Návrh subsystému informačního systému pro bytové družstvo
The design of Subsystem of Information System for the Housing Association

Student: Bc. Jakub Michalík
Vedoucí diplomové práce: Ing. Radek Němec

Ostrava 2013
Zadání diplomové práce

Bc. Jakub Michalík

Studijní program: N6209 Systémové inženýrství a informatika
Studijní obor: 6209T025 Systémové inženýrství a informatika
Téma: Návrh subsystému informačního systému pro bytové družstvo
The Design of Subsystem of Information System for the Housing Association

Zásady pro vypracování:
1. Úvod
2. Teoretická a metodická východiska objektově orientovaného přístupu k analýze a návrhu informačních systémů
3. Analýza současného stavu
4. Návrh subsystému informačního systému bytového družstva
5. Závěr

Seznam použité literatury
Seznam zkratek
Prohlášení o využití výsledků diplomové práce
Seznam příloh
Přílohy

Seznam doporučené odborné literatury:

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: Ing. Radek Němec

Datum zadání: 23.11.2012
Datum odevzdání: 26.04.2013

doc. Ing. Jana Hančlová, CSc.
vedoucí katedry

prof. Dr. Ing. Dana Dluhošová
děkanka fakulty
Děkuji svému vedoucímu diplomové práce panu Ing. Radku Němcovi za vstřícný přístup, podporu a rady k vypracování této práce.
Prohlašuji, že jsem celou práci, včetně všech příloh, vypracoval samostatně.

V Ostravě dne 26.4.2013

[Podpis]

Vlastnoruční podpis autora
Obsah

1 Úvod.......................................................................................................................... 5

2 Objektově orientovaný přístup k analýze a návrhu IS ........................................ 6

  2.1 Jazyk UML ......................................................................................................... 7
    2.1.1 MDA........................................................................................................... 8

  2.2 Nástroje CASE .................................................................................................. 10

  2.3 Struktura jazyka UML ....................................................................................... 12

    2.3.1 Stavební bloky (UML Building Blocks) ...................................................... 13
    2.3.2 Společné mechanismy (Common Mechanism in UML) .............................. 19
    2.3.3 Architektura (UML Architecture) ............................................................... 22

  2.4 UML diagramy .................................................................................................. 23

    2.4.1 Diagram případů užití (Use Case Diagram) .............................................. 23
    2.4.2 Diagram tříd (Class Diagram) .................................................................... 24
    2.4.3 Stavový diagram (State Machine Diagram) ................................................ 26
    2.4.4 Diagram aktivit (Activity Diagram) ............................................................ 27
    2.4.5 Sekvenční diagram (Sequence Diagram) ...................................................... 29

3 Analýza současné situace a požadavky uživatelů ................................................. 30

  3.1 Popis organizace ............................................................................................... 30
    3.1.1 Předmět činnosti ....................................................................................... 30
    3.1.2 Organizační struktura ................................................................................ 30

  3.2 Analýza současné situace .................................................................................. 31

    3.2.1 Osobní zadání úkolu ................................................................................... 31
    3.2.2 Telefonické zadání úkolu .......................................................................... 32
    3.2.3 Zadání úkolu prostřednictvím poštovního klienta ..................................... 32

  3.3 Problémové oblasti ............................................................................................ 33

  3.4 Specifikace softwarových požadavků ............................................................... 34

    3.4.1 Cíl navrhovaného systému ....................................................................... 34
    3.4.2 Specifikace systému .................................................................................... 35
    3.4.3 Funkční požadavky ..................................................................................... 35
    3.4.4 Nefunkční požadavky ................................................................................. 35

4 Návrh systému IS .................................................................................................... 36

  4.1 Diagram případů užití ....................................................................................... 36

    4.1.1 Diagram případů užití aktérů .................................................................... 36
    4.1.2 Diagram případů užití uživatele ................................................................. 37
    4.1.3 Diagram případů užití správce systému ...................................................... 40
4.1.4 Matice trasovatelnosti požadavků ................................................................. 42
4.2 Diagram tříd ........................................................................................................... 42
  4.2.1 Diagram tříd systému ...................................................................................... 43
4.3 Stavový diagram .................................................................................................. 43
  4.3.1 Stavový diagram úkolu .................................................................................... 43
  4.3.2 Stavový diagram události .............................................................................. 44
4.4 Diagram aktivit ...................................................................................................... 45
  4.4.1 Diagram aktivit úkolu .................................................................................... 45
4.5 Sekvenční diagram .............................................................................................. 46
  4.5.1 Sekvenční diagram úkolu ............................................................................ 46
4.6 Návrh rozhraní ...................................................................................................... 47
  4.6.1 Funkční specifikace ....................................................................................... 47
  4.6.2 Dashboard uživatele systému ........................................................................ 48
  4.6.3 Dashboard správce systému ......................................................................... 49
  4.6.4 Tvorba úkolu ................................................................................................ 49
  4.6.5 Evidence úkolů .............................................................................................. 50
  4.6.6 Tvorba události ............................................................................................ 50
  4.6.7 Kalendář ........................................................................................................ 51
5 Závěr ......................................................................................................................... 52
Seznam použité literatury .......................................................................................... 53
Seznam zkratek .......................................................................................................... 54
Seznam obrázků .......................................................................................................... 55
Seznam tabulek .......................................................................................................... 56
1 Úvod

Má diplomová práce se bude zabývat analýzou a návrhem subsystému evidence sledování úkolů pro ORLOVAN, bytové družstvo v obci Orlová-Lutyně. V jednotlivých kapitolách teoretické části diplomové práce bude popsán jazyk UML, jeho struktura, diagramy UML a nástroje CASE. V praktické části bude analyzována současná situace sledování a evidence úkolů v organizaci ORLOVAN, bytové družstvo, specifikovány problémové oblasti, specifikovány softwarové požadavky na systém. V poslední kapitole bude na základě analýzy proveden návrh subsystému ve formě UML diagramů a budou zde také uvedeny návrhy uživatelského rozhraní modelovaného systému.

Téma jsem si zvolil z důvodů, že ve firmách jsou velice často zadávány úkoly poněkud nestandardně a tato skutečnost následně přispívá k jejich nepřehledné evidenci a sledování jejich stavu či průběhu. V zmíněném bytovém družstvě jsou úkoly převážně zadávány osobně nebo telefonicky a následně až jako třetí možnost, navíc ne všemi zaměstnanci, je využita aplikace MS Outlook. Bohužel i tato možnost není zcela ideální, protože MS Outlook je primárně vyvíjen jako poštovní klient, i když v sobě zahrnuje možnost zadávání úkolů nebo vytvoření událostí. Tyto skutečnosti mě vedly k rozhodnutí vytvoření návrhu systému, který bude podporovat evidenci a sledování úkolů v této organizaci.

Cílem mé diplomové práce je navrhnout subsystému evidence a sledování vnitropodnikových úkolů organizace ORLOVAN, bytové družstvo na základě provedené analýzy současného stavu. Systém bude navržen za účelem odstranění bariér nynějších způsobů zadávání, evidování úkolů. Mezi další cíle navrhovaného systému evidence a sledování úkolů patří zrychlení procesu zadávání úkol, jejich evidence, kontroly a především navržení jednoduché a přehledné aplikace.

Systém bude navržen jako webové rozhraní, které bude obsahovat patřičné sekce. Systém bude umístěn do prostředí interního systému družstva a bude sloužit jen pro zaměstnance bytového družstva. Uživatelům bude v systému umožněno vytvořit úkoly, události a dále je bude přehledně evidovat. Úkoly budou s jejich náležitostmi zobrazeny v sekci evidence úkolů a události budou zobrazeny v sekci kalendář.

Jako metoda sběru informací ve fázi analýzy současné situace byla zvolena metoda osobního rozhovoru a pozorování, fáze návrhu byla provedena objektově orientovaným způsobem, přesněji vizuálním jazykem UML.
2 Objektově orientovaný přístup k analýze a návrhu IS


Obr. 2.1 Historie UML.

Metodika Fusion se stala jednou z prvních, která se snažila o sjednocení, ale i přes skutečnost, že byla značně medializována, do její přípravy se nezapojovali tehdejší tržní lídři (Booch, Jacobson, Rumbaugh). Po spojení sil autorů metod Booch a Rumbaugh v organizaci Rational Corporation, jež pracovala na vzniku jazyka UML (Unified Modeling Language), metodika Fusion ukončila své působení na trhu.

---

1 The Booch Method – Jedná se o metodu, kterou vyvinul Grady Booch okolo 1993. Byla hojně využívána v objektově orientované analýze a návrhu. Později byla nahrazena jazykem UML.

2 OMT (Object modeling technique) – Jedná se o metodu, vyvinutou okolo roku 1991 softwarovými inženýry Rumbaugh, Blaha, Premerlani, Eddy a Lorensen, která popisovala objektové modely nebo statickou strukturu systému.

Rok 1996 se stal z pohledu UML velmi důležitým. Ve zminěném roce byla sdružením OMG⁴ (Object Management Group) navržena specifikace RFP ⁵ (Request For Proposal) pro objektově orientovaný jazyk pro vizuální modelování, ve kterém byl jazyk UML navržen jako standard. Jazyk UML byl roku 1997 přijat jako první průmyslový standard objektově orientovaného jazyka pro vizuální modelování. Tato skutečnost se stala důvodem zániku konkurenčních metod a jazyk UML byl veřejností velice kladně přijat.[1]

2.1 Jazyk UML

Vizuální modelovací jazyky, jak už bylo řečeno v předešlé kapitole, se v softwarovém průmyslu vyskytují u delší čas. Základem jejich vzniku se stal fakt, že programovací jazyky neumožňují tak vysokou míru abstrakce ve fázi návrhu systému. Výsledkem snahy analytiků a designérů o vytvoření jazyka, který by umožnil spojit nejlepší existující postupy modelovacích technik, softwarového inženýrství a dále by byl schopen popsat objektově orientovanou analýzu a návrh, se ke konci minulého století stal právě jazyk UML. Tento jazyk byl navržen takovým způsobem, aby jej mohly implementovat všechny nástroje CASE (Computer Aided Software Engineering)

Unifikovaný modelovací jazyk (UML) je považován za univerzální jazyk spadající do rodiny vizuálních notací, jehož cílem je usnadnit specifikaci, návrh a dokumentaci modelovaného softwarového systému, především pak systému, který je objektově orientován. Tento jazyk je obdařen schopností modelovat jednoduché i složité systémy za pomoci stejné syntaxe. Modely vytvořené jazykem UML jsou srozumitelné i pro zadavatele a softwaru umožňují také kvalitní objasnění požadavků uživatelů modelovaného systému.


