Pedagogická fakulta
Faculty of Education

Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

Jihočeská univerzita v Českých Budějovicích
Pedagogická fakulta
Katedra informatiky

Bakalářská práce

# Online systém pro tvorbu pojmových map

Vypracoval: Lukáš Kudrna
Vedoucí práce: PhDr. Milan Novák Ph. D.

České Budějovice 2014

Pedagogická fakulta
Faculty of Education
Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

University of South Bohemia
Faculty of Education
Department of Informatics

Bachelor's thesis

# Online system for creating mind maps

Author: Lukáš Kudrna
Supervisor: PhDr. Milan Novák Ph. D.

Budweis 2014

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH
**Fakulta pedagogická**
Akademický rok: **2011/2012**

# ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

| | |
|---|---|
| Jméno a příjmení: | **Lukáš KUDRNA** |
| Osobní číslo: | **P10347** |
| Studijní program: | **B7507 Specializace v pedagogice** |
| Studijní obor: | **Informační technologie ve vzdělávání** |
| Název tématu: | **Online systém pro tvorbu pojmových map** |
| Zadávající katedra: | **Katedra informatiky** |

Zásady pro vypracování:

Student provede analýzu dostupných online systémů pro návrh pojmových map. Tato analýza bude obsahovat výpis společných funkcionalit, které budou uvedeny v přehledové tabulce. Na základě této analýzy bude proveden procesní návrh nového systému pro online navrhování pojmových map, který bude obsahovat vybrané nástroje.
Druhou částí práce bude praktická realizace systému, která bude vycházet z uvedené teoretické analýzy a návrhu. Dále vybere vhodnou technologii pro realizaci (HTML5, PHP, Adobe Air apod.)

Rozsah grafických prací: **CD ROM**

Rozsah pracovní zprávy: **50**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:


1. **LEUCHNER, Marc, Todd ANDERSON a Matt WRIGHT. Adobe AIR: create-modify-reuse. Indianapolis, IN: Wrox/Wiley Pub., c2008, 457 s. Create-modify-reuse guides. ISBN 0470182075.**
2. **BUZAN, Tony, Barry BUZAN a Matt WRIGHT. The mind map book: how to use radiant thinking to maximize your brain's untapped potential. 1st Plume print. New York: Plume, 1996, 320 s. Plume book. ISBN 04-522-7322-6.**
3. **DOSTÁL, Jiří, Barry BUZAN a Matt WRIGHT. Pojmové mapy kurzů distančního vzdělávání: how to use radiant thinking to maximize your brain's untapped potential. 1st Plume print. Olomouc: Univerzita Palackého, 2009, 320 s. Plume book. ISBN 978-80-244-2349-4.**


Vedoucí bakalářské práce: **Mgr. Milan Novák, Ph.D.**
Ústav aplikované informatiky

Datum zadání bakalářské práce: **12. dubna 2012**

Termín odevzdání bakalářské práce: **26. dubna 2013**

Mgr. Michal Vančura, Ph.D.
děkan

L.S.

doc. PaedDr. Jiří Vaníček, Ph.D.
vedoucí katedry

V Českých Budějovicích dne 12. dubna 2012

## Prohlášení

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně, pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě pedagogickou fakultou elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách.

V Českých Budějovicích dne

## Abstrakt

Bakalářská práce se zabývá online systémy pro návrh a tvorbu pojmových a myšlenkových map. Mapy jsou jednou z metod zápisu informací. Jejich úkolem je informace organizovat, popisovat a třídit. Práce je rozdělena do dvou částí. V teoretické části je provedena analýza již dostupných online systémů pro tvorbu pojmových map. Společné funkcionality těchto systémů jsou uvedeny v přehledové tabulce spolu se stručnými popisy. Tato část také obsahuje výběr nejvhodnějších technologií pro realizaci systému. Praktickou částí je již samotná realizace systému, která vychází z vytvořené teoretické analýzy a procesního návrhu. Závěrečným krokem je uvedení tohoto systému do provozu.

## Abstract

The bachelor thesis deals with online systems for the design and creation of conceptual maps and mind maps. Maps are a method of recording information, helping to organize this information, describe and classify. The task is divided into two parts. The theoretical part is an analysis of already available online systems for creating mind maps. Common functionality of these systems are listed in the summary table along with brief descriptions. This section also contains a selection of the most appropriate technology for system implementation. The practical part is the actual system implementation based on a theoretical analysis and process design. The final step is putting the system into operation.

## Poděkování

Tímto bych velice rád poděkoval vedoucímu mé bakalářské práce panu PhDr. Milanu Novákovi, Ph.D. za jeho odbornou pomoc a cenné rady, které mi předal při zpracovávání této práce.

Dále bych rád poděkoval všem, kteří mi pomáhali či pomáhají při testování systému.

V neposlední řádě patří mé veliké poděkování celé mé rodině, která mě podporovala po celou dobu mého vysokoškolského studia a to nejen finančně. Také nesmím zapomenout poděkovat své přítelkyni za nemalou psychickou podporu.

# Obsah

# 1 Úvod

Pojmové a myšlenkové mapy jsou velice užitečným nástrojem hlavně pro nabývání a osvojování nových vědomostí či rozšiřování již dříve nabytých vědomostí. Ovšem zápis informací do pojmových či myšlenkových map není žádnou novinkou. Nápad vytvořit efektivní metodu zápisu informací nazvanou jako „myšlenkové mapy“ dostal jejich autor Tony Buzan v 60. letech 20. století při studiu na vysoké škole. Zatímco nápad na pojmové mapy dostal autor Joseph D. Novak spolu se svým kolegou D. B. Gowinem přibližně o 10 let později při snaze vytvořit takový materiál, který by dětem pomohl ve správném chápání pojmů.

Ve své podstatě mají jak pojmová tak i myšlenková mapa stejný cíl, čímž je napomáhat učení a přidávání setříděných a organizovaných informací do nových či již existujících struktur. Dalo by se říci, že jsou tyto metody velice podobné, ovšem nikoliv stejné. Každá z těchto metod dochází k jejich stanovenému cíli trochu odlišným způsobem. Myšlenková mapa se snaží využít celý potenciál lidského mozku. Jelikož každá část mozku plní trochu jiný úkol, myšlenková mapa se snaží zapojit co nejvíce částí najednou pomocí propojení psaného slova, grafického znázornění a barevného odlišení. Oproti tomu pojmová mapa se snaží apelovat na správný výklad dané problematiky pomocí vztahů mezi jednotlivými pojmy. Tyto zmíněné vztahy mezi pojmy jsou hlavním rozdílem oproti myšlenkovým mapám.

V teoretické části této práce se čtenář podrobněji seznámí s pojmy, které jsem výše popisoval (konkrétně s pojmovými a myšlenkovými mapami). Dále zde budou popsány a předvedeny různé metody zápisu informací.

V praktické části se práce zabývá analýzou současného stavu online systémů pro tvorbu a správu pojmových či myšlenkových map. Následně zde budou uvedeny výsledky, které ze zmíněné analýzy vzešly. Popsány budou také postupy při výběrech vhodných technologií pro návrh nového systému pro tvorbu pojmových a myšlenkových map. Také bude popsán postupný návrh nového systému.

V závěrečné části budou popsány dílčí části systému, použité programovací jazyky či frameworky a postup práce při případné implementaci tohoto systému do jiných systémů.

# 2 Cíle práce

Hlavním cílem této bakalářské práce je vytvoření funkčního systému pro správu pojmových a myšlenkových map, který bude možné implementovat do jiných webových aplikací, a zároveň bude plně podporovat dotyková zařízení. Systém bude následně spuštěn a otestován na veřejném webovém serveru.

