Zalecenia odnośnie pisania prac magisterskich

Krzysztof Podlaski

1 Uwagi ogólne

- Wszystkie prace muszą spełniać wymagania opisane w regulaminie prac dyplomowych.
- Część pisemna pracy magisterskiej może być tworzona w dowolnej technologii (word, latex, openoffice, ...), proszę o przesyłanie do mnie wersji pdf. Osobiście polecam latex ale jest to tylko sugestia :-).
- W trakcie pracy nad treścią pracy nie poprawiam tekstu, jedynie dodaję komentarze do dokumentu pdf.
- Proszę o założenie na potrzeby projektu repozytorium git/bitbucket, abym miał dostęp do aktualnej wersji kodu programistycznego. Repozytorium może być publiczne lub prywatne, w przypadku prywatnego proszę o dodanie mnie do tego repozytorium.
- Część programistyczna pracy magisterskiej powinna zostać udokumentowana (patrz Sekcja 4).
- Zalecam tworzenie bibliografii w miarę pisania pracy. Podejście "dodam później", niestety stosowane, zwykle wychodzi bokiem.
- To samo dotyczy formatowania, w przypadku korzystania z word bądź openoffice zalecam od samego początku wykorzystywać odpowiednie style, aby można było później łatwo zmieniać format całego dokumentu. Zostawianie tego na sam koniec powoduje często problemy i "rozjeżdżanie się" pracy.

2 Forma

2.1 Język

Praca powinna być napisana poprawnie w języku polskim. Proszę zwrócić uwagę na wszelakie anglicyzmy często występujące w slangu informatycznym. W miarę możliwości stosujmy polskie słowa, jeżeli nie da rady łatwo zastąpić słowa angielskiego, należy tak przeformułować zdanie aby nie była wymagana odmiana słowa, a co za tym idzie konieczności dodawania polskiej końcówki do słowa angielskiego.

Źle: Na potrzeby pracy usługi RESTowe zostały stworzone z wykorzystaniem frameworka Spring.

Poprawnie: Na potrzeby pracy w celu stworzenia usług typu REST wykorzystano framework Spring.

Dodatkowo sugeruję stosowanie twardych spacji w celu unikania wszelkiego rodzaju sierotek, przed ostatecznym wysłaniem pracy do APD proszę o sprawdzenie czy nie występują bękarty, sierotki, wdowy i szewcy [1].

2.2 Rozdziały

Nie ma ogólnych założeń co do podziału logicznego pracy, przykładowy podział pracy na rozdziały poświęcone kolejnym zagadnieniom:

1. Wstęp - powinien zawierać ogólny opis celów pracy
2. Podstawy teoretyczne,
3. Opis metod (algorytmy, założenia, warunki graniczne),
4. Opis technologii wykorzystanych w pracy,
5. Opis implementacji
   - Implementacja części algorytmicznej
   - Dokumentacja projektu jak dla prac inżynierskich (jeśli potrzeba),
6. Opis wykonanych w ramach pracy badań, symulacji, eksperymentów,
7. Analiza otrzymanych wyników,
8. Podsumowanie,
9. Bibliografia.

2.3 Odwołania

2.3.1 Bibliograficzne

W pracy należy stosować jeden z trzech systemów cytowań (Vancouver System, Harvard System, Oxford System) opisanych szczegółowo w [2], regulamin dyplomowania zaleca wykorzystanie systemu Vancouver. W przypadku systemów Vancouver i Oxford bibliografia powinna być posortowana w kolejności pojawiania się odwołań w tekście, natomiast w systemie Harvard alfabetycznie wedle nazwisk pierwszych autorów.

2.3.2 Elementy pływające

Każdy element pływający jak: obrazek, schemat, tabela czy fragment z kodem źródłowym powinien zostać odpowiednio oznaczony (ponumerowany) i zatytułowany. W pracy powinno znaleźć się co najmniej jedno odwołanie do wspomnianego elementu, w innym wypadku sprawia to wrażenie, iż element jest zbędny. Dodatkowo należy pamiętać, iż zasady umieszczania elementów pływających w tekście ograniczają znacząco jaką część strony mogą zawierać, a co za tym idzie omawiana tabelka, obrazek nie muszą znajdować się w bezpośrednim sąsiedztwie tekstu odnoszącego się do obiektu.

