Wydział Podstawowych Problemów Techniki
Politechnika Wrocławska

Wybrane problemy pisania pracy dyplomowej

Imię i nazwisko
Nr indeksu: 123456

Praca magisterska napisana
pod kierunkiem
XXX YYY ZZZ

Politechnika
Wrocławska
Wrocław 2016

Spis treści

Wstęp

Praca dyplomowa inżynierska jest dokumentem opisującym zrealizowany system techniczny. Praca powinna być napisana poprawnym językiem odzwierciedlającym aspekty techniczne (informatyczne) omawianego zagadnienia. Praca powinna być napisana w trybie bezosobowym (w szczególności należy unikać trybu pierwszej osoby liczby pojedynczej i mnogiej). Zdania opisujące konstrukcję systemu informatycznego powinny być tworzone w stronie biernej. W poniższym dokumencie przykłady sformułowań oznaczono kolorem niebieskim. W opisie elementów systemu, komponentów, elementów kodów źródłowych, nazw plików, wejść i wyjść konsoli należy stosować czcionkę stałej szerokości, np: zmienna `wynik` przyjmuje wartość zwracaną przez funkcję `dodaj(a,b)`, dla argumentów `a` oraz `b` przekazywanych ....

Układ poniższego dokumentu przedstawia wymaganą strukturę pracy, z rozdziałami zawierającymi analizę zagadnienia, opis projektu systemu oraz implementację (dobór podrozdziałów jest przykładowy i należy go dostosować do własnej tematyki pracy).

Wstęp pracy (nie numerowany) powinien składać się z czterech części (które nie są wydzielane jako osobne podrozdziały): zakresu pracy, celu, analizy i porównania istniejących rozwiązań oraz przeglądu literatury, oraz opisu zawartości pracy.

Każdy rozdział powinien rozpoczynać się od akapitu wprowadzającego, w którym zostaje w skrócie omówiona zawartość tego rozdziału.

Praca swoim zakresem obejmuje wielowarstwowe rozproszone systemy informatyczne typu „B2B" wspierające wymianę danych pomiędzy przedsiębiorstwami. Systemy tego typu, konstruowane dla dużych korporacji, charakteryzują się złożoną strukturą poziomą i pionową, w której dokumenty ...

Celem pracy jest zaprojektowanie i oprogramowanie aplikacji o następujących założeniach funkcjonalnych:

- wspieranie zarządzania obiegiem dokumentów wewnątrz korporacji z uwzględnieniem ...,
- wspieranie zarządzania zasobami ludzkimi z uwzględnieniem modułów kadrowych oraz ...,
- gotowość do uzyskania certyfikatu ISO ...,
- ....

Istnieje szereg aplikacji o zbliżonej funkcjonalności: ..., przy czym ....

Praca składa się z czterech rozdziałów. W rozdziale pierwszym omówiono strukturę przedsiębiorstwa ..., scharakteryzowano grupy użytkowników oraz przedstawiono procedury związane z obiegiem dokumentów. Szczegółowo opisano mechanizmy .... Przedstawiono uwarunkowania prawne udostępniania informacji ..., w ramach ....

W rozdziale drugim przedstawiono szczegółowy projekt systemy w notacji UML. Wykorzystano diagramy .... Opisano w pseudokodzie i omówiono algorytmy generowania ....

W rozdziale trzecim opisano technologie implementacji projektu: wybrany język programowania, biblioteki, system zarządzania bazą danych, itp. Przedstawiono dokumentację techniczną kodów źródłowych interfejsów poszczególnych modułów systemu. Przedstawiono sygnatury metod publicznych oraz ....

W rozdziale czwartym przedstawiono sposób instalacji i wdrożenia systemu w środowisku docelowym.

Końcowy rozdział stanowi podsumowanie uzyskanych wyników.