Pro úspěšné splnění hlavního cíle, byly stanoveny tyto dílčí cíle:

- Návrh systému vyplyne z teoretické analýzy, v rámci které bude provedena obsahová analýza v současné době nejznámějších online systémů pro tvorbu pojmových a myšlenkových map. Při zpracovávání této obsahové analýzy bude kladen důraz na využívané technologie pro správu map, dále na nabízené verze a nástroje těchto systémů.
- Ze získaných údajů bude provedeno šetření, na základě kterého budou vybrány nejvhodnější technologie, které současná doba nabízí pro vývoj systému stejného či obdobného typu.
- Dalším cílem je samotný návrh struktury systému, který bude zahrnovat diagram případu užití a relační model databáze. Tyto segmenty jednoduše vyjádří plánovanou funkčnost navrhovaného systému.
- Realizovaný systém bude implementován na veřejný server a následně otestován veřejností.
- Do písemné části této práce bude dále sepsán podrobný návod pro případnou implementaci systému do jiných webových aplikací.

# 3 Metodika

Postup práce při navrhování nového systému nejlépe vystihuje vývojový model ADDIE, který umožňuje ucelený pohled na proces návrhu. Zkratka tohoto modulu je tvořena prvními písmeny z anglických slov analysis, design, development, implementation a evaluation, což v překladu znamená analýza, návrh, vývoj, implementace, hodnocení. Významy těchto slov jsou zároveň cyklickými kroky, které budou použity při vypracovávání nového systému.

Analýza se v tomto projektu zabývá současnými online systémy pro tvorbu pojmových map. Analýza těchto systémů, která bude prováděna formou šetření, nám odhalí, jaké technologie tyto systémy využívají pro práci s objekty v mapách, jaké verze tyto systémy umožňují uživatelům využít a také jaké nástroje či funkcionality nabízí v jednotlivých verzích systémů. Výsledky, které vzejdou z tohoto šetření, budou převedeny do přehledové tabulky, která bude součástí této bakalářské práce.

Návrh systému bude vycházet z výsledků provedené analýzy. Z přehledové tabulky budou vybrány nejefektivnější nástroje a funkcionality pro funkčnost nového systému. Vybrané nástroje a funkcionality budou do návrhu systému uvedeny formou diagramu případu užití. Návrh systému dále rozšíří relační model, který zobrazí strukturu databáze. Součástí návrhu bude také wireframe, který bude navržen pro systém umožňující práci s mapami i pomocí dotykových zařízení.

Vývoj nového systému bude samozřejmě vycházet z vytvořeného návrhu systému, a tedy i z vytvořené analýzy. Pro samotný vývoj budou zvoleny technologie, nejlépe splňující stanovené požadavky, které postupně vzešly při vytváření analýzy a návrhu nového systému. Při vývoji systému budou nově naprogramované funkce a nástroje postupně testovány, zda splňují předpokládanou funkčnost. Případné chyby budou opraveny.

Realizovaný systém bude následně implementován na veřejný webový server splňující minimální požadavky, které vzešly z vývoje tohoto systému.

V závěrečné části, kterou je hodnocení, bude systém postupně testován ze strany uživatelů. Případné funkční nedostatky budou cyklicky přecházet zpět na začátek vývojového modelu ADDIE.

## 3.1 Východiska práce

Práce vychází z potřeby analyzování současného stavu online systémů pro tvorbu pojmových a myšlenkových map, na jejímž základě se vyhotoví návrh systému, který by bylo možné využít na všech typech zařízení s možností implementovat jej do jiných systémů.

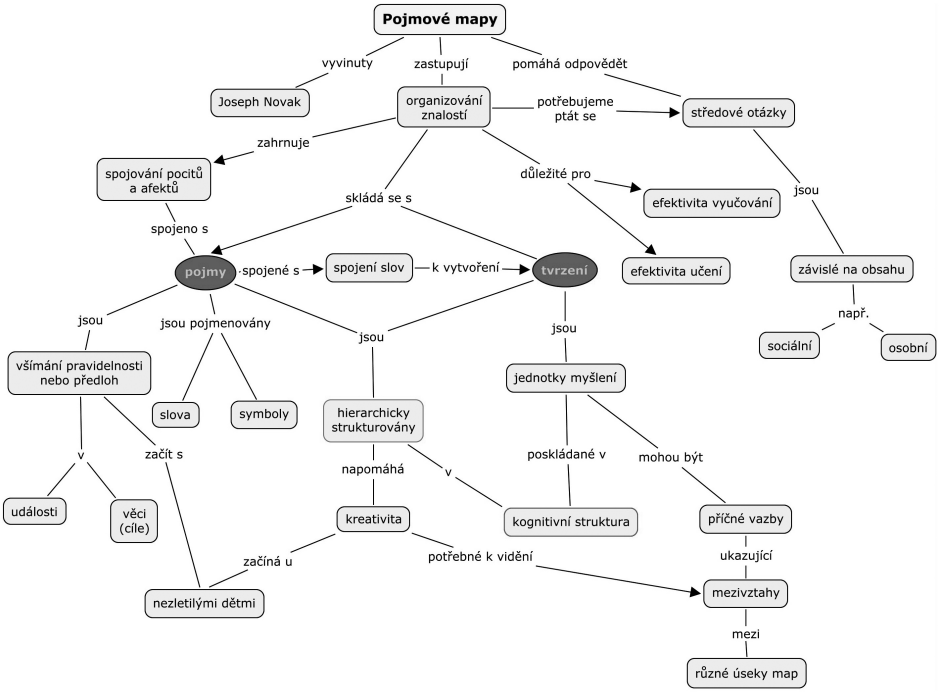
# 4 Úvod do problematiky

Pojmové a myšlenkové mapy jsou jednou z metod zápisu informací. Jejich úkolem je zapisované informace systematizovat a třídit. Zároveň však tato metoda napomáhá učení a přidávání nových informací do existujících mozkových struktur.

## 4.1 Pojmová mapa

### 4.1.1 Co je pojmová mapa?

Podle J. D. Novaka lze pojmovou mapu definovat jako hierarchickou strukturu pojmů a vztahů mezi nimi. Jeden pojem může být spojen vztahem s jedním nebo více pojmy, obdobně tak i více vztahů může vést k jednomu pojmu. Vztah propojuje pojmy v různých hierarchiích. Tento typ vztahů se nazývá vertikální vztah. V pojmové mapě je možný i výskyt pojmů ve stejné hierarchii a jejich propojení horizontálními vztahy.[1, s. 14]




Obrázek 1: Ukázka pojmové mapy [1, s. 15]

Vztahy, které vedou napříč strukturou, se nazývají křížné vztahy. Dokumentují souvislosti jedné části mapy s jinou a jejich vzájemnou provázanost. V pojmové mapě se vyskytují občasně pouze pro demonstraci vztahu jedné částí pojmové mapy k druhé.[1, s. 14]

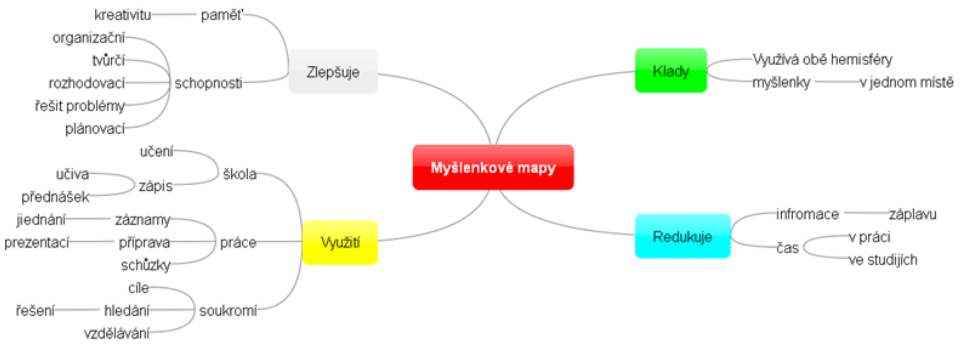
### 4.1.2 K čemu se pojmová mapa využívá?

