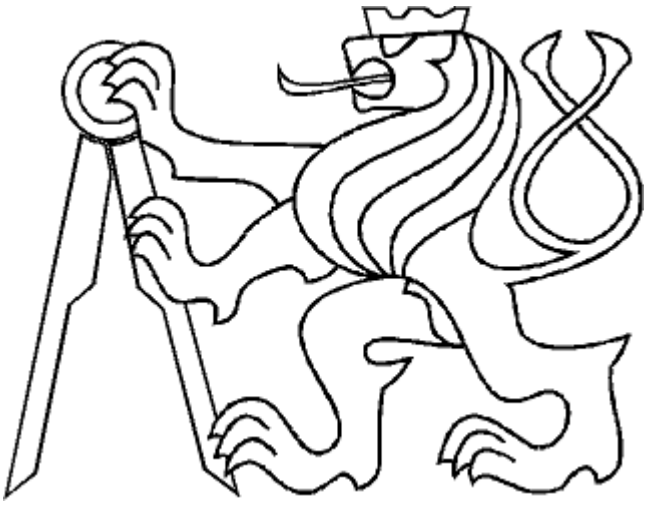
České vysoké učení technické v Praze
Fakulta elektrotechnická



DIPLOMOVÁ PRÁCE

Systém pro digitální předávku práce

Praha, 2014 **Autor: Bc. Armen Hajrapetjan**

České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra řídicí techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Armen Hajrapetjan**

Studijní program: Otevřená informatika (magisterský)
Obor: Počítačové inženýrství

Název tématu: **Systém pro digitalní předávku práce**

Pokyny pro vypracování:

Navrhněte distribuovaný databázový systém pro digitální předávku práce, který bude obsahovat:

* Databázovou část v technologii MS SQL.
* Služby založené na technologii WCF (Windows Communication Foundation).
* Navázaní služeb k databázi pomoci entity framework.
* Relační databází s možností fulltextového vyhledávaní.
* Zabezpečení služeb proti neoprávněným osobám.
* Návrhové vzory.
* Klienta založeného na technologii WPF (Windows Presentation Foundation).
* Testy pro testovaní aplikace.
* Manuál k systému.

Seznam odborné literatury:

[1] Charles Petzold: Mistrovství ve Windows Presentation Foundation. COMPUTER PRESS 2008. 928 s.
[2] Pablo Cibraro, Kurt Claeys, et al.: Professional WCF 4: Windows Communication Foundation with .NET 4. Wiley Publishing, Inc. 2010.

Vedoucí: Ing. Michal Štepanovský, Ph.D.

Platnost zadání: do konce letního semestru 2013/2014

L.S.

prof. Ing. Michael Šebek, DrSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 21. 1. 2013

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 8. 5. 2014

podpis

Poděkování

Děkuji především vedoucímu diplomové práce za jeho podporu a rady, které mi pomohly při vypracování této práce.

Abstrakt

Řešení prezentované v této práci popisuje náhradu současného papírovéhoho systému pro zaznamenávaní vykonané a předávané práce v depu kolejových vozidel ČD a.s. elektronickým informačním systémem. Vyvinutý systém umožňuje evidenci dat o vykonané práci, docházce, vozidlech a zaměstnancích. Mimo jiné nabízí možnost vyhledávat, filtrovat a vytvářet tiskové sestavy. Řešení je uskutečněno formou distribuovaného systému typu klient-server. Serverovou část tvoří tři WCF služby a MS SQL databáze. Klientskou stranu představuje WPF aplikace pro snadné ovládání a správu všech evidencí. Obsahem práce jsou i testy, které prověřily správnou funkčnost celého systému.

Diplomová práce je rozdělená do tří hlavních částí. První část obsahuje analýzu řešení problému, druhá část se zabývá implementací řešení a poslední část obsahuje výsledky testování.

Abstract

This thesis deals with the design, implementation and testing of the digital handover system, which is considered as a replacement of the currently used paper-based system in the railway depot ČD a.s. The system will be used for the logging of performed tasks, collecting the data about finished work, attendance of employees, and monitoring of vehicles. It will allow the user to search, filter and print the data. The solution has a distributed client-server architecture; the server consists of three WCF services and a MS SQL database and the client is designed as a Windows WPF application. The functionality of the system has been verified by a set of tests whose results are also part ot this thesis.

The thesis is divided into three parts. In the first part, the problem is analysed and the conceptual model of the system is designed. The second part deals with the implementation of the system and finally, the last part describes the testing of the system.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Ve společnosti, kde jsem pracoval jako technik elektronických systémů vlaku CityElefant 471, jsme psali veškerou vykonanou a předávanou práci do sešitu. Tento systém vedení záznamu měl několik zásadních omezení. V prvé řadě každý zaměstnanec měl svůj rukopis, který byl pro některé kolegy těžko čitelný. Dalším a velice výrazným problémem bylo vyhledávání záznamu. Pokud bylo požadováno najít určitý záznam a tento záznam byl staršího data, pak to bylo skoro nemožné. Celé depo kolejových vozidel a veškerá oddělení požívaly tento způsob předávání práce, jelikož v depu na konkrétní řadu vozidel existuje jeden mistr (nadřízený), který koriguje a rozdává práci dle typu konkrétním oddělením jako jsou zámečnicí, elektrikáři, zkušebna atd. Je pro ně někdy obtížné držet si přehled o tom, co se již vykonalo a co ne. Tento problém nastává převážně při střídání směn, kdy se mistři na pracovišti mění a kdy si vzájemně nepředají potřebné informace o stavu jednotlivých prací. Po výměně mistra se většinou nový mistr opakovaně dotazuje buď vedoucích oddělení nebo samotných zaměstnanců na to, co je a co není hotové, což někteří zaměstnanci nesnáší dobře. Jeden vedoucí oddělení může mít na starosti více oddělení. Každé oddělení má svoji dílnu a tyto dílny mohou být od sebe poměrně vzdáleny. Z důvodu vzdálenosti jednotlivých dílen dochází převážně k telefonické komunikaci mezi dílnou, vedoucím oddělení a mistrem.

Tématem mé diplomové práce se stala tvorba informačního systému, o který jsem byl požádán zaměstnanci. Systém by měl nahradit nynější sešitovou verzi záznamu směn, přinést větší informovanost jak pro mistry tak i pro vedoucí oddělení, jimž tento navržený systém umožní lépe si udržovat přehled o tom, jaké oddělení co vykonalo, a odstranit zatížení zaměstnanců nadbytečnými telefonními hovory. Další výhodou systému bude možnost vzdáleného přístupu a rychlého vyhledávání záznamů, což doposud nebylo možné.

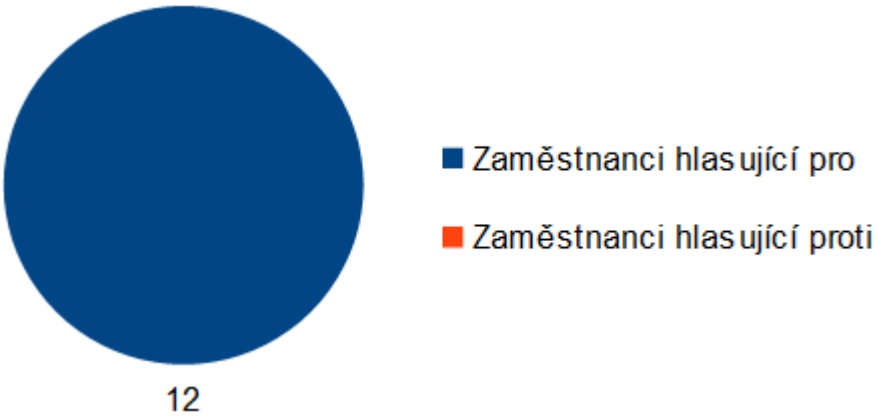
Architektura systému se bude skládat z tenkého klienta, který bude navržen v technologii WPF. Tento klient bude implementován pomocí návrhového vzoru MVVM. Serverová část se bude skládat ze služeb navržených v technologii WCF, které budou hostovány na IIS 7. Data budou uložena v databází MSSQL. K datům se bude přistupovat pomocí EF. Klienti se serverovou částí budou komunikovat přes zabezpečený protokol https.

Kapitola 2

Analýza

2.1 Analýza potřeby

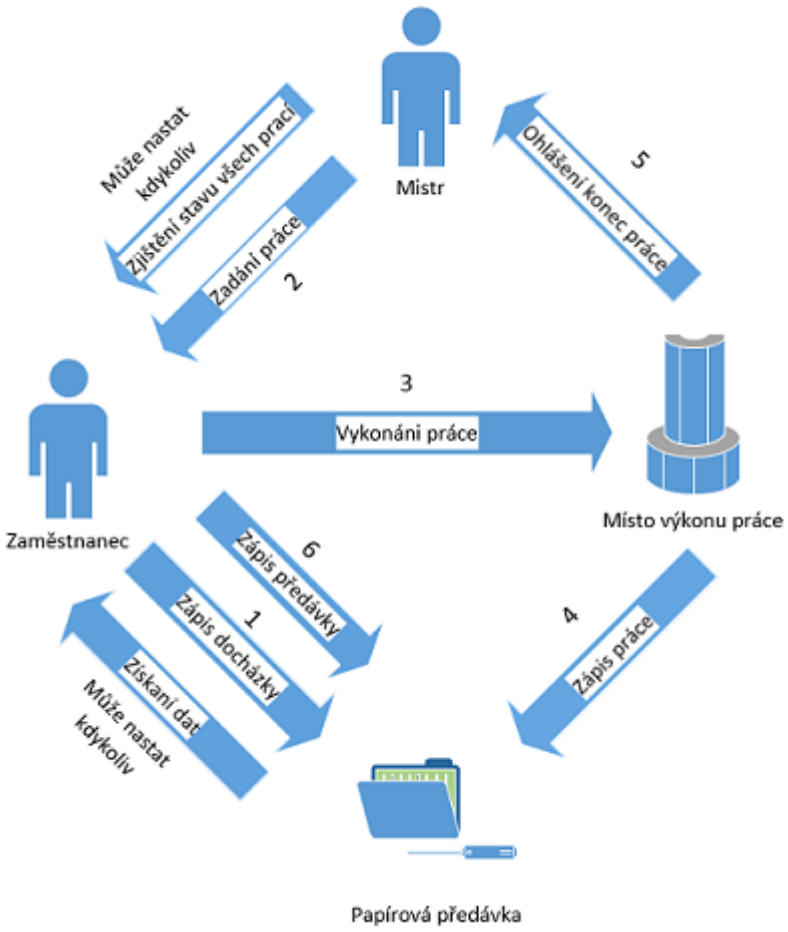
Před zahájením vývoje tohoto systému byla vytvořena anketa, jejímž účelem bylo zjistit zájem zaměstnanců o tento systém. Celkově se zúčastnilo ankety 12 zaměstnanců, kteří tento systém budou užívat. Každý zaměstnanec se mohl svobodně rozhodnout, zdali bude tento systém pro něho přínosem či nikoliv. V anketě byly uvedeny veškeré výhody, které tento systém přinese. Anketa je přiložena v přílohové části této diplomové práce (viz příloha B). Výsledek provedené ankety je zobrazen na obrázku 2.1. Vyhodnocením ankety bylo zjištěno, že všichni dotazovaní zaměstnanci projevili zájem o tento systém. Mezi těmito zaměstnanci byli také starší osoby, kteří se obecně hůře přizpůsobují technickému pokroku. Překvapivým zjištěním z provedené ankety bylo, že i starší osoby souhlasily, aby tento systém byl nasazen do reálného provozu.



Obrázek 2.1: Výsledek ankety

2.2 Pracovní proces s papírovou podobou předávky

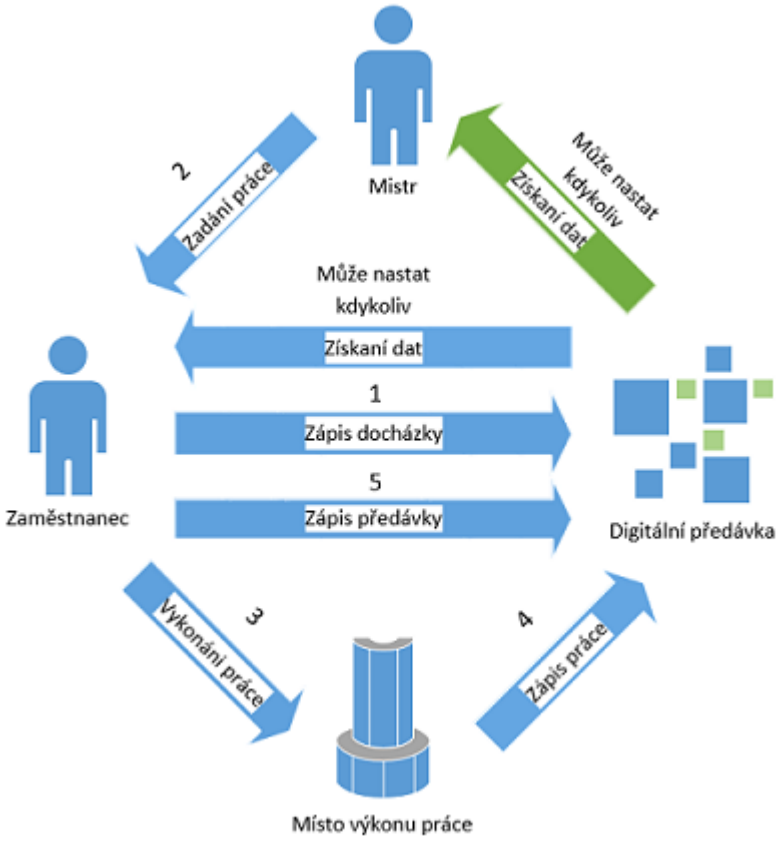
Pracovní proces začíná příchodem zaměstnance do práce. Zaměstnanec zapíše do předávací knihy svojí přítomnost a turnus, do kterého je zařazen. Turnusem se myslí čas počátku a konce pracovní doby zaměstnance. Zaměstnanec zjišťuje z předávací knihy vykonanou a předávanou práci předešlé skupiny (směny). Práci zaměstnanci přiděluje jeho nadřízený (mistr) nebo je práce určená dle předávky v předávací knize. Zaměstnanec zapíše do předávací knihy jím vykonanou práci a oznámí dokončení práce svému nadřízenému. Během směny se nadřízený většinou opakovaně telefonicky informuje o stavu veškerých prací. Práce, které se nestihnou vykonat během směny, jsou zapsaná do předávací knihy jako předávka pro další směnu. Diagram pracovního procesu je vidět na obrázku 2.2, kde čísla určují pořadí průběhu.



Obrázek 2.2: Pracovní proces s papírovou podobou předávky

2.3 Pracovní proces s digitální podobou předávky

Tento proces přináší změnu v komunikaci s nadřízeným. Zatím co ve staré verzi bylo potřeba vše hlásit nadřízenému, nyní jíž mistr muže sám nahlížet do předávky ze své kanceláře bez potřeby opětovného obvolávání podřízených a zjišťování provedených prací. Také muže sám zjistit vykonanou nebo předanou práci předešlé směny. Také muže zjistí mnoho dalších užitečných informací. Nasazení digitální předávky přinese zefektivnění pracovního procesu. Diagram pracovního procesu je vidět na obrázku níže 2.3, kde čísla určují pořadí průběhu.



Obrázek 2.3: Pracovní proces s digitální podobou předávky

2.4 Popis požadavku

Požadavky jsem konzultoval se zaměstnanci, kteří by tento systém měli používat. Každý zaměstnanec měl nápady, jak by se aplikace měla ovládat, jaké funkce by měla umožňovat a jak by se měla chovat. Jedním z hlavních požadavků byla tvorba tenkého klienta v podobě klasické aplikace do počítače, která pro zaměstnance bude pohodlnější k používání. Do budoucna se předpokládá rozšíření této práce o webovou verzi klienta, která není zahrnutá v této diplomové práci. Požadavky na systém jsou sepsány níže.

- Systém musí být lehce ovladatelný, jelikož tento software budou používat i starší generace, pro které je práce s počítačem mnohem obtížnější. Z tohoto důvodu by měl systém umožňovat výběr záznamů ze statisticky nejběžnějších závad a oprav, které se budou generovat při každém spuštění klienta.
- Systém musí umožňovat vyhledávání záznamu s možností zobrazení detailu konkrétního záznamu, ve kterém budou zobrazeny přehledně informace o vozidlu, směně, typu opravy, závadě, způsobu opravy a docházce zaměstnanců v dané směně dle dále uvedených kritérií.
  - Typu závady.
  - Vozidla.
  - Textového výrazu obsaženého v popisu závady.
  - Vymezeném časovém intervalu (od, do).
  - Pokud má uživatel dostatečná oprávnění, pak může vyhledávat záznamy jiných oddělení. V opačném případě se vyhledávají záznamy jen v oddělení, ve kterém je veden jeho účet.
- Systém musí umožňovat tisk a náhled tisku směn pro zálohu záznamů v papírové podobě.
- Po spuštění klienta se má vždy vybrat poslední založená směna.
- Navrhovaný systém musí umožňovat ukládat informace o jednotlivých vozidlech, zaznamenávat docházku zaměstnanců s možností přidání doplňujících informací, přehledně zobrazovat záznamy o zaměstnancích a jejích turnusech, zakládat, editovat a mazat informace o odděleních, turnusech, typech oprav a statusech zaměstnanců pro docházku.

- Tento systém musí být zabezpečen vůči neoprávněnému přístupu pomocí uživatelských účtů a také musí být zabezpečena komunikace se serverem.
- Uživatelské účty by měly umožňovat tuto funkcionalitu:
  - Každý uživatelský účet by mělo být možné navázat na konkrétního zaměstnance.
  - Každý účet by měl být zařazen do konkrétního oddělení.
  - Účet zařazen do konkrétního oddělení by měl umožňovat zápis a čtení záznamu jen z oddělení, do kterého je zařazen.
  - Každý uživatel by měl mít možnost si změnit svoje heslo a vidět svá oprávnění.
  - Každému účtu by mělo být možné nastavit oprávnění pro zápis do systému, pro administrativu a pro zobrazení záznamů z jiných oddělení.
  - Administrátorské účty by měly mít možnost zakládat, editovat a mazat uživatelské účty s možností resetovat hesla jednotlivých uživatelských účtů.
- Mělo by být možné zaznamenávat jak závadu tak i způsob její opravy a typ opravy s možností výběru častých závad a oprav, které se generují při každém spuštění klienta.
- Mělo by být možné zaznamenávat předávanou práci s možností rychlého přesunu do vykonaných prací vybrané směny.

2.5 Uživatelské role (oprávnění)

2.5.1 Bez role

- Registrovaný uživatel bez role bude mít tyto možnosti:
  - Přihlásit se do systému.
  - Prohlížet si směny oddělení, do kterého sám spadá, s možností zobrazení záznamu o provedených prací, docházce i předaných prací.
  - Tisknout sestavy směn.
  - Prohlížet si seznam vozidel.
  - Prohlížet si seznam typů prací.