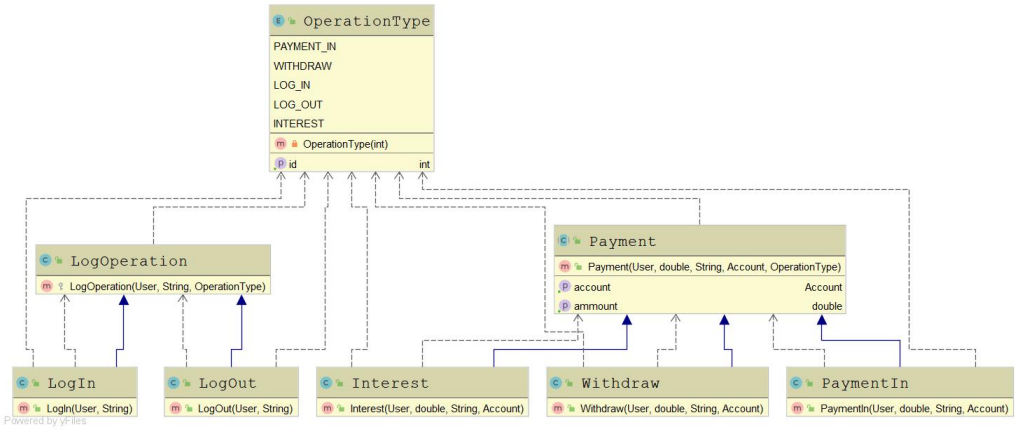
3 Specyfika pracy magisterskiej

Większość z Państwa ma już za sobą pracę inżynierską lub licencjacką więc wiecie w jaki sposób powinna wyglądać praca. Dość istotną kwestią są różnice pomiędzy pracą magisterską a inżynierską/licencjacką. Proszę pamiętać, że praca magisterska jest rodzajem pracy badawczej, a co za tym idzie aspektowi badawczemu powinniście poświęcić maksimum uwagi. Z tego powodu należy zacząć od tak zwanej tezy. Teza pracy jest pewnego rodzaju założeniem wstępnym, którego prawdziwość bądź błędność powinniście wykazać. Dla przykładu teza w postaci "Możliwość efektywnego układania kostki Rubika z wykorzystaniem sieci neuronowych" sugeruje, że w pracy znajdziemy opis próby rozwiązania problemu układania kostki Rubika. W tym celu wykorzystane będą sieci neuronowe. W ramach pracy przebadana zostanie nie tylko możliwość rozwiązania problemu, ale również ocena skuteczności/efektywności zaproponowanej metody. W miarę możliwości proponowane rozwiązanie będzie porównane z innymi podejściami do tego samego problemu. Proszę pamiętać, że kod programistyczny napisany na potrzeby pracy jest tylko narzędziem a nie celem pracy.

Efekty pracy magisterskiej mogą być podmiotem dalszych badań prowadzonych przez pracowników uczelni lub kolejnych dyplomantów. W niektórych wypadkach jest możliwość przygotowania publikacji naukowej opartej o pracę magisterską. Należy dodać, że praca magisterska może być pracą typu pilotażowego. A co za tym idzie, w odróżnieniu do wielu prac naukowych, może w efekcie dać odpowiedź negatywną, to znaczy stwierdzenie: "teza wstępna niestety była niesłuszna i wspomniane w niej podejście nie może stanowić rozwiązania zadanego problemu".

4 Dokumentacja Projektu

Kod programistyczny jest środkiem do uzyskania wyników pracy magisterskiej, nie celem samym w sobie. Z tego powodu większą uwagę należy poświęcić problematyce problemu jak i algorytmicznym problemom podejścia niż kodowi programistycznemu. W pracy magisterskiej powinny zostać zaprezentowane algorytmy działania proponowanego rozwiązania. W tym celu można skorzystać z maszyn stanów (FSM, SDL), algorytmów blokowych, diagramów UML. Sam kod należy jednak również udokumentować.



Rysunek 1: Przykładowy diagram klas w wybranym pakiecie i zależności pomiędzy nimi.