⁴ OMG (Object Management Group) – Jedná se o sdružení dodavatelů, které vzniklo za účelem definování a prosazování specifikací objektů COBRA. Později tomuto sdružení byl svěřen vývoj jazyka UML.
⁵ RFP (Request For Proposal) – Jedná se o způsob tvorby standardů v síti Internet a současně o požadavek na označení takto vzniklého standardu.
Složitost informačních systémů neustále roste. Doba, kdy si člověk sednul a naprogramoval celý systém sám, již skončila. Jazyk UML společně s CASE nástroji dokáže v určitých etapách životního cyklu informačního systému situaci značně usnadnit. Především ve fázi analýzy, kdy jsou specifikovány veškeré požadavky na systém a také ve fázi návrhu systému, kdy je řešen problém jak systém naprogramovat. Existují tři pohledy, jak lze systém modelovat v jazyce UML:

a) UML jako náčrt. Jedná se o neformální způsob využití tohoto jazyka, kdy jsou diagramy náčrtu na papír. Hlavním význam tohoto pohledu jak modelovat systém spočívá v tom, že se jedná o velmi dobrého pomocníka v úvodním vysvětlení základní myšlenky. Tyto náčrty s sebou ovšem nenesou příliš velkou hodnotu a proto se nearchivují. Neformální požadavky jsou obvykle tvořeny na papír, tabuli nebo v kreslicím nástroji typu MS Visio nebo PowerPoint.

b) UML jako plán. Jedná se o přesnější a formálnější způsob, kdy je jazyk UML využíván k detailní specifikaci softwarového systému. UML model je archivován a je důležitou součástí celého projektu. V tomto postupu jsou využity skutečné modelovací nástroje typu CASE.

c) UML jako programovací jazyk. Z podrobného UML diagramu lze vygenerovat šablonu kódu, která je základem pro implementaci. Například prostřednictvím architektury MDA (Model Driven Architecture), kdy lze UML modely použít jako programovací jazyk. Modely UML jsou opatřeny takovými detaily, které z nich přímo umožní překlad modelovaného systému. Z hlediska formálnosti a přesnosti se jedná o nejlepší využití jazyka UML v praxi. Samozřejmostí je vlastnictví CASE nástroje s podporou architektury MDA. Detailnější specifikace architektury MDA bude popsána v následující kapitole.[1][3]

2.1.1 MDA

Prostřednictvím architektury MDA je představován vývoj SW na základě modelů. Základní podstata této architektury spočívá ve skutečnosti, že výroba architektury spustitelného softwaru je řízena modelem. Dnes je díky této architektuře možné setkávat se s takovou miírou automatizace stále více.
Dle architektury MDA je software vytvářen jako sled transformací modelů uskutečňovaných modelovacími nástroji kompatibilními s architekturou MDA. CIM (Computer Independent Model) abstraktní model nezávislý počítači je základem modelu PIM (Platform Independent Model) modelu nezávislého na platformě. Následně je model PIM transformován do modelu PSM (Platform Specific Model) modelu závislého na platformě a ten jen nakonec transformován do finálního kódu. V architektuře MDA je představa modelu velmi obecná a kód je pokládán jen za velice konkrétní druh modelu. Na Obr. 2.2 se nachází graficky znázornění proces transformace v architektuře MDA.

CIM model je velice obecný, jehož prostřednictvím jsou zachycovány na platformě nezávislé klíčové požadavky na systém a oborový slovník. Zachycuje tu část činnosti, jež má za úkol se zautomatizovat.

Skrze model PIM je vyjadřována sémantika činnosti softwarového systému nezávisle na použité platformě (NET apod.) PIM model je obecný a úplný. Je v něm obsažen dostatečný základ pro transformaci do modelu PSM, která je schopna generovat konečný kód.

Následně je ještě potřeba doplnit PSM model o informace související s použitou platformou a výsledkem transformace bude zdrojový kód pro vybranou platformu. Pokud je model PSM dostatečně úplný, je vygenerován všechn zdrojový kód, dokumentace, simulace, program, soubory sestavení a v neposlední řadě deskriptory nasazení. Podmínkou je, aby byl model výpočetně úplný.[1][6]
2.2 Nástroje CASE

CASE nástroje jsou nástroje podpory nejen ve fázi analýzy a návrhu systému a je třeba chápat jako pomocníka. V dnešní době se softwaroví architekti při návrhu rozsáhlejších systémů bez tohoto nástroje neobejdou, přestože je možné UML diagramy zobrazit i za pomoci kreslících programů, jako je například MS Visio. CASE nástroji jsou právě tyto činnosti ve fázi analýzy a návrhu systému podporovány, usnadňují a zbaveny pracných operací.

Díky většině CASE nástrojů je umožněno i modelování procesů jako doplněk k UML technikám. Tyto produkty lze považovat za ideální nástroj pro návrh informačních systémů, jelikož díky nim je poskytnuto jedno prostranství, ve kterém lze identifikovat nejen důležité diagramy UML, ale i také procesní a datové modely potřebné v etapách analýzy a návrhu informačních systémů. Všechny softwarové firmy, jejichž cílem je produkce středních a velkých informačních systémů vyvíjí týmy analytiků, návrhářů a programátorů se bez podpory CASE nástrojů neobejdou. V dnešní době informačních technologií je doba, kdy k analýze postačovala tužka a papír nenávratně pryč. Tyto nástroje se s příchodem grafického modelovacího jazyka UML staly nezbytnou podporou ve všech fázích tvorby informačních systémů.

Možnosti CASE nástrojů

Prostřednictvím CASE nástrojů lze zvýšit kvalitu navrhovaného systému, neboť v nich je obsažena řada automatických kontrol, díky kterým je zamezováno výskytu mnoha chyb. Je zvýšena rychlost návrhu a implementace systému, v počátečních fázích tomu bude spíše naopak, závěrečné fáze jsou ale urychleny a tím je rychlejší také implementace.

Dále je zvýšena kvalita a komplexnost dokumentace systému, protože dokumentace je vedlejším produktem analytické a návrhové činnosti, je tedy kompletní a očištěna od nekonzistentních dat.

Je usnadněno projektové řízení, jelikož projektový tým je informován o nejnovější verzi řešení a jeho dokumentaci. Dále je jednodušší zhodnotit plnění časového plánu a rozpočtu. Je vytvořen standard pro vývoj procesu za podpory řady standardních metod a nástrojů.

Důsledkem kvalitnější dokumentace systému a také struktury odvozené od uplatnění standardních vývojových nástrojů je usnadněna údržba systému.

Díky CASE nástrojům je ulehčena přenositelnost systému na jiné HW a SW platformy.
Rozdělení CASE nástrojů

Na trhu postupným vývojem vzniklo značné množství produktů CASE. Produkty jsou od sebe odlišovány řadou hledisek, jakou jsou funkční možnosti, komplexnost nástrojů, metodické založení, uživatelská přívětivost a především kvalita fungování.[4]

Dle vztahu k etapě životního cyklu:

- PreCASE – umožňuje podporovat činnosti předcházející vývoji IS (stanovení podnikatelské strategie nebo informační strategie)
- UpperCASE – umožňuje podporovat počáteční etapy životního cyklu (identifikace projektu, iniciace, plánování, modelování organizační struktury, celková analýza IS)
- MiddleCASE – umožňuje podporovat detailní analýzu a návrh IS
- LowerCASE – zde jsou obsaženy nástroje pro podporu údržby a implementace IS (kódování)

Dle stupně integrace souboru:

- jednoúčelové nástroje (CASE Tools)
- sady nástrojů:
  - CASE Toolkits – zde je obsažen soubor integrovaných nástrojů pro podporu jedné etapy životního cyklu
  - CASE Workbenches – zde jsou obsaženy integrované nástroje podporující minimálně dvě etapy životního cyklu
  - I-CASE – zde je integrovaný CASE opírající se o centrální repositář, ve kterém jsou obsažena všechna metadata[4]

Komponenty nástrojů CASE

- grafické nástroje
- generátory vstupních a výstupních formulářů
- analytické nástroje CASE
- repositář CASE
- generátory CASE dokumentace
- generátory programového kódu
- ostatní nástroje[4]
2.3 Struktura jazyka UML

Jak již bylo řečeno jazyk UML je považován za univerzální jazyk spadající do rodiny vizuálních notací, jehož cílem je usnadnit specifikaci, návrh a dokumentaci modelovaného softwarového systému, především pak systému, který je objektově orientován.

Pomocí jazyka UML je svět vytvářen jako systém, kde jsou objekty vzájemně ovlivňovány. Objekt lze chápat jako soudržné seskupení dat a funkcí.

