INFORMACJA O OSIĄGNIĘCIACH BĘDĄCYCH PODSTAWĄ NADANIA STOPNIA DOKTORA HABILITOWANEGO

METODY WSPIERAJĄCE PROCES SZACOWANIA PRACOCHŁONNOŚCI W PROJEKTACH INFORMATYCZNYCH

MIROSŁAW OCHODEK

Instytut Informatyki

Wydział Informatyki i Telekomunikacji

Politechnika Poznańska

Spis treści

I. OSIĄGNIĘCIE NAUKOWE BĘDĄCE PODSTAWĄ NADANIA STOPNIA DOKTORA HABILITOWANEGO

Osiągnięcie naukowe obejmuje cykl powiązanych tematycznie artykułów naukowych[1,2] (zgodnie z art. 219 ust. 1. pkt 2b Ustawy):

Metody wspierające proces szacowania pracochłonności w projektach informatycznych

A1. **M. Ochodek**, S. Kopczyńska, M. Staron, *Deep learning model for end-to-end approximation of COSMIC functional size based on use-case names*, Information and Software Technology, vol. 123, 106310, DOI: 10.1016/j.infsof.2020.106310, 2020, ISSN: 0950-5849
MNiSW/MEiN (2019+) – **140 pkt.**, IF=2,726

A2. **M. Ochodek**, M. Staron, W. Meding, J. Bosch, *LegacyPro: A DNA-inspired method for identifying process legacies in software development organizations*, IEEE Software, vol. 37, no. 6, pp. 76-85, DOI: 10.1109/MS.2020.2971894, 2020, ISSN: 0740-7459
MNiSW/MEiN (2019+) – **100 pkt.**, IF=2,589

A3. **M. Ochodek**, R. Hebig, W. Meding, G. Frost, M. Staron, *Recognizing lines of code violating company-specific coding guidelines using machine learning*, Empirical Software Engineering, vol. 25, pp. 220-265, DOI: 10.1007/s10664-019-09769-8, 2020, ISSN: 1382-3256
MNiSW/MEiN (2019+) – **140 pkt.**, IF=3,156

A4. P. Pickerill, H. J. Jungen, **M. Ochodek**, M. Maćkowiak, M. Staron, *PHANTOM: Curating GitHub for engineered software projects using time-series clustering*, Empirical Software Engineering, vol. 25, pp. 2897-2929, DOI: 10.1007/s10664-020-09825-8, 2020, ISSN: 1382-3256
MNiSW/MEiN (2019+) – **140 pkt.**, IF=3,156

A5. **M. Ochodek**, M. Staron, W. Meding, *Simsax: A measure of project similarity based on symbolic approximation method and software defect inflow*, Information and Software Technology, vol. 115, pp. 131-147, DOI: 10.1016/j.infsof.2019.06.003, 2019, ISSN: 0950-5849
MNiSW/MEiN: (2019+) – **140 pkt.**, IF=2,726

A6. S. Kopczyńska, J. Nawrocki, **M. Ochodek**, *An empirical study on catalog of non-functional requirement templates: Usefulness and maintenance issues*, Information and Software Technology, vol. 103, pp. 75-91, DOI: 10.1016/j.infsof.2018.06.009, 2018,

[1] Dla poszczególnych artykułów wskazano do dwóch wartości punktowych z list czasopism publikowanych przez Ministerstwo Nauki i Szkolnictwa Wyższego (MNiSW) / Ministerstwo Edukacji i Nauki (MEiN) – pierwsza zgodna z listą obowiązująca na dzień publikacji danego artykułu oraz druga zgodna z aktualnie obowiązującą listą: MNiSW(2010) - 25.06.2010, MNiSW (2013-2016) – 26.01.2017, MNiSW/MEiN (2019+) – 17-18.12.2019, 29.09.2020 oraz 9.02.2021.

[2] Identyfikatory artykułów (np. A1) będą używane w dalszej części autoreferatu jako odnośniki do poszczególnych publikacji wykazanych w ramach osiągnięcia naukowego.

ISSN: 0950-5849
MNiSW (2013-2016) – **35 pkt.**, MNiSW/MEiN (2019+) – **140 pkt.,** IF=2,921

A7. **M. Ochodek**, S. Kopczyńska, *Perceived importance of agile requirements engineering practices - A survey*, Journal of Systems and Software, vol. 143, pp. 29-43, DOI: 10.1016/j.jss.2018.05.012, 2018, ISSN: 0164-1212
MNiSW (2013-2016) – **35 pkt.**, MNiSW/MEiN (2019+) – **100 pkt.,** IF=2,559

A8. **M. Ochodek**, *Functional size approximation based on use-case names*, Information and Software Technology, vol. 80, pp. 73-88, DOI: 10.1016/j.infsof.2016.08.007, 2016, ISSN: 0950-5849
MNiSW (2013-2016) – **35 pkt.**, MNiSW/MEiN (2019+) – **140 pkt**., IF=2,694

A9. J. Jurkiewicz, J. Nawrocki, **M. Ochodek**, T. Głowacki, *HAZOP-based identification of events in use cases. An empirical study*, Empirical Software Engineering, vol. 20, pp. 82-109, DOI: 10.1007/s10664-013-9277-5, 2015, ISSN: 1382-3256
MNiSW (2013-2016) – **45 pkt.**, MNiSW/MEiN (2019+) – **140 pkt.,** IF=1,393

Wprowadzenie

Szacowanie pracochłonności jest jedną z kluczowych czynności związanych z planowaniem przedsięwzięć informatycznych. Przyjęcie nierealistycznych założeń co do kosztu realizacji projektu stanowi istotne ryzyko dla jego sukcesu. Niestety zarówno niedoszacowanie jak i przeszacowanie pracochłonności może okazać się szkodliwe dla przedsięwzięcia. Niedoszacowanie może doprowadzić do sytuacji, w której poczynione zobowiązania nie mogą zostać spełnione z uwagi na niewystarczające środki finansowe czy zbyt krótki czas na realizację zadań projektowych. Z drugiej strony przeszacowanie może doprowadzić do odrzucenia propozycji projektu przez potencjalnego sponsora z uwagi na pozornie zbyt wysokie koszty, a zatem także do utraty potencjalnych benefitów, które mogłyby wyniknąć z podjęcia się jego realizacji.

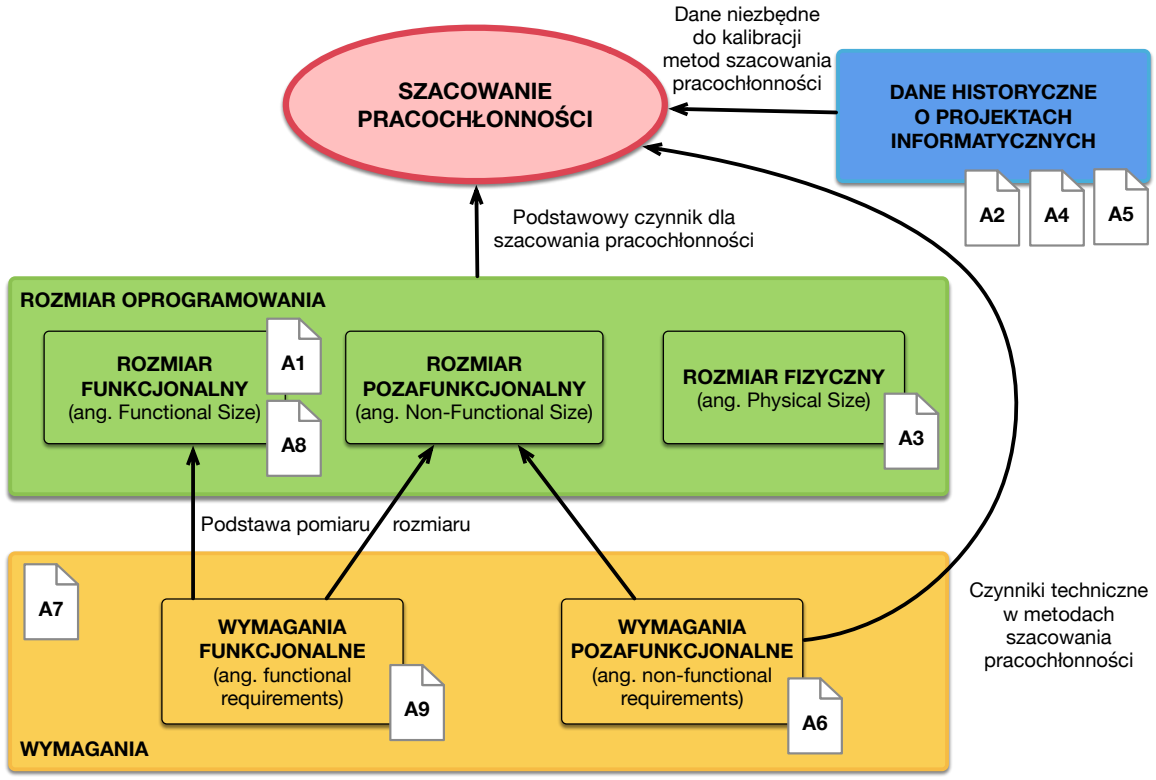
W literaturze znaleźć można przynajmniej kilka rodzin metod szacowania pracochłonności (np. metody algorytmiczne, eksperckie, szacowania przez analogię). Wszystkie one w sposób mniej lub bardziej bezpośredni wykorzystują wiedzę o wcześniej realizowanych projektach odnosząc je do szacowanego przedsięwzięcia. W związku z tym skuteczność większości metod szacowania pracochłonności jest uzależniona od dostępu do wysokiej jakości danych historycznych z podobnych przedsięwzięć do szacowanego. Innym istotnym czynnikiem mającym wpływ na dokładność tych metod to poziom niepewności (niewiedzy) co do projektu, który jest szacowany (np. niepewności co do jego zakresu, szczegółowych wymagań dla rozwijanego produktu, środowiska realizacji projektu). Oczywiście im większy jest poziom niepewności co do szacowanego projektu, tym większego poziomu błędu szacowania pracochłonności należy się spodziewać.

W cyklu powiązanych tematycznie artykułów przedstawionych w niniejszym wniosku, motywem przewodnim są metody wspierające proces szacowania pracochłonności w projektach informatycznych. Poprzez zaproponowane metody starałem się przeciwdziałać

dwóm omówionym wyżej problemom tj. szacowaniem pracochłonności przy wysokim poziomie niepewności w projekcie oraz pozyskiwaniu zbiorów danych o projektach, które mogą posłużyć do trenowania/kalibracji metod szacowania pracochłonności.

Powiązania pomiędzy tematyką poszczególnych artykułów a problemem szacowania pracochłonności przedstawiłem na rysunku 1. Proponowane metody wspierają proces szacowania pracochłonności na trzech płaszczyznach:

1) wspieranie procesu pozyskiwania wymagań funkcjonalnych i pozafunkcjonalnych w celu obniżenia poziomu niepewności co do zakresu projektu [A6, A7, A9],
2) wspieranie pomiaru rozmiaru oprogramowania [A1, A3, A8] oraz
3) wspieranie procesu pozyskiwania i „czyszczenia" zbiorów danych historycznych o projektach informatycznych [A2, A4, A5].



Rysunek 1 Powiązania pomiędzy artykułami w cyklu oraz obszarami związanymi z problemem szacowania pracochłonności projektów informatycznych.

Wspieranie procesu pozyskiwania wymagań

Informacje zawarte w wymaganiach dla produktu informatycznego, który ma zostać opracowany w ramach projektu stanowią podstawowe informacje wejściowe dla wielu metod szacowania pracochłonności. Wymagania produktowe dzielimy na wymagania funkcjonalne i pozafunkcjonalne (zwane także jakościowymi). Te pierwsze opisują tzw. transakcje istotne z perspektywy użytkownika (tj. funkcje, usługi, które wspierają użytkownika w osiąganiu jego celów), natomiast te drugie określają warunki przy jakich udostępniana funkcjonalność jest wartościowa i użyteczna dla użytkownika (np. maksymalny czas odpowiedzi systemu). W przypadku metod szacowania pracochłonności, wymagania funkcjonalne mogą stanowić

podstawę pomiaru rozmiaru funkcjonalnego aplikacji, natomiast wymagania poza-funkcjonalne są używane do określania tzw. czynników wpływu, najczęściej związanych z technicznymi aspektami realizacji przedsięwzięcia (ang. cost drivers / technical drivers).

W tradycyjnym podejściu do organizacji procesu inżynierii wymagań (ang. Requirements Engineering, RE), wymagania dla systemów informatycznych są bardzo często identyfikowane poprzez użycie tzw. podejścia od ogółu do szczegółu (ang. top-down). Definiowanie wymagań zaczyna się od zdefiniowania wysokopoziomowych, ogólnych wymagań (np. używając do tego diagramu przypadków użycia UML), a następnie dekomponuje się te wymagania do bardziej szczegółowych. W niektórych projektach proces ten jest w całości realizowany przed rozpoczęciem prac implementacyjnych. Natomiast wraz z popularyzacją tzw. zwinnych metodyk zarządzania projektami informatycznymi, pojawiło się także nowe określenie związane z procesem inżynierii wymagań – **zwinna inżynieria wymagań** (ang. Agile RE). Niestety termin ten jest **mało precyzyjny i nie posiada jednej konkretnej definicji**. Najczęściej odnosi się on do różnych zbiorów praktyk projektowych rekomendowanych przez zwinne metodyki zarządzania projektami. Z perspektywy rozwoju nowych metod szacowania pracochłonności oraz ich dostosowywania tak, aby mogły być one efektywnie stosowane w nowoczesnych przedsięwzięciach informatycznych, istotnym wydaje się zidentyfikowanie kluczowych praktyk projektowych stosowanych w zwinnych projektach w procesie inżynierii wymagań, a także określenie na ile proces ten różni się w stosunku od tych wcześniej stosowanych.