4.1 Dokumentacja kodu

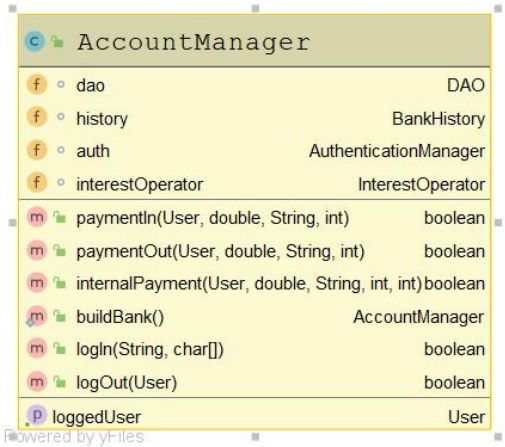
Poprawnym sposobem dokumentacji struktury klas jest diagram klas. Możemy stosować różnego rodzaju diagramy, bardzo ogólny reprezentujący strukturę całej aplikacji Rys. 3, jak i diagramy opisujące poszczególne pakiety

Rys. 1 bądź pojedyncze klasy Rys. 2. Do diagramów klas należy dołączyć bardziej szczegółowy opis klasy, informujący użytkownika za co klasa odpowiada, jakie są jej pola, metody i ich parametry, w tym celu można wykorzystać postać tabelaryczną (Tab. 1) lub przedstawić opis w tekście pracy. W przypadku prac zawierających znaczną ilość klas i ich metod nie wymagam aby wszystkie metody/pola zostały opisane w pracy. W części pisemnej należy opisać jedynie najistotniejsze metody/pola, dodatkowo należy stworzyć dokumentację techniczną klas za pomocą odpowiedniego narzędzia (doxygen, javadoc, ...) i dodać do pracy w postaci załącznika w formie elektronicznej (na płycie CD/DVD).

Nazwa metody	paymentOut
Opis metody	dokonanie przez użytkownika wpłaty na konto bankowe
parametry wejściowe	
user: User	reprezentuje użytkownika dokonującego wpłaty
ammount: double	kwota do wpłacenia
description : String	opis wpłaty
accountId: int	numer id konta na które należy dokonać wpłaty środków
parametry wyjściowe	
typ boolean	czy operacja zakończyła się sukcesem
Uwagi dodatkowe	
możliwe wyjątki	*OperationIsNotAllowedException* wyjątek tworzony w przypadku gdy użytkownik nie ma prawa wypłaty z podanego konta *SQLException* wyjątek tworzony w przypadku kiedy nie ma konta o zadanym accountid
podejmowane działania	W trakcie działania metody wszelkie operacje podlegają logowaniu z wykorzystaniem pola history

Tabela 1: Szczegółowy opis metody paymentOut klasy AccountManager.

Działanie aplikacji, kodu aplikacji, może zostać zaprezentowane z wykorzystaniem diagramów UML [4]: sekwencji (Sequence diagram), aktywności (Activity diagram), stanu (State diagram) bądź schematów blokowych. Oczywiście pomijamy działanie algorytmów gdyż to powinno być opisane w części poświęconej metodzie badawczej a nie implementacji w kodzie. Jeżeli autor



Rysunek 2: Przykładowy diagram opisujący jedną klasę.

pracy uzna, że fragment kodu jest niezbędny, należy zastosować krótki jego fragment (do 10-20 linijek kodu). Proszę o unikanie wklejania do dokumentu fragmentów kodu źródłowego aplikacji. Dodatkowo nie zalecam używania w tym celu zrzutów ekranu z środowiska programistycznego, powoduje to często różne rozmiary czcionek w zależności od rozmiaru wyciętego fragmentu. Dodatkowo kod powinien być kolorowany na białym tle, nie należy stosować motywów typu kolorowy tekst na czarnym tle - wygląda to dużo gorzej na papierze i zużywa mnóstwo tonera, tuszu. Mile widziana jest numeracja linii co ułatwia późniejsze odnoszenie się do przedstawionego kodu źródłowego. Kolorystyka i obramowanie kodu nie musi przypominać tego umieszczonego w dokumencie (Kod źr. 1), jednakże kod źródłowy musi się wyraźnie odróżniać od reszty tekstu pracy.