  - Prohlížet si seznam zaměstnanců.
  - Prohlížet si seznam turnusů.
  - Prohlížet si seznam statusů.
  - Hledat záznamy práce dle těchto parametrů:
    * Typu závady.
    * Vozidla.
    * Textového výrazu obsaženého v popisu závady.
    * Vymezeném časovém intervalu (od, do).
  - Zobrazení detailu vyhledaného záznamu.
  - Zobrazení informace o vlastním účtu.
  - Změna hesla vlastního účtu.

2.5.2 Zápis

- Registrovaný uživatel s rolí „Zápis“ bude mít tyto možnosti:
  - Přihlásit se do systému.
  - Prohlížet, editovat a mazat směny v oddělení, do kterého sám spadá, s možností zobrazení, editace a mazání záznamu práce, docházky i předávaných prací.
  - Převod předávaného záznamu do vykonané práce.
  - Tisknout sestavy směn.
  - Prohlížet, editovat a mazat záznamy vozidel.
  - Prohlížet, editovat a mazat záznamy typu práce.
  - Prohlížet, editovat a mazat záznamy zaměstnanců.
  - Prohlížet, editovat a mazat záznamy turnusů.
  - Prohlížet, editovat a mazat záznamy statusů.
  - Hledat záznamy práce dle těchto parametrů:
    * Typu závady.
    * Vozidla.
    * Textového výrazu obsaženého v popisu závady.

        * Vymezeném časovém intervalu (od, do).
    - Zobrazení detailu vyhledaného záznamu.
    - Zobrazení informace o vlastním účtu.
    - Změna hesla vlastního účtu.

2.5.3 Mistr

- Registrovaný uživatel s rolí „Mistr“ bude mít tyto možnosti:
    - Přihlásit se do systému.
    - Prohlížet si směny oddělení, do kterého sám spadá, s možností zobrazení záznamu práce, docházky i předávané práce.
    - Tisknout sestavy směn.
    - Prohlížet si seznam vozidel.
    - Prohlížet si seznam typu práce.
    - Prohlížet si seznam zaměstnanců.
    - Prohlížet si seznam turnusů.
    - Prohlížet si seznam statusů.
    - Hledat záznamy práce dle těchto parametrů:
        * Typu závady.
        * Vozidla.
        * Oddělení.
        * Textového výrazu obsaženého v popisu závady.
        * Vymezeném časovém intervalu (od, do).
    - Zobrazení detailu vyhledaného záznamu.
    - Zobrazení informace o vlastním účtu.
    - Změna hesla vlastního účtu.

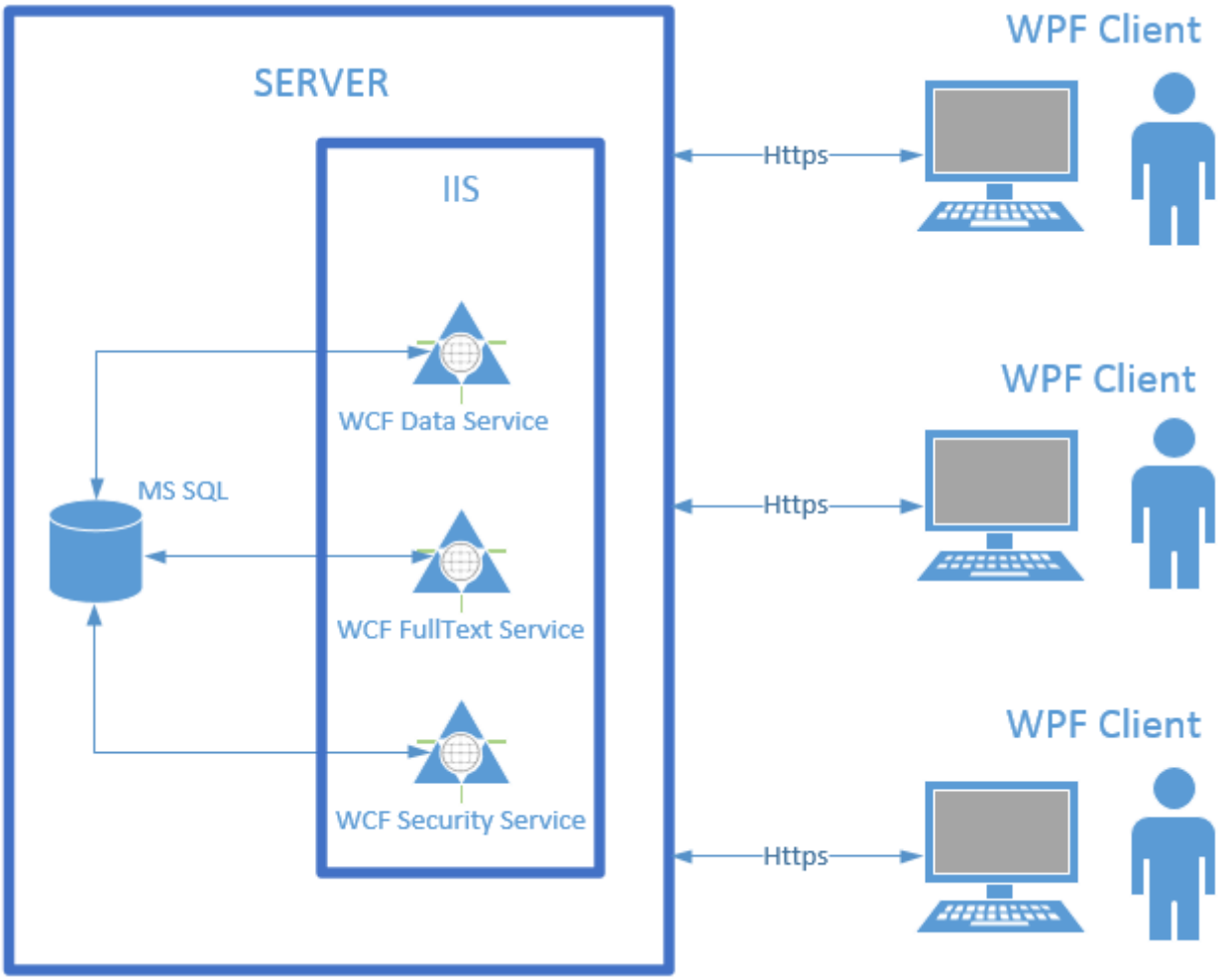
2.5.4 Administrátor

- Registrovaný uživatel s rolí „Administrátor“ bude mít tyto možnosti:

- Přihlásit se do systému.
- Prohlížet si směny oddělení, do kterého sám spadá, s možností zobrazení záznamu práce, docházky i předávané práce.
- Tisknout sestavy směn.
- Prohlížet, editovat a mazat záznamy oddělení.
- Prohlížet si seznam vozidel.
- Prohlížet si seznam typu práce.
- Prohlížet si seznam zaměstnanců.
- Prohlížet si seznam turnusů.
- Prohlížet si seznam statusů.
- Hledat záznamy práce dle těchto parametrů:
  - Typu závady.
  - Vozidla.
  - Textového výrazu obsaženého v popisu závady.
  - Vymezeném časovém intervalu (od, do).
- Zobrazení detailu vyhledaného záznamu.
- Zobrazení informace o vlastním účtu.
- Změna hesla vlastního účtu.
- Prohlížet, zakládat, editovat a mazat uživatelské účty s možností resetovat hesla jednotlivých uživatelských účtů.

2.6 Struktura systému

Strukturu systému jsem návrh tak, aby byla možnost pozdějšího rozšíření na jiné druhy klientů. Systém se skládá z databáze *MSSQL* [1] , která je v express verzi zdarma i pro komerční užití, do které se ukládají veškeré informace. K databázi přistupují tři *WCF* [2] služby, které jsou hostované na *IIS 7* [3] . Ke službám se přistupuje přes https protokol pomocí tenkých klientů. Strukturu systému je možné vidět na obrázku 2.4.



Obrázek 2.4: Struktura systému

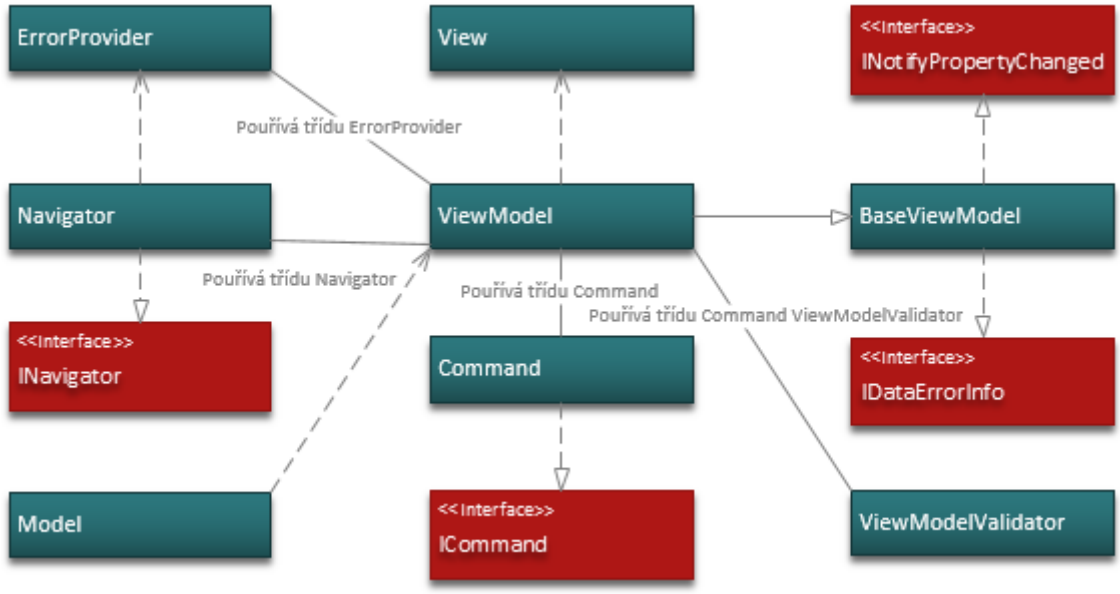
[1]MSSQL (Microsoft SQL Server) je relační databázový server vyvinutý společností Microsoft.

[2]WCF (Windows Communication Foundation) je technologie vyvinuta společností Microsoft pro tvorbu servisně orientovaných aplikací.

[3]IIS 7 (informační internetová služba) je to softwarový webový server vyvinutý společností Microsoft.

2.7 Návrh řešení klienta

Klientská aplikace bude řešena pomocí návrhového vzoru „MVVM“, jelikož se jedna o návrhový vzor, který zpřehledňuje napsaný kód, usnadňuje jeho rozšíření a také umožňuje jednoduchou testovatelnost. Pro lepší přehlednost datového modelu jsem se zaměřil na obecnější popis, a proto zde nejsou zobrazeny jednotlivé třídy „ViewModely“, „Views“ a „Modely“. Všechny tyto třídy jsou znázorněný jen jedním „ViewModelem“, „View“ a „Modelem“. Cely datový model klientské aplikace je možné vidět na obrázku 2.5.

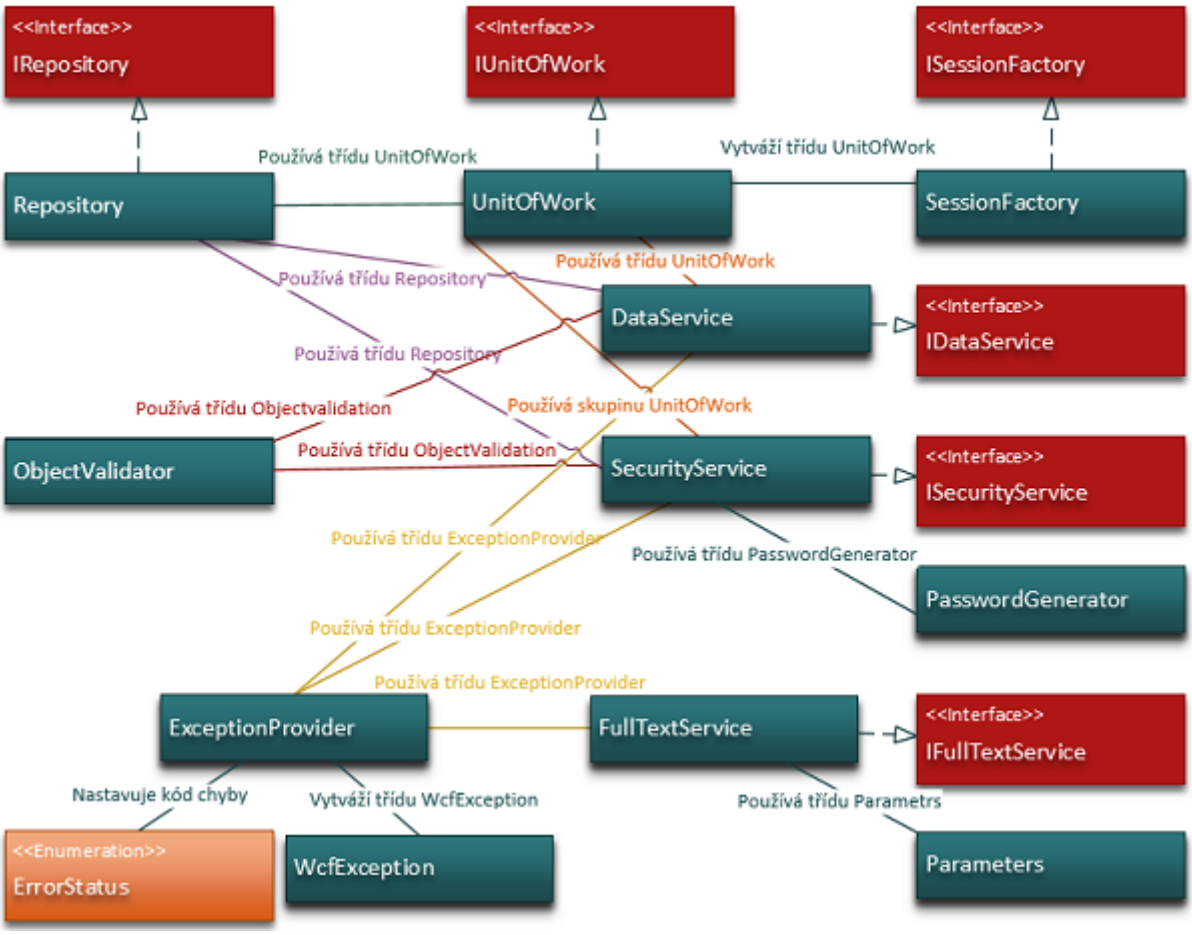


Obrázek 2.5: Datový model klientské aplikace

Na obrázku 2.5 je vidět, že každý „ViewModel“ je nainjektován do „View“ při vytvoření instance třídy „View“. Při tom je automaticky do „ViewModelu“ nainjektována třída „Model“, která reprezentuje data. Pro reagování na uživatelský vstup se ve „ViewModelu“ používá třída „Command“, která implementuje rozhraní „ICommand“. Každý „ViewModel“ také dědí ze třídy „BaseViewModel“, která implementuje rozhraní „INotifyPropertyChanged“ a „IDataErrorInfo“. Rozhraní „INotifyPropertyChanged“ slouží pro aktualizací „View“, pokud se změní některá vlastnost ve „ViewModelu“. Rozhraní „IDataErrorInfo“ slouží pro zobrazení chybového stavu ve „View“, pokud uživatel zadá do vstupu nevalidní data. Validnost dat se kontrolují třídou „ViewModelValidator“. Pro navigací v aplikací slouží třída „Navigator“, která implementuje rozhraní „INavigator“. Pro zobrazení chybových stavů aplikace slouží třída „ErrorProvider“.

2.8 Návrh řešení služeb

Služby budou navržený v technologii „WCF“, kde každá služba bude reprezentována třídou, která implementuje službu a rozhraní, které se zpřístupňuje přes endpoint. Na obrázku 2.6 je vidět datová struktura všech služeb.



Obrázek 2.6: Datový model služeb

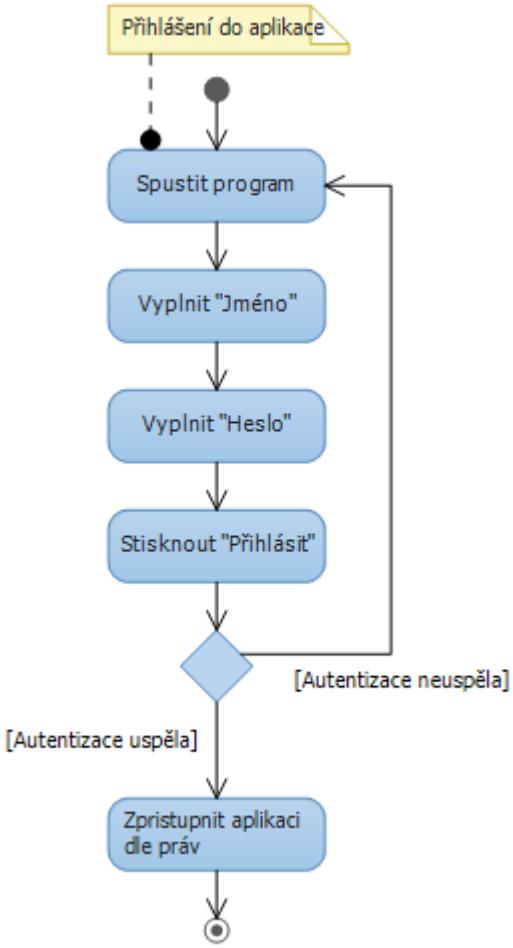
Pro zpracování výjimek používají všechny služby třídu „ExceptionProvider“, která vytváří třídu „WcfException“, která bude odesílána jako „FultContrakt“ klientovi. „ErrorStatus“ je výčet chyb, které mohou nastat na straně služeb. Služby „DataService“ a „SecurityService“ používají třídy „ObjectValidator“, „Repository“, „UnitOfWork“ a „SessionFactory“. Každá z těchto tříd je definována svým rozhraním. Třída „ObjectValidator“ slouží pro validaci objektu, který se předává jako parametr metodám služby. Třída „Repository“ umožňuje provádět základní „CRUD“ operace nad entitami za pomocí třídy „UnitOfWork“, která se stará o transakční zpracování. Třída „SessionFactory“ slouží pro

vytvoření singletonu třídy „UnitOfWork“. „PasswordGenerator“ je třída umožnující generovat náhodná bezpečná hesla a tuto třídu používá jen „SecurityService“, která díky ní generuje hesla novým uživatelům, případně je použita pro resetování hesel stávajícím uživatelům. Služba „FullTextService“ používá třídu „Parameters“, ve které jsou definované veškeré parametry, podle kterých je možné vyhledávat záznamy.

