez Autora częściowo w rozdziale 3, 4 i 5 oraz w rozdziale 6 i 7 rozprawy, odpowiednio na temat praktyki CTDD, praktyki CDP i CBOP, narzędzi zastosowanych do przeprowadzenia eksperymentów dla analizowanych podejść oraz analizy zastosowania podejścia CTDD i CBOP w środowisku przemysłowym.

Ponadto, dyskusja na temat uzyskanych wyników eksperymentów została zaprezentowana pod koniec rozdziału 6 i 7 oraz w podsumowaniu samej rozprawy.

Sposób prezentacji podejmowanych w pracy zagadnień oraz uzyskanych wyników badań jest na ogół jasny i zrozumiały. Autor stosuje dużo różnorodnych symboli na oznaczenie poszczególnych elementów rozpatrywanych praktyk, podejść, projektów i eksperymentów. Wprowadzone symbole są ogólne, czasami nieintuicyjne. Ponadto, część zagadnień, a także wyników pracy Autora poruszona jest w rozprawie bardzo pobieżnie.

Niemniej jednak tekst rozprawy został przygotowany starannie. Strona redakcyjna nie budzi większych zastrzeżeń – moje uwagi zostały przedstawione w kolejnym punkcie recenzji.

Nieprawidłowości, które pojawiły się w rozprawie

Analiza rozprawy nasunęła mi kilka uwag krytycznych:

- Brak jasno zdefiniowanej tezy pracy.
- Rozdziały 3 i 4 są dość krótkie. Z powodzeniem mogłyby być połączone w jeden rozdział.
- Praktyka CTDD została opisana zbyt pobieżnie w rozdziale 3.
- Autorska praktyka CBOP została opisana zbyt pobieżnie w rozdziale 4.
- Zbyt ogólne i przez to niezrozumiałe nazwy niektórych rozdziałów i podrozdziałów oraz sekcji pracy.
- Forma niektórych zdań nasuwa wątpliwości stylistyczne.
- Praca powinna być napisana w formie bezosobowej, chociaż w przeważającej części.
- Numeracja rysunków, tabel jest niezgodna z ogólnie przyjętymi zasadami dla tzw. długich tekstów, jakim jest rozprawa doktorska.
- Tabele w załączniku B, prezentujące wyniki uzyskane podczas przeprowadzenia eksperymentów powinny być lepiej opisane.
- Rysunek 13 jest ogólnie bardzo dobrze znany. Z powodzeniem można się obejść bez niego.
- Brak spisu listingów. Brak indeksu stosowanych w pracy symboli.
- Błędy językowe.

Przedstawione uwagi nie obniżają jednakże mojej pozytywnej oceny pracy.

Ocena, czy rozprawa stanowi oryginalne rozwiązanie problemu naukowego

Uważam, że cel i zakres rozprawy oraz obszar naukowy rozprawy zostały określone jasno i precyzyjnie. Praca ma charakter teoretyczno-implementacyjny. Część teoretyczna obejmuje:

- przeprowadzenie analizy dostępnych źródeł literaturowych na temat poruszanych w rozprawie zagadnień, dotyczących procesu przeprowadzania zwinnych eksperymentów w projektach informatycznych w środowisku przemysłowym (w tym tzw. zwinnego eksperymentowania),
- opracowanie podejścia ciągłego rozwoju oprogramowania opartego na testach CTDD,
- opracowanie podejścia ciągłego przewidywania defektów CDP, a w szczególności autorskiej praktyki ciągłej predykcji wyników budowy oprogramowania CBOP,
- zdefiniowanie i zaprojektowanie środowiska badawczego, które zostało użytego do przeprowadzenia szeregu eksperymentów z zastosowaniem zaproponowanych praktyk rozwoju oprogramowania.

Podejście CTDD to połączenie TDD z CT, gdzie TDD to zwinna praktyka tworzenia oprogramowania sterowana testami, a CT to technika, w której testy są uruchamiane automatycznie w tle IDE programisty. Ideą CTDD jest poszukiwanie efektów synergii pomiędzy CT i TDD poprzez wyeliminowanie konieczności ręcznego uruchamiania testów podczas TDD. Proponowana praktyka CBOP jest lekką implementacją koncepcji CDP. Obie koncepcje zostały zaproponowane przez prof. Madeyskiego i Autora rozprawy. Podstawą CDP jest SDP w połączeniu z ML oraz koncepcja ciągłości informacji zwrotnej do programisty. Jest to realizowane w celu łagodzenia niepożądanych skutków nieprawidłowej kompilacji na serwerze CI.

Część implementacyjna obejmuje stworzenie zestawu narzędzi (NActivitySensor, RSAactivitySensor, AutoTest.Net4CTDD, Jaskier), które zostały użyte do przeprowadzenia eksperymentów na zaproponowanych rozwiązaniach teoretycznych w środowisku przemysłowym, przeprowadzenie eksperymentów oraz przeanalizowanie uzyskanych wyników eksperymentów.