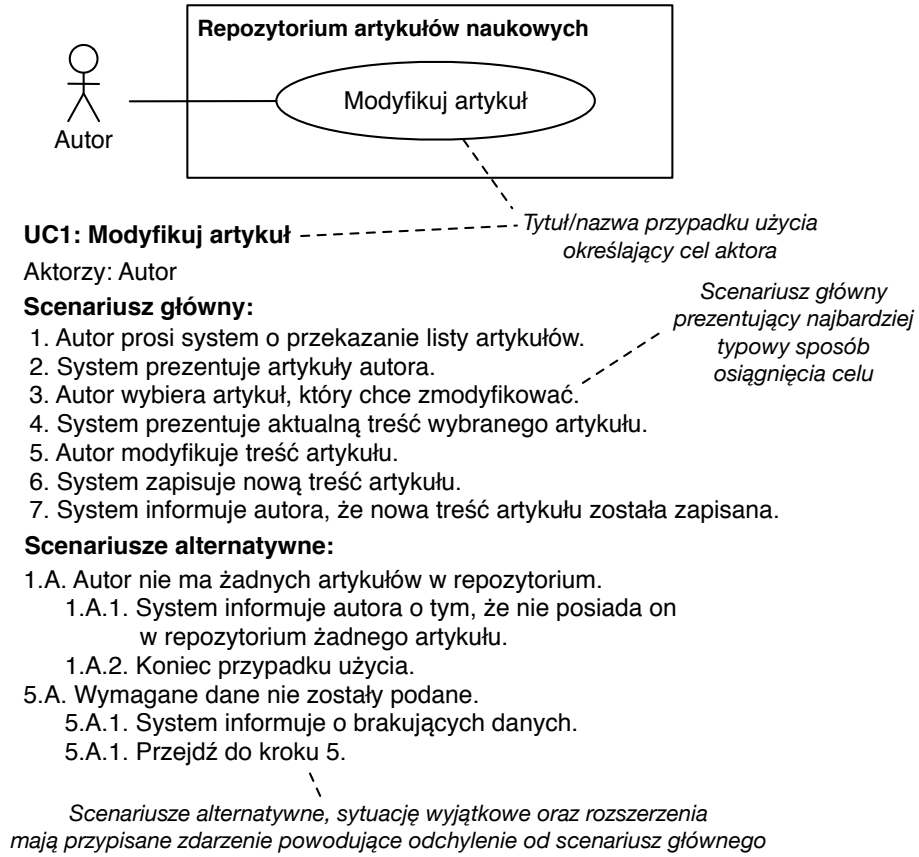
W pracy [A7] przeprowadziliśmy studium literaturowe, które pozwoliło nam zidentyfikować **31 praktyk projektowych zwinnej inżynierii wymagań**. Następnie dokonaliśmy oceny ważności tych praktyk, przeprowadzając badanie ankietowe z udziałem **138 respondentów zawodowo zajmujących się wytwarzaniem oprogramowania**. Na podstawie wyników badania opracowaliśmy ranking praktyk zwinnej inżynierii wymagań z perspektywy ich ważności dla sukcesu przedsięwzięcia, używając do tego opracowanego przez nas algorytmu rankingowego bazującego na metodzie PROMETHEE.[3] Analizując uzyskane wyniki doszliśmy do wniosku, że największy nacisk zwinnej inżynierii wymagań kładziony jest na opracowanie **jasnej wizji produktu oraz na współpracę z interesariuszami w ramach krótkich iteracji, tak aby często otrzymywane sprzężenie zwrotne odnośnie wymagań mogło napędzać i sterować rozwojem produktu**. Takie ustawienie priorytetów zmienia akcenty w planowaniu projektów zwiększając znaczenie planowania w krótkim horyzoncie czasowym (oczywiście nadal pozostawiając konieczność planowania długofalowego). W konsekwencji w projektach zwinnych pojawia się potrzeba częstszego szacowania pracochłonności jako podstawy planowania niż jest to konieczne w projektach realizowanych w sposób „kaskadowy" lub w dłuższych iteracjach. Dodatkowo można także zaobserwować różnice związane z występowaniem wysokiego poziomu niepewności co do wymagań. W projektach zwinnych

[3] Behzadian, M., Kazemzadeh, R., Albadvi, A., Aghdasi, M., PROMETHEE: a comprehensive literature review on methodologies and applications. Eur. J. Oper. Res. 200 (1), DOI: 10.1016/j.ejor.2009.01.021, 2010.

zakres wymagań dla każdej iteracji określany jest na bazie obserwacji poczynionych podczas poprzedzających ją iteracji. W związku z tym w projektach tego typu praca nad pozyskiwaniem i dokumentowaniem wymagań odbywa się w sposób ciągły. W związku z tym potrzebne są metody wspierające proces specyfikowania wymagań m.in pozwalające na podnoszenie kompletności wymagań i przez to redukcję poziomu niepewności związanego z zakresem wymagań na poziomie poszczególnych iteracji projektu.

W moich pracach badawczych związanych ze wsparciem procesu pozyskiwania wymagań skupiłem się nad rozwojem metod mających na celu podnoszenie poziomu kompletności wymagań, a przez to wspieranie procesu szacowania pracochłonności. W szczególności moje zainteresowania badawcze zogniskowane były na opracowywaniu metod wspierających pozyskiwanie i dokumentowanie wymagań funkcjonalnych, wyrażonych w postaci przypadków użycia (ang. use cases) oraz wymagań pozafunkcjonalnych.

Przypadki użycia są uznaną metodą opisu wymagań funkcjonalnych dla systemów informatycznych bazującą na scenariuszach (przykład dokumentacji przypadku użycia przedstawiłem na rysunku 2). Dokumentacja przypadku użycia na poziomie użytkownika (ang. user level) składa się ze zbioru scenariuszy opisujących w jaki sposób użytkownik (aktor) będzie mógł osiągnąć istotny dla niego cel poprzez interakcję z tworzonym systemem informatycznym. Scenariusz, który opisuje najbardziej typowy sposób osiągnięcia takiego celu nazywamy scenariuszem głównym przypadku użycia. Każdy z pozostałych scenariuszy definiowany jest poprzez określenie zdarzenia, które może wystąpić w trakcie realizacji jednego z pozostałych scenariuszy (np. scenariusza głównego) oraz sekwencji kroków, które powinny zostać wykonane w przypadku jego wystąpienia. Specyfikację przypadku użycia rozpoczynamy zazwyczaj od zdefiniowana scenariusza głównego, ponieważ z definicji jest to najbardziej typowy sposób osiągnięcia celu użytkownika. Niestety, jeśli **pominiemy przy tej okazji istotne zdarzenia prowadzące do realizacji scenariuszy alternatywnych to rzeczywista złożoność (rozmiar) takiego przypadku użycia z perspektywy implementacyjnej może zostać znacząco zaniżona a to w konsekwencji może przełożyć się na niedoszacowanie pracochłonności związanej z jego implementacją**.

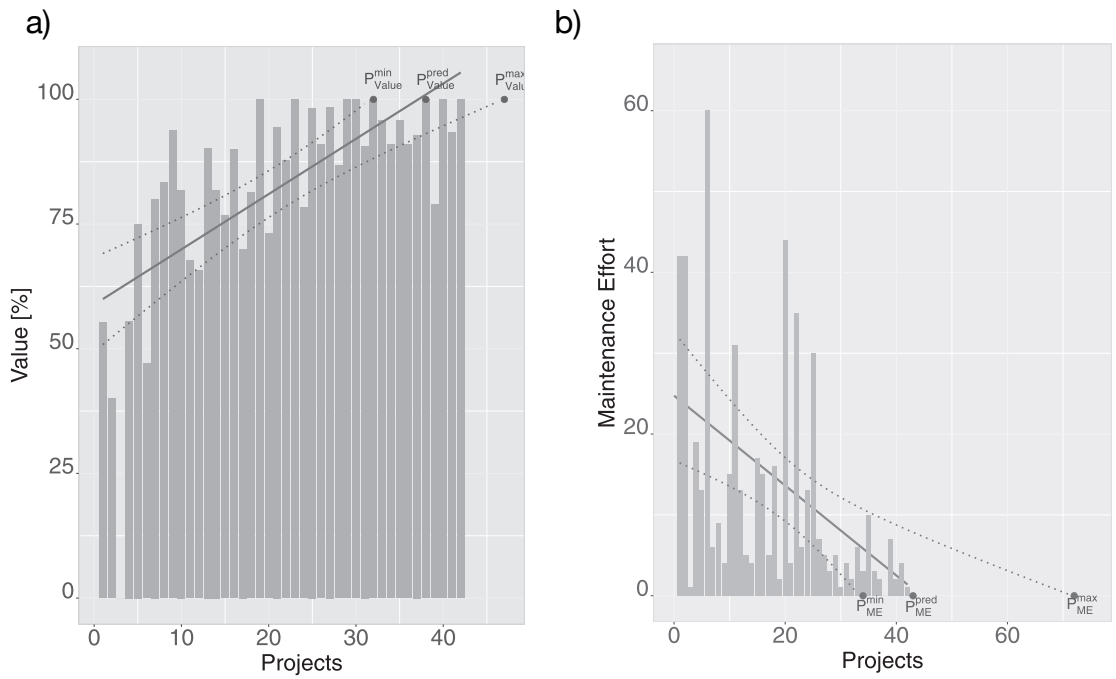


Rysunek 2 Przykład dokumentacji przypadku użycia oraz jego reprezentacja na diagramie przypadków użycia UML.

W pracy [A9] zaproponowaliśmy metodę **HAZOP for use cases (H4U)**, mającą na celu wspomaganie procesu **identyfikacji zdarzeń** w scenariuszach przypadków użycia. Efektywność zaproponowanej metody została oceniona w ramach dwóch eksperymentów (ang. controlled experiment). Każdorazowo efektywność użycia metody H4U we wspomaganiu identyfikacji zdarzeń była porównywana z efektywnością podejścia ad hoc (tzn. dowolnego podejścia zastosowanego przez uczestnika eksperymentu). Pierwszy z eksperymentów został przeprowadzony z udziałem **18 studentów**. Celem badania było sprawdzenie czy i w jakim stopniu metoda H4U może wspomagać pracę osób nieposiadających lub posiadających znikome doświadczenie w pozyskiwaniu i dokumentowaniu wymagań. W kolejnym eksperymencie wzięło udział **82 osób, które zawodowo zajmują się wytwarzaniem oprogramowania**. Celem tego eksperymentu było zbadanie czy użycie metody H4U może wpłynąć na liczbę zdarzeń w scenariuszach przypadków użycia zidentyfikowanych przez profesjonalistów. Wyniki pierwszego eksperymentu pokazały, że podejście do identyfikacji zdarzeń bazujące na metodzie HAZOP pozwala zidentyfikować większą liczbę zdarzeń niż podejście ad hoc (studenci używający metody **H4U** zidentyfikowali średnio o **około 50% więcej zdarzeń** niż używający podejścia ad hoc), natomiast odbyło się to kosztem wydłużenia czasu przeglądu scenariuszy w poszukiwaniu zdarzeń. Wyniki drugiego eksperymentu potwierdziły obserwacje poczynione w ramach pierwszego badania. **Profesjonaliści** używający metody H4U zidentyfikowali **średnio o 27% więcej zdarzeń** niż osoby identyfikujące zdarzenia w dowolny

sposób. Dodatkowo oba eksperymenty pokazały, że osiągnięcie wysokiego poziomu kompletności zdarzeń w przypadkach użycia jest w ogólności zadaniem trudnym. Średnia kompletność zdarzeniowa (definiowana jako stosunek liczby zidentyfikowanych zdarzeń do liczby zdarzeń w tzw. złotym rozwiązaniu wzorcowym) mieściła się w przedziale od 0,15 do 0,26. Wskazuje to na wysokie ryzyko niekompletności przypadków użycia, które niestety może się przełożyć na niedoszacowanie pracochłonności. Jak pokazały wyniki przeprowadzonych przez nas badań użycie metody HAZOP może do pewnego stopnia ograniczyć to ryzyko zwiększając kompletność specyfikacji wymagań bazujących na przypadkach użycia.

Wymagania pozafunkcjonalne (ang. non-functional requirements, NFR) są w przypadku metod szacowania pracochłonności używane do określania tzw. czynników technicznych. Warto zaznaczyć, że mają one zazwyczaj **bardzo istotny wpływ na pracochłonność wytworzenia oprogramowania**. Niestety **wymagania pozafunkcjonalne są także trudne do pozyskania oraz do poprawnego wyspecyfikowania**. W związku z tym część ekspertów w obszarze inżynierii wymagań rekomenduje użycie **szablonów** specyfikacji wymagań pozafukcjonalnych, aby zwiększyć kompletność wymagań i zapewnić wysoką jakość ich specyfikacji. Szablon wymagania określa domyślne brzmienie takiego wymagania, uwzględniając przy tym fragmenty opcjonalne i wymagające modyfikacji, tak aby można go było dopasować do konkretnego kontekstu użycia. Zbiór szablonów tworzy **katalog szablonów** wymagań. Jeśli organizacja decyduje się na stworzenie takiego katalogu, to powinien on być aktualizowany po ukończeniu każdego projektu poprzez dodawanie, modyfikowanie lub usuwanie szablonów wymagań. W pracy [A6] dokonaliśmy empirycznej oceny zastosowania **katalogu wymagań pozafuncjonalnych** z perspektywy jego **użyteczności** oraz **kosztów utrzymania**. W pierwszej kolejności przeprowadziliśmy badanie w postaci kontrolowanego eksperymentu mającego na celu zbadanie użyteczności katalogu szablonów wymagań pozafunkcjonalnych do pozyskiwania i specyfikowania wymagań pozafunkcjonalych. W eksperymencie wzięło udział **107 uczestników**, których zadaniem było wyspecyfikowanie wymagań poza-funkcjonalnych dla przedstawionego im systemu typu e-commerce. Część uczestników pracowała w trybie indywidualnym a część w grupach oraz bez lub z użyciem katalogu szablonów wymagań. Wyniki eksperymentu pokazały, że **użycie katalogu szablonu wymagań pozafunkcjonalnych pozwala zwiększyć zarówno jakość jak i kompletność specyfikowanych wymagań**. W kolejnym kroku przeprowadziliśmy badanie symulacyjne na zbiorze **2231 wymagań pozafunkcjonalnych** z **41 projektów informatycznych**. Celem badania była ocena jak charakterystyki katalogu szablonów wymagań (np. koszt utrzymania) mogą zmieniać się wraz z prowadzeniem kolejnych projektów informatycznych w organizacji. Obserwacje poczynione w trakcie tego badania pozwoliły nam wprowadzić pojęcie **dojrzałości katalogu**, który określa moment, w którym zawartość katalogu stabilizuje się pomiędzy kolejno realizowanymi projektami. Efekt ten można zaobserwować na rysunku 3, który pokazuje jak stopień pokrycia wymagań w projekcie przez szablony dostępne w katalogu (Value) oraz koszty utrzymania (Maintenance Effort) związane z jego aktualizacją zmieniały się w trakcie symulacji realizacji kolejnych projektów.