Kod źródłowy 1: Kod źródłowy obiektu app

```
1  /* Author: Krzysztof Podlaski, University of Lodz */
2  /* based partially on Apache Cordova default app */
3  var app = {
4      initialize: function() {
5          document.addEventListener('deviceready', this.
               onDeviceReady.bind(this), false);
6      },
7      onDeviceReady: function() {
8          this.receivedEvent('deviceready');
9      },
```

```
10      receivedEvent: function(id) {
11          var parentElement = document.getElementById(id);
12          var listeningElement = parentElement.querySelector('.
                listening');
13          var receivedElement = parentElement.querySelector('.
                received');
14          listeningElement.setAttribute('style', 'display:none;')
                ;
15          receivedElement.setAttribute('style', 'display:block;')
                ;
16          console.log('Received␣Event:␣' + id);
17      }
18  };
19  app.initialize();
```

4.2 Dokumentacja baz danych

Praca powinna zawierać informację o strukturze wykorzystywanych baz danych. W przypadku stosowania baz relacyjnych zaleca się postać diagramów opisujących tabele i relacje pomiędzy nimi, w natomiast przypadku baz obiektowych diagramy obiektów.

4.3 Dokumentacja Testów

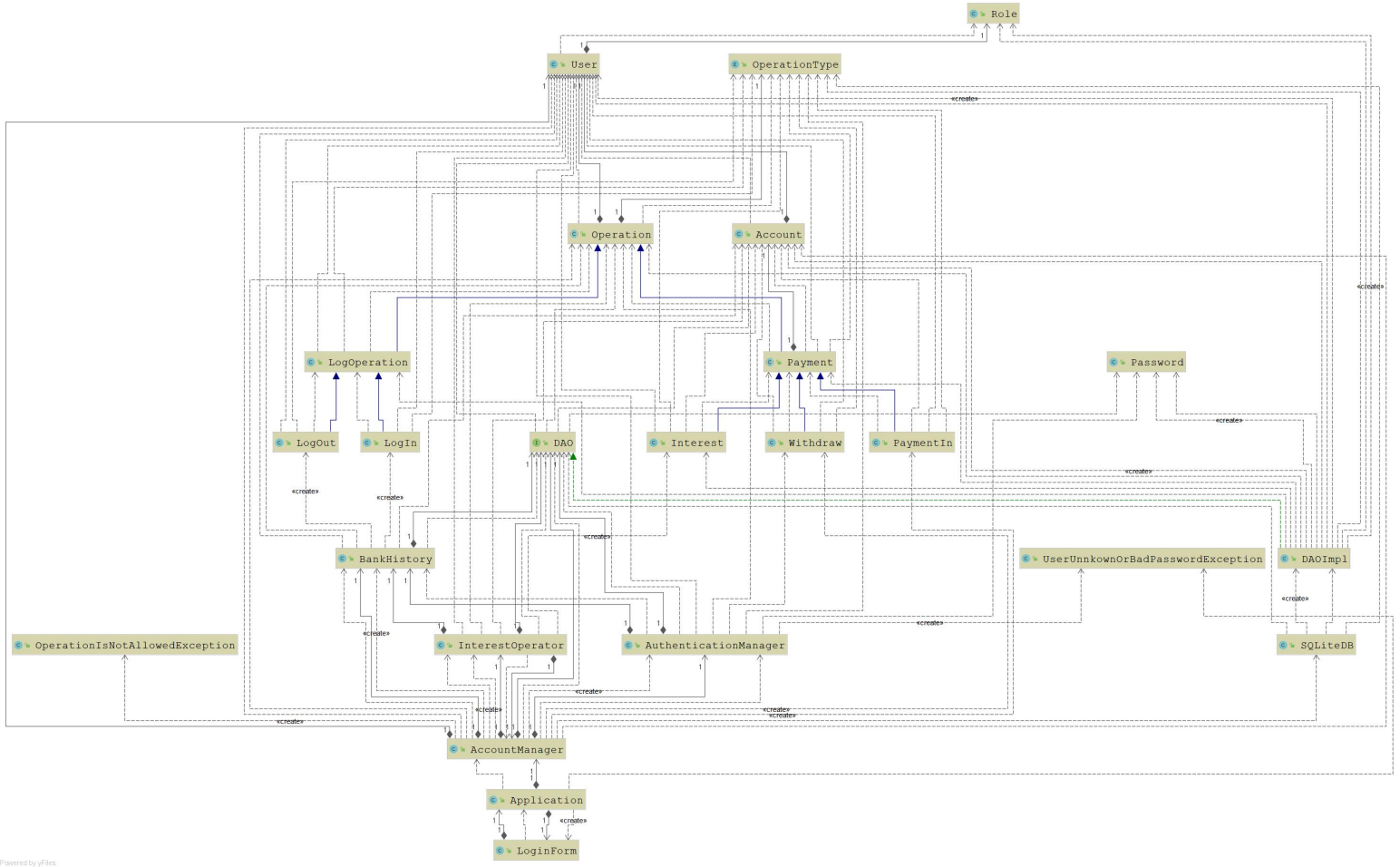
Na potrzeby pracy należy wykonać testy manualne i/lub automatyczne. Mile widziane jest zastosowanie testów jednostkowych, interfejsu użytkownika czy akceptacyjnych. Testy należy opisać w postaci: wyszczególnionych przypadków testowych, każdy przypadek testowy powinien zawierać opis sposobu wykonania, kroków niezbędnych do jego przeprowadzenia.