Pro jazyk UML existují dva aspekty modelu, které nejsou jeden bez druhého úplné a jsou skrze ně zachycovány:

- **Statická struktura**, prostřednictvím které je popsána důležitost typů objektů modelovaného systému a souvislost mezi nimi.
- **Dynamická struktura**, prostřednictvím které je popsán životní cyklus objektů a dále popsán způsob jejich spolupráce cílem dosáhnout požadované funkce modelovaného systému.[1]

**Rozdělení struktury jazyka UML**

Strukturu jazyka UML je možné rozdělit na tři části, viz Obr. 2.3. Jedná se o:

1. **Stavební bloky**, složeny ze základních prvků, relací a diagramů.
2. **Společné mechanismy**, složeny z obecných způsobů, jimiž lze v jazyku UML dospět ke specifickým cílům.
3. **Architektura**, v UML jazyku se zde jedná o pohled na architekturu modelovaného systému.[1]

![Obr. 2.3 Struktura jazyka UML](image-url)
2.3.1 Stavební bloky (UML Building Blocks)

Jazyk UML je založen na třech elementech, jež jsou z pohledu uživatele představovány grafickými značkami v dvourozměrném plošném grafu. Jedná se o elementy:

a) Předměty
b) Relace
c) Diagramy

a) Předměty (Things)

První stavební blok je tvořen předměty. Ty jsou dále rozděleny do čtyř odlišných podkategorií – strukturní abstrakce, chování, seskupení, poznámky.[1]

Strukturní abstrakce (Structural things)

V první podkategorii jsou obsaženy strukturní abstrakce UML modelu (podstatná jména modelu UML), jež jsou tvořeny třídami, rozhraními, spolupraci, případy užití, aktivními třídami, komponenty nebo uzly. V UML diagramech jsou strukturní abstrakce znázorněny jako plošné vždy ukončené tvary, např. obdélníky, elipsami, kružnicemi, kvádry nebo krychlemi. [7][8][1]

Třída (Class) - Třída je reprezentována množinou objektů, jež jsou složeny z identických vlastností. Třída je znázorněna jako obdélník rozdělený na tři části pro zobrazení názvu třídy, atributů a operací viz Obr. 2.4.[8]

Rozhraní (Interface) – Rozhraní je v UML datovým typem, ve kterém jsou obsaženy pouze operace bez implementace tzv. veřejné abstraktní operace. Rozhraní je zobrazováno jako obdélník rozdělený na dvě části pro zobrazení názvu rozhraní a operací. Rozhraní je označováno klíčovým slovem <<interface>> viz Obr. 2.5.[8]
**Spolupráce (Collaboration)** - Spolupráci je definována interakce mezi prvky. Je znázorněna jako elipsa s přerušovaným okrajem viz Obr. 2.6.[8]

![Obr. 2.6 Spolupráce](image)

**Případ užití (Use Case)** - Případ užití je představován akcí, jež je vykonávána systémem k dosáhnutí konkrétního cíle. Je znázorněna jako elipsa s plným okrajem viz Obr. 2.7.[8]

![Obr. 2.7 Případ užití](image)

**Komponenta (Component)** - Komponentou je popisována fyzická část systému. Bývá znázorněna jako obdélník s dalšími menšími obdélníky v pravém horním rohu viz Obr. 2.8.[8]

![Obr. 2.8 Komponenta](image)

**Uzel (Node)** - Uzel může být definován jako existující fyzický prvek v režimu runtime. Jako příklad uzlu může být uveden např. procesor, tiskárna apod. Je znázorněn jako krychle nebo kvádr viz Obr. 2.9.[8]

![Obr. 2.9 Uzel](image)
Chování (Behavioral things)

Druhá podkategorie je tvořena tzv. chováním (slovesa modelu UML), které je v UML diagramech prezentováno vzájemnou komunikací mezi jednotlivými objekty (interkace). Chováním lze také vytvářet stavy, které jsou specifikovány přechody, událostmi a aktivitami. Stav je v diagramech UML znázorňován prostřednictvím obdélníků se zaoblenými rohy nebo kružnicemi. Interakce je znázorňována prostřednictvím různým způsobem konstruovaných a různým způsobem zakončených šipek nebo propojovacích čar.[7][8][1]

Interakce (Interaction) – Interakcí je v UML diagramech definováno chování, které je složeno skupinou zpráv vyměňovaných mezi prvky k dosažení konkrétního úkolu. Je znázorněna jako nepřerušovaná čára s vyplněnou šipkou na konci viz Obr. 2.10 [8]

Stav (State) – Stavem je znázorněna určitá situace v životním cyklu objektu, během kterého jeho splněná určitá podmínka, prováděna určitá operace nebo je očekávána určitá událost. Stav bývá znázorněn jako obdélník se zaoblenými rohy, počáteční stav je znázorněn vyplněnou kružnicí a konečný stav nevyplněnou kružnicí viz Obr. 2.11.[8]

Seskupení (Grouping things)

Třetí podskupina je tvořena seskupením, jehož prostřednictvím jsou podle potřeby sémanticky seskupovány související prvky modelu do kompaktních jednotek. Seskupení je v UML diagramech většinou znázorňováno pomocí stylizované kancelářské složky, kde v levé horní části je umístěn její popis.[7][8][1]

Balíček (Package) – Balíčkem je umožněno seskupit kterékoliv prvky UML do společné jednotky vyšší úrovně. Balíček je znázorněn jako kancelářská složka viz Obr. 2.12.[8]
Poznámky

Poslední podskupina je tvořena poznámkami, jež mají za úkol blíže specifikovat vlastnosti a chování prvků UML diagramů. V naprosté většině jsou poznámky znázorněny obdélníkem s ohnutým pravým horním rohem.[7][8][1]

Poznámka (Note) – Poznámkou lze například znázornit komentář nebo omezení UML prvku viz Obr. 2.13.[8]

b) Relace (Relationship)

Vzhledem ke skutečnosti, že je v diagramech potřeba předměty různým způsobem vzájemně propojit, jsou v UML jazyku specifikovány také relace, tedy vztahy mezi různými předměty diagramu. Prostřednictvím relací je možné zobrazit sémantický vztah mezi předměty.[7][8][1] Jde o vztahy závislosti, asociace, agregace, kompozice, generalizace a realizace.

Závislost (Dependency) – Pokud změna definice jednoho prvku dokáže zapříčinit nutnost změny prvku druhého, je tento stav nazýván závislostí. Závislost je jednosměrným vztahem, znázorněná přerušovanou šipkou víz Obr. 2.14, jenž míří z prezentační třídy do třídy doménové.[3]

Asociace (Association) – Jedná se o obecný sémantický vztah mezi prvky modelu, kterým je specifikováno spojení mezi jejich instancemi. Je znázorněna nepřerušovanou šípkou, víz obr. 2.15, směrem od zdrojového prvku k prvku cílovému.[3]
**Agregace (Aggregation)** – Agregací je vyjadřován vztah, kde je cílový prvek součástí prvku zdrojového. Agregace je znázorněna čarou mezi dvěma prvky s prázdným kosočtvercem u zdrojového prvku viz Obr. 2.16.[3]

![Obr. 2.16 Agregace](attachment:image)

**Kompozice (Composition)** – Kompozicí je vyjadřována silnější forma agregace. Platí zde obecné pravidlo, že se prvek může vyskytovat jak součást jiných prvků, ale každá jeho instance je součástí pouze jednoho vlastníka. Pokud je zrušen vlastník, jsou automaticky zrušeny také jeho vlastněné prvky. Kompozice je znázorněna čarou s plným kosočtvercem u vlastníka viz Obr. 2.17.[3]

![Obr. 2.17 Kompozice](attachment:image)

**Generalizace (Generalization)** – Generalizací je vyjádřen vztah dědičnosti mezi předkem a potomkem, kde jsou veškeré vazby od předka zděděny potomkem. Tento vztah je znázorněn nepřerušovanou a nevyplněnou šipkou od potomka k předkovi viz Obr. 2.18.[3]

![Obr. 2.18 Generalizace](attachment:image)

**Realizace (Realization)** – Realizaci je vyjádřen vztah mezi dvěma prvky, ve kterém mezi těmito prvky vzniká dohoda. Dohoda je jedním prvkem určena (dodavatel) a druhým
prvkem (klient) je její uskutečnění zaručeno. Realizace je znázorněna přerušovanou a nevyplněnou šipkou směřující od klienta k dodavateli viz Obr. 2.19.[3]  

![Diagram realizace](image.png)  
Obr. 2.19 Realizace  

c) Diagramy (Diagrams)  

Ve veškerých CASE nástrojích, které jsou založeny na jazyku UML jsou všechny nově vytvořené předměty nebo relace okamžitě přidávány do vznikajícího modelu. Model je tvořen repositářem všech předmětů a relací vytvářených z důvodu, aby jimi bylo popisováno požadované chování modelovaného systému.  

Diagramy jsou pouze okna nebo pohledy na model. Diagramem není myšlen model. Z diagramu lze odstranit předměty a relace, ale v modelu budou stále dále existovat, dokud nebudou z modelu explicitně odstraněny.  

Jazyk UML 2 je zastoupen celkem třinácti diagramy viz Obr. 2.20, které lze rozdělit na diagramy statických struktur a diagramy dynamických struktur systému. Předměty a strukturní asociace jsou zachycovány statickým modelem. Způsoby vzájemného působení mezi předměty, aby bylo dosaženo požadovaného chování systému, jsou zachycovány dynamickým modelem.  

Pořadí modelování UML diagramů není pevně stanoveno. Jedny z prvních modelovaných diagramů většinou jsou diagram případů užití a diagram tříd. Pravdou ale je, že jsou diagramy mnohdy modelovány souběžně.[1]
2.3.2 Společné mechanismy (Common Mechanism in UML)

V celém jazyku UML jsou konzistentně používány čtyři společné mechanismy. Jsou jimi vysvětleny čtyři strategie cesty k modelování objektů, které jsou v celém jazyku UML opakovaně používány v rozdílných kontextech. Jde o:

a) specifikace
b) ornamenty
c) podskupiny
d) mechanismy rozšiřitelnosti[1]

a) Specifikace (Specification)

Pomocí nástroje CASE, ve kterém jsou obsaženy funkce pro zadávání, prohlížení a úpravy jednotlivých prvků modelu, je sémantický podklad zpravidla udržován.

Jelikož v jazyku UML je implementována velká dávka pružnosti a flexibility pro tvorbu modelů je možné, aby modely byly proškrtané, neúplné nebo nekonzistentní.

V **proškrtaném** modelu je možné, aby určité obsažené prvky v podkladu byly v diagramu skryty, například z důvodu jeho větší jednoduchosti. V **neúplném** modelu existuje možnost úplného neobsažení prvků. V **nekonzistentním** modelu existuje možnost obsazení tzv. protimluv.

Pro jazyk UML je existence těchto volných pravidel úplnosti a konzistence nadmíru důležitá, neboť modely jsou postupně rozvíjeny a je v nich uskutečněno mnoho změn. I přes tyto skutečnosti je největší důraz na tvorbu konzistentních modelů, jež jsou natolik úplné, aby bylo možné vytvořit softwarový systém.