2.9 UML diagramy aktivit

2.9.1 Diagram přihlášení

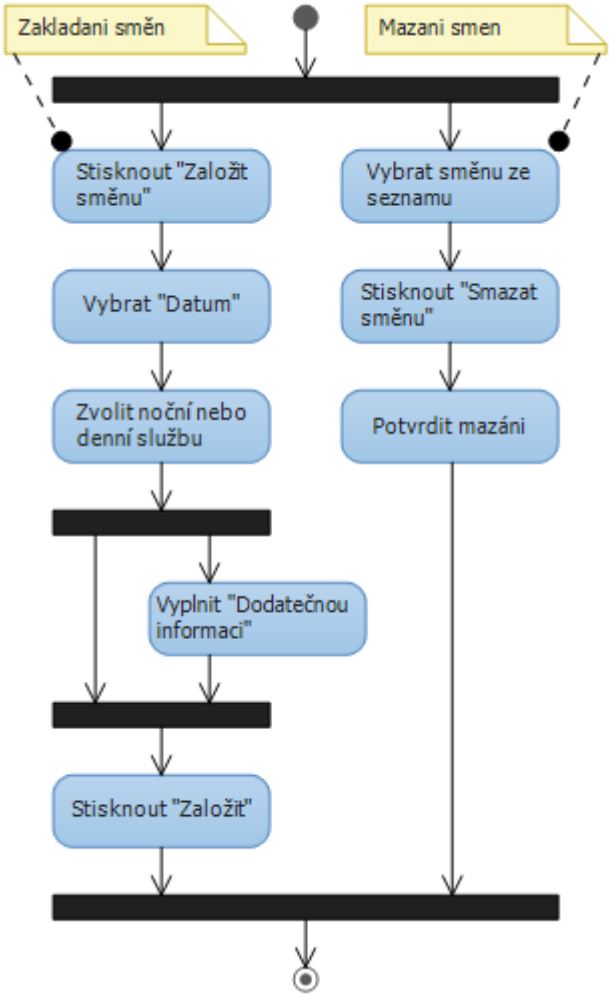
Proces přihlášení je zahájen tím, že se spustí klientská aplikace. Po spuštění se zobrazí dialogové okno pro zadání uživatelského „Jména“ a „Hesla“. Po zadání těchto dvou povinných údajů a po stisknutí tlačítka „Přihlásit“ se začne provádět autentizace. Pokud autentizace proběhne úspěšně, spustí se program s uživatelským prostředím, na které má uživatelský účet práva. V opačném případě se zobrazí zpráva o neúspěšné autentizaci. Pro veškeré další procesy se předpokládá, že je uživatel úspěšně přihlášen.



Obrázek 2.7: Diagram přihlášení

2.9.2 Diagram směn

Proces zakládání směn se zahajuje stisknutím tlačítka „Založit směnu“, na kterou se může kliknout buď z lišty na ikonku *směny* [1] , a nebo přes nabídku „Nástroje“ v hlavním menu. Po kliknutí se otevře dialogové okno pro vytváření směn, kde je třeba vyplnit „Datum“ a vybrat „Denní“ nebo „Noční“ směnu. Jako nepovinný údaj je možné zadat „Dodatečnou informaci“. Po stisku na tlačítko „Založit“ se směna objeví v levém seznamu směn jako vybraná položka. Proces pro mazání začíná výběrem směny, kterou chceme smazat. Po vybrání a kliknutím na tlačítko „Smazat směnu“, kterou lze najít na hlavní liště v podobě ikony *smazat směnu* [2] a nebo přes nabídku „Nástroje“ v hlavním menu, se zobrazí dialogové okno, které je potřeba potvrdit pro dokončení smazání směny.



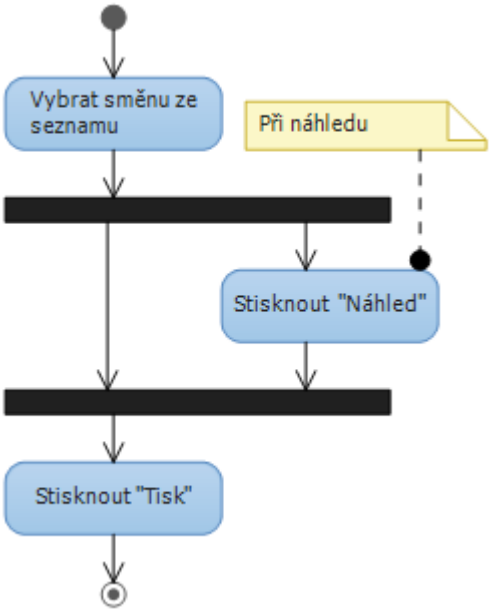
Obrázek 2.8: Diagram směn

[1] ikona založení směny.

[2] ikona odebraní směny.

2.9.3 Diagram tisku

Proces tisku se zahájí výběrem směny. Poté je možné stisknout buď tlačítko „Náhled“ nebo „Tisk“, která se nacházejí v hlavní liště v podobách ikon *tisku* [1] , *náhledu* [2] a nebo přes nabídku „Soubor“ v hlavním menu. Po stisknutí tlačítka „Náhled“ se otevře dialogové okno s náhledem dokumentu, ve kterém se dá stisknout tlačítko „Tisk“. Stisknutím tlačítka „Tisk“ se rovnou nabídne tis vybrané směny, aniž bychom jej před tiskem viděli.



Obrázek 2.9: Diagram tisku

[1] ikona tisku.

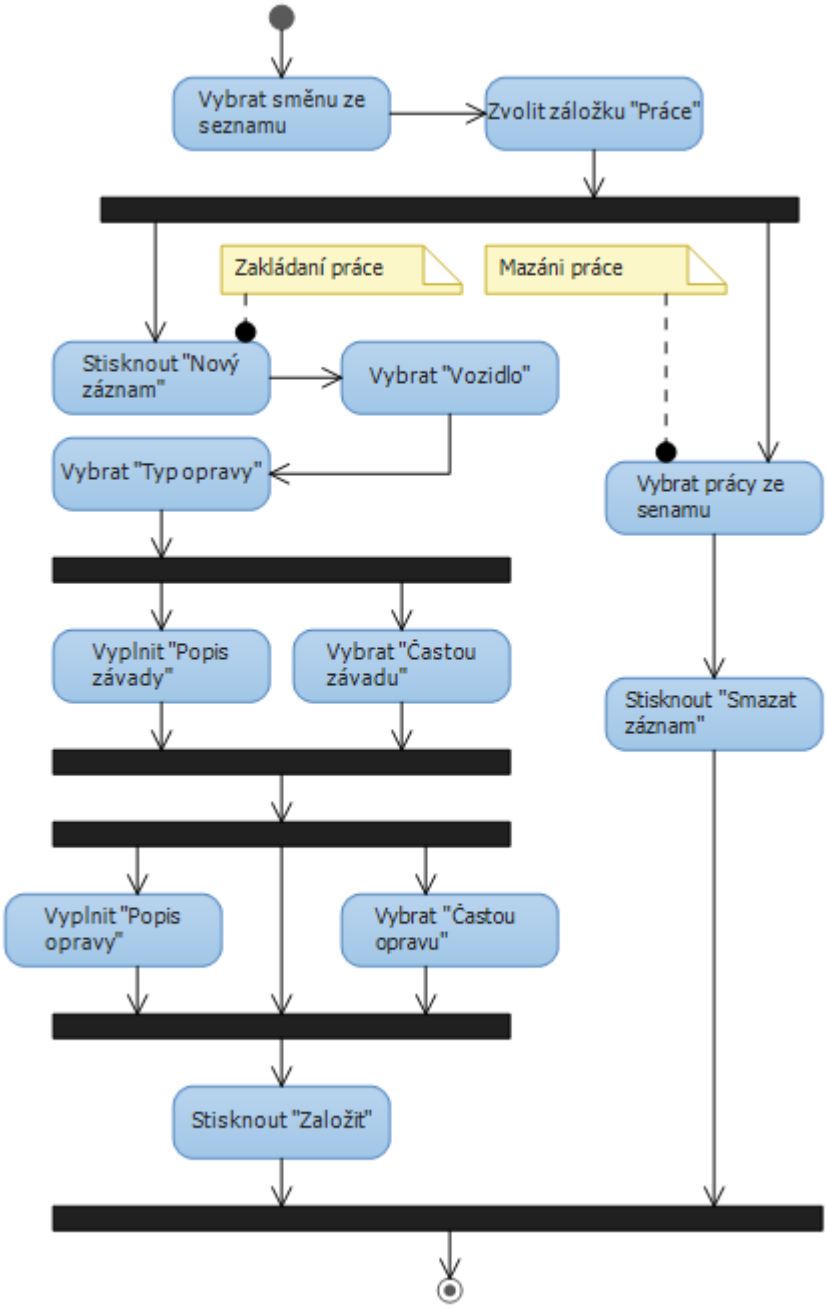
[2] ikona náhledu.

2.9.4 Diagram práce

Proces zakládání práce začíná výběrem směny, do které chceme práci přidat. Také je třeba vybrat záložku „Práce“ v pravé části obrazovky. Nyní už stačí stisknout tlačítko „Nový záznam“, které se nachází v hlavní liště vyobrazené ikonou *nového záznamu* [1] nebo přes nabídku „Nástroje“ v hlavním menu. Po otevření dialogového okna pro zakládání vykonané práce je třeba vyplnit povinné udaje, které jsou označeny červeným rámečkem. Konkrétně se jedná o „Vozidlo“, „Typ opravy“ a „Popis závady“, kde je možné buď vybrat popis „Časté závady“ nebo napsat vlastní. Jako nepovinný údaj je popis „Způsob opravy“, který též lze vybrat buď z „Časté způsoby“ opravy nebo sepsat vlastní v poli „Způsob opravy“. Po vyplnění všech potřebných údajů a stisknutí tlačítka „Založit“ se záznam objeví na pravé straně okna v záložce „Práce“. Proces mazání vykonané práce spočívá ve výběru konkrétního záznamu v záložce „Práce“ a stisknutí tlačítka „Smazat záznam“, které je možné nalézt buď v hlavni liště v podobě ikony *smazat záznam* [2] nebo přes nabídku „Nástroje“ v hlavním menu.

[1] ikona založení záznamu.

[2] ikona odebraní záznamu.



Obrázek 2.10: Diagram práce

2.9.5 Diagram vozidel, typu práce, oddělení a statusu

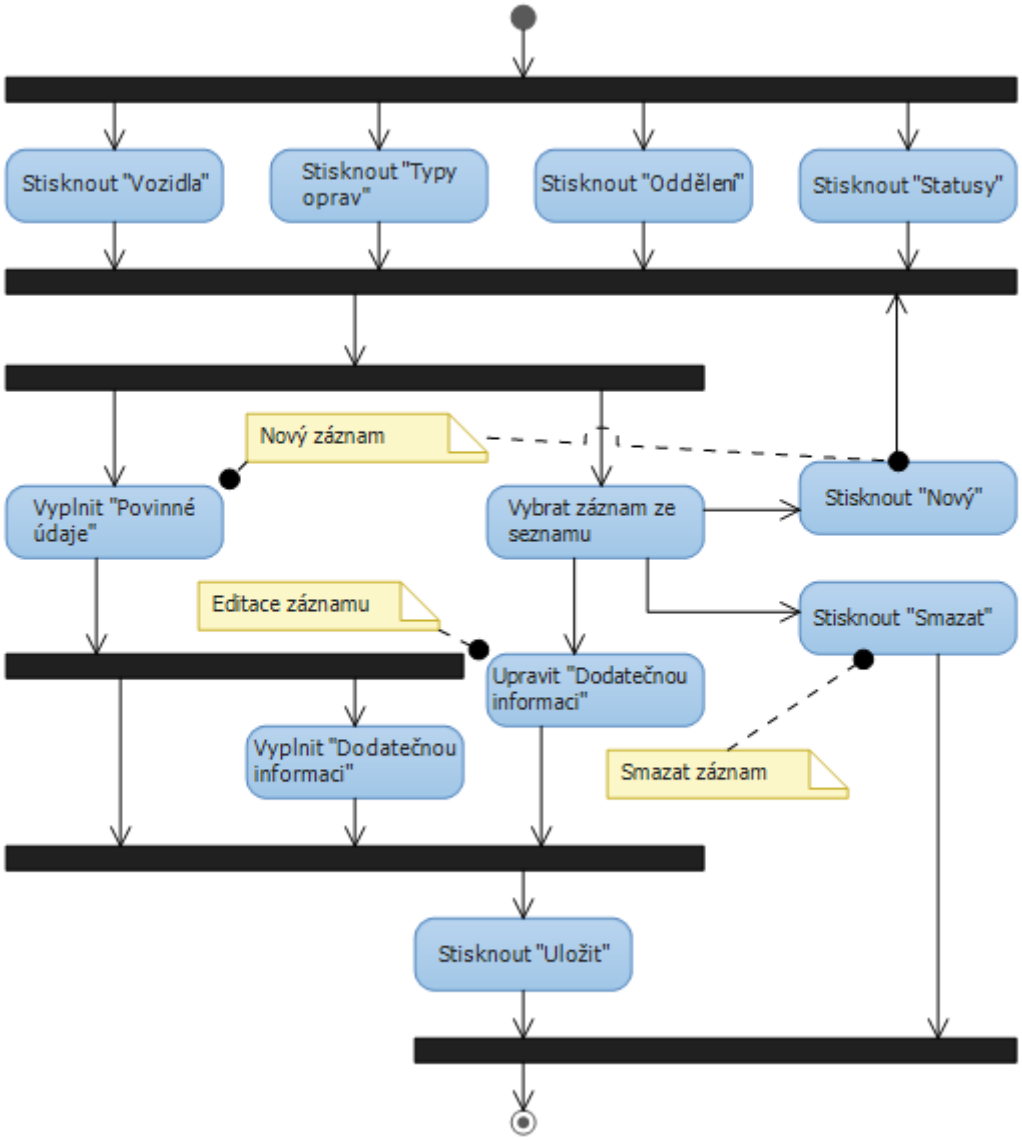
Proces vozidel, typu práce, oddělení a statusu je stejný. Záleží jen na tom, které dialogové okno otevřeme. Veškerá okna jsou buď na hlavní liště aplikace označená ikonami *vozidla* [1] , *typu práce* [2] , *oddělení* [3] , *statusu* [4] , nebo v nabídce „Okna“ v hlavním menu. Po zvolení konkrétního procesu se otevře dialogové okno, do kterého je při zakládání záznamu potřeba vyplnit povinný údaj, jenž je orámovaný červenou barvou. Také je možné doplnit nepovinný údaj v podobě dodatečné informace do pole „Dodatečná informace“. Po vyplnění veškerých údajů a stiskem tlačítka „Uložit“ se záznam objeví na pravé straně seznamu jako vybraná položka. Pokud je nějaká položka vybraná a chceme založit novou, je třeba stisknout tlačítko „Nový“. Smazat záznam je možné výběrem záznamu z pravého seznamu a stiskem tlačítka „Smazat“. Proces editace začíná výběrem záznamu, který chceme upravit. Po vybrání záznamu upravíme pole „Dodatečná informace“ a stisknutím tlačítka „Uložit“ provedené změny uložíme.

[1] ikona vozidel.

[2] ikona typu práce.

[3] ikona oddělení.

[4] ikona statusu.

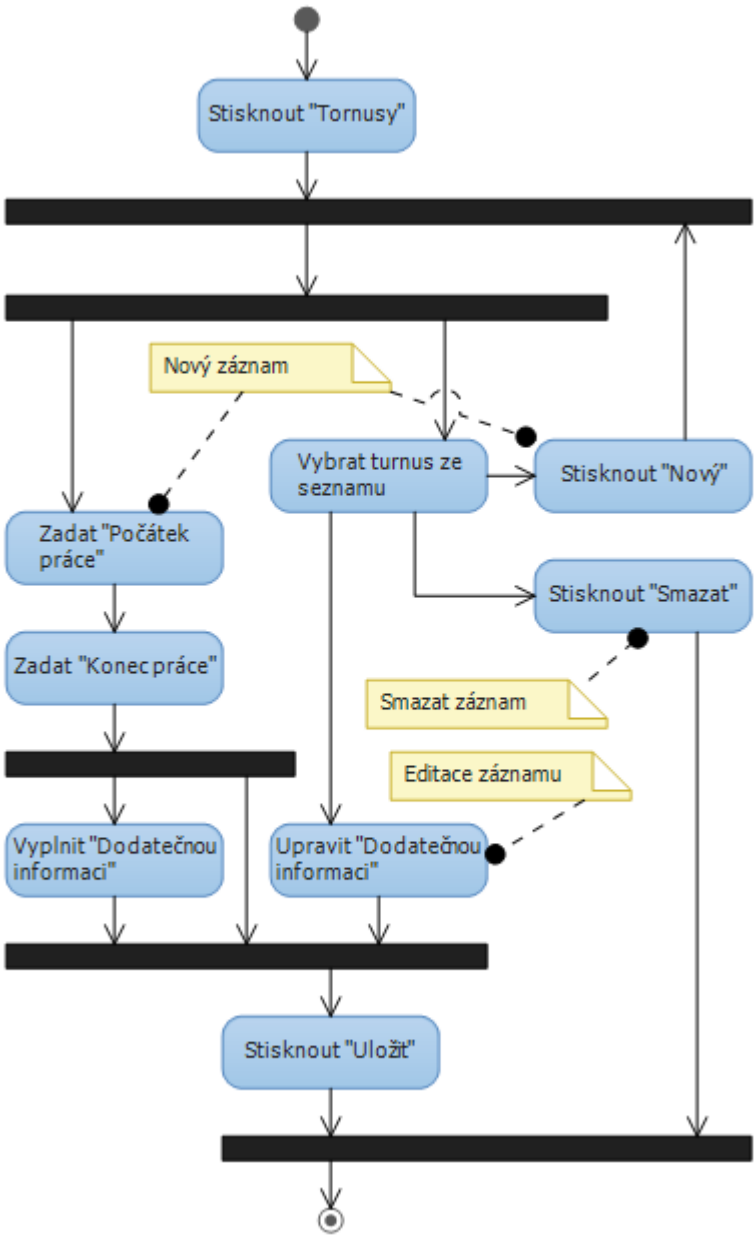


Obrázek 2.11: Diagram vozidel, typu práce, oddělení a statusu

2.9.6 Diagram turnusů

Proces turnusů začíná stiskem tlačítka „Turnusy“, které se nachází na hlavní listě v podobě ikony *turnusů* [1] a nebo v nabídce „Okna“ v hlavním menu, které otevře dialogové okno. Proces zakládání turnusů začíná vyplněním povinných polí „Počátek práce“ a „Konec práce“, jež jsou orámované červenou barvou. Je možné také doplnit dodatečnou informaci o turnusu do pole „Dodatečná informace“. Stisknutím tlačítka „Uložit“ se záznam uloží a současně se objeví v seznamu na pravé straně dialogového okna. Také se nastaví jako aktuálně vybraná položka. Pokud je již vybraný nějaký turnus a chceme zakládat nový, je třeba stisknout tlačítko „Nový“. V případě editace je třeba vybrat z pravého seznamu záznamu turnus, který chceme editovat, a po úpravě pole „Dodatečná informace“ a stisknutí tlačítka „Uložit“ se upravený záznam uloží. V případě smazání stačí vybrat záznam, který chceme smazat, stačí stisknout tlačítko „Smazat“.