Pojmovou mapu lze využít v mnoha oborech či situacích, proto také existuje celá řada variant této metody zápisu informací, a to již od zmíněných hierarchických, systémových až po cyklické. V závislosti na oboru zpracovávané tématiky lze pojmové mapování využít pokaždé trochu jinak. Obecně můžeme říci, že se pojmové mapy využívají pro správné chápání problematiky, vyvozování konkrétních závěrů, přípravu postupu, podporu při rozhodovacím procesu apod.

## 4.2 Myšlenková mapa

### 4.2.1 Co je myšlenková mapa?

Myšlenkové mapy jsou vizuálně velice podobné jako pojmové mapy. Ovšem oproti nim je zde pár principiálních rozdílů. Myšlenkové mapy jsou striktně rozváděny do stromové struktury a mezi jednotlivé pojmy se neuvádí jejich vztahy. To jsou hlavní rozdíly, které na první pohled oddělí myšlenkovou mapu od mapy pojmové.

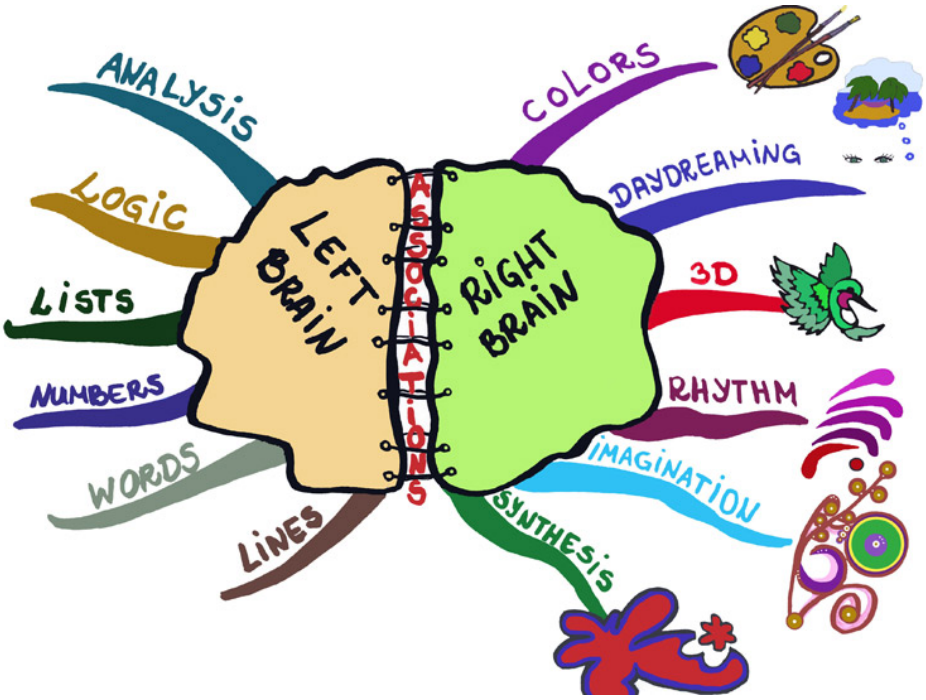



Obrázek 2: Ukázka myšlenkové mapy

Myšlenková mapa (anglicky „mind map“) pracuje s jedním centrálním termínem, který dále rozvíjí a hledá nejrůznější významové a obsahové konotace tohoto slova. Je to tedy jakýsi hrozen informací pojících se k tomuto centrálnímu termínu.[2]

### 4.2.2 K čemu se myšlenková mapa využívá?

Jak již sám název trochu napovídá, myšlenková mapa se používá pro rozvoj či ucelení myšlenek v mozku. K tomuto účelu využívá velmi jednoduchou, avšak vysoce efektivní metodu. Lidský mozek se z fyziologického hlediska dělí do dvou hemisfér. Každá z nich si je schopna zapamatovat jen určité informace.




Obrázek 3: Základní činnosti mozkových hemisfér [3]

Na obrázku je vidět, že zatímco levá hemisféra si například pojem „strom“ zapamatuje formou psaného slova, pravá hemisféra si tento pojem zapamatuje formou vzpomínky či vizuální představy. Myšlenkové mapy tedy zcela záměrně zahrnují jak barvy, obrázky, symboly, tak i psané slovo. V tom případě se do mozku ukládají všechny vnímané aspekty a mezi ukládanými informacemi se vytvářejí asociace, které při vybavení vizuálního aspektu odkážou na příslušnou textovou informaci. Tímto způsobem lze myšlenkové mapy využít pro efektivní učení probírané problematiky.

## 4.3 Metody zápisu poznámek

V této sekci jsou popsány různé metody zápisu informací s jejich krátkými ukázkami.

### 4.3.1 Lineární text, odrážky či číslování

Jedná se o klasický způsob zapisování poznámek. Znázorňuje hierarchii jednotlivých informací. Jeho součástí mohou být jak odrážky, tak číslování. Ústní projev je také koncipován do lineární struktury využívající například odrážek.[4, s. 100]

**Příklad poznámek**

**Počítačové periferie**

- Vstupní
  - Klávesnice
  - Počítačová myš
  - Trackball
  - Joystick
  - Gamepad
  - Scanner
  - Webová kamera
- Výstupní
  - Monitor
  - Tiskárna
  - Reproduktor
  - Plotter
  - Dataprojektor

**Použití**
Tento typ poznámek znázorňuje díky využití nadpisů, podnadpisů a odrážek vztahy mezi jednotlivými informacemi. Tyto zápisky lze snadno zpracovat do textové podoby, díky čemuž je tento způsob zápisu poznámek velmi oblíbený. Vyhovuje zejména těm, kteří mají logické myšlení a nedělá jim problém informace hierarchizovat. Tento formát je proto vhodný pro většinu lidí. Umožňuje informaci rychle, v případě potřeby i doslovně, zaznamenat, ovšem často vyžaduje značnou redukci informací. Pro správné použití této metody je dobré ovládat následující:

- umět dobře naslouchat,
- perfektní schopnost shrnout informace.[4, s. 100]

### 4.3.2 Mřížky či tabulky

Tabulky jsou užitečným nástrojem pro kategorizování informací.

| Datum | Nejvyšší teplota | Nejnižší teplota |
|---|---|---|
| 2. června | 31.6 °$C$ (rok 1901) | 4.6 °$C$ (rok 1928) |
| 3. června | 31.9 °$C$ (rok 1947) | 4.4 °$C$ (rok 1810) |
| 4. června | 32.6 °$C$ (rok 1947) | 3.8 °$C$ (rok 1810) |

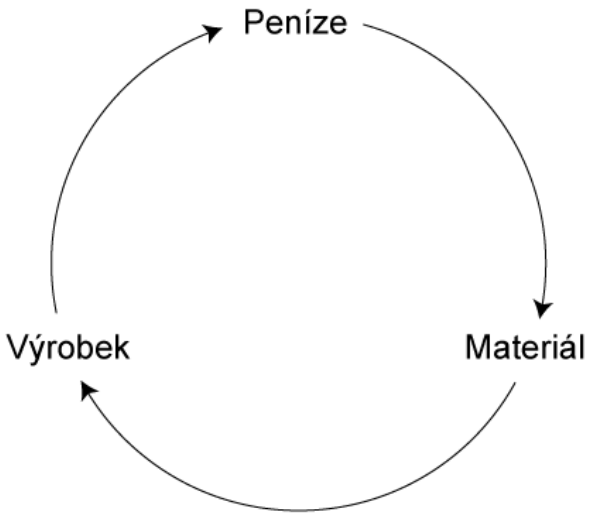
Tabulka 1: Ukázka tabulky

Jak si můžeme všimnout na uvedeném příkladu, tabulka umožňuje systematické zaznamenávání informací a také snadné srovnání.