Na začátku modelování v jazyku UML je běžně vytvořen grafický model jako vizualizace systému. Poté je podklad modelu doplňován sémantikou a takto lze model označit jako užitečný nebo kompletní. [1]

**b) Ornamenty (Adornments)**

V jazyku UML je každý prvek modelu vyjádřen jednoduchým symbolem. Je-li zapotřebí zobrazit diagramem více informací, lze tyto symboly obohatit řadou ornamentů. Tvorbu složitého modelu je možné nejdříve označit několika symboly s párem ornamentů a postupem času ho můžeme vylepšit ornamenty dešetaktivně podrobně. UML diagramy jsou pouze pohledy na daný model, proto by se ornamenty měli používat jen tehdy, dokud zvyšují jeho srozumitelnost a čitelnost.[1]

**c) Podskupiny (Common Divisions)**

Podskupinami jsou znázorněny různé způsoby vidění světa. Jazykem UML jsou rozlišovány dvě takové podskupiny:

- skupina klasifikátorů a instancí
- skupinu rozhraní a implementací[1]

**Skupina klasifikátoru a instancí**

V jazyku UML se vyskytuje předpoklad představy o abstraktním typu předmětu a také představy o specifickém typu předmětu. Klasifikátor je abstraktním vyjádřením, zatímco instance jsou vyjádřením specifickým.
Instance jsou většinou v jazyku UML znázorněny stejným symbolem jako klasifikátor, ale navíc jsou jejich názvy podtrženy.

V jazyku UML 2 existuje třicet tři klasifikátorů, mezi nejpoužívanější lze zařadit aktera, třídu, komponentu, rozhraní, uzel, signál nebo případ užití.[1]

**Skupina rozhraní a implementace**

Při práci v jazyku UML je potřeba vycházet ze zásady oddělení toho, co je předmětem vykonáváno (rozhraní), od toho, jak je to vykonáváno (implementace). Rozhraním je vymezena dohoda, kterou je zaručováno, jak budou jednotlivé implementace řízeny.[1]

d) **Mechanismy rozšiřitelnosti (Extension Mechanism)**

V jazyku UML jsou zabudovány tři jednoduché rozšiřující mechanismy. Jde o mechanismy omezení, stereotypů a označených hodnot.[1]

**Omezení (Constraints)**

Různými omezeními lze přidávat nová pravidla do prvků modelu. Je jimi specifikována určitá podmínka nebo pravidlo prvku modelu. Omezující podmínku je prostý textový řetězec ve složených závorkách { }.[1]

**Stereotypy (Stereotypes)**

Stereotypem je definován nový prvek modelu, jenž je založen na prvku již existujícím. Stereotypem je tedy zastupována určitá varianta existujícího prvku v modelu, která má sice stejnou podobu (atributy, relace), ale je používána s odlišným záměrem. V každém modelu může být obsažen maximálně jeden stereotyp. Mnoho CASE nástrojů má v sobě zabudováno bohatou základnu již předdefinovaných stereotypů. Stereotypy jsou umisťovány do dvojitých ostrých závorek << >>.[1]

**Označené hodnoty (Tagged Values)**

Prostřednictvím označených hodnot je umožněno rozšiřovat prvky modelu o jejich vlastní vlastnosti. Jakákoli hodnota připojená k prvku modelu je jazykem UML považována za vlastnost. Ve většině prvků jsou již předdefinované vlastnosti, některé mohou být zobrazeny v diagramech, jiné jsou obsaženy v sémantickém podkladu modelu.

V některých označených hodnotách jsou pouze obsaženy informace připojené k prvku modelu, jinými hodnotami jsou vyjadřovány vlastnosti nových prvků modelu, které jsou definovány stereotypem. Je-li aplikován náležející stereotyp na model, jsou rovněž do modelu přidány přidružené značky stereotypu. Označená hodnota je zapisována jako klíčové slovo
s připojenou hodnotou, zapisována ve složených závorkách, např. takto \{student = Jakub Michalík\}[1]

2.3.3 Architektura (UML Architecture)

Architekturou systému je definována organizační struktura systému, včetně jeho rozkladu na součásti, propojitelnosti, interakce, mechanismů, směrných zásad, jež pronikají do návrhu systému.

Architekturou jsou zachycovány strategické aspekty vyšší struktury systému. Nejčastěji je na UML architekturu nahlíženo tzv. pohledem 4+1. Jedná se logický pohled, pohled procesů, pohled implementace a pohled nasazení, všechny tyto pohledy jsou sjednoceny pátého pohledu případů užití.[1]

a) Logický pohled

Logickým pohledem je zachycován slovník oblasti problému jako množiny tříd a objektů. Největší důraz je kladen na zobrazení způsobu implementace chování systému objekty a třídami, které jsou základem systému.[1]

b) Pohled procesů

Pohledem procesů jsou modelovány aktivní třídy spustitelnými vlákny a procesy. Jedná se o procesně orientovanou variantu logického pohledu, jež obsahuje také stejné artefakty.[1]

c) Pohled implementace

Pohledem implementace jsou modelovány soubory a komponenty, prostřednictvím kterých je zhotovován zdrojový kód systému. Tímto pohledem jsou zobrazovány závislosti mezi komponentami a zobrazován způsob jak spravovat uspořádání množin, které byly vytvořeny z těchto komponent. Dále umožňuje definici verze systému.[1]

d) Pohled nasazení

Pohledem nasazení je modelováno fyzické nasazení komponent v množině fyzických výpočetních uzlů a dále distribuce komponent na příslušné uzly distribuovaného systému.[1]

e) Pohled případů užití

Pohledem případů užití jsou odvozeny všechny ostatní pohledy. Jsou tak zachycovány základní požadavky kladené na daný systém jako na množinu případů užití. Je tak vytvořen základ tvorby všech dalších pohledů.[1]
2.4 UML diagramy

Jak již bylo řečeno v podkapitole 2.3.1, v jazyku UML existuje celkem třináct diagramů. V této podkapitole budou blíže popsány především ty typy diagramů jazyka UML, které budou později využity v praktické části diplomové práce. Jedná se o diagramy:

a) případů užití
b) diagram tříd
c) stavový diagram
d) diagram aktivit
e) sekvenční diagram[1]

2.4.1 Diagram případů užití (Use Case Diagram)

Diagram případů užití viz Obr. 2.21. je používán k objasnění chování systému z hlediska uživatele. Jsou tak popisovány typy uživatelů systému, činnosti systému, funkční požadavky systému a interakce mezi uživateli systému a systémem.

Diagramy případů užití jsou obvykle konstruovány jako jedny z prvních diagramů ve fázi analýzy a následně návrhu systému. Pro každý konkrétní případ užití je specifikován scénář jako posloupnost kroků, jimiž jsou definovány interakce mezi uživatelem a systémem.

Hlavní kroky modelování diagramu případů užití jsou tvořeny nalezením hranic systému, aktérů, případů užití, specifikace případů užití a alternativních scénářů.[3][9][1]

Prvky diagramu

Diagram případů užití je tvořen hranicí systému, aktéry, případy užití a vztahy mezi aktéry a případy užití. Nejpoužívanějšími vztahy jsou zahrnutí, rozšíření a vztah generализace.

a) Hranice systému (Boundary)

Hranicí systému je definována tím, kým je systém používán tj. aktérem. Hranice systému je vyjádřena rámečkem. Aktéři jsou zakresleni mimo hranici systému a případy užití uvnitř systému.[3][9]

b) Aktér (Actor)

Aktéry jsou v diagramu případů užití nazývání uživatelé systému. Jedná se o roli, kterou má uživatel ve vztahu k systému. Je-li systémem vykonávána služba pro jiný systém, je systém, pro který je vykonáváno také aktérem.[3][9]
c) Případ užití (Use Case)

Případ užití je představován akcí, jež je vykonávána systémem k dosáhnutí konkrétního cíle. Jedná se o sadu scénářů, které jsou vázány oním společným cílem uživatele.[3][9]

a) Zahrnutí (Include)

Tímto vztahem je vyjádřeno, že jedním případem užití je zahrnován nějaký jiný případ užití.[3]

b) Rozšíření (Extend)

Vztahem extend, je vyjádřen odlišný případ užití, který nastane, pokud je splněna daná podmínka.[3]

c) Generalizace (Generalization)

Generalizaci je vyjádřen vztah dědičnosti mezi předkem a potomkem, kde jsou veškeré vazby od předka zděděny potomkem. Správné využití dědičnosti je spojeno s principem nahraditelnosti, tzn potomek by měl být schopen nahradit svého předka.[3]

2.4.2 Diagram tříd (Class Diagram)

Diagramem tříd viz Obr. 2.22 je zobrazen statický pohled na modelovaný systém. Jsou jim popsány typy objektů v systému a vztahy mezi nimi. Vzhledem k faktu, že diagramem tříd jsou znázorňovány pravidla modelovaného systému, jedná se o velmi důležitý podklad jak pro forward engineering, tak pro reverese engineering.

Jsou rozeznávány tři úrovně modelu tříd. Jedná se o konceptuální, designovou a implementační úroveň.[3][10]
a) Konceptuální model

Za účelem analýzy požadavků na systém je vytvářen konceptuální model. Jsou v něm obsaženy tzv. business třídy, prostřednictvím kterých je modelována problémová oblast. V třídách jsou většinou obsaženy pouze klíčové atributy a klíčové metody. Jestliže je diagram tvořen z důvodu znázornění relací mezi třídami, lze atributy i metody vynechat.[10]

b) Designový model

Konceptuální model je základem pro tvorbu modelu designového. Konceptuální model je designovým modelem rozšířen a zpřesněn, například o viditelnost atributů a metod, datových typů apod. Jsou zde také do modelu přidány třídy uživatelského rozhraní a třídy, ve kterých jsou obsaženy systémové události.[10]

c) Implementační model

Účelem implementačního modelu je zaměřit se na návrh a tvorbu programového kódu modelovaného systému.[10]

Prvky diagramu

Nejvíce používanými prvky diagramu tříd jsou prvky třídy a rozhraní. Nejčastější používanými vztahy jsou asociace, generalizace a závislost.

a) Třída (Class)

Třída je reprezentována množinou objektů, jež jsou složeny z identických vlastností. V třídě jsou obsaženy atributy a metody. Atributy jsou představovány vlastnosti dané třídy, metodou jsou představovány operace, které může třída provádět.[8][3]

b) Rozhraní (Interface)

Rozhraní je v UML datovým typem, ve kterém jsou obsaženy pouze operace bez implementace tzv. veřejné abstraktní operace.[3]

c) Asociace (Association)

Jedná se o obecný sémantický vztah mezi prvky modelu, kterým je specifikováno spojení mezi jejich instancemi. Pokud není vztah asociace jasný, je jí přidělen název. U asociace v diagramu tříd je znázorněna také její násobnost.