Rysunek 3 Zmiana wartości (Value) katalogu oraz kosztu jego utrzymania (Maintenance Effort) pomiędzy kolejnymi projektami w ramach badania symulacyjnego [A6].

Po realizacji około **40 projektów** katalog osiągnął punkt dojrzałości (w tym momencie zawierał on **400 szablonów wymagań**). Jak widać na rysunku 3a wartość katalogu (zdefiniowana jako stosunek liczby wymagań pozafunkcjonalnych w projekcie, które można było wywieść z szablonów wymagań w katalogu, do liczby wszystkich wymagań pozafunkcjonalnych w projekcie) rośnie wraz z liczbą realizowanych projektów. Analogicznie koszt utrzymania katalogu (tj. koszt wprowadzania nowych szablonów i zmiany istniejących) stopniowo maleje, oczywiście w zależności od tego jak podobne są do siebie projekty realizowane przez organizację. Na podstawie wyników obu badań można stwierdzić, że użycie szablonów wymagań pozafunkcjonalnych może przyczynić się do poprawy kompletności i jakości wymagań. Oczywiście decydując się na wdrożenie katalogu szablonów do organizacji należy mieć świadomość tego, że utrzymanie takiego katalogu wiążę się z dodatkowym nakładem pracy, zwłaszcza na etapie stabilizacji jego zawartości.

Wspieranie pomiaru rozmiaru oprogramowania

Rozmiar oprogramowania jest podstawowym czynnikiem predykcji używanym w metodach szacowania pracochłonności oprogramowania. W chwili obecnej dominują dwa podejścia do pomiaru rozmiaru oprogramowania: pomiar rozmiaru fizycznego oraz pomiar rozmiaru funkcjonalnego oprogramowania. Istnieje także trzecie podejście określane jako pomiar rozmiaru pozafunkcjonalnego (np. IFPUG SNAP [4]), natomiast nie jest ono powszechnie stosowane w praktyce.

[4] C. Tichenor. *A new software metric to complement function points: the Software Non-functional Assessment Process (SNAP)*. DEFENSE SECURITY COOPERATION AGENCY WASHINGTON DC, 2013.

Metody pomiaru rozmiaru funkcjonalnego takie jak COSMIC[5] czy IFPUG Function Points Analysis (FPA)[6] pozwalają na pomiar rozmiaru oprogramowania na podstawie jego wymagań funkcjonalnych. Natomiast należy wspomnieć, że przeprowadzenie pełnej procedury pomiarowej jest możliwe tylko wtedy, gdy wymagania funkcjonalne są dokładnie wyspecyfikowane. **Niestety problem polega na tym, że w sytuacji, gdy potrzeba szacowania pracochłonności zachodzi na wczesnych etapach projektowych to wymagania dla tworzonego produktu są zazwyczaj zdefiniowane na bardzo wysokim poziomie ogólności**, co skutecznie uniemożliwia dokonanie dokładnego pomiaru rozmiaru funkcjonalnego**.** W takiej sytuacji możemy zastosować jedną z dwóch strategii postępowania. Po pierwsze możemy zainwestować dodatkowy czas i środki, aby już na tym etapie spróbować doprecyzować wymagania funkcjonalne do poziomu szczegółowości wymaganego przez metody pomiaru rozmiaru funkcjonalnego (np. stosując metody opisane we wcześniejszym rozdziale). Alternatywnie możemy spróbować oszacować rozmiar funkcjonalny na bazie dostępnych wysokopoziomowych wymagań, aby uzyskać wystarczająco dobre przybliżenie jego rozmiaru. Drugie podejście jest zdecydowanie bardziej atrakcyjne w przypadku, gdy czas i/lub inne zasoby w projekcie są mocno ograniczone.

W ramach prowadzonych przez mnie prac badawczych skoncentrowałem się na opracowaniu metod pozwalających na szacowaniu rozmiaru funkcjonalnego, na bazie tytułów/nazw przypadków użycia. Tytuł przypadku użycia powinien odzwierciedlać cel jaki użytkownik pragnie osiągnąć poprzez interakcję z systemem (np. „Zgłoś artykuł na konferencję naukową"). W związku z tym tytuły przypadków użycia są często używane do wstępnego określenia zakresu wymagań funkcjonalnych dla systemów informatycznych (taki zakres często prezentowany jest w postaci diagramu przypadków użycia UML). Niestety zarówno metoda COSMIC jak i metoda IFPUG FPA (oraz jej różne warianty) nie pozwalają dokonać pełnego pomiaru rozmiaru funkcjonalnego na podstawie samych tytułów przypadków użycia. Metody te wymagają wiedzy na temat przepływów danych z i do systemu oraz informacji o strukturach danych przetwarzanych przez system. W celu rozwiązania powyższego problemu zaproponowaliśmy zbiór metod pozwalających szacować rozmiar funkcjonalny tylko na podstawie tytułów przypadków użycia. W pracy [A8] zaproponowałem dwie takie metody **Average Use-Case Goal-aware Approximation (AUCG)** oraz **Bayesian Network Use-Case Goal AproxImatioN (BN-UCGAIN)**. Są to metody dwuetapowe. W pierwszym etapie określany jest typ celu przypadku użycia na podstawie jego tytułu (w oparciu o zaproponowaną taksonomię 13 typów celów przypadków użycia). W drugim etapie następuje szacowanie rozmiaru funkcjonalnego na podstawie typu celu przypadku użycia oraz danych historycznych z wcześniej realizowanych projektów. Obie metody powstały w wyniku analizy zbioru **427 przypadków użycia**, która wykazała istnienie zależności pomiędzy typem celu przypadku użycia a jego rozmiarem funkcjonalnym. W tej samej pracy zaproponowałem także

[5] ISO/IEC, ISO/IEC 19761:2011: Software engineering – COSMIC: a functional size measurement method, 2003.

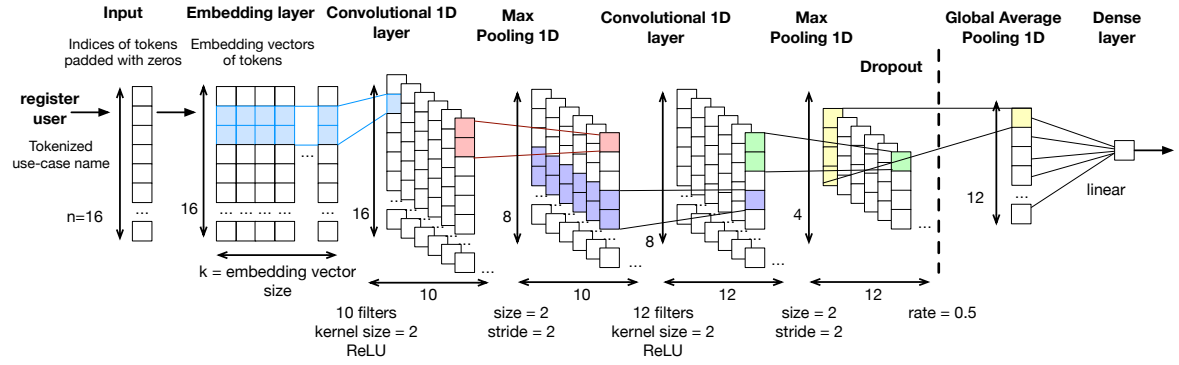
[6] A. Albrecht, Measuring application development productivity, Proc. Joint SHARE/GUIDE/IBM Appl. Dev. Symp., pp. 83–92, 1979.

automatyczne podejście do określania celu przypadku użycia na podstawie jego tytułu. Podejście to bazuje na użyciu narzędzi przetwarzania języka naturalnego i algorytmach uczenia maszynowego. Dokładność szacowania rozmiaru funkcjonalnego proponowanych metod została oceniona na zbiorze **26 projektów** informatycznych i porównania z dokładnością szacowania na podstawie **średniego rozmiar przypadku użycia (AUC)** oraz z dokładnością metody zaproponowanej przez Hussaina, Kosseim i Ormandjievę (**HKO**).[7] Metody te stanowią najbardziej zbliżone alternatywy dla zaproponowanych przeze mnie metod. Przeprowadzone badania wykazało, że metody **AUCG i BN-UCGAIN pozwoliły szacować rozmiar funkcjonalny z większą dokładnością niż metody konkurencyjne (o około 11% do 24%)**. Warto podkreślić, że zaproponowane metody zyskały uznanie konsorcjum COSMIC, które opisało je w oficjalnym podręczniku do przeprowadzania przybliżonych pomiarów metodą COSMIC.[8]

W pracy [A1] przedstawiliśmy wyniki naszych dalszych prac badawczych, które doprowadziły do opracowania nowej metody szacowania rozmiaru funkcjonalnego COSMIC na podstawie nazw/tytułów przypadków użycia o nazwie **DEEP-COSMIC-UC**. Metoda ta bazuje na modelu regresji, mającego postać sztucznej sieci neuronowej (architekturę modelu predykcji przedstawiono na rysunku 4). W przeciwieństwie do poprzednich metod jest to metoda jednoetapowa (tzn. nie wymaga określenia typu celu przypadku użycia). Na wejściem do modelu DEEP-COSMIC-UC podawany jest tekst tytułu przypadku użycia a wyjście stanowi jego szacowany rozmiar funkcjonalny COSMIC. Głównym elementem budującym model predykcji są tzw. warstwy konwolucyjne. W modelu używane są także wstępnie trenowane modele wektorowej reprezentacji słów (ang. word embeddings), pozwalające zamienić tekst tytułu przypadku użycia na postać wektora liczb. Dokładność szacowania nowego modelu została zbadania na zbiorze **438 przypadków użycia z 27 projektów informatycznych**. Proponowany model był w stanie szacować rozmiar COSMIC pojedynczych przypadków użycia z **wyższą dokładnością niż model AUC (o około 20%) oraz dwa poprzednio zaproponowane przeze mnie modele AUCG oraz BN-UCGAIN (o około 5-7%)**. Dodatkowo statystycznie poprawa dokładności szacowania była zauważalna już po szacowaniu dziesięciu przypadków użycia. Istotną zaletą metody DEEP-COSMIC-UC w stosunku do metod AUCG oraz BN-UCGAIN jest to, że nie wymaga ona zbierania danych historycznych innych niż tytuły przypadków użycia wraz z ich rozmiarem funkcjonalnym. W konsekwencji metoda ta może zostać użyta nawet jeśli w bazie danych historycznych nie gromadzimy informacji o typach celów przypadków użycia. Wreszcie możliwość bardziej dokładnego szacowania rozmiaru na poziomie pojedynczych przypadków użycia sprawia, że metoda jest lepiej dostosowana do szacowania pracochłonności krótszych iteracji / wydań charakterystycznych dla zwinnych projektów.

[7] I. Hussain, L. Kosseim, O. Ormandjieva, Approximation of COSMIC functional size to support early effort estimation in Agile, Data & Knowledge Eng. 85, 2–14, 2013.

[8] COSMIC, Early Software Sizing with COSMIC: Experts Guide, 2nd edition, February 27, 2020



Rysunek 4 Architektura modelu szacowania rozmiaru funkcjonalnego COSMIC (DEEP-COSMIC-UC) [A1].

Alternatywnym podejściem do pomiaru rozmiaru oprogramowania jest pomiar rozmiaru fizycznego oprogramowania (tj. pomiar liczby linii kodu źródłowego – LOC). Linie kodu źródłowego są powszechnie stosowane jako czynnik determinujący pracochłonność i od wielu lat stanowią istotny element uznanych metod szacowania pracochłonności (np. LOC stanowi bazowy parametr wejściowy w metodzie COCOMO II[9] – jednej z najbardziej znanych metod szacowania pracochłonności). Niestety podejście polegające na zliczaniu linii kodu źródłowego posiada kilka istotnych wad. Przede wszystkim należy zwrócić uwagę na to, że na etapie planowania budowy nowego systemu informatycznego dysponujemy zazwyczaj jedynie opisem jego wymagań albo wstępnym projektem, a nie kodem źródłowym. Drugim często powtarzanym zarzutem w stosunku do LOC jest trudność pomiaru rozmiaru systemów zaimplementowanych przy użyciu więcej niż jednego języka programowania. Pomimo wyżej wymienionych słabości, pomiar rozmiaru fizycznego wydaje się **mieć zastosowanie w sytuacjach, kiedy szacujemy pracochłonność zadań, dla których bazowy kod źródłowy już istnieje** (np. szacowanie pracochłonności prac związanych z utrzymaniem wdrożonego już systemu informatycznego, czy też szacowanie pracochłonności testowania oprogramowania). W przypadku takich zastosowań może wystąpić potrzeba użycia wyspecjalizowanego **wariantu LOC polegającego na zliczaniu linii kodu istotnych z perspektywy danego zadania**. Niestety katalog specjalizowanych definicji miar LOC dostępnych w literaturze naukowej ogranicza się do kilku definicji takich miar, a opracowanie nowych dedykowanych definicji nie jest zadaniem trywialnym. W odpowiedzi na ten problem zaproponowaliśmy odwrócenie paradygmatu definiowania miar rozmiaru fizycznego oprogramowania z imperatywnego na bazujących na przykładach, tak aby ułatwić definiowanie nowych, specjalizowanych miar LOC. W przeciwieństwie do typowego imperatywnego podejścia, które zakłada konieczność określenia reguł pomiaru (na podstawie których można opracować instrument pomiarowy), nasze podejście bazuje na przykładach pomiarów (fragmentów kodu programów z oznaczonymi liniami, które należy policzyć w danym kontekście użycia) oraz algorytmach uczenia maszynowego pozwalających na automatyczną implementację narzędzi