**Použití**
Tabulky jsou velice vhodným způsobem pro přehledný zápis velkého množství kategorizovaných informací. Pomocí speciálního softwaru, jakou jsou tabulkové procesory, lze zaváděné informace v tabulkách vhodně seřazovat, třídit či formátovat a tak vytvořit systematický zápis informací.

### 4.3.3 Diagramy

Tyto způsoby zápisu informací jsou závislé na druhu informací, se kterými pracujete. Často se používají v přírodních vědách a můžete se s nimi setkat v učebnicích anatomie, fyziky, chemie, psychologie, pedagogiky apod.[4, s. 105]




Obrázek 4: Ukázka cyklického diagramu

**Použití**
Pokud je potřeba shrnout nějaký proces nebo popsat určitý postup, pak jsou diagramy tím nejlepším způsobem znázornění informací. Díky použití šipek lze snadněji pochopit pořadí jednotlivých částí procesu.[4, s. 105]

### 4.3.4 Mapy

Tyto metody zapisování poznámek jsou často velmi náročné. Jsou vhodné pro lidi, kteří se rádi učí vizualizováním informací. Pro svoji přehlednost jsou mapy velmi užitečné pro rychlé osvěžení paměti či opakování informací. Ke klasifikaci a kategorizaci informací můžete použít různé tvary a barvy buněk. Důležitým znakem této metody zápisu je to, že hlavní termín většinou začíná uprostřed mapy. Jak je vidět například na obrázku č. 2, různé roviny informací tvoří nejen samostatné skupiny, ale jsou také rozlišeny tvarem a barvou buňky. Vztahy mezi informacemi je také možné vyznačit pomocí větví, šipek či linek. Tento formát je tedy velmi vhodný pro shrnutí informací a uvedení těchto informací do lépe pochopitelných částí. Výhodou tohoto typu poznámek oproti klasickému formátu je možnost doplnění informací v jakémkoliv stádiu rozpracovanosti.[4, s. 105]

**Použití**
Tento typ poznámek je vhodný především pro lidi s většími zkušenostmi zápisu informací. Pro rychlé zpracování například přednášených informací vyžadují mapy nejen odbornou znalost, ale i velmi hluboké porozumění tématu.[4, s. 106]

# 5 Analýza

V rámci této bakalářské práce byla provedena analýza nejznámějších online systémů pro tvorbu pojmových a myšlenkových map. Analyzováno bylo celkem 21 konkurenčních řešení. Výsledky z této analýzy jsou popsány v následujících podkapitolách.

## 5.1 Analýza konkurenčních řešení

Z dostupných tištěných zdrojů na internetu byly postupně dohledány nejznámější systémy pro tvorbu pojmových a myšlenkových map. Tyto nalezené systémy byly podrobeny šetření, ve kterém bylo zjišťováno, jaké technologie dané systémy využívají pro správu map. Dále pak jaké verze systémy nabízejí k využití a také jaké nástroje a funkcionality je v různých verzích systému možné použít. Veškeré informace byly vkládány do přehledové tabulky. Tato tabulka je v plném rozsahu umístěna na přiloženém CD viz příloha č. 1.

| Název | URL | Technologie pro práci s mapami | | | Dostupné verze | | |
|---|---|---|---|---|---|---|---|
| | | Javascript | Adobe Flex / Flash | Silverlight | Volná verze | Registrace | Placená verze |
| MindMeister Mind Map | http://www.mindmeister.com/ | ■ | - | - | ■ | ■ | - |
| Bubblus | https://bubbl.us/ | - | ■ | - | ■ | ■ | - |
| Mindomo | http://www.mindomo.com/ | - | ■ | - | - | 3 mapy | ■ |
| Mind42 | http://mind42.com/ | ■ | - | - | - | ■ | - |
| TEXT2MINDMAP | http://www.text2mindmap.com/ | - | ■ | - | ■ | - | - |
| WiseMapping | http://www.wisemapping.com/ | ■ | - | - | ■ | ■ | - |
| Comapping | http://go.comapping.com/ | - | ■ | - | - | ■ | - |
| Spider Scribe | http://www.spiderscribe.net/ | - | ■ | - | - | ■ | ■ |
| Mindjet MindManager | http://www.mindjet.com/ | - | ■ | - | - | ■ | ■ |
| MAPMYself | http://www.mapul.com | - | - | ■ | - | ■ | ■ |
| Spicynodes | http://www.spicynodes.org/ | - | ■ | - | ■ | - | - |
| Glinkr | http://www.glinkr.net | - | - | - | ■ | - | - |
| Popplet | http://www.popplet.com | - | ■ | - | - | ■ | - |
| Slatebox | http://www.slatebox.com | ■ | - | - | - | ■ | - |
| Creately | https://www.creately.com | - | ■ | - | ■ | ■ | - |
| Cacoo | https://www.cacoo.com | - | ■ | - | - | ■ | ■ |
| Gliffy | http://www.gliffy.com | - | ■ | - | ■ | ■ | - |
| Lucidcharts | https://www.lucidchart.com | ■ | - | - | ■ | ■ | ■ |
| Slickplan | http://www.slickplan.com/ | ■ | - | - | - | - | ■ |
| Exploretree | http://www.exploratree.org.uk | - | ■ | - | ■ | ■ | - |
| Kerika | https://www.kerika.com/ | - | ■ | - | - | Google.com | - |

Obrázek 5: Náhled přehledové tabulky konkurenčních řešení

Jak je vidět na obrázku č. 5, který zobrazuje 1. část přehledové tabulky konkurenčních řešení, je v tabulce uveden název systému, adresa na úvodní stránku systému, hlavní technologie, kterou systémy využívají pro práci s mapami a seznam verzí, které dané systémy umožňují využít. Dále je pak v tabulce uvedeno, jestli s daným systémem lze vytvářet pojmové a myšlenkové mapy a zbytek tabulky je již zmíněný výpis nástrojů a funkcionalit. Hodnoty u těchto zmíněných položek jsou uvedeny v následujícím textovém formátu:

- hodnoty označené černým čtverečkem či textem jsou funkcionality, které jsou dostupné již pro volnou verzi systému,
- hodnoty označené červeným čtverečkem či textem jsou funkcionality, které jsou dostupné pro uživatele s příslušnou registrací,
- hodnoty označené tyrkysovým čtverečkem či textem jsou funkcionality, které jsou dostupné pro platící uživatele s příslušnou registrací.

Jak je možné si v této tabulce všimnout, technologie pro práci s mapami obsahuje spojené technologie Adobe Flex a Adobe Flash. Jelikož obě tyto technologie vyžadují

kompilaci zdrojových kódů a výsledek této kompilace je možné zobrazit pouze pomocí modulu Adobe Flash Player, není možné se 100% jistotou přesně určit, jakou z těchto technologií jednotlivé systémy využívají. Proto jsou tyto technologie uvedeny v jednom sloupci.

## 5.2 Výsledky analýzy

Z provedené analýzy vyplývá, že většina současných systémů pro tvorbu pojmových a myšlenkových map, jsou po stránce funkčnosti, zároveň i po stránce nabízených nástrojů a funkcionalit, na velice dobré úrovni. Důležité je ovšem zmínit, že ne všechny systémy, které umožňují vytvářet myšlenkové mapy, podporují i tvorbu pojmových map. Další důležitou poznámkou je, že pouze několik systémů dovoluje úpravu map ve webovém prohlížeči i pomocí dotykových zařízení, jako jsou chytré telefony či tablety. Tuto skutečnost lze ovšem vysvětlit tím, že autoři těchto systémů investovali čas i zdroje spíše do vývoje mobilních aplikací. Tyto aplikace jsou z hlediska náročnosti na hardware jistě úspornější než Adobe Flash / Flex, který využívá drtivá většina systémů. Zároveň je pomocí těchto aplikací možné vytvářet mapy, i pokud není uživatel zrovna připojen do internetu, což je jistě nemalé plus pro uživatele.