[1]  ikona turnusů.



Obrázek 2.12: Diagram turnusů

2.9.7 Diagram zaměstnanců

Proces zaměstnanců začíná stisknutím tlačítka „Zaměstnanci“, které se nachází v hlavní liště aplikace reprezentované ikonou *zaměstnanců* [1] , nebo v nabídce „Okna“ hlavního menu, které otevře dialogové okno. Proces zakládání zaměstnance začíná vyplněním povinných údajů o zaměstnanci, které jsou označeny červeným rámečkem. Další nepovinné údaje je možné vyplnit dle potřeb. Jeden z povinných údajů je seznam turnusů zaměstnance, který musí obsahovat aspoň jeden turnus. Předpokládá se, že již byl vytvořen minimálně jeden turnus, který bude možné vybrat a vložit do seznamu turnusu zaměstnance. Po vyplnění všech povinných údajů a případně i některých volitelných stačí pro dokončení procesu založení stisknout tlačítko „Uložit“, a tím se zaměstnanec uloží a zobrazí se v pravém seznamu jako vybraná položka. Pokud je již nějaká položka vybraná a chceme vytvořit nového zaměstnance, stiskneme tlačítko „Nový“. V případě odstranění vybereme zaměstnance a stiskneme tlačítko „Smazat“. Pro editaci vybereme zaměstnance a upravíme některý z nepovinných údajů s možností i editovat seznam turnusu zaměstnance, jenž musí obsahovat minimálně jeden turnus. Stiskem tlačítka „Uložit“ se veškeré změny uloží.

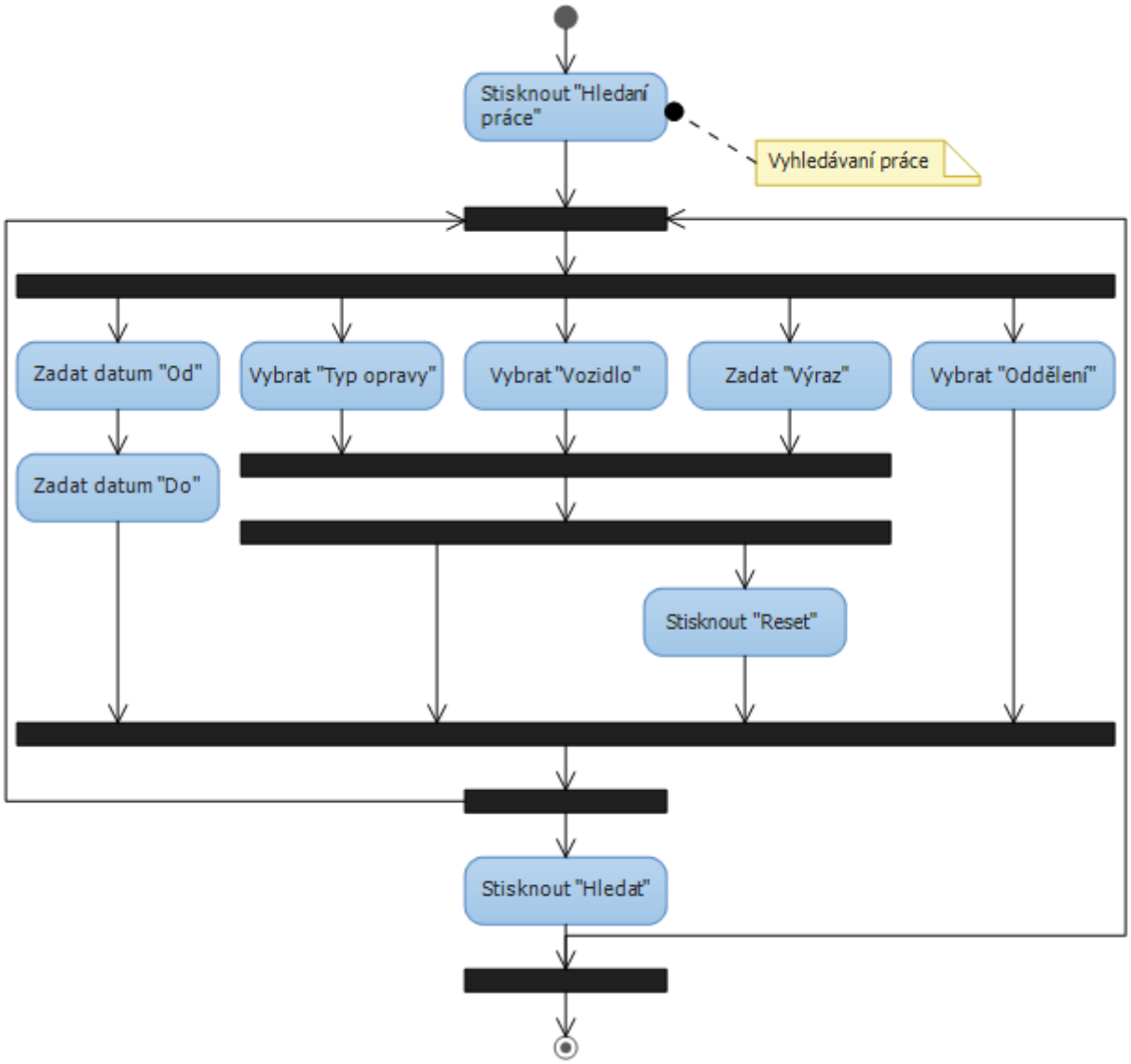
[1]  ikona zaměstnanců.

Obrázek 2.13: Diagram zaměstnanců

2.9.8 Diagram hledání

Proces hledání práce zahájíme stisknutím tlačítka „Hledání práce“, které lze najít jak na hlavní liště v podobě ikony *hledání práce* [1] , tak i v nabídce „Hledat“ v hlavním menu. V otevřeném dialogovém okně lze hledat dle parametru „Vozidlo“, „Typ opravy“, data „Od Do“, a textového výrazu, dle kterého se hledá v popisu závady jednotlivých záznamů, který se píše do pole „Výraz“. Také lze hledat dle „Oddělení“, pokud přihlášený uživatelský účet spadá do role „Mistr“. Po zvolení potřebných parametrů a stisknutí tlačítka „Hledat“ se veškeré nalezené záznamy objeví ve spodní tabulce výsledků. V této tabulce je možné dvojitým poklepáním na záznam zobrazit podrobné údaje. Tlačítkem „Reset“ se resetují vybrané parametry hledání.

[1]  ikona hledání práce.

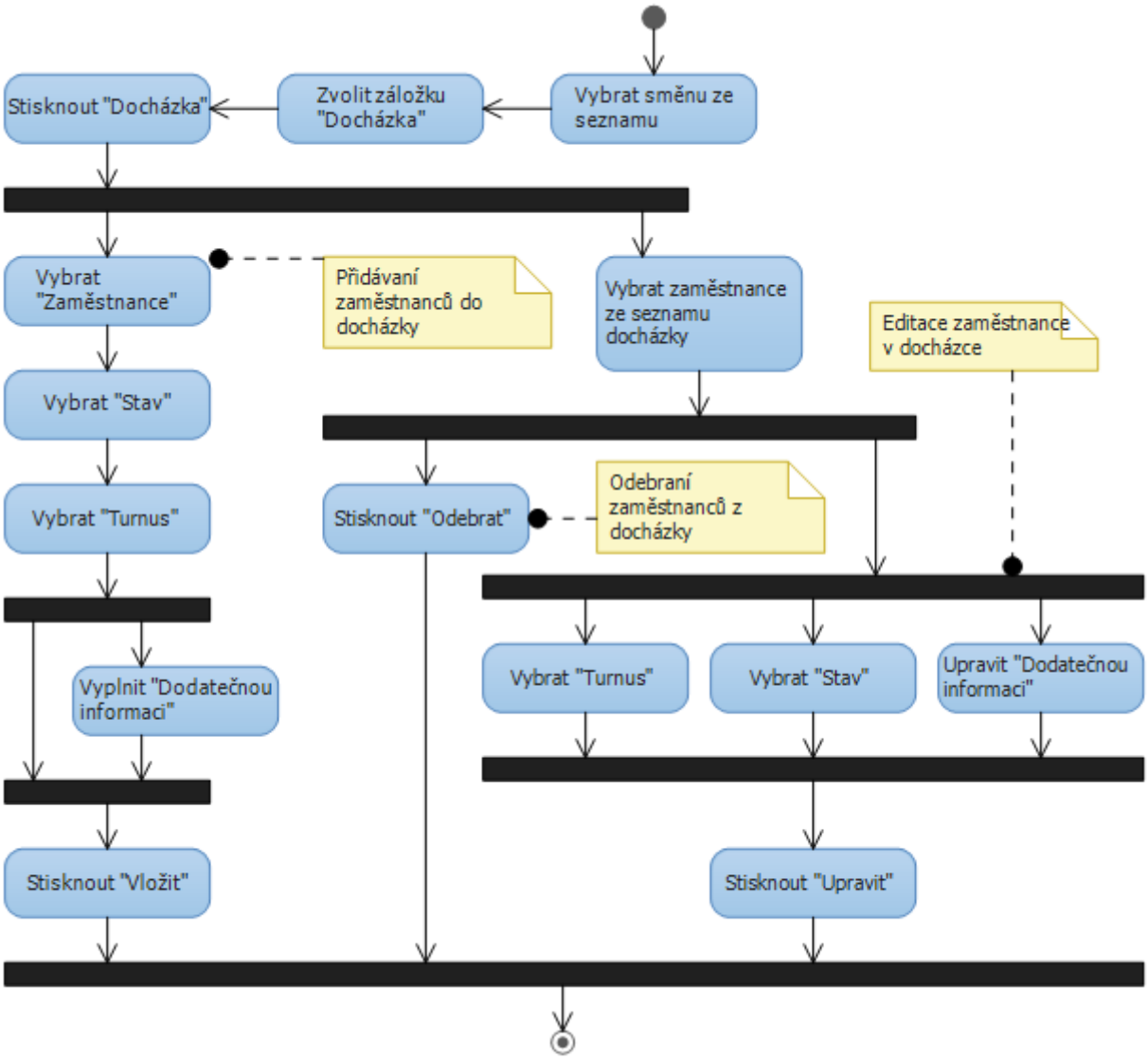


Obrázek 2.14: Diagram hledání

2.9.9 Diagram docházky

Proces docházky zahájíme výběrem směny, pro kterou chceme založit docházku. Po výběru směny je třeba vybrat záložku „Docházka“ na pravé straně hlavního okna. Nyní už stačí stisknout tlačítko „Docházka“, které se nachází na hlavní liště v podobě ikony *docházka* [1] , nebo v nabídce „Nastroje“ v hlavním menu. Po otevření dialogového okna se v pravém seznamu objeví seznam zaměstnanců, který můžeme zařadit do docházky vybrané směny. Po vybrání zaměstnance, kterého chceme přidat do docházky, se červeně orámují povinné údaje, které je nutné doplnit (konkretně se jedná o „Status“ a „Turnus“). Zde se předpokládá, že jsou již vytvořeny „Statusy“ a „Zaměstnanci“, kde jejich diagramy aktivit jsou popsány v jiných diagramech aktivit. Položka „Status“ určuje přítomnost zaměstnance (např. přítomen, nepřítomen, nemocen). Položka „Turnus“ určuje, v jakém turnusu se momentálně zaměstnanec nachází. Zde je umožněno vybrat jen ty turnusy, které má zaměstnanec k sobě přiřazeny. Také je možné vyplnit dodatečnou informaci do pole „Dodatečná informace“, kde může být například uvedeno, že si vzal zaměstnanec na půl dne volno. Po vyplnění všech potřebných údajů a stisknutí tlačítka „Vložit“ se zaměstnanec objeví jako vybraná položka ve spodním seznamu, který je seznamem docházky vybrané směny. Pro odebrání zaměstnance z docházky vybereme zaměstnance ze spodního seznamu a klikneme na tlačítko „Odebrat“. V případě potřeby editace vybereme zaměstnance ve spodním seznamu a upravíme kterýkoliv ze tří údajů („Turnus“, „Status“, „Dodatečná informace“). Po dokončení úprav stiskneme tlačítko „Uložit“.

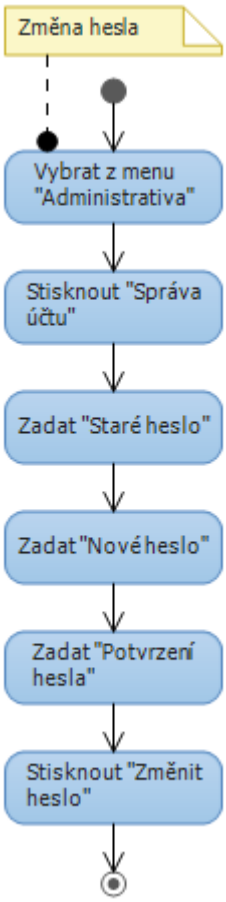
[1]  ikona docházky.



Obrázek 2.15: Diagram docházky

2.9.10 Diagram změny hesla

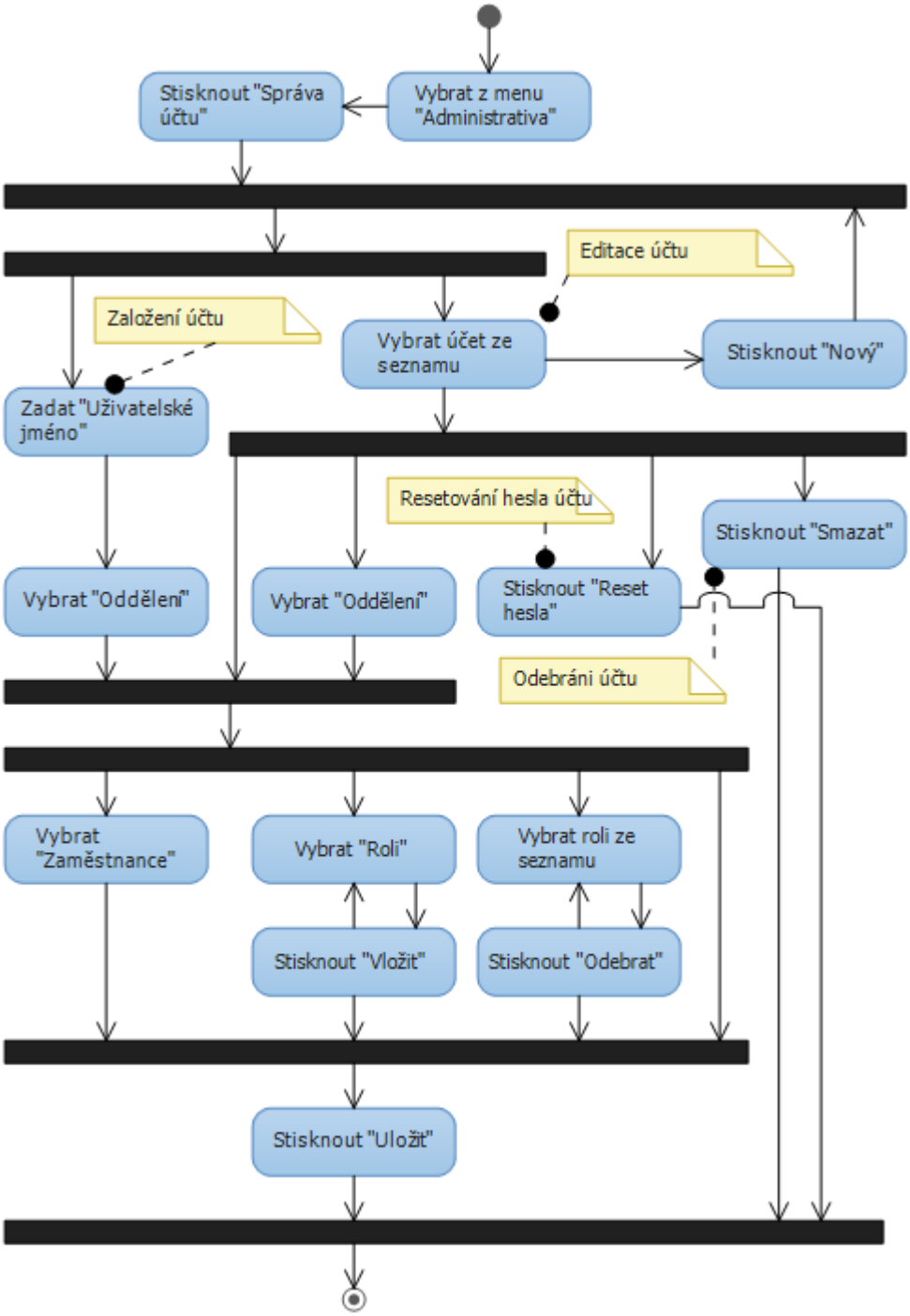
Proces změny hesla zahájíme stisknutím záložky „Můj účet“, která se nachází v nabídce „Administrativa“ v hlavním menu aplikace. Po otevření dialogového okna vyplníme povinné údaje, které jsou označené červeným rámečkem, a stisknutím tlačítka „Změnit heslo“ se začne provádět kontrola zadaných parametrů. Pokud vše proběhne úspěšně, tak se objeví nápis ”Heslo bylo úspěšně změněno!”.



Obrázek 2.16: Diagram změny hesla

2.9.11 Diagram práce s uživatelskými účty

Proces práce s uživatelskými účty zahájíme stisknutím záložky „Správa účtu“, která se nachází v nabídce „Administrativa“ v hlavním menu. V otevřeném dialogovém okně založíme účet tím, že vyplníme minimálně všechna povinná pole orámované červeným rámečkem, kterými jsou „Uživatelské jméno“ a „Oddělení“, do kterého se účet zařadí. Také je možné uživatelský účet svázat s konkrétním zaměstnancem výběrem zaměstnance. To zapříčiní, že pokud se odebere zaměstnanec, tak se automaticky odebere i účet, který je tomuto zaměstnanci přiřazen. Pokud uživatelský účet nechceme mít jen pro vstup do systému a čtení záznamu, je možné mu přiřadit různé role. Role se dají vybrat v dialogovém okně v sekci „Role“ a vkládat je k uživatelskému účtu přes tlačítko „Vložit“. Po zadání všech potřebných parametrů a stisknutím tlačítka „Uložit“ se uživatelský účet zobrazí v pravém seznamu jako vybraná položka. Zároveň se v sekci „Heslo“ objeví vygenerované heslo. Pokud chceme založit nového uživatele a je již vybrán nějaký uživatel, stačí stisknout tlačítko „Nový“. Pro reset hesla uživatele vybereme uživatele z pravého seznamu, kterému chceme vyresetovat heslo, a stiskneme tlačítko „Reset hesla“. Pokud chceme uživatele smazat, vybereme jej z pravého seznamu a stiskneme tlačítko „Smazat“. Pokud chceme editovat uživatele, vybereme jej z pravého seznamu uživatelů a úpravou kterýchkoliv z parametrů „Oddělení“, „Role“, „Zaměstnanec“ a následným stisknutím tlačítka „Uložit“ se veškeré změny uloží.