5 Egzamin dyplomowy – obrona

Komisja egzaminacyjna składa się z trzech osób, przewodniczącego, recenzenta oraz opiekuna pracy. Egzamin składa się z dwu elementów prezentacji efektów pracy dyplomowej oraz faktycznego egzaminu.

5.1 Prezentacja efektów pracy dyplomowej

Dyplomant/dyplomantka jest zwykle proszony o prezentacje efektów pracy dyplomowej, w przypadku pracy magisterskiej są to zwykle efekty części ba-



Rysunek 3: Przykładowy ogólny diagram klas dla całej aplikacji.

dawczej. Student/studentka ma za zadanie w trakcie 5-10 minut zaprezentować najważniejsze elementy pracy.

- Część prezentacyjna może być wsparta prezentacją (ppt, pdf), zalecam taką formę wspomaga ona studenta/studentkę i zapobiega zgubieniu się pod wpływem stresu. W ramach potrzeb można przygotować krótki film wizualizujący działanie metody, prezentacje na żywo zwiększają ryzyko problemów technicznych.

- Przedstawiana praca dyplomowa jest z zakresu informatyki proszę więc skupić się na elementach informatycznych pracy – ale nie chodzi tu o elementy techniczne a fundamentalne jak stosowane algorytmy, struktury danych/klas natomiast nie jaka instrukcja pomogła w wykonaniu np. sortowania.

- W trakcie prezentacji proszę unikać sformułowań czysto slangowych bez wyjaśnienia "JSON", "AJAX", "Spring Beans". Pomimo iż jesteśmy informatykami proszę próbować prezentować z założeniem średniej specjalizacji słuchaczy w dziedzinie pracy dyplomowej.

5.2 Część egzaminacyjna

Student/studentka dostaje trzy pytania, po jednym od każdego z członków komisji egzaminacyjnej. Przewidywany jest czas na przygotowanie do odpowiedzi, nie należy jednak z tym przesadzać normą jest max 5-7 minut przygotowania. Po zakończeniu przygotowania dyplomant/dyplomantka odpowiada na pytania w dowolnej kolejności.

W trakcie, jak i po zakończeniu, odpowiedzi na pytanie członkowie komisji mogą zadawać dodatkowe pytania uszczegóławiające lub wspomagające. Student przechodzi do następnego pytania po jawnej informacji, że komisja nie ma nic do dodania do aktualnego pytania.

Literatura

[1] Strony Wikipedii poświęcone błędom w łamaniu tekstu
`https://pl.wikipedia.org/wiki/B%C4%99kart_(typografia)`

[2] Strona Wikipedii poświęcona metodom cytowania
`https://pl.wikipedia.org/wiki/Cytowanie_pi%C5%9Bmiennictwa`

[3] Ian Sommerville, Software Engineering, 10h ed., Pearsons (2016)
wydanie polskie Ian Sommerville, Inżynieria Oprogramowania, WNT (2003)

[4] Pascal Roques, UML in Practice: The Art of Modeling Software Systems Demonstrated through Worked Examples and Solutions, John Wiley & Sons, 2004
Przykładowy rozdział książki dotyczący wybranych diagramów UML i przypadków użycia `https://people.ok.ubc.ca/bowenhui/310/8-UML.pdf`

[5] Strona domowa projektu PlantUML `https://plantuml.com`