Analiza problemu

Analiza problemu ——————————— W tym rozdziale należy przedstawić analizę zagadnienia, które podlega informatyzacji. Należy zidenty- fikować i opisać obiekty składowe rozważanego wycinka rzeczywistości i ich wzajemne relacje (np. użytkown- ików systemu i ich role). Należy szczegółowo omówić procesy jakie zachodzą w systemie i które będą infor- matyzowane, takie jak np. przepływ dokumentów. Należy sprecyzować i wypunktować założenia funkcjon- alne i poza funkcjonalne dla projektowanego systemu. Jeśli istnieją aplikacje realizujące dowolny podzbiór zadanych funkcjonalności realizowanego systemu należy przeprowadzić ich analizę porównawczą, wskazując na różnice bądź innowacyjne elementy, które projektowany w pracy system informatyczny będzie zawierał. Należy odnieść się do uwarunkowań prawnych związanych z procesami przetwarzania danych w projektowanym sys- temie. Jeśli zachodzi konieczność, należy wprowadzić i omówić model matematyczny elementów systemu na odpowiednim poziomie abstrakcji. W niniejszym rozdziale omówiono koncepcję architektury programowej systemu .... W szczególny sposób .... Omówiono założenia funkcjonalne i niefunkcjonalne podsystemów .... Przedstawiono mechanizmy .... Sklasyfikowano systemy ze względu na .... Omówiono istniejące rozwiązania informatyczne o podobnej funkcjonalności ... (zobacz [1]).

-Omowienie sposobow redundancji danych -RAID -Istniejace rozwiazania -Wybrane rozwiazania -Problem rozbijania danych -Problem wielu replik -Problem synchronizacji -Problem obsługi pełnej funkcjon- alności, np. softlinks Założenia: -Rodzaj wykorzystywanego sposobu redundacji niewidoczny dla uzytkownika -Uzytkownik nie martwi sie replikami poza wyjatkowymi sytuacjami: synchronizacja replik przy zamontowaniu, zatwierdzenie korekty

- Obsługa kilku różnych rozwiązań redundancji
- Obsługa wielu replik dla większego bezpieczeństwa danych
- Wybór najodpowiedniejszej repliki do odczytu danych
- Synchronizacja istniejących danych pomiędzy replikami przy zamontowaniu systemu plików
- Obsługa przynajmniej podstawowych zachowań systemu plików:
  - Czytanie oraz pisanie do plików
  - Tworzenie oraz usuwanie plików
- Detekcja błędów:
  - Sprawdzanie, czy pliki, na których dokonywane są operacje, nie są uszkodzone
  - Sprawdzanie synchronizacji danych między replikami
- Korekcja błędów:
  - Zastosowanie kodów korekcyjnych
  - Całkowite zastępowanie uszkodzonych danych replikami

Projekt systemu

Dla uproszczenia tylko pliki tekstowe, wiec raczej operacje na bajtach W tym rozdziale przedstawiono szczegółowy projekt systemy w notacji UML uwzględniający wymagania funkcjonalne opisane w rozdziale 2. Do opisu relacji pomiędzy składowymi systemu wykorzystano diagramy .... Przedstawiono w pseudokodzie i omówiono algorytmy generowania ....

3.1 Grupy użytkowników i założenia

Architektura systemu ...jest wielowarstwowa i rozproszona, przy czym .... Podsystem ...jest systemem zbiorczym dla danych ...wysyłanych do serwera ....

Taka architektura jest zgodna z wzorcem projektowym MVC[1] (ang. Model-View-Controller). Przetwarzanie danych odbywa się ....

3.2 Przypadki użycia i scenariusze

3.2.1 Montowanie systemu plików

S01	Montowanie systemu plików
Aktor	Użytkownik
Warunki wsępne	Aktor przygotował konfigurację systemu
Przebieg wydarzeń	• Aktor montuje system, podając plik konfiguracyjny • System odczytuje konfigurację • System dostosowuje parametry i montuje system plików
Alternatywny przebieg wydarzeń	• Aktor konfiguruje system przez argumenty wywołania
Sytuacje wyjątkowe	• Aktor podał niepoprawną konfigurację • Katalogi podane przez aktora nie istnieją lub są niepoprawne
Warunki końcowe	System plików jest poprawnie zamontowany pod podanym katalogiem