Obrázek 2.17: Diagram práce s uživatelskými účty

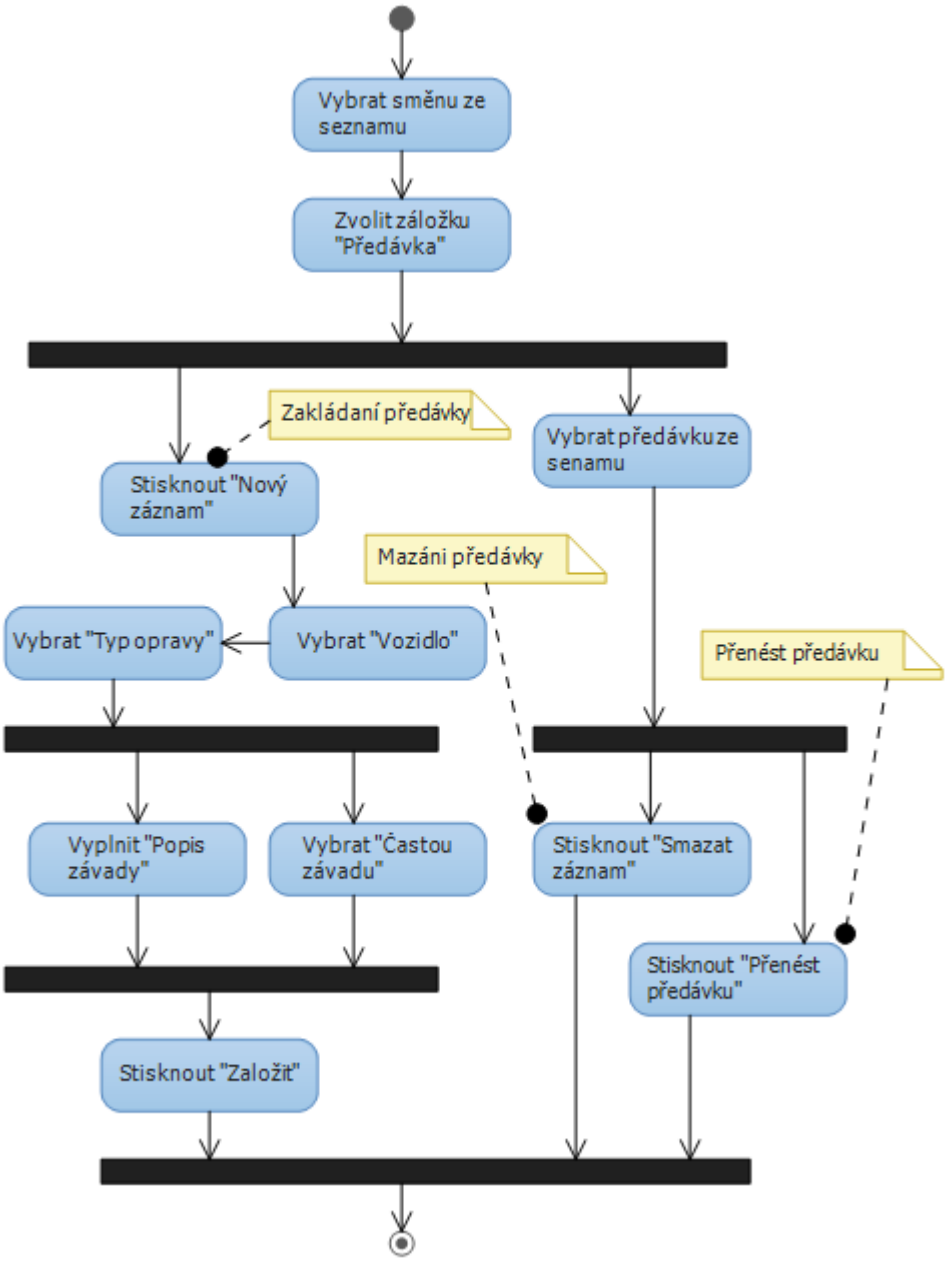
2.9.12 Diagram předávané práce

Proces zakládání předávané práce začíná výběrem směny, která bude určovat údaj, ve které směně byla předávaná práce přidaná. Také je třeba vybrat záložku „Předávka“ v pravé části obrazovky. Nyní už stačí stisknout tlačítko „Nový záznam“, které se nachází v hlavní liště vyobrazené ikonou *nového záznamu* [1] , nebo přes nabídku „Nástroje“ v hlavním menu. Po otevření dialogového okna pro zakládání předávané práce je třeba vyplnit povinné udaje, které jsou orámované červenou barvou. Jedná se o „Vozidlo“, „Typ opravy“ a „Popis závady“, kde je možné buď vybrat popis „Časté závady“ nebo napsat vlastní. Po vyplnění všech potřebných údajů a stisknutí tlačítka „Založit“ se záznam objeví na pravé straně okna v záložce „Předávka“. Proces mazání předávané práce spočívá ve výběru konkrétního záznamu v záložce „Předávka“ a stisknutím tlačítka „Smazat záznam“, které je možné nalézt buď v hlavni liště v podobě ikony *smazat záznam* [2] nebo přes nabídku „Nástroje“ v hlavním menu. Pro převod předávané práce do záznamu vykonané práce musí byt vybraná směna, do které chceme předávanou práci přenést. Také je třeba vybrat záznam v záložce „Předávka“, který má být přenesen do vykonané práce vybrané směny. Stisknutím tlačítka „Přenést“, které se nachází buď v hlavní liště jako ikona *přenést záznam* [3] nebo v nabídce „Nástroje“ hlavního menu, se vybraná předávaná práce přenese do vykonaných prací vybrané směny.

[1] ikona nového záznamu.

[2] ikona odebraní záznamu.

[3] ikona přenesení záznamu.



Obrázek 2.18: Diagram předávané práce

Kapitola 3

Popis řešení

3.1 Použité nástroje

Aplikaci jsem programoval v prostředí „Visual Studio 2012“. Toto prostředí jsem si vybral z důvodu, že je vhodné pro návrh aplikací v jazyce C# a velmi výrazně ulehčuje práci při programování. Postup vývoje jsem verzoval pomocí „Gitu“ (bezplatný systém pro správu verzí projektu) a zálohoval na „SkyDrive“ [1], který je cloudové úložiště, jenž nabízí 7 GB prostoru zdarma. Pro komunikaci přes https jsem si musel vytvořit certifikát. K tomu jsem použil nástroj „SelfCert“ [2] , který umožňuje vytvářet certifikáty sám sebou podepsané. Tyto certifikáty se používají jen pro testovací účely, protože nejsou důvěryhodné. Použil jsem je i při ostrém provozu z finančních důvodů. Celý projekt, který jsem dělal, nikdo nefinancoval, a proto jsem nepoužil certifikát podepsaný autorizační službou. Databázi jsem vytvářel v prostředí „SQL Server managmet studio“. Pro kreslení některých diagramů jsem použil „Microsoft Visio“. Nápovědu pro klientskou aplikaci jsem vytvořil v nástroji „HTML Help WorkShop“. Postup, jak pracovat s nástrojem „HTML Help WorkShop“, je velmi dobře popsán v materiálech [3].

3.2 Bezpečnost

Bezpečnost proti úniku dat zajišťuje autentizace a autorizace, jenž omezují přístup pouze těm, kteří mají uživatelský účet a práva vykonávat určitou operaci. Co se týče komunikace mezi serverem a klientem, zde je bezpečnost zabezpečena protokolem https. Uživatelská hesla se do databáze ukládají jako hash. Hash se vytváří kombinací uživatelského jména a hesla. Tím je zajištěno to, že pokud budou mít dva uživatelé stejná hesla, tak jejich hashe se budou i v tomto případě lišit. K vytváření hashe jsem použil dva algoritmy *MD5* [1] a *SHA1* [2] . Více informací o kryptografické funkcí „SHA“ se lze dočíst v materiálech [4] a o „MD5“ v materiálech [5]. Algoritmem MD5 se vytváří hash kombinace uživatelského jména a hesla. Algoritmem SHA1 se vytváří hash z vygenerovaného MD5 hashe. Použil jsem tento postup pro zajištění vyšší bezpečnosti. Heslo se po síti posílá jen v případě založení účtu nebo resetování hesla, kde služba vygeneruje náhodné důvěryhodné heslo, které odešle klientské aplikaci. Toto heslo je doporučené při prvním přihlášení změnit. Některé metody na službách mohou vyvolávat jen uživatelé s potřebným oprávněním. Při pokusu o vyvolání metody se provádí autorizace, a pokud autorizace proběhne neúspěšně, zakáže se přístup k této metodě.

3.3 Služby

Služby jsem programoval v technologii *WCF* [1] . WCF je technologie, která umožňuje vytvářet servisně orientované aplikace. Skládá se ze seznamu endpointů (koncové body), přes které je možné se službou komunikovat. Každý endpoint se skládá z adresy, bindingu a contractu. Adresa určuje umístění služby, tj. kam se mají odesílat zprávy. Binding určuje chování konkrétního endpointu, tedy dle jakého protokolu se budou posílat zprávy, jaké se použije zabezpečení, kódování, sessions atd. Jako poslední vlastností enpointu je contract, což je rozhraní, které služba poskytuje pro komunikaci. WCF služba komunikuje pomocí *SOAP* [2] zpráv, což je protokol pro přenášení dat ve tvaru *XML* [3] přes síť. Více informací o „WCF“ lze nalézt v knize [6].

[1] MD5 (Message digest algorithm) je kryptografický algoritmus, který ze vstupních dat vytvoří výstup pevné délky, který se nazývá hash.

[2] SHA1 (Secure hash algorithm) je též kryptografickým algoritmem podobný MD5.

3.3.1 Transakční zpracování

Veškeré operace prováděné s daty jsou hlídány pomocí návrhového vzoru „Unit of Work“, která zajišťuje transakční zpracování. To znamená, že se zaznamenávají prováděné změny a až po potvrzení se veškeré změny najednou uloží do databáze. Pokud by některá operace neprošla úspěšně, tak se zahodí veškeré změny. Tím se zajistí, že se do databáze neuloží nekonzistentní data. Návrhový vzor „Unit Of Work“ je popsán v materiálech [7]. „Unit of Work“ přistupuje k datům přes návrhový vzor „Repository“, což je rozhraní, které umožňuje základní *CRUD* [1] operace nad daty. Výhoda „Repozitory“ je v tom, že nabízí rozhraní s práci nad daty, které je stále stejné. Ale přitom nabízí více možností implementace tohoto rozhraní k přístupu k datům. Více o návrhovém vzoru „Repository“ se lze dočíst v materiálech [8]. Moje navržené „Repository“ přistupuje k datům přes *Entity Framework* [2] , který namapuje databázové tabulky na kolekce a umožňuje pracovat s jednotlivými řádky jako s objekty. Více informaci o „Entity Frameworku“ se lze dočíst v knize [9].

3.3.2 Konfigurace služeb

Konfigurace služeb je nastavena v souboru web.config v sekci „services“, Každé službě je nastavené chováni „behaviorConfiguration“ a také koncové body „endpoint“. V koncových bodech je nastaven typ chování komunikace, zabezpečení, certifikát, který se má použít, a rozhraní, které se nabízí pro komunikaci. Pro každou službu jsou přístupny dva koncové body, druhy s rozhraním „ImetadataExchange“ zpřístupňuje metadata, která popisuji samotnou službu. Pomocí těchto dat je možné vygenerovat proxy třídu, která slouží jako prostředník mezi službou a klientem. Také zde jsou uloženy v sekci „connectionStrings“ řetězce pro připojení k databázovému serveru.

[1] WCF (Windows Communication Foundation) je technologie pro tvorbu servisně orientovaných aplikaci.

[2] SOAP (Simple Object Access Protocol).

[3] XML (Extensible Markup Language).

[1] CRUD (Creat,Read,Update,Delete) je zkratka znamenající základní operace nad daty kterými jsou vytvoření, čtení, editace a mazání.

[2] Entity Framework je ORM (object relation maper).

3.3.3 Validace dat

Proto, aby se neposílaly celé entity přes „SOAP“ zprávy a nepřenášely se tak nevyužitá data, jsem vytvořil ořezané objekty pro přenos jen potřebných informací. Tyto vytvořené objekty jsem validoval pomocí anotace tříd, jenž se provádí pomocí atributů, které se vkládají nad konkrétními vlastnostmi dané třídy. Validace se pak provádí vyvoláním metody „IsValid“, které se předá objekt určený pro validaci. Pokud validace neproběhne úspěšně, vyvolá se výjimka, která se odešle pomocí „WCF“ jako „FaultContact“.

3.3.4 Hostování

Služby jsem hostoval na „IIS 7“. Pro bezpečnou komunikaci jsem zvolil protokol https. Aby fungovala bezpečná komunikace, musel jsem vytvořit certifikát pomocí programu „SelfCert“. Poté bylo potřeba vytvořit hostovací prostředí. V „IIS 7“ jsem vytvořil web, kterému jsem nastavil název, IP adresu, protokol „https“, port a vybral vytvořený certifikát. Musel jsem nastavit práva certifikátu, aby ho hostovací web mohl použít. Také jsem musel v databázi vytvořit účet, který umožní přístup k databázi. Pro hostování služeb v „IIS 7“ jsem založil v „Visual Studio 2012 ASP.NET“ projekt, který je v webovým projektem, ve kterém jsou hostované jednotlivé služby. V tomto projektu jsem nareferencoval jednotlivé služby, vytvořil „.svc“ soubory které definují služby a nakonfiguroval je v souboru web.config. Posledním krokem je již jen publikováni projektu do adresáře s vytvořeným webem na „IIS 7“.

3.3.5 Bezpečnostní služba

Práci s uživatelskými účty zajišťuje samostatná služba, která umožňuje prohlížet, zakládat, editovat a mazat uživatelské účty s možností resetovat hesla jednotlivých uživatelských účtů a to za podmínky, že přihlášený uživatel má práva k těmto funkcím. Pro vygenerování bezpečného hesla jsem naprogramoval funkci, která z pole znaků vygeneruje náhodné heslo zadané délky. Tato funkce se používá při zakládání nebo resetování hesla. Seznam znaků, ze kterých se vytváří heslo a kód generátoru, je možné vidět v kódu 3.3.5.

```
private const string Chars =
@"abcdefghijkmnopqrstuvwxyzABCDEFGHJKLMNOPQRSTUVWXYZ0123456789!#$%&'()*+,-./:;<=>?@[\]_";

private void GeneratePassword(int length)
{
        Random rand = new Random((int)DateTime.Now.Ticks & 0x0000FFFF);
        var password = new char[length];
        for (var i = 0; i < length; i++)
        {
                password[i] = Chars[rand.Next(0, Chars.Length)];
        }
        Password = new String(password);
}
```

Služba nabízí pro komunikaci rozhraní „ISecurityService“, které umožňuje volat následující metody. Pro volání některých metod je třeba mít účet, který je zařazen do určité role.

Dostupné metody pro práci s uživatelskými účty:

- ***Objects.User AddUser(Objects.User user)*** založí v databázi uživatele a vrací tento objekt s vygenerovaným heslem. Tuto metodu mohou volat jen uživatele s rolí *Administrator*.
- ***void DeleteUser(Int32 userID)*** smaže uživatelský účet s odpovídajícím ID. Tuto metodu mohou volat jen uživatelé s rolí *Administrator*.
- ***string ResetPassword(Int32 userID)*** vygeneruje a nastaví heslo uživateli se zadaným ID a zároveň vygenerované heslo vrací jako řetězec. Tuto metodu mohou volat jen uživatele s rolí *Administrator*.
- ***void UpdateUser(Objects.User user)*** aktualizuje uživatelský účet dle předaného objektu. Tuto metodu mohou volat jen uživatele s rolí *Administrator*.
- ***List<Objects.Role> GetRoles()*** vrací kolekci uživatelských rolí. Tuto metodu mohou volat jen uživatelé s rolí *Administrator*.
- ***List<Objects.User> GetUsers()*** vrací kolekci uživatelských účtu bez hesel. Tuto metodu mohou volat jen uživatele s rolí *Administrator*.
- ***Objects.User GetUserInfo(string userName)*** vrací uživatelský účet bez hesla, kterému odpovídá uživatelské jméno v zadaném parametru. Tuto metodu může volat kterýkoliv uživatel.

- ***bool ChangePassword(Objects.User user)*** změní uživatelské heslo které je je obsaženo v předaném objektu. Toto heslo je jíž v podobě hashe. Metoda vrací informaci, zda se změna hesla povedla. Tuto metodu může volat kterýkoliv uživatel.

3.3.6 Vyhledávací služba

Vyhledávání záznamu zajišťuje samostatná služba poskytující hledání záznamu podle různých parametrů. Je možné libovolně nastavovat a hledat podle parametrů: vozidla, typu opravy, data od do, textového výrazu, který je obsažen v popisu závady. Pro uživatelské účty s rolí „Mistr“ je možné hledat záznamy v konkrétních odděleních. Účty, které nejsou zařazeny do role „Mistr“, toto právo nemají a vyhledávají záznamy vždy jen v oddělení, ve kterém jsou samy zařazeny.

Služba nabízí pro komunikaci rozhraní „IFullTextService“, která umožňuje volat metodu:

List<Objects.FullTextWork> FindWork(Parameters parameters). Tato metoda umožňuje vyhledávání vykonané práce dle zadaných parametru, které jsou předané metodě. Tuto metodu muže volat jakýkoliv uživatel.

3.3.7 Datová služba

Další služba zajišťuje veškerou datovou komunikaci, kterými jsou např. zápis, editace a mazání směn, vozidel, turnusů, typu oprav, zaměstnanců a oddělení.

Dostupné metody pro práci s docházku:

- ***void AddAttendance(Objects.Attendance attendance)*** založí docházku dle předaného objektu. Tuto metodu mohou volat jen uživatele s rolí *Zápis*.

- ***void DeleteAttendance(Objects.Attendance attendance)*** smaže docházku s odpovídající kombinaci ID směny a ID zaměstnance, které jsou obsahem předaného objektu. Tuto metodu mohou volat jen uživatele s rolí *Zápis*.