Při pohledu na přehledovou tabulku je jasně patrné, že valná většina systémů využívá technologie Adobe Flex / Flash. Jelikož tyto technologie od sebe nelze s jistotou rozlišit, mohu se jen domnívat, že nejpoužívanější technologií je technologie Adobe Flex. Oproti Adobe Flash není potřeba žádného drahého vývojového prostředí pro vytváření aplikací, stačí i jednoduchý textový editor.

Z vytvořené přehledové tabulky byly vybrány nejvhodnější technologie, kterými disponují konkurenční systémy. Tyto funkcionality byly zahrnuty do návrhu nového systému pomocí digramu případu užití, viz kapitola 6.3.1.

# 6 Návrh řešení

V této kapitole je popsán rozbor technologií pro správu map, použitých v některém ze systémů, které byly podrobeny teoretické analýze. Z těchto technologií byla následně vybírána nejvhodnější technologie pro nový systém takovéhoto typu. Proces tohoto výběru je popsán v následujících podkapitolách. Z vybraných technologií je proveden rozbor podpory webových prohlížečů. V závěrečné podkapitole je popsán návrh systému formou diagramu případu užití, relačního modelu a wireframu.

## 6.1 Výběr technologií pro realizaci nového systému

I když valná většina systémů dle mého názoru využívá technologie Adobe Flex, bylo nutné zjistit, zda je tato technologie v současnou dobu opravdu tou nejvhodnější pro systém takovéhoto typu a zda by splňovala požadavky, které byly pro nový systém stanoveny.

### 6.1.1 Výběr technologie pro práci s mapami

**Technologie, které využívají konkurenční systémy**
V této kapitole jsou popsány technologie, které využil minimálně jeden z testovaných konkurenčních systémů.

**Adobe Flash**
Adobe Flash je technologie založená na vektorové grafice s možností využití časové osy. Tato technologie využívá programovacího jazyku Action Script (AS). Jedná se o jednoduchý, objektově orientovaný programovací jazyk, syntaxí podobný javascriptu. Pomocí tohoto jazyku je možné manipulovat s objekty na scéně, ovládat prezentaci jako celek, vytvářet formuláře, atp. Ke svému chodu využívá technologie Flash softwaru Adobe Flash Player, což je software vytvořený společností Adobe. Tento software umožňuje prohlížení interaktivního obsahu a aplikací na webu.[5]

**ZÁPORY**

- Nutnost nainstalovaného softwaru Adobe Flash Player,
- vysoké nároky na hardware uživatele.

**Adobe Flex**
Flex je vysoce produktivní, open source aplikační framework pro vytváření a udržování působivých webových aplikací.[6] Tato technologie využívá značkovacího jazyku MXML, který je syntaxí podobný HTML, ovšem s daleko větší sémantikou. Adobe Flex již v základu obsahuje spoustu hotových komponent pro aplikační vývoj. Ke svému chodu využívá technologie Flex také softwaru Adobe Flash Player. V některých případech je zapotřebí i software Adobe AIR.

**ZÁPORY**

- Nutnost nainstalovaného softwaru
  - Adobe Flash Player,
  - Adobe AIR.

**Microsoft Silverlight**

Microsoft Silverlight je výkonná technologie pro poskytování médií a vytváření internetových aplikací na webu. Jelikož je tato technologie založená na Microsoft .NET Frameworku, je možné pro vývoj využít několik programovacích jazyků jako jsou například C#, Visual Basic .NET a Delphi. Ke svému chodu využívá technologie Microsoft Silverlight plug-inu stejného názvu. V současnou dobu je tuto technologii možné využít na platformách Microsoft Windows a Macintosh.[7]

**ZÁPORY**

- Nutnost nainstalovaného plug-inu Silverlight.

**HTML 5 s použitím javascriptu**

Technologie HTML5 je používána jako základ všech webových aplikací. HTML5 je nejnovější specifikace standardu HTML. Obě tyto technologie je možné využít na všech webových prohlížečích bez nutnosti instalace dalších softwarů či plug-inů. Použití javascriptu je možné ve většině prohlížečů dodatečně vypnout. Ovšem většina uživatelů tuto možnost nevyužívá.

**ZÁPORY**

- Nutnost zapnutého javascriptu v prohlížeči na straně uživatele.

**Požadavky na technologii pro práci s mapami**

Ze zadaných cílů této bakalářské práce byly stanoveny tyto požadavky. Technologie pro co možná největší množinu uživatelů (multiplatformnost), podpora dotykových zařízení a implementovatelnost. Jako další požadavek jsem si určil alespoň minimální zkušenost s danou technologií.

**Vyhodnocení technologií pro práci s mapami**

Adobe Flash či Adobe Flex jsou bezesporu velice silné technologie. Přejdeme-li nutnost mít nainstalovaný software Adobe Flash Player, který stejně většina uživatelů desktopových počítačů nainstalovaný má, je zde další velice nepříjemná skutečnost. Software není od 15. srpna 2012 oficiálně dostupný pro uživatele OS Android. Aplikace Flash Player byla tímto datem stažena z Google Play (obchodu aplikací pro operační systém

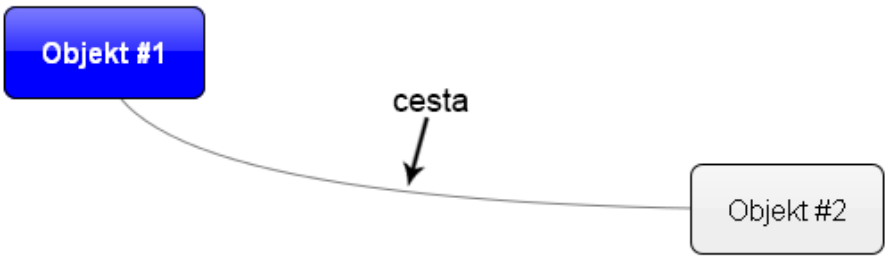
Android) a zároveň se zastavil jeho další vývoj. Jelikož uživatelů využívající OS Android je poměrně velká množina, je tato skutečnost v rozporu se stanovenými cíli, a proto byly tyto technologie uznány za nevyhovující.

Dále zde byla možnost využití technologie Microsoft Silverlight. Pokud se ale podíváme na problémy pojící se v současnou dobu s touto technologií, jako je nejistý další vývoj a nefunkčnost této technologie na mobilních zařízeních, nebyly pro mě tyto problémy natolik malicherné, abych tuto technologii zvolil pro vývoj nového systému.

Zbývající možnost, využití technologie HTML5 s použitím javascriptu, neobsahovala žádné fatální zápory, které by byly v rozporu se stanovenými cíli. Proto jsem tuto možnost zvolil jako technologii pro práci s mapami. Využití javascriptu jsem se rozhodl zjednodušit použitím z mé strany oblíbeného javascriptového frameworku jQuery.

### 6.1.2 Výběr technologie pro vykreslování cest

S výběrem technologie HTML5 a použitím javascriptu se objevila další otázka, a to jakou technologii zvolit pro vykreslování cest, tedy čar spojujících objekty v mapě.




Obrázek 6: Ukázka cesty

**HTML prvky, které je možné využít pro vykreslování cest**
V této kapitole jsou popsány prvky, které je možné použít ve značkovacím jazyce HTML5 pro vykreslení čar, a tedy i cest.

**Prvek <canvas>**
Canvas je HTML prvek, který může být použitý pro kreslení bitmapové grafiky na plátno pomocí skriptovacího jazyku (obvykle javascriptu). Kreslící plátno může být využito pro vykresování grafů, herní grafiky a ostatních vizuálních prvků.[8][9]

**Prvek <svg>**
Prvek <svg> využívá SVG značkovací jazyk pro tvorbu vektorové grafiky. V jazyce SVG jsou zápisem elementů vkládány jednotlivé grafické prvky do plátna. Atributy jednotlivých elementů pak určují danému grafickému prvku vlastnosti, jako jsou například umístění a barva.[10]

**Požadavky na prvek pro vykreslování cest**
Jelikož v době dokončování analýzy nebyl standard HTML5 v prohlížečích plně podporován, a některé funkce tohoto standardu se teprve testovaly, bylo důležité zjistit, jaké z výše uvedených prvků budou již natolik funkční, aby je bylo možné bez obav použít. Dále bylo dobré tyto funkce otestovat ohledně úpravy již nakreslených objektů.