Násobnost může nabývat hodnot:

- 0..1 - žádná nebo jedná instance
- 0..* - žádná nebo několik instancí
- 1 - právě jedna instance
- 1..* - jedná nebo několik instancí[3]
d) Závislost (Dependency)

Pokud změna definice jednoho prvku dokáže zapříčinit nutnost změny prvku druhého, je tento stav nazýván závislostí.[3]

![Diagram of Class and Dependency](image)

Obr. 22 Class Diagram

2.4.3 Stavový diagram (State Machine Diagram)

Stavovým diagramem viz Obr. 2.23 je popisováno chování systému jedné třídy, kterým je zobrazováno chování jednoho objektu v rámci celého životního cyklu. Jsou zde zachycovány konkrétní stavy objektů a přechody mezi nimi.[3][11]

Prvky diagramu

Nejvíce používané prvky stavového diagramu jsou počáteční stav, konkrétní stav, přechod a koncový stav.

a) Počáteční stav (Initial State)

Jedná se o počáteční pseudostav, který vede k prvnímu konkrétnímu stavu.[3]

b) Stav (State)

Stavem je nazývána situace v životním cyklu objektu, ve které je objektem splněna určitá podmínka, prováděna určitá operace nebo je očekávána určitá událost.[11]

c) Přechod (Transitions)

Přechodem je vyjadřována změna stavu z jednoho do jiného. V diagramu jsou značeny linií vedoucí od jednoho stavu k jinému. U každého přechodu je obsažena jmenovka, která je složena ze tří částí a to spouštěče, podmínky a aktivity. Všechny části, ale nejsou povinné. Spouštěčem je zobrazena jedna událost, prostřednictvím které je spuštěna potencionální
změna stavu. Podmínkou je uveden výraz, který musí být pravdivý, aby mohl nastat přechod. Aktivitou je zobrazeno určité chování, jež bude vykonáno během přechodu.[3]

d) Koncový stav (Final State)

Jedná se o konečný pseudostav, do kterého je veden přechod ze stavu v poslední fázi jeho životního cyklu. Je jim naznačen konec procesu stavového diagramu.[3]

Obr. 23 State Machine Diagram

2.4.4 Diagram aktivit (Activity Diagram)

Diagramy aktivit viz Obr. 2.24 jsou určeny k popisu procedurální logiky, business procesů a toku práce. V mnoha ohledech je diagram aktivit podobný diagramu stavovému. V UML 1 také byl součástí diagramu stavů. V UML 2 se již jedná o samostatný diagram, který navíc umožňuje znázornění paralelních procesů a je modelován v jazyce Petri Nets.[1][3][12]

Prvky diagramu

Nejčastěji používanými prvky jsou počáteční uzel, rozvětvení, spojení, rozhodování, splynutí, plavecké dráhy koncový uzel.

a) Počáteční uzel (Initial Node)

Jedná se o počáteční pseudostav, který vede k prvnímu akci.[3]

b) Rozvětvení (Fork)

Rozvětvením je umožněn běh paralelních procesů. Je spojen s jedním vstupním tokem a několika toky výstupními.[3]
c) **Spojení (Join)**

Pokud jsou modelovány paralelní procesy, je zapotřebí je zase spojit, prostřednictvím spojení.[3]

d) **Rozhodování (Decision)**

Rozhodováním je umožněno znázornit podmíněné chování. Je spojen s jedním vstupním tokem a několika podmíněnými toky výstupními. Každý výstupní tok je spojen s určitou podmínkou, které se musí navzájem vylučovat. Vždy když se dospěje k rozhodování, lze pokračovat jen jedním výstupním tokem.[3]

e) **Splynutí (Merge)**

Splynutím je umožněno spojit vícenásobné vstupní toky a vytvořit tak jeden tok výstupní.[3]

f) **Plavecké dráhy (Swim Lanes)**

Plaveckými dráhami je umožněno diagram aktivit rozdělit na části, aby bylo objasněno, kzym jsou aktivity prováděny.[3]

g) **Koncový uzel (Final Node)**

Jedná se o konečný pseudostav, do kterého je vedena poslední aktivita a diagram aktivit je tím tak ukončen.[3]
2.4.5 Sekvenční diagram (Sequence Diagram)

Sekvenční diagram viz Obr. 2.25 je součástí skupiny diagramu interakcí. Jedná se asi o nejvíce používaný diagram této skupiny. S jeho pomoci je zobrazováno chování a spolupráce mezi jednotlivými objekty pro jeden konkrétní případ užití.[3][11]

Prvky diagramu

Sekvenční diagram je nejčastěji tvořen klasifikátory, čárami života, zprávami a poznámkami.[3][11]

a) Klasifikátoři (Classifiers) Zprávy (Messages)

Klasifikátoři jsou složeni z objektů, tříd nebo objektů, mezi kterými jsou zasílány zprávy. Každý klasifikátor je spojen s čárou života.[3][11]

b) Čáry života (Lifeline)

Čáry života jsou orientovány svisle. Každá čára života je spojena s pruhem aktivace, prostřednictvím kterého je znázorněno, kdy je klasifikátor v interakci aktivní. Aktivita je spojena s jednou z metod nebo operací, která je klasifikátoru k dispozici.[3]

c) Zprávy (Messages)

Zprávy jsou tvořeny metodami konkrétních tříd nebo operacemi. První zpráva, ke které není přidružen klasifikátor a je tak poslána neurčitým zdrojem se nazývá nalezená zpráva.[3]

d) Poznámky (Notes)

Konkrétní zprávy mezi objekty, bývají doprovázeny poznámkami s popisem dané sekvence.[13]
3 Analýza současné situace a požadavky uživatelů

V této kapitole diplomové práce bude stručně specifikována organizační struktura a předmět činnosti organizace ORLOVAN, bytové družstvo, pro kterou je IS evidence a sledování úkolů navrhován. Dále bude analyzována současné situace řešeného problému, její nedostatky a určení požadavků na nový systém. Informace byly zjištěny na základě rozhovoru a pozorování zaměstnanců družstva.

3.1 Popis organizace


K 1. prosinci 2004 bylo družstvo zapsáno v obchodním rejstříku. Stalo se vlastníkem 166 domů s 3 612 bytovými jednotkami, které se staly členským vkladem Města Orlová, se splatností od 1. 7. 2005. Na základě stanov družstva se dalším členem družstva stává každý dosavadní nájemce bytu, kterému vlastník práv a povinností k bytu Město Orlová tato práva převede.

Řádné volby do orgánů družstva se uskutečnili dne 26. 9. 2007 na shromáždění delegátů družstva.

3.1.1 Předmět činnosti

Předmět činnosti bytového družstva tvoří především správa bytových a nebytových objektů ve vlastnictví nebo spoluvlastnictví družstva.

Jedná se zejména o tyto činnosti:

- údržba
- výměny
- opravy
- rekonstrukce
- modernizace

3.1.2 Organizační struktura

Organizační hierarchie je složena ze čtyř pracovních úseků, kde úsek vedení družstva, je ostatním nadřazen viz níže Obr. 3.1.
3.2 Analýza současné situace

V současné době v organizaci bytového družstva neexistuje systém, který by byl specializován na evidenci a sledování vnitropodnikových úkolů. Veškeré operace vedoucí k zadání úkolu zaměstnancům jsou prováděny třemi způsoby a to osobně nebo telefonicky. Relativně krátkou dobu mohou být úkoly zadány taktéž elektronickou formou prostřednictvím poštovního klienta MS Outlook. Tento poštovní klient ale není nainstalován na veškerém dostupném počítačovém vybavení, což vede k neefektivnímu řízení úkolů v organizaci zmíněným způsobem.

Evidence a sledování úkolů

V organizaci ORLOVAN , bytové družstvo jsou úkoly zadávány a řízeny osobně, telefonicky nebo prostřednictvím poštovního klienta, jejichž klady a zápory budou vysvětleny níže.

3.2.1 Osobní zadání úkolu

Zadavatel musí osobně dojít k realizátorovi a sdělit mu obsah úkolu, což vede k lepšímu vysvětlení úkolu i jeho pochopení realizátorem. Na druhou stranu zadavatel dočasně opouští místo svého pracoviště, odlučuje se od své práce a je nemožné jej kontaktovat z venkovního prostředí. To může vést ke zhoršení image organizace, pokud se zadávání úkolu prodlouží. Dále zadavatel i realizátor jsou nuceni si dělat osobní poznámky, kde tato situace následně při neustále se zvětšujícím množství úkolů, vede k celkové nepřehlednosti a vysoké časové náročnosti.
3.2.2 Telefonické zadání úkolu

Realizátorovi je zadavatelem zadán úkol prostřednictvím telefonní linky. Jedná se o rychlejší způsob než je osobní zadání úkolu, zadavatel neopouští místo svého pracoviště, ale i zde je relativně vysoká časová náročnost. Tato možnost je neosobní, zadavatel i realizátor jsou nuceni si dělat vlastní poznámky, které sice vedou k vyšší hodnotě srozmýšleností, ale i k zminěné vyšší časové náročnosti. Dále jsou blokovány příchozí hovory ať už zvenčí nebo zevenčí.

3.2.3 Zadání úkolu prostřednictvím poštovního klienta

Tato možnost zadání úkolu je oproti výše uvedeným způsobům modernější a pro zadavatele také pohodlnější. Její využitelnost bohužel snižuje nepřítomnost zmíněného klienta na většině počítačového vybavení bytového družstva, navíc mezi zaměstnanci chybí standard pro prováděné operace.

Při užívání poštovního klienta MS Outlook pro zadávání úkolu zaměstnanci využívají tři způsoby jak jej doručit realizátorovi. A to prostřednictvím navigačního panelu zasláním klasické e-mailové pošty - karta pošta, sdílením události - karta kalendář nebo vytvořením úkolu - karta úkoly.