Tablica 3.1: Montowanie systemu plików

[1] Należy odnieść się do wykorzystywanych wzorców projektowych

3.2.2 Odczyt pliku

S02.1	Odczyt pliku przy wykorzystaniu pojedynczej repliki korekcyjnej
Aktor	Użytkownik
Warunki wsępne	System plików jest zamontowany, replika wspiera korekcję błędów
Przebieg wydarzeń	1. Aktor podaje plik do odczytu 2. System sprawdza spójność pliku w replice 3. System dokonuje korekty istniejących uszkodzeń pliku przed odczytem danych 4. Aktor dostaje odczytane dane
Alternatywny przebieg wydarzeń	3. System nie może dokonać korekty błędów 4. Aktor zostaje poinformowany o błędzie przez 5. Aktor dostaje odczytane dane
Sytuacje wyjątkowe	• Dany plik nie istnieje w replice • Nieudany odczyt pliku • Dysk z repliką został odmontowany lub uszkodzony
Warunki końcowe	System zwrócił aktorowi odczytane dane

Tablica 3.2: Odczyt pliku przy wykorzystaniu pojedynczej repliki korekcyjnej

S02.2	Odczyt pliku przy wykorzystaniu pojedynczej repliki bez korekty danych
Aktor	Użytkownik
Warunki wsępne	System plików jest zamontowany, replika nie wspiera korekty błędów
Przebieg wydarzeń	1. Aktor podaje plik do odczytu 2. System sprawdza spójność pliku w replice 3. System informuje o znalezionych błędach 4. Aktor dostaje odczytane dane
Alternatywny przebieg wydarzeń	Brak
Sytuacje wyjątkowe	• Dany plik nie istnieje w replice • Błąd oczytu • Dysk z repliką został odmontowany lub uszkodzony
Warunki końcowe	System zwrócił aktorowi odczytane dane

Tablica 3.3: Odczyt pliku przy wykorzystaniu pojedynczej repliki bez korekty

S02.1	Odczyt pliku przy wykorzystaniu pojedynczej repliki korekcyjnej
Aktor	Użytkownik
Warunki wsępne	System plików jest zamontowany, replika wspiera korekcję błędów
Przebieg wydarzeń	1. Aktor podaje plik do odczytu 2. System sprawdza spójność pliku w replice 3. System dokonuje korekty istniejących uszkodzeń pliku przed odczytem danych 4. Aktor dostaje odczytane dane
Alternatywny przebieg wydarzeń	3. System nie może dokonać korekty błędów 4. Aktor zostaje poinformowany o błędzie przez 5. Aktor dostaje odczytane dane
Sytuacje wyjątkowe	• Dany plik nie istnieje w replice • Nieudany odczyt pliku • Dysk z repliką został odmontowany lub uszkodzony
Warunki końcowe	System zwrócił aktorowi odczytane dane

Tablica 3.4: Odczyt pliku przy wykorzystaniu pojedynczej repliki korekcyjnej

S02.2	Odczyt pliku przy wykorzystaniu pojedynczej repliki bez korekty danych
Aktor	Użytkownik
Warunki wsępne	System plików jest zamontowany, replika nie wspiera korekty błędów
Przebieg wydarzeń	1. Aktor podaje plik do odczytu 2. System sprawdza spójność pliku w replice 3. System informuje o znalezionych błędach 4. Aktor dostaje odczytane dane
Alternatywny przebieg wydarzeń	Brak
Sytuacje wyjątkowe	• Dany plik nie istnieje w replice • Błąd oczytu • Dysk z repliką został odmontowany lub uszkodzony
Warunki końcowe	System zwrócił aktorowi odczytane dane