Tematyka rozprawy sytuuje ją w obszarze badawczo-eksperymentalnym, związanym z poszukiwaniem efektywnych metod i praktyk, dotyczących budowy oprogramowania, w szczególności budowy i rozwoju oprogramowania w środowisku przemysłowym.

Uważam, że podjęcie tematu pracy doktorskiej w rozpatrywanym obszarze i zakresie jest celowe i w pełni uzasadnione potrzebą zapewniania i zwiększania jakości współcześnie tworzonego oprogramowania, a w szczególności złożonych i skomplikowanych w swojej strukturze i działaniu systemów informatycznych. Jest to w pełni uzasadnione ze względów teoretycznych, poznawczych i praktycznych na tle obecnego stanu wiedzy w rozpatrywanym obszarze.

Praktyczne zastosowanie uzyskanych wyników badań

Autor w swoich badaniach połączył aspekty teoretyczne i praktyczne. W ramach realizacji pracy doktorskiej Autor przeprowadził eksperymenty, dotyczące dwóch zaproponowanych podejść: CTDD (Ciągły rozwój oparty na testach) i CDP (Ciągłe przewidywanie defektów). Podczas pierwszego eksperymentu zmierzono wydajność pojedynczego programisty przy użyciu CTDD. Wydajność mierzono i oceniano za pomocą wprowadzonej zmiennej RTG. CDP to rozszerzenie Software Defect Prediction, które wykorzystuje uczenie maszynowe (ML) do przewidywania wszelkich możliwych defektów i zapewnia natychmiastową informację zwrotną dla programisty. W pracy Autora podejście CDP zostało udoskonalone i zawężone do przewidywania wyników kompilacji ciągłej integracji (CI) – stworzono lekką implementację CDP, nazwaną ciągłym przewidywaniem wyników kompilacji (CBOP).

Metoda Agile Experimentation okazała się skuteczna w obu eksperymentach. Eksperymenty przeprowadzono w rzeczywistych projektach rozwoju oprogramowania zorientowanych na biznes. Do przeprowadzenia eksperymentów przygotowano odpowiedni zestaw narzędzi typu open source, które są dostępne w GitHub. Wstępną ocenę akceptacji narzędzi opracowanych do wspierania CTDD i CBOP przeprowadzono za pomocą ankiety opartej na TAM.
Wyniki eksperymentu CTDD wydają się potwierdzać skuteczność zaproponowanej praktyki w kontekście tworzenia oprogramowania przemysłowego. W przypadku eksperymentu CBOP wyniki wskazują na negatywny wpływ praktyki na pożądany rezultat. Informacje zwrotne od

twórców oprogramowania zebrano na podstawie wstępnej oceny akceptacji, aby ocenić wpływ biznesowy pomysłów i narzędzi.

Wyniki rozprawy mają zatem znaczenie dla dalszego rozwoju nauki w dziedzinie inżynierii oprogramowania oraz posiadają wartość aplikacyjną i praktyczną dla rzeczywiście realizowanych projektów informatycznych.

Wniosek końcowy

W moim przekonaniu Autor w wystarczającym stopniu przeanalizował aktualny stan wiedzy w rozpatrywanym temacie, jasno sformułował problemy, a następnie w zadawalającym stopniu rozwiązał je w swojej pracy.

Oryginalnym rezultatem rozprawy, a tym samym samodzielnym i oryginalnym dorobkiem Autora według mojej oceny jest opracowanie metody CBOP, wspomagającej budowę systemów informatycznych, która oparta jest na zwinnym eksperymentowaniu, ciągłym rozwoju opartym na testach i ciągłym przewidywaniu defektów oraz przeprowadzenie szeregu eksperymentów, celem oceny podejścia CTDD i CBOP w środowisku przemysłowym. Dorobek Autora powiększa zestaw narzędzi opisanych w rozdziale 5, które zostały użyte do przetestowania zaproponowanych rozwiązań w rzeczywiście realizowanych projektach informatycznych.

Uważam, że przedstawiona mi do recenzji rozprawa prezentuje ogólną wiedzę teoretyczną kandydata w dyscyplinie informatyka techniczna i telekomunikacja oraz umiejętność samodzielnego prowadzenia pracy naukowej.

W mojej ocenie, Pan mgr Marcin Kawalerowicz w swojej rozprawie trafnie zdefiniował problem badawczy, a następnie rozwiązał go w odpowiednim zakresie. Pozytywnie oceniam wszystkie wymienione powyżej elementy pracy.

Uważam, że rozprawa spełnia wymagania stawiane rozprawom doktorskim przez obowiązujące przepisy o stopniach i tytułach naukowych. W związku z tym wnoszę o dopuszczenie jej do publicznej obrony.