Karta pošta

Zadavatel se nachází v klasickém prostředí elektronické pošty, kde v procesu tvorby nové poštovní zprávy vybere realizátora (příjemce pošty), vyplní další požadované náležitosti zprávy a do pole pro text zprávy vepíše zadání úkolu s jeho termínem splnění a následně zprávu odešle. Hlavním nedostatkem této možnosti zadání úkolu je nemožnost sledování průběhu či stavu úkolu, zkreslování skutečnosti realizátorem, pokud si zprávu přečte v okně náhledu, kdy se zpráva neoznačí jako přečtená. V případě že se realizátor nachází momentálně mimo pracoviště nebo zrovna čerpá dovolenou, dojde k přečtení zprávy se značným zpožděním. Dále realizátor nedisponuje rychlou volbou odmítnutí úkolu s příslušným odúvodněním. Pokud mu není úkol srozmýšlený z důvodu neexistence šablon, je tato situace řešena opět e-mailem, telefonicky nebo osobně. V neposlední řadě mezi hlavní nedostatky patří existence limitovaného prostoru e-mailové schránky, kdy při zaplnění přiděleného prostoru dochází k dočasné blokací zaslání nebo přijetí nového e-mailu. Navíc v této situaci může při čistění přiděleného prostoru dojít k nechtěnému odstranění zpráv ze schránky a to i nedávno přiděleného úkolu.
**Karta kalendář**

Prostřednictvím kalendáře zadavatel vytvoří na požadovaný den v roce událost s rozsáhlým výborem vlastností téže události. Může zde nastavit čas zahájení a ukončení, místo konání, připomenutí atd. Dále vybere realizátory (účastníky) této události a odešle ji. Realizátorovi nebo skupině realizátorů se následně doručí e-mail s obsahem události a zapiše se jím do kalendáře. Zde se znovu vyskytuje problém limitovaného prostoru e-mailové schránky. Dalším nedostatkem považuji nepříliš šťastné řešení konfliktů jednotlivých událostí, kdy může lehce dojít k jejímu přehlédnutí.

**Karta úkoly**


**3.3 Problémové oblasti**

Tato část diplomové práce se bude zabývat shrnutím všech nedostatků stávající situace při evidenci a sledování úkolu bytového družstva, která vychází z předešlé analýzy. Situace bude znázorněna tabulkou, viz Tab. 3.2, kde bude uveden způsob evidence, stručný popis a zmíněné zápory jednotlivé situace.
Tab. 3.1 Specifikace problémů

3.4 Specifikace softwarových požadavků

V této fázi analýzy budou specifikovány veškeré požadavky na navrhovaný systém. Bude zde objasněn cíl a specifikace systému. Uživatelské požadavky budou kategorizovány na funkční a nefunkční požadavky

3.4.1 Cíl navrhovaného systému

Cílem je navrhnout informační systém, který bude umístěn v intranetovém prostředí organizace ORLOVAN, bytové družstvo. Systém bude přehledný, jednoduchý, usnadní zaměstnancům družstva evidovat a sledovat vnitropodnikové úkoly a tím dosáhnout vyšších úspor času.

---

Karta úkoly v MS Outlook umožňuje možnosti sledování průběhu, stavu i filtraci úkolu.
3.4.2 Specifikace systému

Aplikace bude vytvořena jako subsystém intranetu bytového družstva a bude sloužit pouze pro jejich zaměstnance. Po spuštění intranetu bude uživatel automaticky identifikován a přihlášen do všech částí intranetu, tedy i do subsystému evidence a sledování úkolů. Po úspěšném přihlášení bude uživateli umožněné využívat veškeré funkcionality na základně jemu přidělených práv.

3.4.3 Funkční požadavky

1. systém umožní zadávání úkolu do IS
2. systém umožní vytvářet události
3. systém umožní editaci úkolu, události
4. systém umožní vytvářet šablony úkolů
5. systém umožní export úkolů
6. systém bude obsahovat uživatelské profily
7. systém bude evidovat všechny úkoly (potvrzené, nepotvrzené, rozpracované, dokončené, nedokončené) a události
8. systém bude obsahovat opatření proti odstranění úkolu
9. systém bude signalizovat přijetí úkolu
10. systém bude zobrazovat stav úkolu
11. systém bude zobrazovat rozpracovanost úkolu
12. systém umožní filtrace úkolu
13. systém bude obsahovat „rychlou volbu“ (odmítnout, specifikovat)
14. systém umožní editovat události
15. systém bude obsahovat správce systému (spravovat pozice, úseky, oprávnění, družstvo, uživatele)

3.4.4 Nefunkční požadavky

1. systém bude realizován formou webového rozhraní a bude umístěn do prostředí intranetu družstva (sub systém)
2. systém bude obsahovat úvodní obrazovku (Dashboard)
3. systém bude obsahovat kalendář
4. rychlost, jednoduchost vytváření
5. přehlednost
6. odezva systému
7. průměrná HW/SW náročnost
4 Návrh systému IS

V této kapitole diplomové práce bude vytvořen návrh subsystému evidence a sledování úkolů organizace ORLOVAN, bytového družstvo, který bude vycházet z analýzy provedené v předchozí kapitole.

Návrh bude tvořen modelovacím jazykem UML, který slouží jako univerzální jazyk pro vizuální modelování systémů.

Veškeré UML diagramy jsou vytvořeny v prostředí Enterprise Architect, které bylo vybráno z důvodu jeho relativně častého výskytu v pracovních požadavcích firem, zabývajících se softwarovým inženýrstvím.

4.1 Diagram případů užití

Diagram případů užití je používán k objasnění chování systému z hlediska uživatele. Jsou tak popisovány typy uživatelů systému, činnosti systému, funkční požadavky systému a interakce mezi uživateli systému a systémem.[3]

Pro systém evidence a sledování úkolů jsou navrženy tři diagramy případů užití:

- diagram případů užití aktérů viz Obr. 4.1
- diagram případů užití uživatele viz Obr. 4.2
- diagram případů užití správce systému viz Obr. 4.3

4.1.1 Diagram případů užití aktérů

Obr. 27 Diagram případů užití aktérů
Specifikace diagramu případů užití aktérů

<table>
<thead>
<tr>
<th>Název aktéra</th>
<th>Popis aktéra</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uživatel</td>
<td>Uživatelem je každý zaměstnanec bytového družstva, který je přihlášen do systému evidence a sledování úkolů v prostředí Intranetu zmíněného družstva a využívá jeho běžných funkcionalit.</td>
</tr>
<tr>
<td>Správce systému</td>
<td>Správcem systému je vybraný zaměstnanec nebo zaměstnanci, kteří jsou přihlášeni do systému evidence a sledování úkolů. Mají přidělena vyšší oprávnění oproti uživateli, pro operace s daty.</td>
</tr>
</tbody>
</table>

Tab. 4.1 Specifikace diagramu případů užití

4.1.2 Diagram případů užití uživatele

Obr. 4.28 Diagram případů užití uživatele

Specifikace diagramu případů užití uživatelo

<table>
<thead>
<tr>
<th>Název případu užití</th>
<th>UC1 Vytvářet úkoly</th>
</tr>
</thead>
<tbody>
<tr>
<td>Popis</td>
<td>Systém evidence a sledování úkolů umožňuje vytvořit úkol.</td>
</tr>
</tbody>
</table>
| Aktéři              | • Uživatel  
 |                     | • Správce systému  
 |                     | • Systém |
| Podmínky spuštění   | Uživatel musí být v systému přihlášen a zadat volbu pro vytvoření úkolu. |
1. Systém vygeneruje okno pro vytvoření úkolu.
2. Uživatel vyplní náležitosti úkolu, jméno realizátora a odešle jej.
   3.1. Pokud data nejsou validní, systém uživatele upozorní a nepovolí úkol odeslat.
   3.2. Uživatel opraví nevalidní data a úkol odešle.
4. Systém úkol uloží.

Je vytvořen úkol a uložen do sekce Evidence úkolů

### Název případu užití | UC2 Vytvářet událost
---|---
**Popis** | Systém evidence a sledování úkolů umožňuje vytvořit událost.
**Aktéři** | • Uživatel
• Správce systému
• Systém

Základní tok
1. Systém vygeneruje okno pro vytvoření události.
2. Uživatel vyplní náležitosti události, jméno realizátora a odešle ji.
   3.1. Pokud data nejsou validní, systém uživatele upozorní a nepovolí úkol odeslat.
   3.2. Uživatel opraví nevalidní data a úkol odešle.
4. Systém úkol uloží

Podmínky dokončení
Je vytvořena událost a uložena v sekci Kalendář

### Název případu užití | UC3 Editovat vytvořené úkoly/události
---|---
**Popis** | Systém evidence a sledování úkolů umožňuje editovat vytvořené úkoly/události.
**Aktéři** | • Uživatel
• Správce systému
• Systém

Základní tok
1. Systém otevře okno s vytvořeným úkolem/událostí.
2. Uživatel edituje náležitosti úkolu/události a odešle ji.
2.1. Uživatel zvolí možnost smazání úkolu.
3. Systém zkontroluje validitu dat úkolu/události nebo úkol smaže.
   3.1. Pokud data nejsou validní, systém uživatele upozorní a nepovolí úkol/událost odeslat.
   3.2. Uživatel opraví nevalidní data a úkol/událost odešle.
4. Systém úkol/událost uloží

Podmínky dokončení
Je editován úkol/událost a uložen v příslušené sekci.
Název případu užití | UC4 Vytvářet šablony
--- | ---
**Popis** | Systém evidence a sledování úkolu umožňuje vytvářet šablony úkolů.

**Aktéři** | • Uživatel  
• Správce systému  
• Systém

**Podmínky spuštění** | Uživatel musí být v systému přihlášen a zadat volbu pro vytvoření šablony úkolů.

**Základní tok** | 1. Systém vygeneruje okno pro vytvoření šablony úkolů.  
2. Uživatel vyplní náležitosti šablony.  
4. Po skončení procesu vytváření systém šablony uloží.

**Podmínky dokončení** | Je vytvořena šablona úkolu a uložena v sekci Evidence úkolů

---

Název případu užití | UC5 Export úkolů
--- | ---
**Popis** | Systém evidence a sledování úkolu umožňuje export úkolů.

**Aktéři** | • Uživatel  
• Správce systému  
• Systém

**Podmínky spuštění** | Uživatel musí být v systému přihlášen a zadat volbu pro export úkolů.

**Základní tok** | 1. Systém vygeneruje náhled úkolů ve formátu PDF.  
1.1. Uživatel v náhledu úkolů vybere možnost uložení  
1.2. Uživatel v náhledu úkolů vybere možnost tisku  
1.3. Uživatel v náhledu úkolů vybere možnost odeslání e-mailem  
2. Systém úkoly uloží, vytiskne nebo pošle e-mailem

**Podmínky dokončení** | PDF soubor s úkoly je uložen, vytištěn nebo poslán na e-mail.

---

Název případu užití | UC6 Editovat svůj profil
--- | ---
**Popis** | Systém evidence a sledování úkolu umožňuje editovat uživatelský profil.