Tablica 3.5: Odczyt pliku przy wykorzystaniu pojedynczej repliki bez korekty

3.2.3 Zapis pliku

S02.1	Odczyt pliku przy wykorzystaniu pojedynczej repliki korekcyjnej
Aktor	Użytkownik
Warunki wsępne	System plików jest zamontowany, replika wspiera korekcję błędów
Przebieg wydarzeń	1. Aktor podaje plik do odczytu 2. System sprawdza spójność pliku w replice 3. System dokonuje korekty istniejących uszkodzeń pliku przed odczytem danych 4. Aktor dostaje odczytane dane
Alternatywny przebieg wydarzeń	3. System nie może dokonać korekty błędów 4. Aktor zostaje poinformowany o błędzie przez 5. Aktor dostaje odczytane dane
Sytuacje wyjątkowe	• Dany plik nie istnieje w replice • Nieudany odczyt pliku • Dysk z repliką został odmontowany lub uszkodzony
Warunki końcowe	System zwrócił aktorowi odczytane dane

Tablica 3.6: Odczyt pliku przy wykorzystaniu pojedynczej repliki korekcyjnej

S02.2	Odczyt pliku przy wykorzystaniu pojedynczej repliki bez korekty danych
Aktor	Użytkownik
Warunki wsępne	System plików jest zamontowany, replika nie wspiera korekty błędów
Przebieg wydarzeń	1. Aktor podaje plik do odczytu 2. System sprawdza spójność pliku w replice 3. System informuje o znalezionych błędach 4. Aktor dostaje odczytane dane
Alternatywny przebieg wydarzeń	Brak
Sytuacje wyjątkowe	• Dany plik nie istnieje w replice • Błąd oczytu • Dysk z repliką został odmontowany lub uszkodzony
Warunki końcowe	System zwrócił aktorowi odczytane dane

Tablica 3.7: Odczyt pliku przy wykorzystaniu pojedynczej repliki bez korekty

3.3 Diagramy klas

W tej sekcji należy przedstawić diagramy klas dla odpowiednich elementów systemu zidentyfikowane na podstawie wcześniejszych rozważań

3.4 Diagramy sekwencji

W tej sekcji należy przedstawić diagramy sekwencji dla obiektów systemu zidentyfikowanych na podstawie wcześniejszych rozważań. Należy wykorzystać nazewnictwo wprowadzone w poprzednich rozdziałach, w szczególności odpowiadające definicjom wprowadzonych klas.

3.5 Diagramy stanów

W tej sekcji należy przedstawić diagramy stanów w których może znaleźć się system. Diagramy te są szczególnie istotne przy projektowaniu systemów czasu rzeczywistego.

3.6 Projekt bazy danych

W tej sekcji należy przedstawić projekt bazy danych. Należy omówić wycinek rzeczywistości i odpowiadające mu zidentyfikowane elementy systemu, których wartości będą podlegać utrwalaniu. Należy przedyskutować wybór typów danych dla atrybutów poszczególnych obiektów. Należy uzasadnić wybór platformy DBMS. Dla relacyjnych baz danych należy przedyskutować jej normalizację.

3.7 Opis protokołów

W tej sekcji należy omówić protokoły wykorzystywane przez komponenty systemu. Omówić formaty komunikatów i zilustrować je przykładami.

3.8 Opis algorytmów

W tej sekcji należy wymienić i przedyskutować algorytmy wykorzystywane w systemie. Algorytmy należy przedstawić w pseudokodzie (wykorzystać pakiet `algorithm2e`). Omówienia poszczególnych kroków algorytmów powinny zawierać odwołania do odpowiednich linii pseudokodu. Dla zaproponowanych autorskich algorytmów należy przeprowadzić analizę ich złożoności czasowej i pamięciowej.

Algorytm bąblowania jest przedstawiony w Pseudokodzie 3.1.

Pseudokod 3.1: Wyporność przez bąblowanie

```
   Input: Zbiór bąbli B
   Output: Wyporność W
1  foreach b ∈ B do
2      Process(b);
3      for i ← 1 to |B| do
4          if Calculate(EW(i,b)) ⩽ 0 then
5              b ← 2 ∗ b;
6  while B ≠ ∅ do
7      for j ← 1 to |B| do
8          if Calculate(FT(j,b̂)) ⩽ 0 then
9              w ← 2 ∗ b̂;
10             W ← W ∪ {w};
11             B ← B \ {b};
```

Implementacja systemu

W niniejszym rozdziale przedstawiono szczegóły implementacyjne zaproponowanych rozwiązań. Kompletny kod źródłowy wraz z komentarzami znajduje się w załączniku do pracy.