[9] Boehm, B. W., Abts, C., Brown, A. W., Chulani, S., Clark, B. K., Horowitz, E., ... & Steece, B., Software cost estimation with COCOMO II (Vol. 1). Upper Saddle River, NJ: Prentice Hall, 2000.

pomiarowych. W wyniku prowadzonych prac badawczych opracowaliśmy oprogramowanie narzędziowe **Flexible Code Counter/Classifier (CCFLex)**, które pozwala na tworzenie instrumentów pomiarów dla specjalizowanych miar rozmiaru oprogramowania tylko na bazie dostarczonych przykładów pomiarów. Oprogramowanie to jest rozwijane na zasadach wolnego oprogramowania (ang. open source). W ramach wstępnych badań nad proponowanym podejściem do pomiaru rozmiaru fizycznego sprawdziliśmy czy możliwe jest wytrenowania instrumentów pomiarowych na bazie przykładów dla znanych i często stosowanych miar LOC.[10] W ramach dalszych prac badawczych [A3], wprowadziliśmy znaczne usprawnienia do zaproponowanego narzędzia CCFLex oraz zbadaliśmy możliwość jego zastosowania do identyfikacji (zliczania) **linii kodu źródłowego, które naruszają wewnętrzne wytyczne stosowane w dwóch dużych skandynawskich firmach wytwarzających oprogramowanie**. Linie naruszające takie zasady mają istotne znaczenie z perspektywy szacowania pracochłonności i kosztów utrzymania aplikacji (ang. maintenance effort). Przeprowadzone badanie typu **action research** (pol. badanie w działaniu) obejmowało: (a) analizę wyzwań i potrzeb obu firm w zakresie wykrywania linii kodu niezgodnych ze stosowanymi przez nie wytycznymi, (b) zbadanie możliwości użycia narzędzia CCFlex do wykrywania linii niezgodnych z dwoma popularnymi standardami kodowania zaproponowanymi przez firmy Sun oraz Google w kodzie trzech dużych projektów typu open source w języku Java, (c) ocenę skuteczności narzędzia CCFlex w warunkach przemysłowych na kodzie programów podlegających ciągłej ewolucji oraz (d) poszukiwanie strategii trenowania zaproponowanego narzędzia z perspektywy ograniczenia kosztów związanych z przygotowaniem przykładów treningowych dla klasyfikatorów linii kodu. W przypadku **projektów open source** i standardów kodowania **Sun'a i Google'a** zaproponowane narzędzie CCFlex osiągnęło średnią **dokładność (accuracy) powyżej 99%** i **średnią wartość miary F-score na poziomie 0,80** w przypadkach gdy do jego trenowania użyto odpowiednio dużego zbioru danych (**40 tyś. linii kodu źródłowego**). Uzyskaliśmy zbliżone wyniki (**średnia wartość F-score na poziomie 0,78**) dla **kodu przemysłowego**, ale tym razem **używając tylko do 700 linii kodu jako bazy do trenowania narzędzia**. Proponowane podejście uzyskiwało **najlepsze rezultaty dla wytycznych, które wymagały zrozumienia kontekstu pojedynczej lub kilku linii kodu programów (często pozwalając osiągnąć F-score na poziomie 0,90 i wyższym)**. Otrzymane rezultaty pozwoliły nam stwierdzić, że proponowane podejście do pomiaru rozmiaru fizycznego oprogramowania może zostać zastosowane w warunkach przemysłowych do tworzenia specjalizowanych definicji miary LOC, które w dalszym etapie mogą zostać użyte do budowy modeli szacowania pracochłonności, dedykowanych do konkretnych typów zadań w projektach informatycznych.

[10] 9. M. Ochodek, M. Staron, D. Bargowski, W. Meding, R. Hebig. Using machine learning to design a flexible LOC counter, In 2017 IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE), pp. 14-20. IEEE, DOI: 10.1109/MALTESQUE.2017.7882011, 2017.

Wspieranie pozyskiwania zbiorów danych historycznych

Trzeci nurt badawczy w ramach prezentowego przeze mnie cyklu artykułów dotyczy możliwości wsparcia procesu pozyskiwania oraz „czyszczenia" danych historycznych o projektach informatycznych. Moje zainteresowanie tą problematyką wynika z faktu, że większość metod szacowania pracochłonności wymaga kalibracji lub wręcz uczenia modeli predykcji od podstaw w oparciu o dane historyczne.

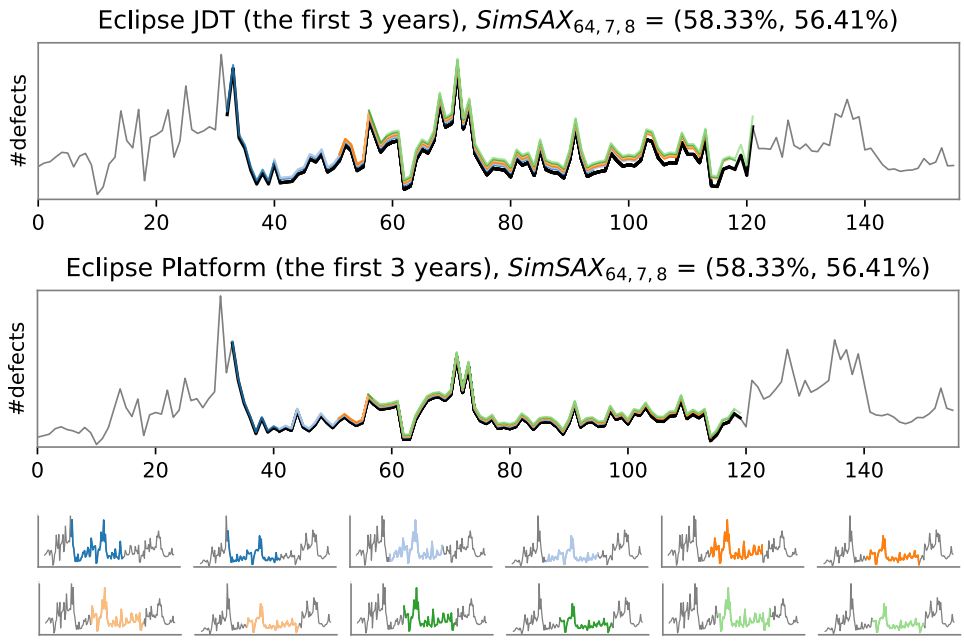
W przypadku metod szacowania pracochłonności pożądane jest, aby projekty w zbiorze danych historycznych użytym do kalibracji/trenowania metody szacowania pracochłonności były jak najbardziej zbliżone do projektów, których pracochłonność będzie szacowana w przyszłości (np. aby dotyczyły one tej samej dziedziny biznesowej, miały na celu rozwijać podobne typy aplikacji, czy też były realizowane według podobnych procesów). Moje zainteresowania badawcze w tym obszarze ogniskowały się na dwóch zagadnieniach: na procesie „czyszczenia" danych zawartych w zbiorach historycznych o projektach oraz na wyszukiwaniu podobieństw pomiędzy projektami informatycznych, uwzględniając przy tym istotne zmiany w tych projektach zachodzące w czasie ich realizacji (np. zmiany w procesach czy praktykach stosowanych przez zespół projektowy).

Aby stworzyć jednorodny (homogeniczny) zbiór danych historycznych niezbędny jest algorytm pozwalający na ocenę podobieństwa pomiędzy projektami. W większości baz danych projektowych (np. w bazie ISBSG [11]) pojedyncze projekty opisywane są pewnym stałym zestawem cech (np. pracochłonność, liczba osób w zespole, stosowany język programowania). Cechy te mogą zostać użyte do oceny podobieństwa pomiędzy projektami oraz na tej podstawie do wyszukiwania projektów pasujących do danego kontekstu realizacji przedsięwzięcia. Niestety podejście to zakłada, że cechy opisujące projekt są stałe dla całego okresu jego realizacji ignorując tym samym fakt, że **wiele organizacji rozwija swoje produkty w sposób ciągły, nawet bez wyraźnego rozgraniczenia pomiędzy projektami** (jest to w dużej mierze konsekwencją wdrożenia przez nie zwinnych metodyk zarządzania projektami). Biorąc pod uwagę, że dany produkt może być rozwijany i utrzymywany przez wiele lat trudno **jest scharakteryzować go używając jednego zestawu cech** (np. liczba osób w zespole czy proces wytwarzania oprogramowania może podlegać ciągłym zmianom). W efekcie istnieje potrzeba poszukiwania nowych sposobów oceny podobieństwa pomiędzy projektami informatycznymi, które wezmą pod uwagę także zachodzące zmiany w trakcie ich realizacji. Uwzględnienie perspektywy czasowej umożliwiłoby poszukiwanie podobieństw zarówno dla całych projektów, ale także pomiędzy ich etapami realizacji. W pracy [A5] podjęliśmy próbę rozwiązania tego problem proponując **nową miarę oceny podobieństwa projektów informatycznych** o nazwie **SimSAX**. Ocena podobieństwa dokonywana jest na podstawie analizy tzw. profilu napływu zgłoszeń o defektach w oprogramowaniu (ang. defect-inflow profile). Taki profil ma postać szeregu czasowego (ang. time series) liczby defektów wykrytych w kolejnych tygodniach realizacji projektu. W ramach procedury pomiarowej profile defektów

[11] ISBSG - https://www.isbsg.org

transformowane są do sekwencji symboli (z określonego alfabetu) oraz porównywane w celu wykrycia tzw. **motywów** (powtarzających się, podobnych podsekwencji). W ramach prowadzonych prac badawczych opracowaliśmy procedurę kalibracji parametrów pomiaru podobieństwa z użyciem miary SimSAX. Przeprowadziliśmy także wstępną kalibrację metody pomiarowej na podstawie danych z **dwóch dużych projektów przemysłowych**. Bazując na danych pochodzących z wywiadów z uczestnikami tych projektów byliśmy w stanie określić rzeczywiste podobieństwa pomiędzy projektami w trakcie ich realizacji oraz skonfrontować je z podobieństwami wskazywanymi przez SimSAX. W kolejnym kroku rozszerzyliśmy badania o analizę podobieństw **pięciu projektów przemysłowych i sześciu projektów open source**. Przeprowadzone analizy wykazały, że miara SimSAX może zostać użyta do znajdowania podobieństw między projektami, zwłaszcza gdy wyszukiwane są długie motywy (32-tygodniowe i dłuższe) oraz kiedy użyty alfabet ma przynajmniej 5 symboli. Na rysunku 5 przedstawiłem wizualizację przykładowego porównania dwóch profili defektów, w tym przypadku dla wybranego okresu rozwoju Eclipse Platform oraz Eclipse JDT oraz motywów znalezionych przy użyciu SimSAX. W okresie objętym analizą oba projekty miały bardzo zbliżone procesy wytwórcze. Wreszcie zaproponowana metoda kalibracji pozwala dostosować miarę do konkretnego kontekstu organizacji czy też projektu dzięki czemu może posłużyć znajdowaniu wzorców podobieństwa w warunkach przemysłowych.

W kolejnej pracy [A2] zaproponowaliśmy metodą o nazwie **LegacyPro**, która bazuje na użyciu miary SimSAX do znajdowania „zaszłości" w sposobie pracy zespołów programistycznych po przeprowadzeniu transformacji w organizacji (np. po wprowadzeniu zwinnych metod zarządzania projektami). Metoda LegacyPro może także zostać użyta do ciągłego monitorowania sposobu pracy zespołów projektowych. W przypadku wykrycia znaczących zmian w sposobie pracy należałoby dokonać ponownej analizy danych historycznych pod kątem ich adekwatności do nowej sytuacji i ew. dokonać re-kalibracji używanej metody szacowania pracochłonności. LegacyPro bazuje na wykrywaniu dwóch typów motywów w profilach defektów – **motywów wewnętrznych** w ramach danego projektu (ang. internal motifs) oraz **motywów współdzielonych,** czyli występujących w następujących po sobie projektach (ang. shared motifs). Motywy wewnętrzne pozwalają określić na ile powtarzalny jest sposób pracy danego zespołu oraz czy nie uległ on zmianie na przestrzeni czasu. Natomiast motywy współdzielone pozwalają wykrywać zmiany (lub ich brak) na poziomie całej organizacji. W ramach prowadzonych prac badawczych dokonaliśmy walidacji proponowanej metody na danych z **czterech projektów przemysłowych i pięciu projektów open source,** charakteryzujących się bardzo długim czasem rozwoju. Użycie metody pozwoliło wykryć podobieństwa oraz różnice w sposobie rozwoju tych projektów, które znajdowały swoje odzwierciedlenie w faktach historycznych. Dodatkowe badanie o charakterze symulacyjnym pozwoliło stwierdzić, że istnieje możliwość zamiany standardowego wyjścia metody LegacyPro do postaci klasyfikatora binarnego, który można użyć do automatycznego wykrywania zmian w sposobie realizacji przedsięwzięć informatycznych. W kontekście dostosowywania metod szacowania pracochłonności użycie metody może pozwolić na utrzymanie jednorodności i aktualności zbiorów danych historycznych gromadzonych w ramach danej organizacji.



Rysunek 5 Przykłady podobieństw znalezionych przy użyciu SimSAX pomiędzy profilami liczby zgłoszonych defektów dla projektów Eclipse Platform oraz Eclipse JDT (w tym okresie, oba projekty były realizowane w podobny sposób) [A5].