**Aktéři** | • Uživatel  
• Správce systému  
• Systém

**Podmínky spuštění** | Uživatel musí být v systému přihlášen a zadat volbu pro editaci svého profilu.

**Základní tok** | 1. Systém zobrazí údaje o uživateli.  
2. Uživatel edituje vybrané údaje.  
3. Systém po ukončení editace oznámí správci systému požadavek na změnu údajů.  
3.1. Pokud správce systému změny potvrdí, systém údaje změní.  
3.2. Pokud správce systému změny nepotvrdí, systém nezmění.

**Podmínky dokončení** | Jsou změněny uživatelské údaje.
4.1.3 Diagram případů užití správce systému

[Diagram případů užití správce systému]

Specifikace diagramu případů užití správce systému

<table>
<thead>
<tr>
<th>Název případu užití</th>
<th>UC8 Spravovat pozice</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Popis</strong></td>
<td>Systém evidence a sledování úkolů umožňuje správci systému editovat pracovní pozice bytového družstva.</td>
</tr>
</tbody>
</table>
| **Aktěři**          | • Správce systému  
                    | • Systém |
| **Podmínky spuštění** | Správce systému musí být v systému přihlášen a zadat volbu pro editování pozic. |
| **Základní tok**    | 1. Systém zobrazí okno se seznamem pozic.  
                        2. Správce systému edituje vybranou pozici.  
                        3.1. Pokud data nejsou validní, systém správce systému upozorní a nepovolí upravená data uložit.  
                        3.2. Správce systému opraví nevalidní data.  
                        4. Systém pracovní pozici uloží. |
| **Podmínky dokončení** | Je editována pracovní pozice. |

Tab. 4.9 Specifikace dat pozic

<table>
<thead>
<tr>
<th>Název případu užití</th>
<th>UC9 Spravovat úseky</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Popis</strong></td>
<td>Systém evidence a sledování úkolů umožňuje správci systému editovat pracovní úseky bytového družstva.</td>
</tr>
</tbody>
</table>
| **Aktěři**          | • Správce systému  
                    | • Systém |
| **Podmínky spuštění** | Správce systému musí být v systému přihlášen a zadat volbu pro editování pracovních úseků |
| **Základní tok**    | 1. Systém zobrazí okno se seznamem pracovních úseků. |
2. Správce systému edituje vybraný pracovní úsek.
3. Systém zkontroluje validitu dat úseku.
3.1. Pokud data nejsou validní, systém správce systému upozorní a nepovolí upravená data uložit.
3.2. Správce systému opraví nevalidní data.
4. Systém pracovní úsek uloží.

**Podmínky dokončení**
Je editován pracovní úsek.

<table>
<thead>
<tr>
<th>Název případu užití</th>
<th>UC10 Spravovat oprávnění</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Popis</strong></td>
<td>Systém evidence a sledování úkolů umožňuje správci systému měnit uživatelská práva uživatele systému.</td>
</tr>
</tbody>
</table>
| **Aktéři** | • Správce systému  
• Systém |
| **Podmínky spuštění** | Správce systému musí být v systému přihlášen a zadat volbu pro úpravu oprávnění uživatele. |
| **Základní tok** | 1. Systém zobrazí okno se seznamem uživatelů systému.  
2. Správce systému mění uživatelská práva.  
3. Systém zkontroluje validitu dat oprávnění.  
3.1. Pokud data nejsou validní, systém správce systému upozorní a nepovolí upravená data uložit.  
3.2. Správce systému opraví nevalidní data.  
4. Systém uloží oprávnění uživatele. |
| **Podmínky dokončení** | Jsou upravena uživatelská práva. |

<table>
<thead>
<tr>
<th>Název případu užití</th>
<th>UC11 Spravovat uživatele</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Popis</strong></td>
<td>Systém evidence a sledování úkolů umožňuje správci systému editovat uživatele systému.</td>
</tr>
</tbody>
</table>
| **Aktéři** | • Správce systému  
• Systém |
| **Podmínky spuštění** | Správce systému musí být v systému přihlášen a zadat volbu pro editaci uživatele. |
| **Základní tok** | 1. Systém zobrazí okno se seznamem uživatelů systému.  
2. Správce systému edituje uživatele.  
3. Systém zkontroluje validitu dat uživatele.  
3.1. Pokud data nejsou validní, systém správce systému upozorní a nepovolí upravená data uložit.  
3.2. Správce systému opraví nevalidní data.  
4. Systém uloží uživatele. |
| **Podmínky dokončení** | Jsou upraveny data uživatele. |

<table>
<thead>
<tr>
<th>Název případu užití</th>
<th>UC12 Spravovat družstvo</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Popis</strong></td>
<td>Systém evidence a sledování úkolů umožňuje správci systému editovat informace o družstvu.</td>
</tr>
</tbody>
</table>
| **Aktéři** | • Správce systému  
• Systém |
| **Podmínky spuštění** | Správce systému musí být v systému přihlášen a zadat volbu pro editaci družstva. |
Základní tok
1. Systém zobrazí okno s informacemi družstva.
2. Správce systému edituje data družstva.
3.1. Pokud data nejsou validní, systém správce systému upozorní a nepovolí upravená data uložit.
3.2. Správce systému opraví nevalidní data.
4. Systém uloží informace o družstvu.

Podmínky dokončení
Jsou upraveny informace družstva.

<table>
<thead>
<tr>
<th>Tab. 4.13 Specifikace dat družstva</th>
</tr>
</thead>
</table>

4.1.4 Matice trasovatelnosti požadavků

<table>
<thead>
<tr>
<th>Požadavky \ Případy užití</th>
<th>1.</th>
<th>2.</th>
<th>3.</th>
<th>4.</th>
<th>5.</th>
<th>6.</th>
<th>7.</th>
<th>8.</th>
<th>9.</th>
<th>10.</th>
<th>11.</th>
<th>12.</th>
<th>13.</th>
<th>14.</th>
<th>15.</th>
</tr>
</thead>
<tbody>
<tr>
<td>UC1</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC2</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC3</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td>x</td>
<td></td>
<td>x</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC4</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC5</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC6</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC7</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC11</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UC12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tab. 4.14 Matice trasovatelnosti požadavků

4.2 Diagram tříd

Diagramem tříd je zobrazen statický pohled na modelovaný systém. Jsou jim popsány typy objektů v systému a vztahy mezi nimi. Vzhledem k faktu, že diagramem tříd jsou znázorňovány pravidla modelovaného systému, jedná se o velmi důležitý podklad jak pro forward engineering, tak pro reverse engineering.[3][10]
4.2.1 Diagram tříd systému

Na Obr. 4.4 je navržen diagram tříd pro evidenci a sledování úkolu bytového družstva, který zobrazuje třídy a jejich vztahy mezi jednotlivými třídami.

4.3 Stavový diagram

Stavovým diagramem je popisováno chování systému jedné třídy, kterým je zobrazováno chování jednoho objektu v rámci celého životního cyklu. Jsou zde zachycovány konkrétní stavy objektů a přechody mezi nimi.[3][11]

4.3.1 Stavový diagram úkolu

Pro systém evidence a sledování úkolů stavový diagram zachycuje změny stavu vytvořeného nebo editovaného úkolu viz Obr a vytvořené události viz Obr 4.5.
4.3.2 **Stavový diagram události**

Na Obr. 4.6 jsou znázorněny jednotlivé stavy, v nichž se můžou jednotlivé události nacházet.
4.4 Diagram aktivit

Diagramy aktivit jsou určeny k popisu procedurální logiky, business procesů a toku práce. V mnoha ohledech je diagram aktivit podobný diagramu stavovému. V UML 1 také byl součástí diagramu stavů. V UML 2 se již jedná o samostatný diagram, který navíc umožňuje znázornění paralelních procesů a je modelován v jazyce Petri Nets.[1][3][12]

4.4.1 Diagram aktivit úkolu

Na Obr. 4.7 se nachází diagram aktivit, který znázorňuje posloupnost aktivit v procesu tvorby úkolu v systému bytového družstva. V diagramu jsou pro lepší srozumitelnost zobrazeny plavecké dráhy. Jedná se o plavecké dráhy pro zadavatele, realizátora a systém.
4.5 Sekvenční diagram

Sekvenční diagram je součástí skupiny diagramu interakcí. Jedná se asi o nejvíce používaný diagram této skupiny. S jeho pomocí je zobrazováno chování a spolupráce mezi jednotlivými objekty pro jeden konkrétní případ užití.[3][11]

4.5.1 Sekvenční diagram úkolu

Sekvenčním diagramem úkolu na Obr. 4.8 je znázorněna situace interakce mezi zadavatelem úkolu a systémem, kdy si v procesu tvorby úkolu mezi sebou zasílájí zprávy.
4.6 Návrh rozhraní

V této podkapitole diplomové práce bude objasněna funkční specifikace systému a budou zde obsaženy návrhy jednotlivých rozhraní subsystému evidence a sledování úkolů bytového družstva.

Jedná se o návrhy rozhraní:
- Dashboardu uživatele systému
- Dashboardu správce systému
- tvorby úkolu
- evidence úkolů
- tvorby události
- kalendáře

4.6.1 Funkční specifikace

Funkční specifikace vychází z analýzy uživatelských požadavků, ze kterých vyplivají následující funkce informačního systému.
Informační systém bude sloužit k evidenci a sledování vnitropodnikových úkolů, jejich stavu a rozpracovanosti. Systém bude obsahovat Dashboard s hlavními sekcemi Vytvořit úkol, Vytvořit událost, Evidence úkolů a Kalendář.

V sekci Vytvořit úkol, bude zadavateli umožněno vytvoření nového úkolu a jeho následné odeslání realizátorovi nebo skupině realizátorů. Dále je zde možnost vytvořit šablonu úkolu.

V sekci Evidence úkolů bude seznam zadaných úkolů, jejich stav (potvrzený, nепotvrzený, dokončený, nedokončený), rozpracovanost (25%, 50%, 75%, 100%) a možnost exportu.

V sekci Kalendář bude umožněno vytvářet, editovat a zobrazit události včetně jejich detailu.

Oprávnění editovat událost nebo úkol bude mít pouze jejich zadavatel (autor úkolu). Každá ze sekcí bude umožňovat rychlé přepnutí do jiné sekce.