**Vyhodnocení prvků pro vykreslování cest**
Z postupné analýzy a testování byly zjištěny tyto údaje.

| Název prvku | Správná funkčnost | Typ grafiky | DOM |
|---|---|---|---|
| ***canvas*** | + | bitmapová | - |
| ***svg*** | + | vektorová | + |

Tabulka 2: Porovnání prvků pro vykreslování cest

Jak je vidět v tabulce č. 2, funkčnost uvedených prvků již byla bezproblémová na všech testovaných prohlížečích. V nově navrhovaném systému jsem chtěl využít lépe vypadající vektorovou grafiku, kterou podporuje prvek svg, ale ani tento aspekt nebyl rozhodující pro konečný výběr. Rozhodujícím faktorem bylo možné využití tzv. „DOM“ (Document Object Model), která mi umožňuje upravit pouze cílené části v kreslícím plátně. Tuto možnost měl pouze prvek svg, který mi touto skutečností umožňuje překreslit pouze cílené cesty a nemusím tedy při jakékoliv změně, jako by to bylo v případě využití značky canvas, překreslit celé plátno. Proto byl zvolen prvek svg jako prvek pro vykreslování cest.

## 6.2 Podpora prohlížečů s vybranými technologiemi

Tak jako u všeho i v tomto systému platí tvrzení, že je pouze tak silný, jak je silný jeho nejslabší článek. V tomto systému a v současnou dobu je to technologie HTML5 a její podpora na straně webových prohlížečů. Novější a jistě i budoucí verze webových prohlížečů budou použité funkce bez problémů ovládat, ovšem u starších a dosud používaných webových prohlížečů, jako je například Internet Explorer 7 a 8, nejsou všechny použité funkce technologie HTML5 podporovány, a tak není možné tyto webové prohlížeče využít pro práci s mapami.

| Internet Explorer | Mozilla Firefox | Google Chrome | Safari | Opera | iOS Safari |
|---|---|---|---|---|---|
| 9+ | 12+ | 7+ | 5.1+ | 9+ | 5.1+ |

Tabulka 3: Podpora desktopových prohlížečů

| Android Browser | Blackberry Browser | Opera Mobile | Firefox for Android | Chrome for Android | IE Mobile |
|---|---|---|---|---|---|
| 3.0+ | 7.0+ | 12.1+ | 26+ | 33+ | 10+ |

Tabulka 4: Podpora mobilních prohlížečů

Webové prohlížeče jsou uvedeny v takových verzích, které začali funkčně podporovat použité funkcionality standardu HTML5.

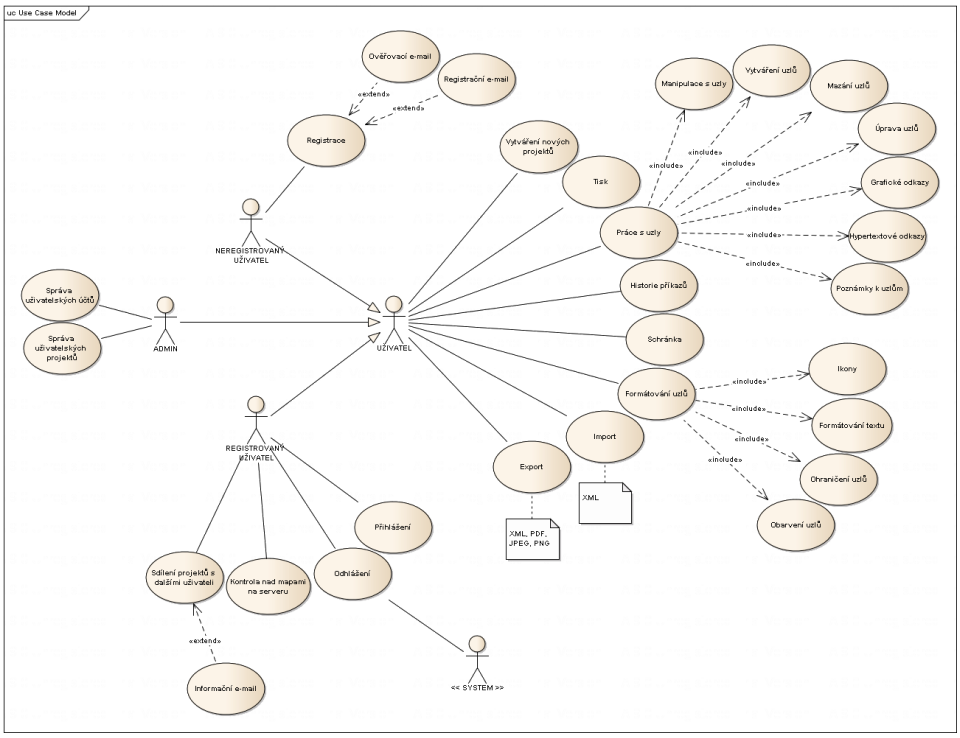
Jak je vidět v tabulce č. 3, s vybranými technologiemi je Internet Explorer podporován až od verze 9. Tato skutečnost je ovšem nepříjemná pro příznivce tohoto webového prohlížeče využívající dosud oblíbený operační systém Windows XP, jelikož zmíněný operační systém podporuje Internet Explorer v nejvyšší verzi 8. Společnost Microsoft tímto tahem zřejmě chtěla přispět k tomu, aby uživatelé operačního systému Windows XP aktualizovali svůj operační systém na novější verzi. Dá se ovšem předpokládat, že uživatelů využívající Internet Explorer 8 bude postupně ubývat. Datem 8. dubna 2014 společnost Microsoft zrušila technickou podporu pro již zmíněný operační systém Windows XP. Dalším aspektem mého předpokladu je plánované konečné schválení specifikace HTML 5.0 v posledním čtvrtletí tohoto roku a tím pádem nárůst webových aplikací využívající stejné či podobné technologie, jako v tomto systému. Uživatelům Internet Exploreru tak nakonec nezbude jiná možnost, než využít některého z neustále vyvíjených webových prohlížečů jako jsou například Google Chrome, Mozilla Firefox, Opera nebo jiných, které Windows XP stále podporují nebo aktualizovat svůj operační systém na novější.

## 6.3 Objektový návrh systému

Z výsledků zpracované teoretické analýzy dostupných online systémů pro tvorbu pojmových a myšlenkových map byl vytvořen návrh systému pomocí diagramu případu užití, relačního modelu a wireframu.