Wraz z rozwojem platform służących do współdzielenia kodu takich jak GitHub [12] czy SourceForge [13] społeczność pracująca nad rozwojem systemów informatycznych zyskała dostęp do dużego korpusu danych o projektach informatycznych (tylko w ramach samej platformy GitHub w 2020 roku udostępniono ponad 190 milionów repozytoriów kodu). Dane te mogą zostać użyte do budowy różnego rodzaju modeli predykcyjnych – w tym do trenowania / kalibracji modeli szacowania pracochłonności. Niestety jakość danych dostępnych w ramach tego typu platform jest często kwestionowana (zresztą zasadnie). W związku z tym istnieje potrzeba opracowania metod pozwalających na identyfikację repozytoriów zawierających „rzeczywiste" projekty (ang. engineered projects) oraz takie zawierające projekty nieistotne w kontekście budowy modeli predykcji w warunkach przemysłowych (np. dane z mini projektów studenckich, projektów realizowanych w ramach samodoskonalenia). W 2017 roku Munaiah i inni[14] zaproponowali metodę oraz narzędzie Reaper pozwalające filtrować „rzeczywiste" projekty w zbiorze repozytoriów GitHub. Metoda bazowała na algorytmach uczenia maszynowego z nadzorem i swoją skutecznością przewyższyła wcześniej stosowane podejścia. W ramach prowadzonych prac dokonano automatycznej klasyfikacji zbioru 1,8 miliona repozytoriów dostępnych w ramach platformy GitHub. Chociaż było to istotne osiągnięcie, to zaproponowane przez autorów podejście nie pozostawało bez wad. Po pierwsze wymagało ono analizy wielu artefaktów w ramach

[12] GitHub - https://github.com

[13] SourceForge - http://sourceforge.net

[14] Munaiah N, Kroh S, Cabrey C, Nagappan M, Curating GitHub for engineered software projects. Empirical Software Engineering, vol. 22, pp. 3219–3253, https://doi.org/10.1007/s10664-017-9512-6, 2017

pojedynczego projektu (tj. logów Git, plików konfiguracyjnych, kodu źródłowego, rejestru spraw – GitHub issues, licencji, kodu testów jednostkowych). W konsekwencji zaproponowane narzędzie miało **bardzo wygórowane wymagania co do zasobów sprzętowych** (wspomniana wcześniej analiza wymagała użycia 200 węzłów obliczeniowych i zajęła miesiąc czasu). Po drugie niektóre z artefaktów wymagały opracowania dedykowanych analizatorów uzależnionych od języka programowania oraz bibliotek programistycznych użytych w danym projekcie. W związku z tym opracowane **narzędzie musi zostać dostosowane do użycia w konkretnym kontekście**. W pracy [A4] zaproponowaliśmy metodę oraz narzędzie alternatywne do powyższych o nazwie **PHANTOM** (Project History Analysis of Time-Series Method). Metoda bazuje na algorytmach uczenia maszynowego bez nadzoru i **pozwala odseparować repozytoria zawierające „rzeczywiste" projekty od pozostałych, nawet przy bardzo dużych wolumenach danych.** Metoda bazuje wyłącznie na analizie częstotliwości zmian w historii rozwoju projektów na podstawie danych z logów Git. Dzięki temu nie wymaga ona użycia zaawansowanego sprzętu komputerowego. Bazując na zbiorze danych udostępnionym przez Munaiah i innych pokazaliśmy, że PHANTOM jest wstanie odtworzyć porządek klasyfikacyjny na zbiorze referencyjnym oraz sklasyfikować przykłady ze zbioru testowego **z precyzją równą 0,87** (ang. precision) **oraz czułością na poziomie 0,94** (ang. recall). PHANTOM pobrał i przetworzył metadane z **1 786 601 repozytoriów GitHub w czasie 21,5 dni używając do tego typowego komputera osobistego**, co czyni go zdecydowanie bardziej wydajnym niż metoda i narzędzie zaproponowane przez Munaiah i innych (jak już wcześniej wspomniałem podobna analiza opisana we wcześniejszej pracy zajęła miesiąc czasu i wymagała użycia klastra obliczeniowego złożonego z 200 węzłów). W ramach tego samego badania zweryfikowaliśmy możliwość użycia metody/narzędzia PHANTOM w warunkach przemysłowych badając **100 repozytoriów należących do dwóch firm informatycznych**. Wyniki badań pokazały, że PHANTOM jest podejściem konkurencyjnym w stosunku do zaproponowanego wcześniej, w odniesieniu do jego dokładności a przy tym jego wydajność jest o dwa rzędy wyższa. Zatem metoda / narzędzie to może być stosowane do pozyskiwania i oczyszczania danych dotyczących projektów, w oparciu o duże zbiory zarówno w warunkach przemysłowych jak i w kontekście otwartego oprogramowania.

Podsumowanie

Zaprezentowany cykl artykułów obejmuje dziewięć pozycji opublikowanych w czasopismach indeksowanych w ramach Journal Citation Reports – JCR (wszystkie one należą do kategorii A listy czasopism publikowanej przez Ministerstwo Edukacji i Nauki / Ministerstwo Nauki i Szkolnictwa Wyższego). Artykuły prezentują wyniki badań mających na celu opracowywanie metod wspierających proces szacowania pracochłonności w projektach informatycznych. W szczególności koncentrują się one na trzech obszarach badawczych: wsparciu procesu pozyskiwania wymagań, pomiaru / szacowaniu rozmiaru oprogramowania oraz pozyskiwaniu zbiorów danych historycznych o projektach informatycznych.

Chociaż zaprezentowane cykl artykułów prezentuje moje najbardziej istotne osiągnięcia naukowe związane z problematyką szacowania pracochłonności to chciałbym zaznaczyć w tym

miejscu, że nie stanowią one całości mojego dorobku w tym obszarze. Zaprezentowany zbiór prac może zostać uzupełniony o kilka innych prac (głównie rozdziały w monografiach oraz artykuły w materiałach konferencyjnych) przedstawionych w rozdziale II „wykaz osiągnięć naukowych stanowiących znaczny wkład w rozwój określonej dyscypliny" będącym jednym z pozostałych załączników do wniosku.

II. WSPÓŁPRACA MIĘDZYNARODOWA

Od 2016 roku prowadzę bardzo intensywną współpracę z naukowcami z **University of Gothenburg (GU) / Chalmers University of Technology** należącymi do jednej z najbardziej renomowanych grup badawczych w obszarze inżynierii oprogramowania. Na przestrzeni ostatnich lat regularnie gościłem na uniwersytecie w Goteborgu. Odwiedziłem tę uczelnie dwukrotnie w 2017 roku (w marcu oraz na przełomie listopada i grudnia) a w kolejnym roku byłem tam zatrudniony na stanowisku starszego wykładowcy przez okres **sześciu miesięcy** (od 25 stycznia do 30 czerwca). Jeszcze tego samego roku ponownie odwiedziłem grupę badawczą w Goteborgu w ramach programu Erasmus+ Staff for Training (STT).

Dzięki współpracy z GU/Chalmers miałem możliwość prowadzenia badań we współpracy z dużymi firmami wytwarzającymi oprogramowanie zlokalizowanymi w Skandynawii i współpracującymi w ramach **Software Center**.[15] Pełniłem także rolę „głównego badacza" (ang. principal investigator) w projekcie "Anomaly Detection in Wireless Networks using Machine Learning" realizowanego w ramach Software Center przy współpracy z firmami Ericsson AB, Axis Communication AB oraz Grundfos.

Rezultaty prac badawczych prowadzonych we współpracy z naukowcami z Goteborga zostały przedstawione w dziesięciu publikacjach naukowych (część z tych prac należy do cyklu artykułów przedstawionych w rozdziale IV). Jedna z naszych prac prezentowana na konferencji Software Quality Days 2019 w Wiedniu uzyskała nagrodę „Best Paper Award."

Miałem też zaszczyt być członkiem zespołu, obejmującego dwóch pracowników uczelni (GU oraz Politechniki Poznańskiej) oraz dwóch pracowników firmy Ercisson AB, który w 2018 roku otrzymał nagrodę **Emerald's Real Impact Award** [16] w uznaniu za praktyczne znaczenie prowadzonych badań w obszarze pomiaru oprogramowania oraz wdrożone rozwiązania w firmach sektora ICT.

[15] Software Center - https://www.software-center.se

[16] Emerald's Real Impact Awards 2018: https://www.emeraldgrouppublishing.com/topics/news/winners-first-ever-real-impact-awards-revealed.

III.INFORMACJE NAUKOMETRYCZNE

Przedstawione informacje naukometryczne bazują na analizie dorobku sporządzonej przez Bibliotekę Politechniki Poznańskiej (załączona do wniosku): Google Scholar (11.02.2021), Scopus (11.02.2021), Web of Science (10.02.2021) oraz analiza dorobku (15.02.2021).

Informacja o punktacji Impact Factor (w dziedzinach i dyscyplinach, w których parametr ten jest powszechnie używany jako wskaźnik naukometryczny)

- **Impact Factor** (wszystkie artykuły) = **29,691**
- **Impact Factor** po uzyskaniu stopnia doktora (10 artykułów) = 26,479
- **Impact Factor** przed uzyskaniem stopnia doktora (4 artykuły) = 3,212

Informacja o liczbie cytowań publikacji wnioskodawcy, z oddzielnym uwzględnieniem autocytowań

Google Scholar

- **447** cytowań (włączając autocytowania).

Scopus

- **263** cytowań, w tym **23** autocytowań**,**
- **258** cytowań włączając publikacje nieindeksowane w bazie Scopus (bez autocytowań).

Clarivate Web of Science™

- **107** cytowań (bez autocytowań)
- **123** cytowań włączając w to publikacje nieindeksowane w WoS (bez autocytowań).

Informacja o posiadanym indeksie Hirscha

Google Scholar

- h-index = **11**,
- i10-index = **13**.

Scopus

- h-index = **8**.

Clarivate Web of Science™

- h-index = **5,**
- h-index = **6** włączając publikacje nieindeksowane w WoS.

Informacja o liczbie punktów MNiSW/MNiE

I) Punktacja zgodnie z bieżącymi wytycznymi **MNiSW/MEiN (2019+)** za:

- **publikacje w czasopismach naukowych**:
  - po uzyskaniu stopnia doktora (11 artykułów) = 1320 pkt.
  - przed uzyskaniem stopnia doktora (5 artykułów) = 340 pkt.
  - **Łącznie 1660 pkt.**
- **Rozdziały w monografiach naukowych/publikacje w materiałach konferencyjnych**
  - po uzyskaniu stopnia doktora (18 publikacji) = 595 pkt.
  - przed uzyskaniem stopnia doktora (7 publikacji) = 290 pkt.
  - **Łącznie 885 pkt.**

1. II) Punktacja zgodnie z wytycznymi **MNiSW/MEiN (2019+, 2013-2016)** obowiązującymi w roku publikacji za:

- **publikacje w czasopismach naukowych**
  - po uzyskaniu stopnia doktora (11 artykułów) = 854 pkt.
  - przed uzyskaniem stopnia doktora (5 artykułów) = 95 pkt.
  - **Łącznie 943 pkt.**

IV.OSIĄGNIĘCIA DYDAKTYCZNE, ORGANIZACYJNE ORAZ MAJĄCE NA CELU POPULARYZUJĄCE NAUKI

W trakcie mojej pracy na **Wydziale Informatyki i Telekomunikacji** Politechniki Poznańskiej (wcześniej były to Wydziały Informatyki oraz Wydział Informatyki i Zarządzania) prowadziłem liczne kursy akademickie obejmujące zagadnienia związane z inżynierią oprogramowania (wykłady, zajęcia laboratoryjne i projektowe) zarówno na pierwszym jak i drugim stopniu studiów (m.in. inżynierii oprogramowania, zarządzanie projektami, inżynieria wymagań, empiryczna inżynieria oprogramowania, zarządzanie jakością). Prowadzone przeze mnie zajęcia były doceniane przez studentów i uzyskiwały wysokie oceny w prowadzonych na uczelni anonimowych badaniach ankietowych wśród studentów. Moja średnia ocena jako prowadzącego zajęcia na przestrzeni lat 2014/15 i 2019/20 wynosiła 4,53 (skala: 1–5, gdzie 1 oznacza ocenę minimalną). Dodatkowo przez ostatnie dziesięć lat prowadziłem kurs dotyczący inżynierii wymagań w ramach studium podyplomowego inżynierii oprogramowania (SPIO) [17] na Politechnice Poznańskiej.

Od 2011 roku jestem jednym z dwóch nauczycieli akademickich prowadzących **Studio Rozwoju Oprogramowania** (SDS) na Politechnice Poznańskiej. [18] SDS ma charakter kursu akademickiego realizowanego symultanicznie na dwóch stopniach nauczania w formie projektu (ang. capstone project). Studenci tworzą zespoły projektowe, które obejmują dwóch/trzech studentów studiów magisterskich specjalności Software Engineering oraz czterech studentów pierwszego stopnia kierunku informatyka. Wspólnymi siłami i pod opieką mentora, realizują oni projekt informatyczny na zlecenie rzeczywistego odbiorcy (firmy,

[17] SPIO - http://www.cs.put.poznan.pl/spio

[18] Software Development Studio - http://sds.cs.put.poznan.pl

jednostki administracji publicznej czy uczelni). Projekt ten staje się także projektem dyplomowym studentów pierwszego stopnia. W ciągu ostatnich dziesięciu lat zrealizowaliśmy w tej formule 45 projektów, w tym wiele we współpracy z firmami (np. Roche Polska Sp. z o.o., Wunderman Thompson Technology Sp. z o.o./Cognifide Sp. z o.o., HighSolutions Sp. z o.o., Forcom Sp. z o.o.). W uznaniu wkładu w rozwój tej inicjatywy trzykrotnie w latach 2013/14, 2015/16 oraz 2018/19 otrzymywałem nagrodę Rektora Politechniki Poznańskiej za osiągnięcia dydaktyczne.

Podczas mojego sześciomiesięcznego pobytu na **University of Gothenburg** i pracy na stanowisku **starszego wykładowcy** (Senior Lecturer) prowadziłem zajęcia dydaktyczne dotyczące architektury oprogramowania oraz zwinnych metodyk wytwarzania oprogramowania, a także promowałem i recenzowałem prace licencjackie.