4.1 Opis technologii

System plików implementujący całą funkcjonalność przedstawionego systemu wykorzystuje Filesystem in Userspace. Interfejs FUSE umożliwia tworzenie systemów plików w przestrzeni użytkownika. Bez schodzenia do poziomu kodu jądra systemu, interfejs ten pozwala skupić się na żądanej funkcjonalności.

FUSE jest wykorzystywany do tworzenia wirtualnych systemów plików (dalej nazywanych VFS). Niniejszy projekt jest klasyfikowany jako VFS, co ma znaczenie w dalszej części pracy.

4.2 Podział systemu na moduły

System plików został zaimplementowany z myślą i dalszym rozwoju projektu. Każda istniejąca replika może być zamontowana jako osobny system plików, główny moduł stanowi połączenie między tymi replikami. Dzięki temu rozwiązaniu, działanie całego systemu jest niezależne od zastosowanych rozwiązań redundancji.

obrazki

4.3 Omówienie kodów źródłowych

Kod źródłowy 4.1 przedstawia opisy poszczególnych metod interfejsu: `WSPodmiotRejestracjaIF`. Kompletne kody źródłowe znajdują się na płycie CD dołączonej do niniejszej pracy w katalogu `Kody` (patrz Dodatek A).

Kod źródłowy 4.1: Interfejs usługi Web Service: `WSPodmiotRejestracjaIF`.

```
package erejestracja.podmiot;
import java.rmi.RemoteException;
// Interfejs web serwisu dotyczącego obsługi podmiotów i rejestracji.
public interface WSPodmiotRejestracjaIF extends java.rmi.Remote{
// Pokazuje informacje o danym podmiocie.
// parametr: nrPeselRegon - numer PESEL podmiotu lub numer REGON firmy.
// return: Podmiot - obiekt transportowy: informacje o danym podmiocie.
public Podmiot pokazPodmiot(long nrPeselRegon) throws RemoteException;
// Dodaje nowy podmiot.
// parametr: nowyPodmiot - obiekt transportowy: informacje o nowym podmiocie.
// return: true - jeśli podmiot dodano, false - jeśli nie dodano.
public boolean dodajPodmiot(Podmiot nowyPodmiot) throws RemoteException;
// Usuwa dany podmiot.
// parametr: nrPeselRegon - numer PESEL osoby fizycznej lub numer REGON firmy.
// return: true - jeśli podmiot usunięto, false - jeśli nie usunięto.
public boolean usunPodmiot(long nrPeselRegon) throws RemoteException;
// Modyfikuje dany podmiot.
```

```
// parametr: podmiot - obiekt transportowy: informacje o modyfikowanym podmiocie.
// return: true - jeśli podmiot zmodyfikowano, false - jeśli nie zmodyfikowano.
public boolean modyfikujPodmiot(Podmiot podmiot) throws RemoteException;
// Pokazuje zarejestrowane podmioty na dany dowód rejestracyjny.
// parametr: nrDowoduRejestracyjnego - numer dowodu rejestracyjnego.
// return: PodmiotRejestracja[] - tablica obiektów transportowych: informacje o
// wszystkich zarejestrowanych podmiotach.
public PodmiotRejestracja[] pokazZarejestrowanePodmioty(
String nrDowoduRejestracyjnego) throws RemoteException;
// Nowa rejestracja podmiotu na dany dowód rejestracyjny.
// parametr: nrDowoduRejestracyjnego - numer dowodu rejestracyjnego.
// parametr: nrPeselRegon - numer PESEL podmiotu lub numer REGON firmy.
// parametr: czyWlasciciel - czy dany podmiot jest właścicielem pojazdu.
// return: true - jeśli zarejestrowano podmiot, false - jeśli nie zarejestrowano.
public boolean zarejestrujNowyPodmiot(String nrDowoduRejestracyjnego,
long nrPeselRegon, boolean czyWlasciciel) throws RemoteException;
// Usuwa wiązanie pomiędzy danym podmiotem, a dowodem rejestracyjnym.
// parametr: nrDowoduRejestracyjnego - numer dowodu rejestracyjnego.
// parametr: nrPeselRegon - numer PESEL podmiotu lub numer REGON firmy.
// return: true - jeśli podmiot wyrejestrowano, false - jeśli nie wyrejestrowano.
public boolean wyrejestrujPodmiot(String nrDowoduRejestracyjnego,
long nrPeselRegon) throws RemoteException;
```