### 6.3.1 Diagram případu užití

Diagram případů užití (Use Case Diagram) se používá k popisu chování systému z hlediska uživatele a zachycuje, které typy uživatelů se systémem pracují a jaké činnosti v rámci systému vykonávají.[11] Umožňuje znázornit funkční požadavky na systém tím, že popisuje interakci mezi ním a uživateli.[12]

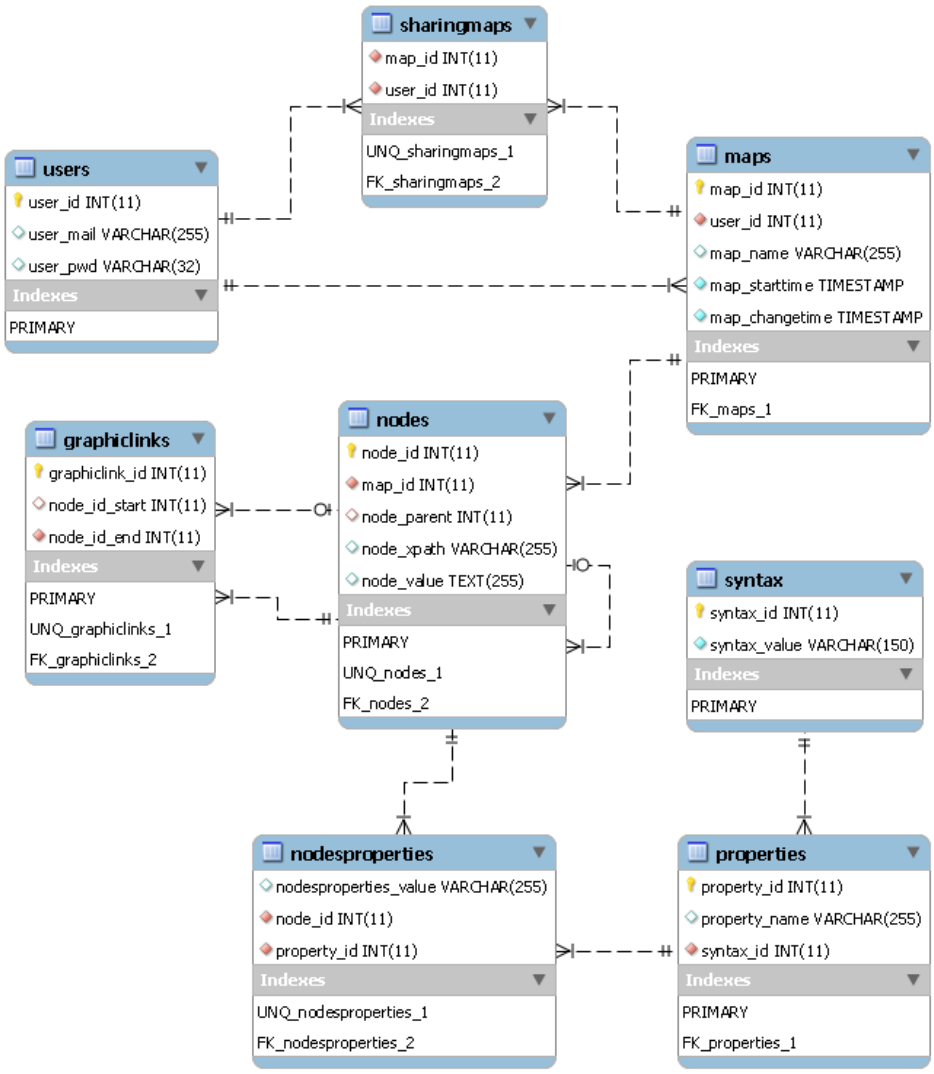



Obrázek 7: Diagram případu užití

Na tomto obrázku je vidět diagram případu užití, který zobrazuje jednotlivé aktéry (uživatele) a jejich možnosti práce v systému. Hlavním aktérem je tedy „uživatel“, který může vytvářet a editovat mapy pomocí uvedených funkcionalit. Pro neregistrovaného uživatele je zde dále možnost registrace, která mimo jiné uživateli umožní přístup a kontrolu nad svými mapami uložených na serveru, a dále možnost zabezpečeného sdílení projektu s jinými registrovanými uživateli.

### 6.3.2 Relační model

Relační datový model popisuje databázi, její strukturu a vlastnosti i manipulaci s daty pomocí matematických pojmů, především z teorie množin a matematické logiky. Zavádí přesné definice ke všem pojmům, dosud popisovaným jen slovně. Definuje vlastnosti správně navržené databáze, popisuje chyby ve strukturách tabulek databáze.[13]



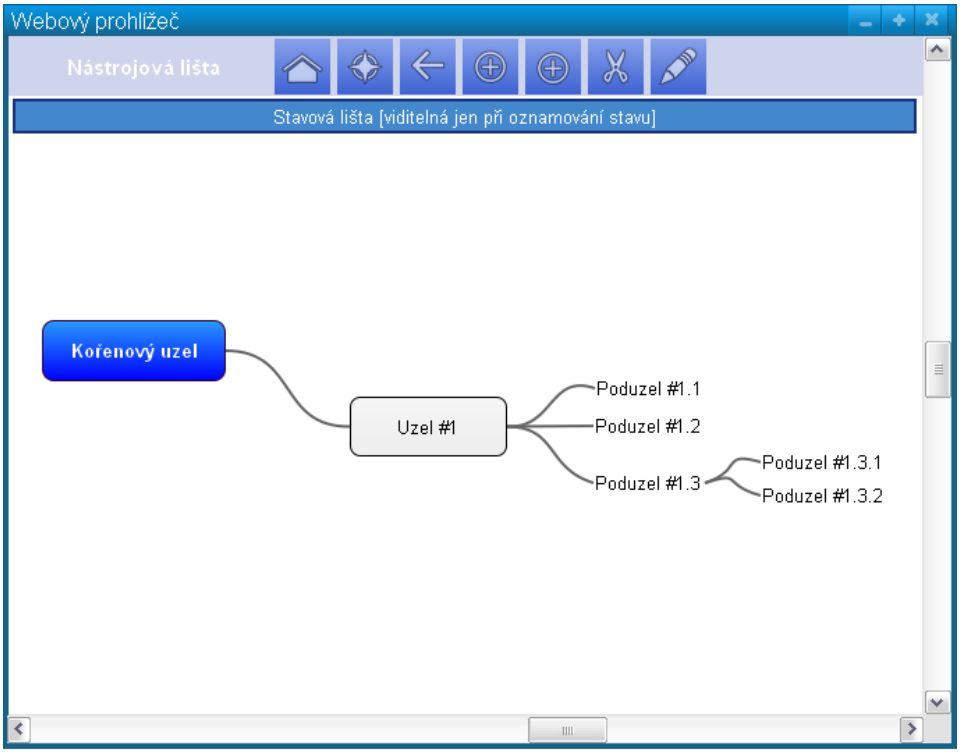

Obrázek 8: Relační model

Na tomto obrázku je vidět relační model, který zobrazuje návrh databáze. Jednou ze základních tabulek je tabulka „user“, která obsahuje registrované uživatele. Tato tabulka může být nahrazena jinou tabulkou obsahující registrované uživatele a jejich ID. K jednotlivým uživatelům se váže tabulka „maps“, která obsahuje základní údaje o myšlenkové mapě, jako jsou název, datum vytvoření a poslední změny. K jednotlivým mapám se váže tabulka „nodes“, obsahující základní údaje pro uzly, jako jsou hodnota, kterou daný uzel právě nabývá a absolutní cestu uzlu v rámci mapy, ve které je obsažen. Na tyto dvě tabulky se váže tabulka „sharingmaps“, která reprezentuje případné sdílení map s uživateli, kteří nejsou vlastníci daných map. Na jednotlivé uzly se vážou 2 tabulky. Tabulka „graphiclinks“ obsahující odkazy na uzly pro vytvoření grafického odkazu mezi uzly z jiné části stromové struktury mapy. Druhá tabulka je „nodesproperties“ obsahující odkaz na samotný uzel, odkaz na tabulku „properties“, která obsahuje vlastnosti (např. barvu pozadí, barvu ohraničení) a samotnou hodnotu vlastnosti. Na

tabulku „properties“ se váže již jen tabulka „syntax“, která jednotlivým vlastnostem z tabulky „properties“ udává jaká má být syntaxe pro hodnotu dané vlastnosti formou regulárního výrazu.