Przez cały okres mojej pracy na uczelni starałem się rozwijać metody dydaktyczne, aby ułatwić studentom przyswajanie umiejętności i wiedzy z zakresu inżynierii oprogramowania. Niektóre z opracowanych metod zostały opisane w artykułach naukowych. Na przykład, wspólnie z moimi kolegami zaproponowaliśmy podejście do nauczania zagadnień związanych z oceną architektury oprogramowania bazujące na odgrywaniu ról, które nazwaliśmy technical drama.[19] Zaproponowałem także autorską metodę nauczania zagadnień inżynierii oprogramowania bazującą na wzmocnieniu synergii treści przekazywanych w ramach wykładów, zajęć laboratoryjnych i projektowych (**Scrum-centric framework for organizing software engineering academic courses**).[20] Metoda ta została wdrożona w ramach bazowego kursu inżynierii oprogramowania na studiach pierwszego stopnia na kierunku informatyka Politechniki Poznańskiej uzyskując bardzo wysokie oceny wśród studentów (średnia ocena kursu to 4,58, skala: 1–5, gdzie 1 jest najniższą oceną). W sposób ciągły pracuje także nad doskonaleniem zajęć projektowych w ramach Studio Rozwoju Oprogramowania, m.in. badając wpływ poziomu realizmu takich projektów na efektywność przyswajania wiedzy.[21]

W ramach pracy indywidualnej ze studentami kładę duży nacisk na zachęcanie ich do poszerzania swoich horyzontów i zgłębiania zagadnień dotyczących inżynierii wymagań poprzez włączenia się w realizację prac badawczych w naszym zespole. Niejednokrotnie współpraca taka zaowocowała powstaniem i opublikowaniem artykułów naukowych, których współautorami byli studenci.[22]

---

[19] B. Michalik, J. Nawrocki, M. Ochodek. "3-step knowledge transition: a case study on architecture evaluation." Proceedings of the 30th international conference on Software engineering. ACM, 2008.

[20] M. Ochodek. "A Scrum-centric framework for organizing software engineering academic courses." Towards a Synergistic Combination of Research and Practice in Software Engineering. Springer, Cham, 2018. 207-220.

[21] S. Kopczyńska, J. Nawrocki, M. Ochodek. "Software development studio: bringing industrial environment to a classroom." Proceedings of the First International Workshop on Software Engineering Education Based on Real-World Experiences. IEEE Press, 2012.

[22] Prace przedstawione w “wykazie osiągnięć naukowych stanowiących znaczny wkład w rozwój określonej dyscypliny” – II.2 – 4, 8, 11, 12, 14, 15; II.4 – 4, 12

Od uzyskaniu stopnia doktora nauk technicznych w 2011 roku miałem przyjemność promować ponad **50 dyplomantów na poziomie magisterskim oraz inżynierskim/licencjackim** na Politechnice Poznańskiej oraz na University of Gothenburg. Pełniłem także rolę **promotora pomocniczego w trzech zakończonych pomyślnie przewodach doktorskich:**

- S. Kopczyńska, *Supporting Non-functional Requirements Elicitation with Templates*, promotor: J. Nawrocki, promotor pomocniczy: M. Ochodek, Poznań, Polska, 2019.
- B. Alchimowicz, *Automatic generation of user manual for web applications*, promotor: J. Nawrocki, promotor pomocniczy: M. Ochodek, Poznań, Polska, 2015
- J. Jurkiewicz, *Identification of Events in Use Cases*, promotor: J. Nawrocki, promotor pomocniczy: M. Ochodek, Poznań, Polska, 2014

Równolegle z obowiązkami naukowymi i dydaktycznymi wykonywałem prace o charakterze organizacyjnym. W latach 2011-2013 pełniłem role kierownika zadania w projekcie **TECH-INFO** (finansowanego ze środków unijnych) na Politechnice Poznańskiej.[23] W ramach tego projektu byłem odpowiedzialny m.in. za organizację 45 płatnych staży studenckich oraz organizację 12 projektów studenckich (52 studentów) realizowanych wspólnie z firmami wytwarzającymi oprogramowanie.

W latach 2012[24] do 2015 pełniłem role **Sekretarza Komitetu Narodowego ds. Współpracy z Międzynarodową Federacją Przetwarzania Informacji (IFIP)** działającym przy Komitecie Informatyki PAN.

Przez wiele lat byłem także aktywnie zaangażowany w umiędzynarodowienie studiów na Politechnice Poznańskiej. Przez ostatnie dziesięć lat pełniłem rolę członka komisji rekrutacyjnej dla studentów zagranicznych na Wydziale Informatyki i Telekomunikacji (oraz wcześniejszych wydziałach). W latach 2020-2022 pełniłem rolę **Koordynatora Wydziałowego dla programu Erasmus+**.

Od 1 stycznia 2022 objąłem funkcję **Z-cy Dyrektora Instytutu Informatyki** na Wydziale Informatyki i Telekomunikacji Politechniki Poznańskiej.

Ponadto, uczestniczyłem w organizacji kilku konferencji oraz warsztatów (CEE-SET 2007, KKIO 2014, IFIP Doctoral Seminar on Software-Intensive Systems 2015). Byłem także członkiem komitetów programowych kilkunastu konferencji i warsztatów (włączając w to konferencje EASE – CORE Rank A oraz SOFSEM - CORE Rank B). W 2014 roku pełniłem rolę współprzewodniczącego komitetu programowego XVI KKIO Software Engineering Conference. Pełnię także rolę członka komitetu redakcyjnego czasopisma e-Informatica Software Engineering Journal (EISEJ).

Regularnie udzielam się także jako recenzent w uznanych czasopismach naukowych publikujących prace naukowe o tematyce związanej z inżynierią oprogramowania[25] (np.

[23] Tech-Info - http://www.cs.put.poznan.pl/zamawiane

[24] Powołany przez Komitet Informatyki PAN na posiedzeniu 15 marca 2012 roku.

[25] Publons profile - https://publons.com/researcher/1297867/mirosaw-ochodek/peer-review/

Empirical Software Engineering, Information and Software Technology, Journal of Systems and Software, Science of Computer Programming).

Właściwie od początku mojej kariery naukowej podejmowałem także starania mające na celu popularyzację i transfer osiągnięć naukowych w obszarze inżynierii oprogramowania do firm z sektora IT m.in. poprzez organizację warsztatów i szkoleń (m.in. dla Volkswagen Polska sp. z o.o., Bank Zachodni WBK S.A., Comarch S.A., POSTDATA S.A.) oraz publikując artykuły w czasopismach branżowych (np. w magazynie "Industrial monitor - Produkcja i Utrzymanie Ruchu").

V.INFORMACJA O WSPÓŁPRACY Z OTOCZENIEM GOSPODARCZYM

Informacja o współpracy z sektorem gospodarczym

Od początku swojej kariery naukowej współpracuje z firmami w zakresie transferu wiedzy w obszarze inżynierii oprogramowania. W latach 2005-2007 prowadziłem warsztaty dla firm informatycznych z województwa wielkopolskiego w ramach projektu **InMoST** (tematyka związana z szacowaniem pracochłonności i pomiarem rozmiaru oprogramowania). W trakcie kolejnych czterech lat prowadziłem prace badawcze w ramach **Konsorcjum XPrince**[26] (włączając w to realizację projektu Ventures/2008-1/3).

Od 2011 roku **prowadziłem liczne warsztaty świadcząc także usługi konsultacyjne dotyczące metod pomiaru rozmiaru funkcjonalnego oprogramowania, szacowania pracochłonności, a także metodyk zarządzania projektami informatycznymi** dla firm oraz jednostek administracji publicznej (np. Bank Zachodni WBK S.A., Comarch S.A., POSTDATA S.A., Volkswagen Polska sp. z o.o., Ministerstwo Rozwoju Regionalnego, Ministerstwo Finansów). Jestem także współautorem kilku **ekspertyz** dotyczących pomiaru rozmiaru funkcjonalnego i wyceny systemów informatycznych dla Ministerstwa Rozwoju Regionalnego oraz PKP PLK S.A.

Od 2018 roku aktywnie współpracuje z dużymi firmami zrzeszonymi w **Software Center** (https://www.software-center.se). Pełniłem rolę głównego badacza (principal investigator) w ramach projektu „Anomaly Detection in Wireless Networks using Machine Learning" realizowanego wspólne z firmami Ericsson AB, Axis Communication AB oraz Grundfos.

Informacje o wdrożonych technologiach

Oprogramowanie opisane w pkt. II.5 obejmuje dwa rozwiązania informatyczne, które zostały wdrożone is są nadal wykorzystywane: oprogramowanie do wykrywania anomalii w sieciach telefonii komórkowej w firmie Ericsson AB oraz system WOODy, który jest użytkowany na Politechnice Pozańskiej.

[26] XPrince - https://pl.wikipedia.org/wiki/XPrince

Informacja o wykonanych ekspertyzach lub innych opracowaniach wykonanych na zamówienie instytucji publicznych lub przedsiębiorców

- **MindsEater Sp. z o. o.:** M. Ochodek - *Analiza stopnia pokrycia wymagań funkcjonalnych w implementacji systemu IT Project Place (ITPP)*, 2018
- **PKP PLK S.A.**: S. Kopczyńska, M. Ochodek, *Wyceny systemów informatycznych PKP PLK S.A.* przez Izbę Rzeczoznawców Polskiego Towarzystwa Informatycznego, 2015
- **Ministerstwo Rozwoju Regionalnego:** M. Ochodek**,** *Pomiar rozmiaru funkcjonalnego IFPUG FPA oraz pozafunkcjonalnego IFPUG SNAP dla systemu Centralny System Informatyczny SL2014,* 2014
- **Urząd Miasta Poznania**: J. Jurkiewicz, M. Ochodek, B. Walter, *Zasady tworzenia specyfikacji wymagań dla systemów informatycznych wraz z przykładową specyfikacją* – na podstawie analizy Archiwum Dokumentów Elektronicznych w Urzędzie Miasta Poznania, 2009

VI.INFORMACJA O AKTYWNOŚCI NAUKOWEJ

Wykaz opublikowanych rozdziałów w monografiach naukowych[27]

1. M. Staron, W. Meding, O. Söder, **M. Ochodek**. *Improving Quality of Code Review Datasets–Token-Based Feature Extraction Method*. In *International Conference on Software Quality*, pp. 81-93. Springer, Cham., 2021
   MNiSW/MEiN (2019+) - **20 pkt.**
2. K.W. Al-Sabbagh, M. Staron, **M. Ochodek**, R. Hebig, W. Meding. *Selective Regression Testing based on Big Data: Comparing Feature Extraction Techniques*. In *2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pp. 322-329. IEEE, 2020.
   MNiSW/MEiN (2019+) - **20 pkt.**
3. M. Staron, **M. Ochodek**, W. Meding, O. Söder. *Using machine learning to identify code fragments for manual review*. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 513-516. IEEE, 2020.
   MNiSW/MEiN (2019+) - **70 pkt.** (CORE B)
4. A. Sadaj, **M. Ochodek**, S. Kopczyńska, J. Nawrocki, *Maintainability of Automatic Acceptance Tests for Web Applications—A Case Study Comparing Two Approaches to Organizing Code of Test Cases*, In SOFSEM 2020: Theory and Practice of Computer Science: 46th International Conference on Current Trends in Theory and Practice of Informatics, SOFSEM 2020, Limassol, Cyprus, January 20–24, 2020, Proceedings, pp. 454-

[27] W związku z tym, że zasady punktowania rozdziałów w monografiach i publikacji w materiałach konferencyjnych zmieniały się znacząco na przestrzeni ostatnich dziesięciu lat, w zestawieniu uwzględniono jedynie aktualnie obowiązującą zasady punktowania tego typu prac naukowych (MNiSW/MNiE 2019+).

466. Springer, Cham, DOI: 10.1007/978-3-030-38919-2_37, 2020.
MNiSW/MEiN (2019+) - **70 pkt.** (CORE B)

5. **M. Ochodek**, S. Kopczyńska, J. Nawrocki, *A Case Study on a Hybrid Approach to Assessing the Maturity of Requirements Engineering Practices in Agile Projects (REMMA)*, In SOFSEM 2020: Theory and Practice of Computer Science: 46th International Conference on Current Trends in Theory and Practice of Informatics, SOFSEM 2020, Limassol, Cyprus, January 20–24, 2020, Proceedings, pp. 689-698. Springer, Cham, DOI: 10.1007/978-3-030-38919-2_58, 2020.
MNiSW/MEiN (2019+) - **70 pkt.** (CORE B)
6. **M. Ochodek**, M. Staron, W. Meding, *On Identifying Similarities in Git Commit Trends—A Comparison Between Clustering and SimSAX*, In Software Quality: Quality Intelligence in Software and Systems Engineering: 12th International Conference on Software Quality, SWQD 2020, Vienna, Austria, January 14–17, 2020 : Proceedings, pp. 109-120. Springer, Cham, DOI: 10.1007/978-3-030-35510-4_7, 2020.
MNiSW/MEiN (2019+) - **20 pkt.**
7. S. Kopczyńska, **M. Ochodek**, J. Nawrocki, *On importance of non-functional requirements in agile software projects—a survey*, In Integrating Research and Practice in Software Engineering, pp. 145-158. Springer, Cham, DOI: 10.1007/978-3-030-26574-8_11, 2020.
MNiSW/MEiN (2019+) - **20 pkt.**
8. S. Halali, M. Staron, **M. Ochodek**, W. Meding, *Improving Defect Localization by Classifying the Affected Asset Using Machine Learning*, In Software Quality: The Complexity and Challenges of Software Engineering and Software Quality in the Cloud: 11th International Conference on Software Quality, SWQD 2019, Vienna, Austria, January 15–18, 2019, Proceedings, pp. 106-122. Springer, Cham, DOI: 10.1007/978-3-030-05767-1_8, 2019.
MNiSW/MEiN (2019+) - **20 pkt.**
9. S. Kopczyńska, J. Nawrocki, **M. Ochodek**, *When NFR Templates Pay Back? A Study on Evolution of Catalog of NFR Templates*, In Product-Focused Software Process Improvement: 20th International Conference, PROFES 2019, Barcelona, Spain, November 27–29, 2019 : Proceedings, pp. 145-160. Springer, Cham, DOI: 10.1007/978-3-030-35333-9_11, 2019.
MNiSW/MEiN (2019+) - **70 pkt.** (CORE B)