### 4.6.2 Dashboard uživatele systému

Na Obr. 4.9 je zображeno uživatelské rozhraní Dashboardu (úvodní obrazovka) standardního uživatele systému. Po spuštění aplikace běžným uživatelem to je vždy první uživatelské rozhraní, které je zobrazeno. Rozhraní je rozděleno na tři části. V horní části je zobrazeno jméno uživatele, jeho avatar (fotka uživatel) a název aplikace a organizace. V dolní levé části jsou znázorněny nabídky akcí uživatele. V dolní pravé části je zobrazen přehled, jehož cílem je uživatele informovat o aktuálním datu, čase a jeho úkolech. Toto rozhraní se vztahuje k případům užití uživatele UC1, UC2, UC3, UC6 a třídě Uzivatel.

![Dashboard uživatele systému](image.png)
4.6.3 **Dashboard správce systému**

Na Obr. 4.10 je zobrazeno uživatelské rozhraní Dashboardu uživatele, který má oprávnění správce subsystému. Pro správce systému je to první rozhraní, které mu bude po spušťení aplikace zobrazeno. Rozdíl v rozhraní oproti běžnému uživateli je v nabídce akcí. Správci systému je umožněno vykonat o pět akcí více. K tomuto rozhraní se vztahují případy užití správce systému UC8, UC9, UC10, UC11, UC12 a třída SpraveceSystemu.

Obr. 4.36 Dashboard správce systému

4.6.4 **Tvorba úkolu**

Na Obr. 4.11 je znázorněno uživatelské rozhraní pro tvorbu úkolů. Rozhraní je systémem zobrazeno po vyvolání akce **VYTVOŘIT ÚKOL**. Jeho hlavní část je tvořena nástrojovou lištou, náležitostmi úkolu a polem pro zadání textového obsahu úkolu. Po vyplnění náležitostí úkolu je uživateli umožněno vytvořený úkol uložit do systému a tím jej odeslat realizátorovi, skupině realizátorů nebo jej zrušit. K tomuto rozhraní se vztahuje třída Ukol a případy užití uživatele UC1, UC4, UC6.

Obr. 4.37 Tvorba úkolu
4.6.5 Evidence úkolů

Na Obr. 4.12 je znázorňeno uživatelské rozhraní evidence úkolů. Rozhraní je systémem zobrazeno po vyvolání akce EVIDENCE ÚKOLŮ. Jeho hlavní část je tvořena nástrojovou lištou a seznamem všech úkolů. Uživateli jsou k dispozici mimo evidence úkolů také funkce exportu, filtrace, a vytvoření úkolu. K rozhraní se vztahují případy užití uživatele UC1, UC3, UC4, UC5, UC6 a třída EvidenceUkolu.

Obr. 4.38 Evidence úkolů

4.6.6 Tvorba události


Obr. 4.39 Tvorba události
4.6.7 Kalendář

Na Obr. 4.14 je znázorněno uživatelské rozhraní kalendáře. Rozhraní je systémem zobrazeno po vyvolání akce KALENDÁŘ. Jeho hlavní části jsou tvořeny nástrojovou lištou, denním, týdenním nebo měsíčním formátem zobrazení událostí a v pravé menší části je umístěna miniatura měsíčního kalendáře. K tomuto rozhraní se vztahuje třída Kalendar a případy užití uživatele UC2, UC3, UC6.
5 Závěr

Zvolené téma diplomové práce vycházelo z potřeby odstranění bariér způsobů zadávání a evidování úkolů v organizaci ORLOVAN, bytové družstvo v obci Orlová-Lutyně. Cílem práce bylo navrhnout nové řešení ve formě subsystému evidence a sledování úkolů na základě uskutečněné analýzy, tento cíl byl splněn v praktické části diplomové práce.

Systém byl navržen na míru dle potřeb organizace ORLOVAN, bytové družstvo. Systém byl navržen jako webové rozhraní, aby mohlo být umístěno do interního systému družstva, kde bude pro všechny zaměstnance k dispozici.

V první části práce jsem se po krátkém popisu družstva zabýval analýzou současného stavu, tedy jak nyní dochází k zadávání a evidování úkolu v organizaci, dále analýzou požadavků na budoucí systém. Ze získaných informací jsem specifikoval problémové oblasti, následně také požadavky na nový systém, které jsem rozdělil na funkční a nefunkční požadavky.


Analýzy současného stavu, požadavků uživatelů a následný návrh systému jsou náročné činnosti. Diplomová práce by mohla být pro ještě větší srozumitelnost rozšířena i o další UML diagramy, ale tento návrh by měl být již nyní plně dostačující k následné implementaci.
Seznam použité literatury

Odborná kniha


Elektronické dokumenty


<table>
<thead>
<tr>
<th>Zkratka</th>
<th>Ověřený název</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASE</td>
<td>Computet Aided Software Engineering</td>
</tr>
<tr>
<td>CIM</td>
<td>Computer-independent Model</td>
</tr>
<tr>
<td>MDA</td>
<td>Model Driven Architecture</td>
</tr>
<tr>
<td>MS</td>
<td>Microsoft</td>
</tr>
<tr>
<td>OMG</td>
<td>Object Management Group</td>
</tr>
<tr>
<td>OMT</td>
<td>Object Modeling Technique</td>
</tr>
<tr>
<td>PIM</td>
<td>Platform Independent Model</td>
</tr>
<tr>
<td>PSM</td>
<td>Platform Specific Model</td>
</tr>
<tr>
<td>RFP</td>
<td>Request For Proposal</td>
</tr>
<tr>
<td>SW</td>
<td>Software</td>
</tr>
<tr>
<td>UC</td>
<td>Use Case</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modeling Language</td>
</tr>
<tr>
<td>UP</td>
<td>Unified Process</td>
</tr>
</tbody>
</table>
Seznam obrázků

Obr. 2.1 Historie UML ................................................................. 6
Obr. 2.2 Transformace v architektuře MDA ........................................ 9
Obr. 2.3 Struktura jazyka UML .....................................................12
Obr. 2.4 Třída ............................................................................. 13
Obr. 2.5 Rozhraní ........................................................................ 13
Obr. 2.6 Spolupráce .................................................................... 14
Obr. 2.7 Případ užití ................................................................. 14
Obr. 2.8 Komponenta ................................................................. 14
Obr. 2.9 Uzel .............................................................................. 14
Obr. 2.10 Zpráva ........................................................................ 15
Obr. 2.11 Stav ............................................................................ 15
Obr. 2.12 Baliček ........................................................................ 16
Obr. 2.13 Poznámka .................................................................... 16
Obr. 2.14 Závislost ..................................................................... 16
Obr. 2.15 Asociace ..................................................................... 17
Obr. 2.16 Agregace ..................................................................... 17
Obr. 2.17 Kompozice ................................................................. 17
Obr. 2.18 Generalizace ............................................................... 17
Obr. 2.19 Realizace ................................................................. 18
Obr. 2.20 Diagramy UML ............................................................. 19
Obr. 2.21 Use Case Diagram ......................................................... 24
Obr. 2.22 Class Diagram ............................................................... 26
Obr. 2.23 State Machine Diagram .................................................. 27
Obr. 2.24 Activity Diagram ............................................................ 28
Obr. 2.25 Sequence Diagram .......................................................... 29
Obr. 3.1 Organizační struktura organizace ORLOVAN, bytové družstvo ........................................ 31
Obr. 4.1 Diagram případů užití akterů ........................................... 36
Obr. 4.2 Diagram případů užití uživatele ....................................... 37
Obr. 4.3 Diagram případů užití správce systému ......................... 40
Obr. 4.4 Analytický diagram tříd systému .................................... 43
Obr. 4.5 Stavový diagram úkolu .................................................... 44
Obr. 4.6 Stavový diagram události ............................................... 45
Obr. 4.7 Diagram aktivit úkolu ...................................................... 46
Obr. 4.8 Sekvenční diagram úkolu ................................................. 47
Obr. 4.9 Dashboard uživatele systému .......................................... 48
Obr. 4.10 Dashboard správce systému ............................................ 49
Obr. 4.11 Tvorba úkolu ............................................................... 49
Obr. 4.12 Evidence úkolů ............................................................ 50
Obr. 4.13 Tvorba událostí ........................................................... 50
Obr. 4.14 Kalendář ..................................................................... 51
Seznam tabulek

Tab. 3.1 Specifikace problémů ........................................................................................................ 34
Tab. 4.1 Specifikace diagramu případů užití ............................................................................... 37
Tab. 4.2 Specifikace tvorby úkolu ................................................................................................. 38
Tab. 4.3 Specifikace tvorby události ............................................................................................. 38
Tab. 4.4 Specifikace editování úkolu/události .............................................................................. 38
Tab. 4.5 Specifikace tvorby šablony ............................................................................................ 39
Tab. 4.6 Specifikace exportu úkolu ............................................................................................... 39
Tab. 4.7 Specifikace editace profilu ............................................................................................... 39
Tab. 4.9 Specifikace dat pozic ...................................................................................................... 40
Tab. 4.10 Specifikace dat úseků .................................................................................................... 41
Tab. 4.11 Specifikace dat oprávnění ............................................................................................. 41
Tab. 4.12 Specifikace dat uživatele ............................................................................................. 41
Tab. 4.13 Specifikace dat družstva ............................................................................................... 42
Tab. 4.14 Matice trasovatelnosti požadavků ................................................................................ 42
Prohlášení o využití výsledků diplomové práce

Prohlašuji, že

- jsem byl seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. – autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obrázků, v rámci školních představení a užití díla školního a § 60 – školní dílo;

- beru na vědomí, že Vysoká škola běžská – Technická univerzita Ostrava (dále jen VŠB-TUO) má právo nevydělečně, ke své vnitřní potřebě, diplomovou práci užit (§ 35 odst. 3);

- souhlasím s tím, že diplomová práce bude v elektronické podobě archivována v Ústřední knihovně VŠB-TUO a jeden výtisk bude uložen u vedoucího diplomové práce. Souhlasím s tím, že bibliografické údaje o diplomové práci budou zveřejněny v informačním systému VŠB-TUO;

- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona;

- bylo sjednáno, že užití svého dílo, diplomovou práci, nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat příměněný příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).

V Ostravě dne 26. 4. 2013

[signature]

jméno a příjmení studenta