Kod źródłowy 4.2 przedstawia procedurę przetwarzającą żądanie. Hasz utrwalany `%granulacja` wykorzystywany jest do komunikacji międzyprocesowej.

Kod źródłowy 4.2: Przetwarzanie żądania - procedura `process_req()`.

```
sub process_req(){
  my($r) = @_;
  $wyn = "";
  if ($r=~/get/i) {
        @reqest = split("_",$r);
        $zad = $reqest[0];
        $ts1 = $reqest[1];
        $ts2 = $reqest[2];
        @date1 = split(/\D/,$ts1);
        @date2 = split(/\D/,$ts2);
        print "odebralem:_$r";
        $wyn = $wyn."zadanie:_$zad\n";
        $wyn = $wyn."czas_od:_"."$date1[0]"."-"."$date1[1]"."-"."$date1[2]"."_"."$date1[3]".":"."$
        $wyn = $wyn."czas_do:_"."$date2[0]"."-"."$date2[1]"."-"."$date2[2]"."_"."$date2[3]".":"."$
        $wyn = $wyn.&sym_sens($ts1,$ts2);
        return $wyn;
  }
  if ($r=~/set gt/i) {
        @reqest = split("_",$r);
        $zad = $reqest[0];
        $ts1 = $reqest[1];
        $ts2 = $reqest[2];
        $gt = $reqest[2];
        dbmopen(%granulacja,"granulacja_baza",0644);
        $granulacja{"gt"}=$gt;
        dbmclose(%granulacja);
        $wyn = "\'GT\'_zmienione_na:_$gt";
  }
}
```

Instalacja i wdrożenie

W tym rozdziale należy omówić zawartość pakietu instalacyjnego oraz założenia co do środowiska, w którym realizowany system będzie instalowany. Należy przedstawić procedurę instalacji i wdrożenia systemu. Czynności instalacyjne powinny być szczegółowo rozpisane na kroki. Procedura wdrożenia powinna obejmować konfigurację platformy sprzętowej, OS (np. konfiguracje niezbędnych sterowników) oraz konfigurację wdrażanego systemu, m.in. tworzenia niezbędnych kont użytkowników. Procedura instalacji powinna prowadzić od stanu, w którym nie są zainstalowane żadne składniki systemu, do stanu w którym system jest gotowy do pracy i oczekuje na akcje typowego użytkownika.

Podsumowanie

W podsumowanie należy określić stan zakończonych prac projektowych i implementacyjnych. Zaznaczyć, które z zakładanych funkcjonalności systemu udało się zrealizować. Omówić aspekty pielęgnacji systemu w środowisku wdrożeniowym. Wskazać dalsze możliwe kierunki rozwoju systemu, np. dodawanie nowych komponentów realizujących nowe funkcje.

W podsumowaniu należy podkreślić nowatorskie rozwiązania zastosowane w projekcie i implementacji (niebanalne algorytmy, nowe technologie, itp.).

Bibliografia

[1] J. Cichoń, M. Klonowski, Łukasz Krzywiecki, B. Różański, P. Zieliński. On the number of nodes and their distribution in chord. 2006.

Zawartość płyty CD

W tym rozdziale należy krótko omówić zawartość dołączonej płyty CD.