10. **M. Ochodek**, *A Scrum-centric framework for organizing software engineering academic courses*, In Towards a Synergistic Combination of Research and Practice in Software Engineering, pp. 207-220. Springer, Cham, DOI: 10.1007/978-3-319-65208-5_15, 2018.
MNiSW/MEiN (2019+) - **20 pkt.**
11. **M. Ochodek**, K. Koronowski, A. Matysiak, P. Miklosik, S. Kopczyńska, *Sketching use-case scenarios based on use-case goals and patterns*, In Software Engineering: Challenges and Solutions: Results of the XVIII KKIO 2016 Software Engineering Conference 2016

held at September 15-17 2016 in Wroclaw, Poland, pp. 17-30. Springer, Cham, DOI: 10.1007/978-3-319-43606-7_2, 2017.
MNiSW/MEiN (2019+) - **20 pkt.**

12. **M. Ochodek**, M. Staron, D. Bargowski, W. Meding, R. Hebig. *Using machine learning to design a flexible LOC counter*, In 2017 IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE), pp. 14-20. IEEE, DOI: 10.1109/MALTESQUE.2017.7882011, 2017.
MNiSW/MEiN (2019+) - **20 pkt.**

13. **M. Ochodek,** *Approximation of COSMIC functional size of scenario-based requirements in Agile based on syntactic linguistic features—a replication study*, In 2016 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement, pp. 201-211. IEEE, DOI: 10.1109/IWSM-Mensura.2016.8, 2016.
MNiSW/MEiN (2019+) - **20 pkt.** (CORE C)

14. H. Mansoor, **M. Ochodek**, *Towards Semi-Automatic Size Measurement of User Interfaces in Web Applications with IFPUG SNAP*, 2016 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement, pp. 191-194. IEEE, DOI: 10.1109/IWSM-Mensura.2016.7, 2016.
MNiSW/MEiN (2019+) - **20 pkt.** (CORE C)

15. **M. Ochodek**, B. Ozgok, *Functional and Non-functional Size Measurement with IFPUG FPA and SNAP—Case Study*, In Software Engineering in Intelligent Systems: Proceedings of the 4th Computer Science On-line Conference 2015 (CSOC2015), Vol. 3, pp. 19-33, Springer, Cham, DOI: 10.1007/978-3-319-18473-9_3, 2015.
MNiSW/MEiN (2019+) - **20 pkt.**

16. J. Nawrocki, **M. Ochodek**, J. Jurkiewicz, S. Kopczyńska, B. Alchimowicz, *Agile requirements engineering: A research perspective*, In SOFSEM 2014: Theory and Practice of Computer Science: 40th International Conference on Current Trends in Theory and Practice of Computer Science, Nový Smokovec, Slovakia, January 26-29, 2014: Proceedings, pp. 40-51. Springer, Cham, DOI: 10.1007/978-3-319-04298-5_5, 2014.
MNiSW/MEiN (2019+) - **70 pkt.** (CORE B)

17. J. Jurkiewicz, P. Kosiuczenko, L. Madeyski, **M. Ochodek**, C. Orłowski, Ł. Radliński, *Recent Polish achievements in Software Engineering, Software Engineering from Research and Practice Perspective*, Scientific Papers of the Polish Information Processing Society Scientific Council, pp. 15-38, ISBN 978-83-63919-16-0, 2014.
MNiSW/MEiN (2019+) - **5 pkt.**

18. S. Kopczyńska, J. Nawrocki, **M. Ochodek**, *Software development studio—Bringing industrial environment to a classroom*, In 2012 First International Workshop on Software Engineering Education based on Real-World Experiences (EduRex), pp. 13-16. IEEE, DOI:

10.1109/EduRex.2012.6225698, 2012.
MNiSW/MEiN (2019+) - **20 pkt.**

Po uzyskaniu stopnia doktora ↑

19. **M. Ochodek**, B. Alchimowicz, J. Jurkiewicz, J. Nawrocki, *Reliability of transaction identification in use cases*, In FSM '10: Proceedings of the Workshop on Advances in Functional Size, pp. 1-8, ACM, DOI: 10.1145/1921705.1921710, 2010.
MNiSW/MEiN (2019+) - **20 pkt.**
20. B. Alchimowicz, J. Jurkiewicz, J. Nawrocki, **M. Ochodek**, *Towards use-cases benchmark*, In Huzar Z., Koci R., Meyer B., Walter B., Zendulka J. (eds) Software Engineering Techniques. CEE-SET 2008. Lecture Notes in Computer Science, vol. 4980, pp. 20-33. Springer, Berlin, Heidelberg, DOI:10.1007/978-3-642-22386-0_2, 2008.
MNiSW/MEiN (2019+) - **20 pkt.**
21. Ł. Olek, J. Nawrocki, **M. Ochodek**, *Enhancing use cases with screen designs*, In In Huzar Z., Koci R., Meyer B., Walter B., Zendulka J. (eds) Software Engineering Techniques. CEE-SET 2008. Lecture Notes in Computer Science, vol. 4980, pp. 48-61. Springer, Berlin, Heidelberg, DOI:10.1007/978-3-642-22386-0_4, 2008.
MNiSW/MEiN (2019+) - **20 pkt.**
22. B. Michalik, J. Nawrocki, **M. Ochodek**, *3-Step Knowledge Transition: a Case Study on Architecture Evaluation*, In 2008 ACM/IEEE 30th International Conference on Software Engineering, pp. 741-748. IEEE,
DOI: 10.1145/1368088.1368193, 2008.
MNiSW/MEiN (2019+) - **200 pkt.** (CORE A*)
23. **M. Ochodek**, J. Nawrocki, *Automatic transactions identification in use cases*, In Meyer B., Nawrocki J.R., Walter B. (eds) Balancing Agility and Formalism in Software Engineering. CEE-SET 2007. Lecture Notes in Computer Science, vol. 5082, pp. 55-68, Springer, Berlin, Heidelberg, DOI:10.1007/978-3-540-85279-7_5, 2007.
MNiSW/MEiN (2019+) - **20 pkt.**
24. Ł. Olek, B. Michalik, J. Nawrocki, **M. Ochodek**, *Quick prototyping of web applications*, In Software Engineering in Progress, ed. by Madeyski, Lech and Ochodek, Mirosław and Weiss, Dawid and Zendulka, Jaroslav, pp. 124-137, NAKOM. 2007.
MNiSW/MEiN (2019+) - **5 pkt.**
25. J. Nawrocki, T. Nędza, **M. Ochodek**, Ł. Olek, *Describing business processes with use cases*, In Proceedings of the Business Information Systems Conference, ed. by Abramowicz, Witold, vol. P-85, pp. 13-27, Koellen Druck+Verlag. Lecture Notes in Informatics. 2006.
MNiSW/MEiN (2019+) - **5 pkt.**

Informacja o członkostwie w redakcjach naukowych monografii

1. P. Kosiuczenko, L. Madeyski, **M. Ochodek**, A. Paszkiewicz (eds.): *Software Engineering Research for the Practice*, Polish Information Processing Society Scientific Council.

Scientific Papers of the Polish Information Processing Society Scientific Council. (ISBN: 978-83-946253-5-1). 2017.

2. L. Madeyski, **M. Ochodek** (eds.): *Software Engineering from Research and Practice Perspective*, Polish Information Processing Society Scientific Council. Scientific Papers of the Polish Information Processing Society Scientific Council, Nakom (ISBN: 978-83-63919-16-0). 2014. http://zbc.ksiaznica.szczecin.pl/dlibra/docmetadata?id=31905
3. L. Madeyski, **M. Ochodek** (eds.): *Inżynieria oprogramowania: badania i praktyka*, Rada Naukowa Polskiego Towarzystwa Informatycznego. Zeszyty Rady Naukowej Polskiego Towarzystwa Informatycznego, Nakom (ISBN: 978-83-63919-15-3). 2014. http://zbc.ksiaznica.szczecin.pl/dlibra/docmetadata?id=31904

Po uzyskaniu stopnia doktora ↑

4. T. Hruska, L. Madeyski, **M. Ochodek** (eds.): *Software Engineering Techniques in Progress,* Oficyna Wydawnicza Politechniki Wrocławskiej. 2008.
5. L. Madeyski, **M. Ochodek**, D. Weiss, J. Zendulka, (eds.): *Software Engineering in Progress*, Poznan, Poland. Nakom, 2007.

Wykaz opublikowanych artykułów w czasopismach naukowych (z zaznaczeniem pozycji niewymienionych w pkt I)

1. **M. Ochodek**, S. Kopczyńska, M. Staron, *Deep learning model for end-to-end approximation of COSMIC functional size based on use-case names*, Information and Software Technology, vol. 123, 106310, DOI: 10.1016/j.infsof.2020.106310, 2020, ISSN: 0950-5849
   MNiSW (2019) – **140 pkt.,** IF=2,726
2. **M. Ochodek**, M. Staron, W. Meding, J. Bosch, *LegacyPro: A DNA-inspired method for identifying process legacies in software development organizations*, IEEE Software, vol. 37, no. 6, pp. 76-85, DOI: 10.1109/MS.2020.2971894, 2020, ISSN: 0740-7459
   MNiSW (2019) – **100 pkt.,** IF=2,589
3. **M. Ochodek**, R. Hebig, W. Meding, G. Frost, M. Staron, *Recognizing lines of code violating company-specific coding guidelines using machine learning*, Empirical Software Engineering, vol. 25, pp. 220-265, DOI: 10.1007/s10664-019-09769-8, 2020, ISSN: 1382-3256
   MNiSW (2019) – **140 pkt.,** IF=3,156
4. P. Pickerill, H. J. Jungen, **M. Ochodek**, M. Maćkowiak, M. Staron, *PHANTOM: Curating GitHub for engineered software projects using time-series clustering*, Empirical Software Engineering, vol. 25, pp. 2897-2929, DOI: 10.1007/s10664-020-09825-8, 2020, ISSN: 1382-3256
   MNiSW (2019) – **140 pkt.,** IF=3,156
5. **M. Ochodek**, M. Staron, W. Meding, *Simsax: A measure of project similarity based on symbolic approximation method and software defect inflow*, Information and Software Technology, vol. 115, pp. 131-147, DOI: 10.1016/j.infsof.2019.06.003, 2019, ISSN: 0950-

5849
MNiSW (2019) – **140 pkt.,** IF=2,726

6. S. Kopczyńska, J. Nawrocki, **M. Ochodek**, *An empirical study on catalog of non-functional requirement templates: Usefulness and maintenance issues*, Information and Software Technology, vol. 103, pp. 75-91, DOI: 10.1016/j.infsof.2018.06.009, 2018, ISSN: 0950-5849
MNiSW (2013-2016) – **35 pkt.**, MNiSW (2019) – 140 pkt., IF=2,921
7. **M. Ochodek**, S. Kopczyńska, *Perceived importance of agile requirements engineering practices - A survey*, Journal of Systems and Software, vol. 143, pp. 29-43, DOI: 10.1016/j.jss.2018.05.012, 2018, ISSN: 0164-1212
MNiSW (2013-2016) – **35 pkt.**, MNiSW (2019) – 100 pkt., IF=2,559
8. **M. Ochodek**, *Functional size approximation based on use-case names*, Information and Software Technology, vol. 80, pp. 73-88, DOI: 10.1016/j.infsof.2016.08.007, 2016, ISSN: 0950-5849
MNiSW (2013-2016) – **35 pkt.**, MNiSW (2019) – 140 pkt., IF=2.694
9. J. Jurkiewicz, J. Nawrocki, **M. Ochodek**, T. Głowacki, *HAZOP-based identification of events in use cases. An empirical study*, Empirical Software Engineering, vol. 20, pp. 82-109, DOI: 10.1007/s10664-013-9277-5, 2015, ISSN: 1382-3256
MNiSW (2013-2016) – **45 pkt.**, MNiSW (2019) – 140 pkt., IF=1,393

Artykuły w czasopismach naukowych włączone do osiągnięcia naukowego w pkt I ↑

10. L. Madeyski, **M. Ochodek**, *Introduction to the Special Issue on Software Engineering Methods, Tools and Products Improvement and Evaluation*, Foundations of Computing and Decision Sciences, vol. 43(4), pp. 247-250, 2018, ISSN: 0867-6356*, a special-issue introduction paper*
11. M. Maćkowiak, J. Nawrocki, **M. Ochodek**, *On some end-user programming constructs and their understandability*, Journal of Systems and Software, vol. 142, pp. 206-222, DOI: 10.1016/j.jss.2018.03.064, 2018, ISSN: 0164-1212
MNiSW (2013-2016) – **35 pkt.**, MNiSW (2019) – 100 pkt., IF=2,559
12. J. Kowalska, **M. Ochodek**, *Supporting Analogy-based Effort Estimation with the Use of Ontologies*, e-Informatica Software Engineering Journal, vol. 8(1), pp. 53-64, DOI: 10.5277/e-Inf140104, 2014, ISSN: 1897-7979
MNiSW (2013-2016) – **9 pkt.**, MNiSW (2019) – 40 pkt.

Artykuły w czasopismach naukowych po uzyskaniu stopnia doktora ↑

13. **M. Ochodek**, B. Alchimowicz, J. Jurkiewicz, J. Nawrocki, *Improving the reliability of transaction identification in use cases*, Information and Software Technology, vol. 53(8), pp. 885-897, DOI: 10.1016/j.infsof.2011.02.004, 2011, ISSN: 0950-5849
MNiSW (2011) – **30 pkt.**, MNiSW (2019) – 140 pkt., IF=1,250
14. **M. Ochodek**, J. Nawrocki, K. Kwarciak, *Simplifying effort estimation based on Use Case Points*, Information and Software Technology, vol. 53(3), pp. 200-213, ISSN: 0950-5849 DOI: 10.1016/j.infsof.2010.10.005, 2011,
MNiSW (2011) – **30 pkt.**, MNiSW (2019) – 140 pkt., IF=1,250

15. B. Alchimowicz, J. Jurkiewicz, **M. Ochodek**, J. Nawrocki, *Building Benchmarks for Use Cases*, Computing and Informatics, vol. 29(1), pp. 27-44, 2010
    MNiSW (2010) – **13 pkt.**, MNiSW (2019) – 20 pkt., IF=0,356
16. Ł. Olek, **M. Ochodek**, J. Nawrocki, *Enhancing Use Cases with Screen Designs. A Comparison of Two Approaches*, Computing and Informatics, vol. 29(1), pp. 91-106, 2010
    MNiSW (2010) – **13 pkt.,** MNiSW (2019) – 20 pkt., IF=0,356
17. **M. Ochodek**, J. Nawrocki, *Enhancing use-case-based effort estimation with transaction types*, Foundations of Computing and Decision Sciences, vol. 35(2), pp. 91-105, 2010
    MNiSW (2010) – **9 pkt.**, MNiSW (2019) – 20 pkt.