- ***void UpdateAttendance(Objects.Attendance attendance)*** aplikuje změny docházky s odpovídající kombinaci ID zaměstnance a ID směny, které jsou také obsahem předaného objektu. Tuto metodu mohou volat jen uživatele s rolí *Zápis*.
- ***List<Objects.Attendance> FindAttendances(Int64 dateID,Int32 detachmentID)*** vrací kolekci docházky, které jsou obsažené v zadané směně a oddělení. Tuto metodu může volat jakýkoliv uživatel.

Dostupné metody pro práci se směnami:

- ***Int64 AddDate(Objects.Date date)*** založí směnu dle předaného objektu a vrátí ID založené směny. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void DeleteDate(Int64 dateID)*** smaže směnu s odpovídajícím ID. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void UpdateDate(Objects.Date date)*** aktualizuje směnu s odpovídajícím ID dle předaného objektu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***List<Int32> GetYear();*** vrací kolekci roků, ve kterých jsou založeny směny. Tuto metodu může volat jakýkoliv uživatel.
- ***List<Int32> GetMonth(Int32 year)*** vrací kolekci měsíců zadaného roku, ve kterém jsou založeny směny. Tuto metodu může volat jakýkoliv uživatel.
- ***List<Objects.Date> FindDates(Int32 year,Int32 month)*** vrací kolekci směn, které jsou obsažené v zadaném roku a měsíci. Tuto metodu může volat jakýkoliv uživatel.

Dostupné metody pro práci s předávanou prací:

- ***Int32 AddChangeover(Objects.Changeover changeover)*** založí předávanou práci dle předaného objektu a vrátí ID založené předávané práce. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void DeleteChangeover(Int32 changeoverID)*** smaže předávanou práci se zadaným ID. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void UpdateChangeover(Objects.Changeover changeover);*** aktualizuje předávanou práci s odpovídajícím ID dle předaného objektu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.

- ***Objects.Work TransferChangeover(Int64 dateID,Int32 changeoverID)*** převede předávanou práci s odpovídajícím ID jako vykonanou práci do směny s odpovídajícím ID. Metoda též vrací objekt převedeného záznamu v podobě vykonané práce. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.

- ***List<Objects.Changeover> FindChangeovers(Int32 detachmentID)*** vrací kolekci předávaných prací, které spadají do oddělení se zadaným ID. Tuto metodu může volat jakýkoliv uživatel.

Dostupné metody pro práci s oddělenými:

- ***Int32 AddDetachment(Objects.Detachment detachment)*** založí oddělení dle předaného objektu a vrátí ID založeného oddělení. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.

- ***void DeleteDetachment(Int32 detachmentID)*** smaže oddělení se zadaným ID. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.

- ***void UpdateDetachment(Objects.Detachment detachment)*** aktualizuje oddělení s odpovídajícím ID dle předaného objektu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.

- ***Objects.Detachment FindDetachment(Int32 detachmentID)*** vrací objekt oddělení dle zadaného ID. Tuto metodu může volat jakýkoliv uživatel.

- ***List<Objects.Detachment> GetDetachments()*** vrací kolekci oddělení. Tuto metodu může volat jakýkoliv uživatel

Dostupné metody pro práci s turnusy:

- ***Int32 AddTour(Objects.Tour tour)*** založí turnus dle předaného objektu a vrátí ID založeného turnusu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.

- ***void DeleteTour(Int32 tourID)*** smaže turnus se zadaným ID. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.

- ***void UpdateTour(Objects.Tour tour)*** aktualizuje turnus s odpovídajícím ID dle předaného objektu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.

- ***List<Objects.Tour> GetTours()*** vrací kolekci turnusu. Tuto metodu může volat jakýkoliv uživatel

Dostupné metody pro práci s vozidly:

- ***Int32 AddVehicle(Objects.Vehicle vehicle)*** založí vozidlo dle předaného objektu a vrátí ID založeného turnusu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void DeleteVehicle(Int32 vehicleID)*** smaže vozidlo se zadaným ID. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void UpdateVehicle(Objects.Vehicle vehicle)*** aktualizuje vozidlo s odpovídajícím ID dle předaného objektu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***List<Objects.Vehicle> GetVehicles()*** vrací kolekci vozidel. Tuto metodu může volat jakýkoliv uživatel.

Dostupné metody pro práci s pracemi:

- ***Int64 AddWork(Objects.Work work)*** založí práci dle předaného objektu a vrátí ID založené práce. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void DeleteWork(Int64 workID)*** smaže práci se zadaným ID. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void UpdateWork(Objects.Work work)*** aktualizuje práci s odpovídajícím ID dle předaného objektu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***List<Objects.Work> FindWorks(Int64 dateID,Int32 detachmentID)*** vrací kolekci prací, které spadají do oddělení se zadaným ID a také do směny se zadaným ID . Tuto metodu může volat jakýkoliv uživatel.
- ***List<string> GetTopFaultWorks(Int32 count,Int32 detachmentID)*** vrací kolekci nejčastějších závad, které jsou uloženy v databázi. Kde „count“ určuje maximální množství záznamů, které se má vracet, a „detachmendID“ určuje, v jakém oddělení se mají hledat nejčastější závady.
- ***List<string> GetTopCauseWorks(Int32 count,Int32 detachmentID)*** vrací kolekci nejčastějších oprav, které jsou uloženy v databázi. Kde „count“ určuje maximální množství záznamů, které se má vracet, a „detachmendID“ určuje, v jakém oddělení se mají hledat nejčastější opravy.

Dostupné metody pro práci se zaměstnanci:

- ***Int32 AddWorker(Objects.Worker Detailworker)*** založí zaměstnance dle předaného objektu a vrátí ID založeného zaměstnance. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void DeleteWorker(Int32 workerID)*** smaže zaměstnance se zadaným ID. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void UpdateWorker(Objects.Worker Detailworker)*** aktualizuje zaměstnance s odpovídajícím ID dle předaného objektu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***Objects.WorkerDetail FindWorker(Int32 workerID)*** vrací detailní informace o zaměstnanci se zadaným ID. Tuto metodu může volat jakýkoliv uživatel.
- ***List<Objects.Worker> FindWorkers(Int32 detachmentID)*** vrací kolekci zaměstnanců, kteří spadají do oddělení se zadaným ID. Tuto metodu může volat jakýkoliv uživatel.

Dostupné metody pro práci se stavy zaměstnanců:

- ***Int32 AddWorkerState(Objects.WorkerState workerState)*** založí stav zaměstnance dle předaného objektu a vrátí ID založeného stavu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void DeleteWorkerState(Int32 workerStateID)*** smaže stav zaměstnance se zadaným ID. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void UpdateWorkerState(Objects.WorkerState workerState)*** aktualizuje stav zaměstnance s odpovídajícím ID dle předaného objektu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***List<Objects.WorkerState> GetWorkerStates()*** vrací kolekci stavů zaměstnanců. Tuto metodu může volat jakýkoliv uživatel.

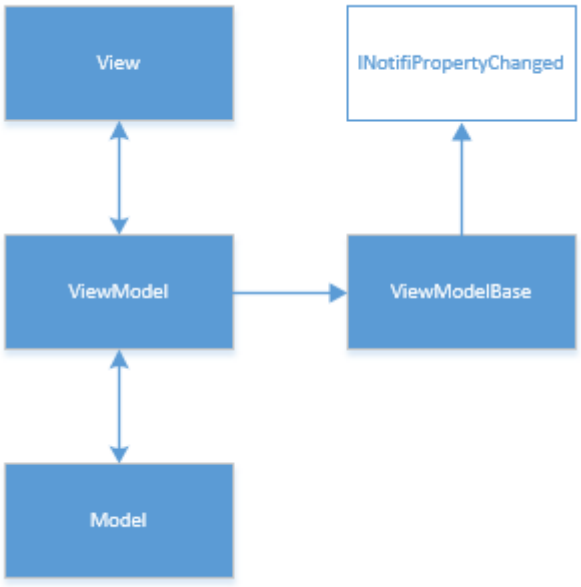
Dostupné metody pro práci s typy oprav:

- ***Int32 AddWorkType(Objects.WorkType workType)***založí typ opravy dle předaného objektu a vrátí ID založeného typu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.

- ***void DeleteWorkType(Int32 workTypeID)***smaže typ opravy se zadaným ID. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***void UpdateWorkType(Objects.WorkType workType)***aktualizuje typ opravy s odpovídajícím ID dle předaného objektu. Tuto metodu mohou volat jen uživatelé s rolí *Zápis*.
- ***List<Objects.WorkType> GetWorkTypes()***vrací kolekci typů oprav. Tuto metodu může volat jakýkoliv uživatel.

3.4 Klient

Klientská aplikace je naprogramovaná pomocí technologie *WPF* [1] .Více informací o „WPF“ lze nalézt v knize [10] a materiálech [11]. Při implementaci klientské aplikace jsem musel často upravovat některé části kódu hlavně z důvodu zjištění, že můj způsob programování nebyl zrovna ten správný, který se běžně používá. Jelikož jsem chtěl mít aplikaci, která bude jednoduše rozšířitelná, a také abych se naučil správně programovat, dával jsem na tyto věci velký důraz a tyto částí přepisoval. Aby se aplikace při provádění nějakého dotazu na server nezasekávala, tak veškerá dotazování se prováděly asynchronně. Prováděním jakékoliv asynchronní operace je uživatel upozorněn obrazovkou, která se překreslí přes aktuálně otevřené okno s nápisem, že program pracuje. Jako návrhový vzor se použil „MVVM“, který umožňuje oddělit logiku aplikace od jejího vzhledu. Tento model se skládá ze tři části „View“, „Model“ a „ViewModel“. „View“ je grafické uživatelské rozhraní. „View“ se ve „WPF“ píše v jazyce „XAML“, který je pro tento účel velmi dobře přizpůsobený. Model tvoří konkrétní data, se kterými aplikace pracuje. V mém případě jsou to služby, ke kterým se aplikace připojuje a žádá, případně odesílá data. Model by neměl nic vědět o „View“. „ViewModel“ je nejdůležitější částí, jelikož zprostředkovává komunikaci mezi uživatelským rozhraním „View“ a daty „Modelem“. Pro svázání „ViewModelu“ a „View“ se používá „Binding“ a pomocí „Commandu“ je možné z „View“ vyvolat metodu ve „ViewModelu“, která se provede třeba při stisku tlačítka. Další informace o návrhovém vzoru „MVVM“ se lze dočíst v knize [12].

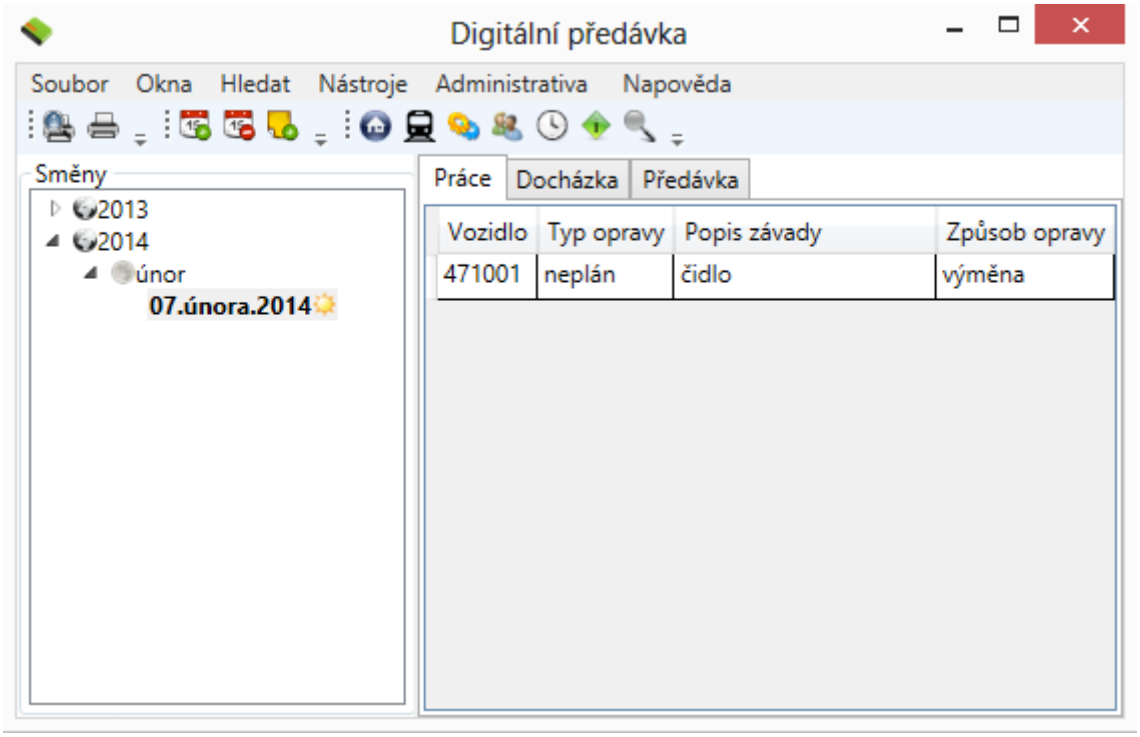


Obrázek 3.1: Návrhový vzor MVVM

[1] WPF (Windows Presentation Foundation) je podmnožina .NET frameworku, která používá pro definici vzhledu jazyk XAML [zaml].

3.4.1 Uživatelské rozhraní

Uživatelské rozhraní bylo psáno v jazyce, který se nazývá *XAML* [1] . Vše, co lze napsat v jazyce „XAML“, lze napsat i v jazyce „C#“ přímo do kódu. Umožňuje velmi rychle vytvářet různorodou a zajímavou grafiku. Více informací o „XAML“ se lze dočíst v knize [10]. Návrh, který jsem vytvořil, je podobný běžným aplikacím. Veškeré ikony které zde byly použity, jsou určeny k použití zdarma i pro komerční užití. Ikony použité v aplikaci byly stažený ze stránek [13]. Klient po přihlášení automaticky rozpozná uživatelská práva a dle toho přizpůsobí uživatelské rozhraní. Ikony se také mění dle toho, jaká záložka je momentálně vybrána. V aplikaci jsou směny uchovány v přehledné stromové struktuře, jak je zobrazeno na obrázku 3.2, kde dle data je možné velmi rychle najít konkrétní směnu. Směny mohou být jak noční tak i denní, kde jejich druh zaznamenává ikona ve smyslu slunce (denní) nebo měsíce (noční). Napravo se pak objevují ve vybrané záložce záznamy vybrané směny.



Obrázek 3.2: Uživatelské rozhraní

[1] XAML je značkovací jazyk podobný HTML založený na XML, který je vyvíjen Microsoftem, a je součástí .NET frameworku.

3.4.2 Konfigurace klienta

Proto, aby klient mohl komunikovat se službami, je třeba neimplementovat pro komunikaci s každou službou proxy třídu, která funguje jako prostředník mezi službou a klientem. Proxy třídu můžeme buď napsat sami nebo si ji můžeme nechat vygenerovat z metadat. V mém případě jsem použil pro vygenerovaní proxy tříd „Visual Studio 2012“, kde stačí nareferencovat služby a „Visual Studio 2012“ už sám pomocí metadat vygeneruje jednotlivé proxy třídy. Přitom se automaticky vygeneruje i konfigurace do souboru app.config, ve kterém jsou nastavené jednotlivé koncové body.

3.4.3 Nápověda

Nápovědu pro klientskou aplikaci jsem vytvářet pomoci programu „HTML Help WorkShop“, který dokáže z „HTML“ stránek vygenerovat soubor typu „.chm“, který se standardně ve windows používá pro nápovědu. Nástroj „HTML Help WorkShop“ umožňuje vytvářet stromovou strukturu pro prohlížení jednotlivých souboru „HTML“, také se v něm dá vytvořit rejstřík a zapnout možnost pro vyhledávání v celé nápovědě. Jak pracovat s nástrojem „HTML Help WorkShop“ je velmi dobře popsán v materiálech [3]. Nápovědu lze spustit z klientské aplikace přes nabídku „O aplikaci“ stisknutím položky „Nápověda“ v hlavním menu.

3.4.4 Validace vstupu

Veškerá zadávací pole se validují pomocí „Fluent Validation“. „Fluent Validation“ je validační knihovna nabízející rozhraní pro plynulou validaci objektů pomocí lambda výrazu. „Fluent Validation“ je popsána v materiálech [14]. V projektu jsou vytvořeny pro většinu „ViewModelu“ validační třídy, které dědí z generické validační třídy „AbstractValidator“, která umožňuje vyvolat validaci nad „ViewModelem“. Každý „ViewModel“ dědí z „BaseViewModelu“, který také implementuje validační rozhraní „IDataErrorInfo“. Toto rozhraní umožňuje validovat konkrétní vlastnosti „ViewModelu“, které jsou nabaindovány ve „View“. Pokud validace neproběhne úspěšně nad některou z vlastností, která je nabaindována na ovládací prvek, tak se zobrazí error template, který jsem si nadefinoval jako červený rámeček kolem ovládací kontrolky s popisem chyby v tooltypu.

3.4.5 Instalace klienta

Klientská aplikace je publikovaná pomocí technologie „ClickOnce“ jako webová služba na serveru, která je hostovaná na „IIS 7“. Z tohoto webu je možné si nainstalovat klientskou aplikaci po stisknutí tlačítka „Install“. Instalátor sám rozpozná potřebu instalace „Net frameworku“ a nainstaluje ji dle potřeby. Teprve poté se nainstaluje klientská aplikace. Po každém spuštění klientské aplikace se zkontroluje, zda není na serveru nová verze. Pokud ano, nabídne se uživateli k instalaci. Více informací o „ClickOnce“ se nalézá v materiálech [15].

3.4.6 Odinstalace klienta

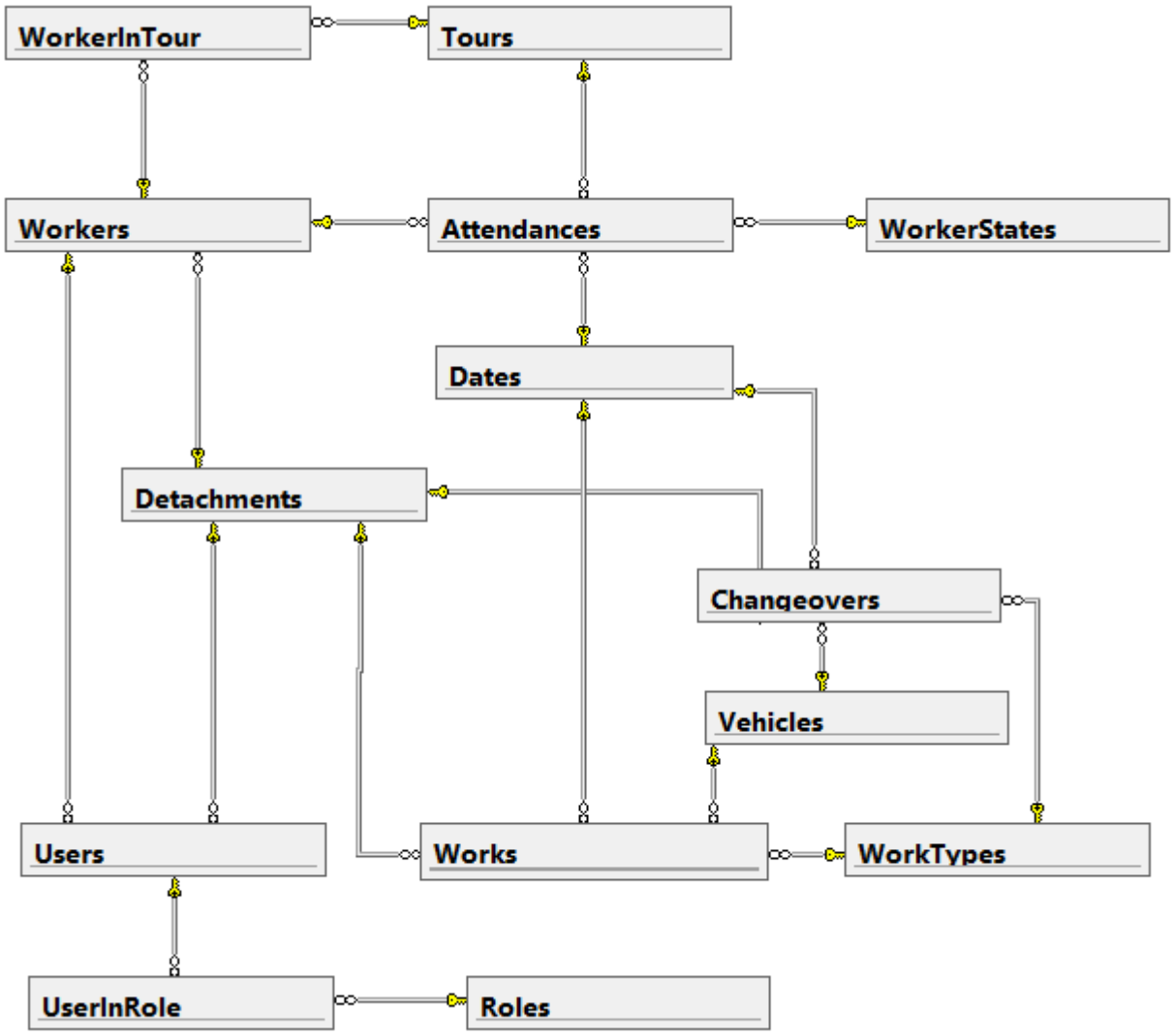
Klientskou aplikaci je možné odinstalovat jako každou jinou aplikaci ve windows. Pokud v průběhu provozu klientské aplikace byly nainstalovány nové verze, je možné se při odinstalování vrátit ke starším verzím programu. Tato volba se zobrazí při odinstalování klientské aplikace.

3.4.7 Verzování klientské aplikace

Jak již bylo zmíněno výše, technologie „ClickOnce“ umožňuje verzovat publikovaný software. Tím pádem je možné si nastavit automatické zvýšení čísla verze po každé publikaci softwaru. Takto publikovaný software si vytváří pro každou verzi novou složku, do které se tato verze uloží.

3.5 Datový model databáze

Databáze je navržená v technologii „MS SQL“ obsahující 14 tabulek. K databázi se nepřistupuje přímo ale pomoci „EntityFramework“-u, který namapuje tabulky na objekty. To znamená, že se dá přistupovat k jednotlivým tabulkám jako ke generickým kolekcím a že se dá přistupovat k jednotlivým řádkům jako k jednotlivým objektům dané tabulky, kde každý objekt má vlastnosti, které jsou ekvivalentem k atributům tabulky. Celý model databáze je možné vidět na obrázku 3.3. Více informací o databázi „MS SQL“ se lze dočíst v materiálech [16].



Obrázek 3.3: Datový model databáze

3.5.1 Atributy datového modelu databáze

Jednotlivé sloupce v tabulkách mají významy popsané níže:

- **Klič** – PK určuje, že se jedná o primární klíč. FK značí, že se jedná o cizí klíč. PFK značí, že se jedná o primární cizí klíč.
- **Název** – určuje název sloupce v tabulce.
- **Typ** – určuje datový typ sloupce.
- **Not null** – Hodnota ano značí, že sloupec musí obsahovat hodnotu, a hodnota ne značí, že sloupec nemusí obsahovat hodnotu.
- **Unikátní** – Hodnota ano značí, že sloupec musí obsahovat unikátní data, a hodnota ne značí, že se ve sloupci mohou objevovat stejná data. Sloupce označené jako AnoS označují skutečnost, že kombinace údajů v těchto sloupcích musí být unikátní.
- **Autoinkrementace** – Honota ano značí, že se automaticky generuje klíč sloupce při každém přidání záznamu.
- **Komentář** – příklad toho, co se dá do konkretního sloupce vložit.

3.5.2 Tabulka Attendances

Tabulka „Attendances“ obsahuje veškeré udaje o docházce, primárním klíčem je zde kombinace atributu „AttendanceWorkerID“, který určuje id zaměstnance, a atributu „AttendanceDateID“, který určuje id směny. Tím je zaručeno, že zaměstnanec může být zařazen do docházky směny jen jednou. Atributem „AttendanceDateID“ se určuje směna, pro kterou je určen záznam o docházce. Zaměstnanec, který patří do záznamu, je určen atributem „AttendanceWorkerID“. Stav zaměstnance v docházce se určuje atributem „AttendanceWorkerStateID“ a turnus atributem „AttendanceWorkerTourID“. Do posledního atributu „AttendanceDescription“ je možné uložit dodatečnou informaci.

Klič	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PFK	AttendanceDateID	bigint	ANO	ANOS	NE	
PFK	AttendanceWorkerID	int	ANO	ANOS	NE	
FK	AttendanceWorkerStateID	int	ANO	NE	NE	
FK	AttendanceWorkerTourID	int	ANO	NE	NE	
	AttendanceDescription	nvarchar(MAX)	NE	NE	NE	1/2 dne dovolena

Tabulka 3.1: Attendances

3.5.3 Tabulka Dates

Tabulka „Dates“ obsahuje veškeré informace o směnách. Datum směny se ukládá do sloupce „DateDate“, sloupec „DateIsNight“ určuje, o jakou směnu se jedná. Pokud je hodnota sloupce „DateIsNight“ true, tak se jedná o noční směnu, jinak se jedná o denní směnu. Také je možné do sloupce „DateDescription“ vložit dodatečné informace o směně, například zda se jedná o směnu, která se vykonává ve svátek.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	DateID	bigint	ANO	ANO	ANO	
	DateDate	date	ANO	ANOS	NE	12.12.2012
	DateIsNight	bit	ANO	ANOS	NE	false
	DateDescription	nvarchar(MAX)	NE	NE	NE	Svátek

Tabulka 3.2: Dates

3.5.4 Tabulka Detachments

Tabulka „Detachments“ udržuje veškeré informace o jednotlivých odděleních, kde se jména oddělení ukládají do sloupce „DetachmentName“. Do sloupce „DetachmentDescriptor“ je možné vložit dodatečnou informaci o oddělení.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	DetachmentID	int	ANO	ANO	ANO	
	DetachmentName	nvarchar(50)	ANO	ANO	NE	Zkušebna
	DetachmentDescription	nvarchar(MAX)	NE	NE	NE	Praha 4, Chodovská 35

Tabulka 3.3: Detachments

3.5.5 Tabulka Changeovers

Tabulka „Changeovers“ udržuje informace o předávané práci, která se v dané směně nestihla vykonat. Atribut „ChangeoverDateID“ určuje, v jaké směně byl záznam o předávce vytvořen. Atribut „ChangeoverVehicleID“ určuje vozidlo, o které se jedná, atributem „ChangeoverWorkTypeID“ se určuje, o jaký typ závady se jedná, a atribut „ChangeoverDetachmentID“ určuje, k jakému oddělení záznam patří. Pro popis závady slouží atribut „ChangeoverDescription“.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	ChangeoverID	int	ANO	ANO	ANO	
FK	ChangeoverDateID	bigint	ANO	NE	NE	
FK	ChangeoverVehicleID	int	ANO	NE	NE	
FK	ChangeoverWorkTypeID	int	ANO	NE	NE	
FK	ChangeoverDetachmentID	int	ANO	NE	NE	
	ChangeoverDescription	nvarchar(MAX)	ANO	NE	NE	Nastavit tachometr

Tabulka 3.4: Changeovers

3.5.6 Tabulka Roles

Tabulka „Roles“ obsahuje seznam dostupných rolí pro uživatelé. Tato tabulka slouží pro autorizační účely uživatelských účtů. Role jsou již na začátku přesně dány a z klientské časti je není možné měnit. Každá role umožňuje spouštět různé operace na straně služeb. Do sloupce „RoleName“ se vkládá jméno role a do sloupce „RoleDescription“ je možné vložit popis této role.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	RoleID	int	ANO	ANO	ANO	
	RoleName	nvarchar(30)	ANO	ANO	NE	Admin
	RoleDescription	nvarchar(MAX)	NE	NE	NE	Práva pro účty

Tabulka 3.5: Roles

3.5.7 Tabulka Tours

Tabulka „Tours“ obsahuje seznam turnusů. Do těchto turnusů pak mohou být zařazeny jednotliví zaměstnanci. Sloupec „TourStartTime“ určuje začátek pracovní doby a sloupec „TourEndTime“ určuje konec pracovní doby. Do sloupce „TourDescription“ je možné vložit informaci o turnusu.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	TourID	int	ANO	ANO	ANO	
	TourStartTime	time(0)	ANO	ANOS	NE	06:15:00
	TourEndTime	time(0)	ANO	ANOS	NE	18:15:00
	TourDescription	nvarchar(MAX)	NE	NE	NE	Noční směna

Tabulka 3.6: Tours

3.5.8 Tabulka UserInRole

Tabulka „UserInRole“ je vazební tabulkou, která je určená pro vztah m:n tabulek „Users“ a „Roles“. Definuje, že každému uživateli je možné přiřadit více rolí a v konkrétní

roli může být obsaženo více uživatelů.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PFK	UserInRoleRoleID	int	ANO	ANOS	NE	
PFK	UserInRoleUserID	int	ANO	ANOS	NE	

Tabulka 3.7: UserInRole

3.5.9 Tabulka Users

Tabulka „Users“ obsahuje informace o uživatelských účtech. Pomocí atributu „UserWorkerID“ je možné k uživatelskému účtu přidružit konkrétního zaměstnance. Tato vazba umožňuje automaticky smazat přidružený uživatelský účet, pokud se z databáze smaže zaměstnanec, se kterým je tento účet svázán. Atribut „UserDetachmentID“ určuje, do jakého oddělení je uživatelský účet přidružen. Tím je zajištěno, že uživatel může manipulovat s daty jen v oddělení, do kterého je zařazen. Ve sloupci „UserName“ je uloženo uživatelské jméno, ve sloupci „UserPassword“ je uloženo heslo ve tvaru hashe a do sloupce „UserDescription“ je možné doplnit dodatečné informace.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	UserID	int	ANO	ANO	ANO	
FK	UserWorkerID	int	NE	NE	NE	
FK	UserDetachmentID	int	ANO	NE	NE	
	UserName	nvarchar(30)	ANO	ANO	NE	User
	UserPassword	nvarchar(MAX)	ANO	NE	NE	
	UserDescription	nvarchar(MAX)	NE	NE	NE	Účet pro zaměstnance

Tabulka 3.8: Users

3.5.10 Tabulka Vehicles

Tabulka „Vehicles“ obsahuje veškeré informace o vozidlech. Identifikace vozidla se vkládá do sloupce „VehicleNumber“ a veškeré další informace o vozidlech se ukládají do sloupce „VehicleDescription“.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	VehicleID	int	ANO	ANO	ANO	
	VehicleNumber	nvarchar(20)	ANO	ANO	NE	471001
	VehicleDescription	nvarchar(MAX)	NE	NE	NE	Chyby telerail

Tabulka 3.9: Vehicles

3.5.11 Tabulka WorkerInTour

Tabulka „WorkerInTour“ je vazební tabulkou, která je určená pro vztah m:n tabulek „Workers“ a „Tours“. Definuje, že každý zaměstnanec muže pracovat v rozdílných turnusech a že pod konkrétním turnusem může pracovat více zaměstnanců.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PFK	WorkerInTourWorkerID	int	ANO	ANOS	NE	
PFK	WorkerInTourTourID	int	ANO	ANOS	NE	

Tabulka 3.10: WorkerInTour

3.5.12 Tabulka Workers

Tabulka „Workers“ obsahuje informace o zaměstnancích. Atribut „WorkerDetachmentID“ určuje, do jakého oddělení zaměstnanec patří. Jméno zaměstnance se ukládá do atributu „WorkerFirstName“ a jeho příjmení se ukládá do atributu „WorkerLastName“. Atribut „WorkerServiceNumber“ určuje osobní číslo zaměstnance, atribut „WorkerSapNumber“ určuje identifikační číslo zaměstnance. Jako volitelné údaje je možné vložit služební email do sloupce „WorkerServiceEmail“, soukromý email do sloupce „WorkerPersonalEmail“, služební telefonní číslo do sloupce „WorkerServicePhone“, soukromé číslo do sloupce „WorkerPersonalPhone“ a pro dodatečnou informací slouží sloupec „WorkerDescription“. Atribut „WorkerPhoto“ slouží pro vložení obrázku zaměstnance.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	WorkerID	int	ANO	ANO	ANO	
FK	WorkerDetachmentID	int	ANO	NE	NE	
	WorkerFirstName	nvarchar(20)	ANO	NE	NE	Armen
	WorkerLastName	nvarchar(30)	ANO	NE	NE	Hajrapetjan
	WorkerServiceNumber	nvarchar(20)	ANO	ANO	NE	752220
	WorkerSapNumber	nvarchar(20)	ANO	ANO	NE	300600
	WorkerServiceEmail	nvarchar(50)	NE	NE	NE	hajrapetjan@cd.cz
	WorkerPersonalEmail	nvarchar(50)	NE	NE	NE	hajrapetjan@gmail.com
	WorkerServicePhone	nvarchar(20)	NE	NE	NE	+420725082207
	WorkerPersonalPhone	nvarchar(20)	NE	NE	NE	+420725055455
	WorkerDescription	nvarchar(MAX)	NE	NE	NE	Mistr
	WorkerPhoto	image	NE	NE	NE	

Tabulka 3.11: Workers

3.5.13 Tabulka WorkerStates

Tabulka „WorkerStates“ obsahuje seznam stavů zaměstnanců. Tyto údaje slouží pro určení stavu zaměstnance při tvorbě docházky, kde se určuje, zda je zaměstnanec přítomen,

nepřítomen, nemocen atd.. Sloupcem „WorkerStateName“ se definuje stav zaměstnance a do sloupce „WorkerStateDescription“ se dá vložit dodatečná informace.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	WorkerStateID	int	ANO	ANO	ANO	
	WorkerStateName	nvarchar(30)	ANO	ANO	NE	Přítomen
	WorkerStateDescription	nvarchar(MAX)	NE	NE	NE	

Tabulka 3.12: WorkerStates

3.5.14 Tabulka Works

Tabulka „Works“ obsahuje záznamy všech vykonaných prácí. Atribut „WorkDateID“ určuje, do jaké směny patří záznam o vykonané práci. Atribut „WorkVehicleID“ určuje, o jaké vozidlo se jedná. Atributem „WorkWorkTypeID“ se určuje, o jaký typ závady se jedná, a atribut „WorkDetachmentID“ určuje, ke kterému oddělení záznam patří. Pro popis závady slouží atribut „WorkFaultDescription“ a pro popis způsobu opravy slouží atribut „WorkCauseDescription“.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	WorkID	bigint	ANO	ANO	ANO	
FK	WorkDateID	bigint	ANO	NE	NE	
FK	WorkVehicleID	int	ANO	NE	NE	
FK	WorkWorkTypeID	int	ANO	NE	NE	
FK	WorkDetachmentID	int	ANO	NE	NE	
	WorkFaultDescription	nvarchar(MAX)	ANO	NE	NE	Nesvítí reflektor
	WorkCauseDescription	nvarchar(MAX)	NE	NE	NE	Výměna lampy

Tabulka 3.13: Works

3.5.15 Tabulka WorkTypes

Tabulka „WorkTypes“ obsahuje seznam typů prací. Tyto udaje slouží pro určení typu práce při zakládání vykonané práce, kde se určuje, zda byla práce vykonána neplánovaně, plánovaně, telefonicky atd.. Sloupcem „WorkTypeName“ se definuje typ práce a do sloupce „WorkTypeDescription“ se dá vložit dodatečná informace.

Klíč	Název	Typ	Not null	Unikátní	Autoinkrementace	Komentář
PK	WorkTypeID	int	ANO	ANO	ANO	
	WorkTypeName	nvarchar(30)	ANO	ANO	NE	E0
	WorkTypeDescription	nvarchar(MAX)	NE	NE	NE	Technická kontrola

Tabulka 3.14: WorkTypes

3.6 Školení zaměstnanců

Pro zaměstnance, kteří budou používat tento informační systém, bylo provedeno školení v používání tohoto informačního systému. Na školení byly zaměstnancům založeny uživatelské účty. Byla jim představena struktura celého systému. Zaměstnanci se naučili, jak klienta nainstalovat, jak se přihlásit do systému a jak si změnit heslo. Naučili se vytvářet a upravovat záznamy o vozidlech, odděleních, typech oprav, zaměstnancích, turnusech a statusech zaměstnanců. Naučili se vytvářet směny, přidávat a odebírat záznamy práce a docházky do směny. Také se dozvěděli, jak vytvářet předávku a jak je přesouvat do vykonané práce. Bylo jim ukázáno, jak pracovat s vyhledáváním záznamů a zobrazení jejich detailů. Další částí školení byla správa uživatelských účtů a tisk, kde jim bylo ukázáno, jak vytisknout směnu a jak vytvářet a upravovat uživatelské účty. Celé školení bylo provedeno v praktickém pojetí. Školení probíhalo v prostorách ČD.

Kapitola 4

Testy

4.1 Testování během vývoje

Během vývoje a při psaní jednotlivých funkcionalit jsem osobně zkoušel funkčnost vytvořeného informačního systému. Funkce napsané na straně služeb jsem zkoušel pomocí nástroje „WCF Test Client“ [17], který je integrován do „Visual Studia 2012“. Pomocí tohoto nástroje je možné volat jednotlivé metody služeb a tím je testovat aniž by bylo třeba psát vlastní klientskou aplikaci. Co se týče testování klientské aplikace, tak i tu jsem během vývoje testoval a to již pomocí navrženého uživatelského rozhraní, se kterým je funkční část svázána.