Wykaz osiągnięć projektowych, konstrukcyjnych, technologicznych

- **CCFlex** – Flexible Lines of Code Counter/Classifier – oprogramowanie pozwalające na pomiar rozmiaru fizycznego oprogramowania oraz automatyczną klasyfikację linii kodu źródłowego programów (np. linii niezgodnych ze standardem kodowania). Jestem głównym architektem oraz programistą tego oprogramowania. Dostępne publicznie: https://github.com/mochodek/py-ccflex
- **WOODy** – system informatyczny wspierający organizację i przeprowadzanie egzaminów dyplomowych na Politechnice Poznańskiej (w użyciu od ośmiu lat).
  W pierwszym okresie rozwoju systemu byłem zaangażowany w projekt jako mentor/promotor (projekt realizowany w ramach Studio Rozwoju Oprogramowania na Politechnice Poznańskiej). Następnie przez około pięciu lat prowadziłem dalsze prace rozwojowe i utrzymaniowe tego systemu.
- **Oprogramowanie identyfikujące anomalie w sieciach telefonii komórkowej na bazie sztucznych sieci neuronowych** – pakiet oprogramowania powstały w ramach projektu *Anomaly Detection in Wireless Networks using Machine Learning* (Software Center - https://www.software-center.se). Został on włączony jako element jednego z produktów rozwijanych przez firmę Ericssona AB (pakiet ten jest w dalszym ciągu rozwijany wewnętrznie przez firmę).

Informacja o wystąpieniach na krajowych lub międzynarodowych konferencjach naukowych

- **46th International Conference on Current Trends in Theory and Practice of Informatics (SOFSEM),** 20-24.01.2020, Limassol, Cyprus
  - *Maintainability of Automatic Acceptance Tests for Web Applications—A Case Study Comparing Two Approaches to Organizing Code of Test Cases*
  - *A Case Study on a Hybrid Approach to Assessing the Maturity of Requirements Engineering Practices in Agile Projects (REMMA)*
- **IEEE Workshop on Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE),** 21.02.2017, Klagenfurt, Austria - *Using machine learning to design a flexible loc counter*

- **XIX KKIO Software Engineering Conference**, 14-16.09.2017, Rzeszów, Poland - *A Scrum-centric framework for organizing software engineering academic courses*
- **XVIII KKIO Software Engineering Conference**, 15-17.09.2016, Wrocław, Poland - *Sketching Use-Case Scenarios Based on Use-Case Goals and Patterns* (**Best Presentation Award**)
- **26th International Workshop on Software Measurement (IWSM) and the 11th International Conference on Software Process and Product Measurement (Mensura), IWSM-Mensura 2016**, 5-7.10.2016, Berlin, Germany
    - *Approximation of COSMIC functional size of scenario-based requirements in Agile based on syntactic linguistic features-a replication study*
    - *Towards semi-automatic size measurement of user interfaces in web applications with IFPUG SNAP*
- **4th Computer Science On-line Conference 2015 (CSOC2015)**, 27-30.04.2015, Czech Republic, Czech Republic - *Functional and Non-functional Size Measurement with IFPUG FPA and SNAP - Case Study*
- **XVI KKIO Software Engineering Conference**, 22-24.09.2014, Poznań, Poland - *Supporting Analogy-based Effort Estimation with the Use of Ontologies*
- **1st International Workshop on Software Engineering Education Based on Real-World Experiences (EduRex)** - co-located with ICSE 2012, 9.06.2012, Zurich, Switzerland - *Software development studio—Bringing industrial environment to a classroom*

Po uzyskaniu stopnia doktora ↑

- **Workshop on Advances in Functional Size Measurement and Effort Estimation - co-located with ECOOP 2010**, 21-25.06.2010, Maribor, Slovenia - *Reliability of transaction identification in use cases*
- **3rd IFIP TC 2 Central and East European Conference on Software Engineering Techniques (CEE-SET 2008)**, 13-15.10.2008, Brno, Czech Republic - *Towards use-cases benchmark*
- **2nd IFIP TC 2 Central and East European Conference on Software Engineering Techniques, CEE-SET 2007**, 10-12.10.2007, Poznań, Poland - *Automatic transactions identification in use cases* (**Best Presentation Award**)

Informacja o udziale w komitetach organizacyjnych i naukowych konferencji krajowych lub międzynarodowych

Przewodniczący Komitetu Programowego:

- **KKIO** - Krajowa Konferencja Inżynierii Oprogramowania / KKIO Software Engineering Conference
    - KKIO 2014, 22-24.09.2014, Poznań, Poland

Członek Komitetu Sterującego:

- **KKIO** - Krajowa Konferencja Inżynierii Oprogramowania / KKIO Software Engineering Conference

  - KKIO 2017 – 14-16.09.2017, Rzeszów, Poland
  - KKIO 2016 – 15-17.09.2016, Wrocław, Poland

Członek Komitetu Programowego:

- **EASE** - Evaluation and Assessment in Software Engineering (Emerging Results and Vision) - **CORE Rank A,** MNiSW/MEiN (2019+) - 140 pkt.
  - EASE 2021 – 21-23.06.2021, Trondheim, Norway
  - EASE 2020 – (postponed and merged with EASE 2021 due to COVID-19)
  - EASE 2019 – 15-17.04.2019, Copenhagen, Denmark
- **SOFSEM** - International Conference on Current Trends in Theory and Practice of Computer Science conference - **CORE Rank B,** MNiSW/MEiN (2019+) - 70 pkt.
  - SOFSEM 2020 – 20-24.01.2020, Limassol, Cyprus
  - SOFSEM 2019 – 27-30.01.2019, Nový Smokovec, Slovakia
  - SOFSEM 2018 – 29.01-02.02.2018, Krems an der Donau, Austria
  - SOFSEM 2013 – 26-31.01.2013, Špindlerův Mlýn, Czech Republic
- **MaLTeSQuE** - Machine Learning Techniques for Software Quality Evaluation
  - MaLTeSQuE 2020 – 16.11.2020, Sacramento, USA
  - MaLTeSQuE 2019 – 27.08.2019, Talinn, Estonia
  - MaLTeSQuE 2018 – 20.03.2018, Campobasso, Italy
  - MaLTeSQuE 2017 – 21.02.2017, Klagenfurt, Austria
- **RePa** - International Workshop on Requirements Patterns in conjunction with IEEE International Requirements Engineering Conference
  - RePa 2017 – 4-8.09.2017, Lisbon, Portugal
- **CEE-SET** - IFIP TC2 Central and Eastern European Conference on Software Engineering Techniques
  - CEE-SET 2011 – 25-26.08.2011, Debrecen, Hungary
- **KKIO** - Krajowa Konferencja Inżynierii Oprogramowania / KKIO Software Engineering Conference
  - KKIO 2019 – 11-13.09.2019, Białystok, Poland
  - KKIO 2018 – 27-28.09.2018, Pułtusk, Poland
  - KKIO 2017 – 14-16.09.2017, Rzeszów, Poland
  - KKIO 2016 – 15-17.09.2016, Wrocław, Poland
  - KKIO 2015 – 17-19.09.2015, Międzyzdroje, Poland
- **CSOC** - Computer Science On-line Conference (on-line conference)
  - 2015 – teraz - https://csoc.openpublish.eu
- **CoMeSyso** - Computational Methods in Systems and Software (on-line conference)
  - 2017 – teraz - https://comesyso.openpublish.eu

Informacja o uczestnictwie w pracach zespołów badawczych

- **IGRE** (UDA-POIG.01.03.01-00-132/08-00)- *Opracowanie indeksu gatunkowego i optymalizacji technologii produkcji wybranych roślin energetycznych*, projekt współfinansowany ze środków Europejskiego Funduszu Rozwoju Regionalnego w ramach Programu Operacyjnego Innowacyjna Gospodarka, 2012 (rola: ekspert

w obszarze specyfikacji wymagań dla systemów informatycznych – praca przy systemie BIOPOWER rozwijanym w ramach projektu).

Po uzyskaniu stopnia doktora ↑

- **Ventures/2008-1/3** - *Szacowanie rozmiaru oprogramowania we wstępnych fazach projektów informatycznych*, projekt realizowany w ramach programu Ventures Fundacji na rzecz Nauki Polskiej (FNP) i współfinansowany ze środków Europejskiego Funduszu Rozwoju Regionalnego w ramach Programu Operacyjnego Innowacyjna Gospodarka, 2008-2010 (rola: kierownik projektu)
- **MNiSW N516 001 31/0269** – *Generowanie testów i szacowanie pracochłonności na podstawie przypadków użycia*, 2006-2008 (rola: prowadzenie pac badawczych)

Członkostwo w międzynarodowych lub krajowych organizacjach i towarzystwach naukowych wraz z informacją o pełnionych funkcjach

- **IEEE** - Institute of Electrical and Electronics Engineers, Polish Section (numer legitymacji członkowskiej: #90808971), 2016-2020
- **IFPUG** - International Function Point Users Group (IFPUG) (numer legitymacji członkowskiej: 23160) – 2011-2017

Informacja o odbytych stażach w instytucjach naukowych lub artystycznych, w tym zagranicznych, z podaniem miejsca, terminu, czasu trwania stażu i jego charakteru

- 27.08.2018 – 31.08.2018 (**5 dni**) - **University of Gothenburg** – wizytacja w ramach programu Erasmus+ Staff Mobility for Training (STT)
- 25.01.2018 – 30.06.2018 (**~6 miesięcy**) – **University of Gothenburg** –zatrudnienie na stanowisku starszego wykładowcy oraz udział w pracach badawczych.
- 27.11.2017 – 01.12.2017 (**5 dni**) - **University of Gothenburg** – wizyta naukowa dotycząca wspólnych prac badawczych dot. zastosowania algorytmów uczenia maszynowego do rozwiązywania problemów z obszaru inżynierii oprogramowania.
- 06.03.2017 – 10.03.2017 (**5 dni**) - **University of Gothenburg** – wizyta naukowa dotycząca wspólnych prac badawczych dot. zastosowania algorytmów uczenia maszynowego do rozwiązywania problemów z obszaru inżynierii oprogramowania.

Członkostwo w komitetach redakcyjnych i radach naukowych czasopism wraz z informacją o pełnionych funkcjach (np. redaktora naczelnego, przewodniczącego rady naukowej, itp.)

- 2018 – **redaktor gościnny** dla *Special Issue on Software Engineering Methods, Tools and Products Improvement and Evaluation*, vol. 43(4), **Foundations of Computing and Decision Sciences**, MNiSW/MNiE (2019+) – 20 pkt.
- 2014 – teraz – **Członek rady naukowej** czasopisma **e-Informatica Software Engineering Journal**, Scopus CiteScore (2019) = 3.7, MNiSW/MNiE (2019+) – 40 pkt.

Informacja o recenzowanych pracach naukowych lub artystycznych, w szczególności publikowanych w czasopismach międzynarodowych[28]

- **Information and Software Technology**, Elsevier, (12 recenzje)
- **Journal of Systems and Software**, Elsevier (9 recenzji)
- **IET Software**, IET (5 recenzji)
- **Software Quality Journal**, Springer (4 recenzje)
- **Software and Systems Modeling**, Springer (2 recenzje)
- **IEEE Software**, IEEE (2 recenzje)
- **Science of Computer Programming**, Elsevier (2 recenzje)
- **Empirical Software Engineering**, Springer (2 recenzje)
- **Journal of Software: Evolution and Process**, Wiley (2 recenzje)
- **IEEE Access**, IEEE (2 recenzje)
- **PLOS One**, PLOS (1 recenzja)
- **Computer Science and Information Systems**, ComSIS (1 recenzja)
- **e-Informatica Software Engineering Journal**, Politechnika Wrocławska (12 recenzji)
- **Foundations of Computing and Decision Sciences**, Politechnika Poznańska (3 recenzje) – niezweryfikowane na profilu Publons

Informacja o uczestnictwie w programach europejskich lub innych programach międzynarodowych

Od 2018 roku aktywnie działam w ramach międzynarodowego projektu **Software Center** (https://www.software-center.se) działającego na zasadzie partnerstwa uczelni wyższych oraz dużych firm/koncernów (np. Bosch, Siemens, Ericsson, Scania, Volvo, SAAB).

Brałem udział w następujących projektach współfinansowanych ze środków Unii Europejskiej:

- **TECH-INFO** (UDA-POKL-04.01.02-00-189/10-00) – *Rozwój i doskonalenie kształcenia na Politechnice Poznańskiej w zakresie technologii informatycznych i ich zastosowań w przemyśle – TECH-INFO*, Europejski Fundusz Społeczny, Program Operacyjny Kaptiał Ludzki, 2010-2014
- **IGRE** (UDA-POIG.01.03.01-00-132/08-00) *Opracowanie indeksu gatunkowego i optymalizacji technologii produkcji wybranych roślin energetycznych*, Europejski Fundusz Rozwoju Regionalnego, Program Operacyjny Innowacyjna Gospodarka, 2012

Po uzyskaniu stopnia doktora ↑

- **Ventures/2008-1/3** - *Szacowanie rozmiaru oprogramowania we wstępnych fazach projektów informatycznych*, Fundacji na rzecz Nauki Polskiej (FNP), Europejski Fundusz Rozwoju Regionalnego, Programu Operacyjnego Innowacyjna Gospodarka , 2008-2010

[28] Profil publiczny Publons - https://publons.com/researcher/1297867/mirosaw-ochodek/peer-review/

- **InMoST***: Wielkopolska sieć współpracy w zakresie innowacyjnych metod wytwarzania oprogramowania,* Europejski Fundusz Społeczny, ZPORR, 2005-2007