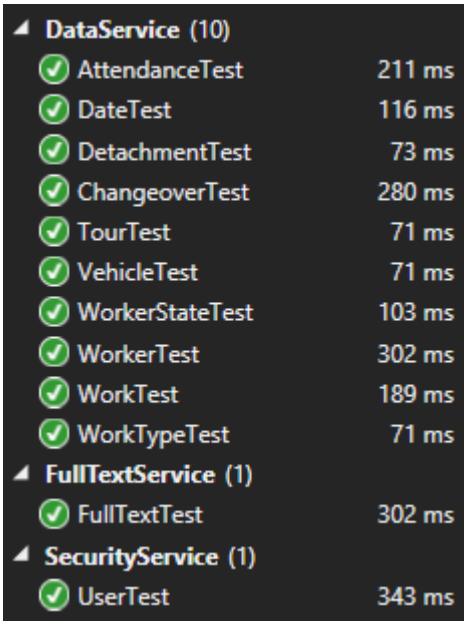
4.2 Funkční test

Funkčním testem se ověřovala správná funkce implementovaného systému, tedy zda splňuje veškeré požadavky, pro které byla navržena. V tomto testu se obecně testovala veškerá napsaná funkcionalita, zda správně funguje a zda splňuje požadavky zaměstnanců.

Ve zkušebním provozu systému sami zaměstnanci zkoušeli veškerou funkcionalitu systému a zkoušeli, zda veškerá funkcionalita, která byla uvedena v zadání, je funkční. Co se týče tohoto testu, tak provoz proběhl bez jakýchkoliv problémů.

4.3 Testy služeb

Pro otestování služeb byly navrženy unit testy, které otestovaly každou metodu služby. Výsledky testů jsou zobrazeny na obrázku 4.1 i s dobou jejich trvání. Více informací o „Unit Testech“ se lze dočíst v materiálech [18].



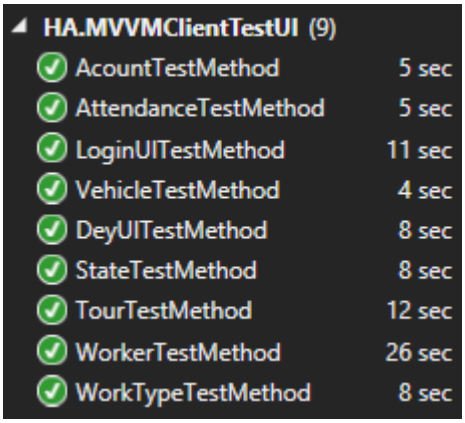
Obrázek 4.1: Výsledky testu služeb s dobou jejich trvání

- Služba DataService se testovala nad základními operacemi *CRUD* [1] .
- Služba SecurityService byla též testována nad operacemi CRUD a také funkcí pro resetování hesla.
- Služba FullTextService byla testována na vyhledávaní záznamu jak pomocí parametru tak i pomocí výrazu.

[1] CRUD (create, read, update, delete) je zkratkou pro základní operace nad daty. Každé písmeno znamená jednu operaci C (create – založení), R (read – čtení), U (update – editace), D (delete- mazání).

4.4 Testy uživatelského rozhraní

Pro testováni uživatelského rozhraní jsem použil Coded UI Test, který je automatizovaným testem zahrnujícím i funkční testy ovládacích prvků. Tento způsob testování umožňuje potvrdit, že celá aplikace funguje tak, jak má i s uživatelským rozhraním. V tomto testu v podstatě stačí naklikat postup práce s aplikací, který se postupně nahrává, a poté zvolit, zda chceme z tohoto postupu vygenerovat testovací metodu, kterou poté můžeme spustit, a postup zopakovat. Jak vytvářet „Coded UI Testy“ se lze dočíst v materiálech [19]. V klientské aplikaci se testovaly jednotlivé obrazovky, kde se kontrolovala funkčnost jednotlivých komponent. Výsledky testů je možné vidět na obrázku 4.2 i s dobou jejich trvání.



Obrázek 4.2: Výsledky testů uživatelského rozhraní s dobou jejich trvání

4.5 Pilotní test

Po dokončení implementace projektu byl celý systém nasazen do zkušebního provozu v reálném prostředí pro vyzkoušení spolehlivosti a funkčnosti systému jako celku. Systém běží v tomto zkušebním provozu již několik měsíců a zatím nebyl objeven žádný problém. Zkušební provoz zatím nebyl ukončen z důvodu hledání prostředků pro zálohování databáze pro zajištění vyšší bezpečnosti uložených dat.

Kapitola 5

Závěr

Při návrhu této práce jsem v první fázi zjišťoval, zda o systém jeví zájem většina zaměstnanců, kteří by jej měli využívat. Zjištění zájmu bylo provedeno formou ankety, kde se každý zaměstnanec mohl svobodně rozhodnout, zda o tento systém jeví zájem, či nikoliv. Dalším krokem bylo sepsání požadavků na systém, vytvoření návrhů datových modelů služeb, klientské aplikace a databázového modelu. Také jsem připravil UML diagram aktivit, které popisují manipulaci se systémem. Po dokončení analýzy celého systému jsem začal s implementační částí, během níž jsem jednotlivé části kódu pokryl unit testy pro kontrolu jejich plné funkčnosti. Po dokončení implementace byla provedena řada testů, které měly zaručit bezchybné chování celého systému jako celku.

Celý informační systém byl po navržení a realizaci nasazen v depu kolejových vozidel a spuštěn do zkušebního provozu. V tomto zkušebním provozu se testovala její funkčnost v reálném prostředí a zjišťovaly se potřebné úpravy a doplnění funkcionality do systému.

Tento systém po nasazení zefektivnil pracovní proces díky rozmanitému a rychlému vyhledávání záznamů s možností zjištění popisu závady a způsobu její opravy. Také snížil zátěž na zaměstnance neustálým obvoláváním a zjišťováním stavu vykonaných prací.

Co se týče tohoto projektu, neskončilo to jen zhotovením této diplomové práce, ale pořád se tento systém rozšiřuje o nové funkce.

Z mého hlediska mě velice těší, že práce, kterou jsem vykonal v reálné praxi, funguje a umožňuje ulehčit práci jiným lidem. Také jsem se mnohému naučil při programování této práce (návrhové vzory a různé technologie jakými jsou WPF, WCF). Také jsem se naučil pracovat s relační databází MSSQL a s IIS 7.

Literatura

[1] Sky Drive:
http://windows.microsoft.com/cs-cz/onedrive/onedrive-help#onedrive=windows-8

[2] SelfCert: Create a Self-Signed Certificate:
http://blog.pluralsight.com/selfcert-create-a-self-signed-certificate-interactively-gui-or-programmatically-in-net

[3] Rudolf Jalovecký: HTML HELP a jeho začlenění do Visual FoxPro 2006.

[4] Secure Hash Algorithm:
http://cs.wikipedia.org/wiki/Secure_Hash_Algorithm

[5] Message-Digest algorithm:
http://cs.wikipedia.org/wiki/Message-Digest_algorithm

[6] Pablo Cibraro and Kurt Claeys: Professional WCF 4 : Windows Communication Fundation with .NET 4. Wiley Publishing 2010.

[7] Unit Of Work Pattern:
http://voho.cz/wiki/informatika/oop/navrhovy-vzor/unit-of-work
http://blog.netcorex.cz/tag/unitofwork/
http://www.codeproject.com/Articles/581487/Unit-of-Work-Design-Pattern
http://msdn.microsoft.com/en-us/magazine/dd882510.aspx

[8] Generic Repository Pattern:
http://www.tugberkugurlu.com/archive/generic-repository-pattern-entity-framework-asp-net-mvc-and-unit-testing-triangle

[9] Julia Lerman: Programming Entity Framework. O'Reilly Media 20120

[10] Charles Petzold: Mistrovství ve Windows Presentation Fundation. COMPUTER PRESS 2008.

[11] Windows Presentation Fundation:
http://msdn.microsoft.com/en-us/library/ms754130.aspx

[12] Ryan Vice and Muhammad Shujaat Siddiqi: MVVM Survival Guide for Enterprise Architectures in Silverlight and WPF. Packt Publishing 2012

[13] Ikony:
https://www.iconfinder.com/
http://thenounproject.com/

[14] Fluent Validation Framework:
http://fluentvalidation.codeplex.com/wikipage?title=
Validators&referringTitle=Documentation&ANCHOR#NotEmpty

[15] ClickOnce Security and Deployment:
http://msdn.microsoft.com/cs-cz/library/t71a733d.aspx

[16] MS SQL:
http://www.dotnetportal.cz/clanek/140/Seznameni-a-instalace-
Microsoft-SQL-Serveru
http://www.dotnetportal.cz/clanek/141/Sprava-sluzeb-a-komponent-
Microsoft-SQL-Serveru
http://www.dotnetportal.cz/clanek/142/Prvni-kroky-s-databazi
http://www.dotnetportal.cz/clanek/143/Fyzicke-soubory-databaze-
datove-soubory-a-transakcni-log
http://www.dotnetportal.cz/clanek/145/Tabulky
http://www.dotnetportal.cz/clanek/149/Datove-typy

[17] WCF Test Client:
http://msdn.microsoft.com/cs-cz/library/bb552364(v=vs.110).aspx

[18] Verifying Code by Using Unit Tests:
http://msdn.microsoft.com/en-us/library/dd264975.aspx

[19] Verifying Code by Using UI Automation:
http://msdn.microsoft.com/en-us/library/dd286726.aspx

Seznám příloh

- Příloha A obsahuje soupis požadavku.
- Příloha B obsahuje Originál ankety a seznám zaměstnanců přítomných na školení.
- Příloha C obsahuje obsah přiloženého CD.
- Příloha D obsahuje technickou dokumentaci klientské aplikace. Tato příloha je obsažena na CD.
- Příloha E obsahuje technickou dokumentaci serveru. Tato příloha je obsažena na CD.
- Příloha F obsahuje uživatelskou dokumentaci k systému. Tato příloha je obsažená na CD.

Příloha A

Soupis požadavku

- Systém musí být lehce ovladatelný, jelikož tento software budou používat i starší generace, pro které je práce s počítačem mnohem obtížnější. Z tohoto důvodu by měl systém umožňovat výběr záznamů ze statisticky nejběžnějších závad a oprav, které se budou generovat při každém spuštění klienta.
- Systém musí umožňovat vyhledávání záznamu s možností zobrazení detailu konkrétního záznamu, ve kterém budou zobrazeny přehledně informace o vozidlu, směně, typu opravy, závadě, způsobu opravy a docházce zaměstnanců v dané směně dle dále uvedených kritérií.
  - Typu závady.
  - Vozidla.
  - Textového výrazu obsaženého v popisu závady.
  - Vymezeném časovém intervalu (od, do).
  - Pokud má uživatel dostatečná oprávnění, pak může vyhledávat záznamy jiných oddělení. V opačném případě se vyhledávají záznamy jen v oddělení, ve kterém je veden jeho účet.
- Systém musí umožňovat tisk a náhled tisku směn pro zálohu záznamů v papírové podobě.
- Po spuštění klienta se má vždy vybrat poslední založená směna.
- Navrhovaný systém musí umožňovat ukládat informace o jednotlivých vozidlech, zaznamenávat docházku zaměstnanců s možností přidání doplňujících informací,

přehledně zobrazovat záznamy o zaměstnancích a jejích turnusech, zakládat, editovat a mazat informace o odděleních, turnusech, typech oprav a statusech zaměstnanců pro docházku.

- Tento systém musí být zabezpečen vůči neoprávněnému přístupu pomocí uživatelských účtů a také musí být zabezpečena komunikace se serverem.
- Uživatelské účty by měly umožňovat tuto funkcionalitu:
  - Každý uživatelský účet by mělo být možné navázat na konkrétního zaměstnance.
  - Každý účet by měl být zařazen do konkrétního oddělení.
  - Účet zařazen do konkrétního oddělení by měl umožňovat zápis a čtení záznamu jen z oddělení, do kterého je zařazen.
  - Každý uživatel by měl mít možnost si změnit svoje heslo a vidět svá oprávnění.
  - Každému účtu by mělo být možné nastavit oprávnění pro zápis do systému, pro administrativu a pro zobrazení záznamů z jiných oddělení.
  - Administrátorské účty by měly mít možnost zakládat, editovat a mazat uživatelské účty s možností resetovat hesla jednotlivých uživatelských účtů.
- Mělo by být možné zaznamenávat jak závadu tak i způsob její opravy a typ opravy s možností výběru častých závad a oprav, které se generují při každém spuštění klienta.
- Mělo by být možné zaznamenávat předávanou práci s možností rychlého přesunu do vykonaných prací vybrané směny.

A.1 Uživatelské role (oprávnění)

A.1.1 Bez role

- Registrovaný uživatel bez role bude mít tyto možnosti:
  - Přihlásit se do systému.
  - Prohlížet si směny oddělení, do kterého sám spadá, s možností zobrazení záznamu o provedených prací, docházce i předaných prací.

  - Tisknout sestavy směn.
  - Prohlížet si seznam vozidel.
  - Prohlížet si seznam typů prací.
  - Prohlížet si seznam zaměstnanců.
  - Prohlížet si seznam turnusů.
  - Prohlížet si seznam statusů.
  - Hledat záznamy práce dle těchto parametrů:
    * Typu závady.
    * Vozidla.
    * Textového výrazu obsaženého v popisu závady.
    * Vymezeném časovém intervalu (od, do).
  - Zobrazení detailu vyhledaného záznamu.
  - Zobrazení informace o vlastním účtu.
  - Změna hesla vlastního účtu.

A.1.2 Zápis

- Registrovaný uživatel s rolí „Zápis“ bude mít tyto možnosti:
  - Přihlásit se do systému.
  - Prohlížet, editovat a mazat směny v oddělení, do kterého sám spadá, s možností zobrazení, editace a mazání záznamu práce, docházky i předávaných prací.
  - Převod předávaného záznamu do vykonané práce.
  - Tisknout sestavy směn.
  - Prohlížet, editovat a mazat záznamy vozidel.
  - Prohlížet, editovat a mazat záznamy typu práce.
  - Prohlížet, editovat a mazat záznamy zaměstnanců.
  - Prohlížet, editovat a mazat záznamy turnusů.
  - Prohlížet, editovat a mazat záznamy statusů.
  - Hledat záznamy práce dle těchto parametrů:

    - Typu závady.
    - Vozidla.
    - Textového výrazu obsaženého v popisu závady.
    - Vymezeném časovém intervalu (od, do).
  - Zobrazení detailu vyhledaného záznamu.
  - Zobrazení informace o vlastním účtu.
  - Změna hesla vlastního účtu.

A.1.3 Mistr

- Registrovaný uživatel s rolí „Mistr“ bude mít tyto možnosti:
  - Přihlásit se do systému.
  - Prohlížet si směny oddělení, do kterého sám spadá, s možností zobrazení záznamu práce, docházky i předávané práce.
  - Tisknout sestavy směn.
  - Prohlížet si seznam vozidel.
  - Prohlížet si seznam typu práce.
  - Prohlížet si seznam zaměstnanců.
  - Prohlížet si seznam turnusů.
  - Prohlížet si seznam statusů.
  - Hledat záznamy práce dle těchto parametrů:
    - Typu závady.
    - Vozidla.
    - Oddělení.
    - Textového výrazu obsaženého v popisu závady.
    - Vymezeném časovém intervalu (od, do).
  - Zobrazení detailu vyhledaného záznamu.
  - Zobrazení informace o vlastním účtu.
  - Změna hesla vlastního účtu.

A.1.4 Administrátor

- Registrovaný uživatel s rolí „Administrátor“ bude mít tyto možnosti:
  - Přihlásit se do systému.
  - Prohlížet si směny oddělení, do kterého sám spadá, s možností zobrazení záznamu práce, docházky i předávané práce.
  - Tisknout sestavy směn.
  - Prohlížet, editovat a mazat záznamy oddělení.
  - Prohlížet si seznam vozidel.
  - Prohlížet si seznam typu práce.
  - Prohlížet si seznam zaměstnanců.
  - Prohlížet si seznam turnusů.
  - Prohlížet si seznam statusů.
  - Hledat záznamy práce dle těchto parametrů:
    - Typu závady.
    - Vozidla.
    - Textového výrazu obsaženého v popisu závady.
    - Vymezeném časovém intervalu (od, do).
  - Zobrazení detailu vyhledaného záznamu.
  - Zobrazení informace o vlastním účtu.
  - Změna hesla vlastního účtu.
  - Prohlížet, zakládat, editovat a mazat uživatelské účty s možností resetovat hesla jednotlivých uživatelských účtů.

Příloha B

Originál ankety a seznám přítomností na školení

Originál ankety je přiložen na následujících stránkách této přílohy.

Seznam zaměstnanců který byli přítomní na školení je přiložen na následujících stránkách této přílohy.

Anketa zjištění zájmu o návrh systému pro zaznamenávaní vykonané práce v elektronické podobě

Prosím zaměstnance, aby se každý svobodně vyjádřil, zda by měl zájem nahradit sešitovou verzi zaznamenávaní práce za digitální systém. Tento systém by nabízel výhody jak v čitelnosti záznamu, tak v přehledností a hlavně ve vyhledávání.

Jméno	Příjmení	Mám zájem	Nemám zájem
Robert	Hlušička		
David	Lněnička		
Jiří	Bína		
Bohuslav	Kuta		
Tomáš	Musil		
Jiří	Novotný		
Antonín	Procházka		
Milan	Trojánek		
Michal	Tůma		
Ladislav	Živný		
Petr	Hudec		
Vojtěch	Pán		

Účast na školení o aplikaci „Digitální předávka“
28.9.2013

Jméno	Příjmení	Podpis
Robert	Hlušička	
David	Lněnička	
Jiří	Bína	
Bohuslav	Kuta	
Tomáš	Musil	
Jiří	Novotný	
Antonín	Procházka	
Milan	Trojánek	
Michal	Tůma	
Ladislav	Živný	
Petr	Hudec	
Vojtěch	Pán	

Příloha C

Obsah přiloženého CD

Složky

- **HA.MVVMClient**
  - Obsahuje zdrojové soubory klientské aplikace.
- **Diagrams**
  - Obsahuje zdrojové soubory diagramů aktivit.
- **HA.MVVMClientTestUI**
  - Obsahuje zdrojové soubory testu klientské aplikace.
- **HA.HostWeb**
  - Obsahuje zdrojové soubory hostovacího prostředí pro služby.
- **HA.Services**
  - Obsahuje zdrojové soubory služeb.
- **HA.ServicesTest**
  - Obsahuje zdrojové soubory testu služeb.

Soubory

- **ScriptDb.sql**
  - Obsahuje skript pro tvorbu databáze.
- **DiplomovaPrace.pdf**
  - Obsahuje elektronickou podobu této diplomové práce.
- **Help.chm**
  - Obsahuje uživatelskou dokumentaci.
- **ServerDokumentace.pdf**
  - Obsahuje technickou dokumentaci serveru.
- **KlientDokumentace.pdf**
  - Obsahuje technickou dokumentaci klientské aplikace.