### 6.3.3 Wireframe

Wireframe je síťový model webových stránek, který zobrazuje základní rozložení jednotlivých textových a grafických částí na stránce.[14]




Obrázek 9: Wireframe systému

Obrázek č. 9 zobrazuje základní rozmístění hlavních ovládacích a informačních bloků pracovního prostředí systému. V horní části se bude nacházet nástrojová lišta, která v sobě bude mít vloženy ovládací prvky systému. Tato nástrojová lišta se bude na dotykových zařízeních automaticky přizpůsobovat aktuálnímu přiblížení pracovního prostředí, zároveň se bude v případě potřeby reorganizovat na nejhlavnější ovládací prvky. Zbylé ovládací prvky bude možné zobrazit pomocí tlačítka, které při aktivaci tyto prvky vysune. Pod nástrojovou lištou se bude nacházet stavová lišta, které se bude zobrazovat jen v případě oznamování určitých stavů, jako jsou například neúspěšné uložení, chyby při zpracování požadavků a případně tipy pro využívání systému. Tyto zmíněné segmenty budou umístěny na svých pozicích ve webovém prohlížeči. Zbylá část bude již samotná pracovní plocha, ve které se bude moci pracovat s uzly, poduzly, apod.

# 7 Vývoj systému

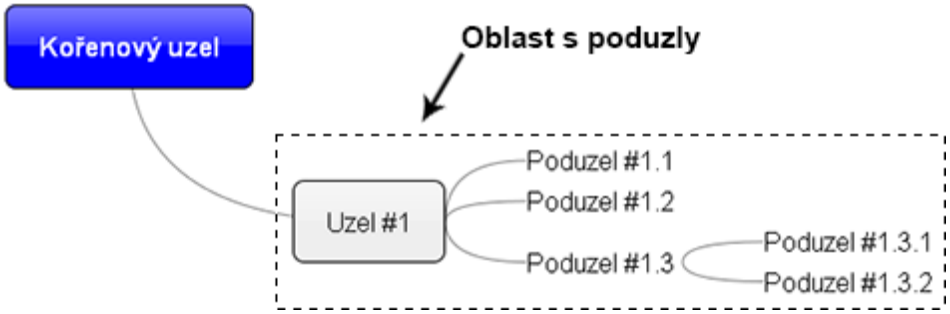
Dle vytvořeného návrhu byl realizován systém s názvem „Mindlook“. Tato kapitola se zabývá popisem problematiky a řešení jeho vybraných částí. Dále jsou zde uvedeny a vysvětleny použité programovací jazyky a frameworky. Následně je popsána adresářová struktura komprimovaného souboru, který je ke stažení na oficiálních stránkách systému Mindlook (http://www.mindlook.ic.cz). V závěrečné části je popsán postup při implementaci systému do jiných webových aplikací.

## 7.1 Technické řešení vybraných částí systému

V rámci realizace nového systému byly řešeny určité technické problémy. Mezi řešené technické problémy patřila například problematika volného manipulování s uzly. S touto problematikou se také pojil problém kontroly případných dotyků mezi oblastmi s poduzly. V rámci optimalizace systému pro dotyková zařízení nastal problém s regulací velikosti nástrojové lišty. Jeden z největších technických problémů byl proces ukládání provedených změn do databáze. Především těmto vyjmenovaným problémům se věnují následující podkapitoly.

### 7.1.1 Přesouvání oblastí s poduzly

Mapa obsahuje 2 základní objekty. Hlavní objekt je tzv. „kořenový uzel“, což je centrální uzel celé mapy, ke kterému se následně pojí tzv. „oblasti s poduzly“. Oba zmíněné objekty jsou na stránce umístěny v absolutní pozici, standardně je to pozice vzhledem k levému, hornímu okraji celé stránky.




Obrázek 10: Ukázka oblasti s poduzly

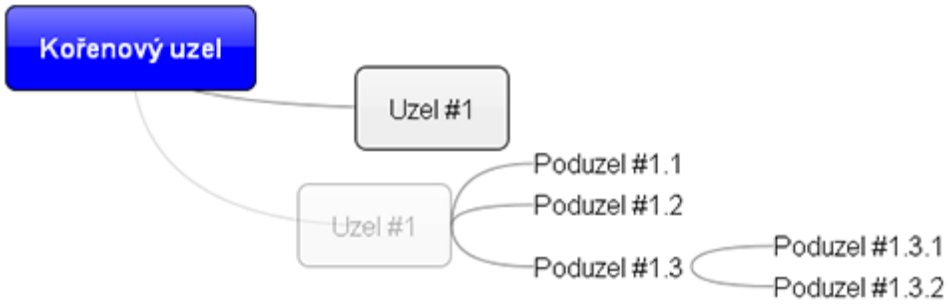
Jak je vidět na obrázku č. 10, dají se i oblasti s poduzly dělit do 2 částí, a to na uzel a jeho poduzly. Poduzly jsou k danému uzlu vázány striktně do stromové struktury, tudíž tato možnost je použitelná především pro myšlenkové mapy. Tento zmíněný uzel je důležitý především z hlediska přesouvání celé oblasti s poduzly, jelikož pouze tímto objektem lze danou oblast s poduzly přesunout na jinou pozici. Pro každý systémem vytvořený uzel, je vyvolána následující funkce, která webovému prohlížeči zavádí posluchač (listener) pro možné přesouvání tohoto uzlu a následné přesunutí jeho poduzlů.

```
/**
 * Zavedeni posluchace (listeneru) pro presouvani zadaneho elementu
 * @param Object[div.node]   thisElm   hlavni uzel [.node]
 */
function dragndrop(thisElm) {
   thisElm.draggable({
      start: function(event, ui) {
         // Ziskani ID zadaneho uzlu
         nodeClass = $(this).attr('class'),
         nodeID = nodeClass.match(/node([0-9]+)/)[1];

         // Pridani klonu [.node] a [.path{nodeID}]
         svgElm = svg.getElementById('path' + nodeID);
         pathTemp = svg.clone(svgElm)[0];
         svg.change(pathTemp, {id: 'pathTemp'});
         $(this).clone().appendTo('body').attr('id', 'nodeTemp').css({
            position: "absolute",
            left: ui.offset.left + 'px',
            top: ui.offset.top + 'px'
         });
         dragStartX = $('.node0').offset().left + ($('.node0').outerWidth
             () / 2),
         dragStartY = $('.node0').offset().top + ($('.node0').outerHeight
             () / 2);
      },
      drag: function(event, ui) {
         // Vypocteni stredu presouvaneho uzlu
         var endX = ui.offset.left + (ui.helper.outerWidth() / 2),
         endY = ui.offset.top + (ui.helper.outerHeight() / 2);
         // Uprava cesty k aktualni pozici pretahovaneho uzlu
         svg.change(svgElm, {d: 'M'+dragStartX+','+dragStartY+'Q'+
             dragStartX+','+endY+','+endX+','+endY});
      },
      stop: function(event, ui) {
         // Konecna velikost presunu
         var addX = parseInt(ui.offset.left),
         addY = parseInt(ui.offset.top);

         // Odebrani klonu
         svg.remove(pathTemp);
         $('#nodeTemp').remove();

         // Odstraneni pomocnych attributu u .node potrebnych pro
             dragndrop
         $(this).css({
            left: "",
            top: ""
         });

         // Zavedeni puvodnich pozic do seznamu pro zpetne kroky
         addToBackwardSteps("UPDATE", "left", nodeID, nodesProperties[
             nodeID]['position'][0], false);
         addToBackwardSteps("UPDATE", "top", nodeID, nodesProperties[
             nodeID]['position'][1], true);

         // Presun a korekce ostatnich poduzlu
         moveNodeArea(nodeID, addX, addY);
```

```
52
53          // Zavedeni nove pozice do logu pro zmeny
54          createRequestToStore("UPDATE", "left", nodeID, nodesProperties[
                nodeID]['position'][0]);
55          createRequestToStore("UPDATE", "top", nodeID, nodesProperties[
                nodeID]['position'][1]);
56
57          // Test pripadnych kolizi
58          collisionTest(nodeID, [nodeID], false);
59
60          // Zavedeni dat z logu pro zmeny do DB
61          saveChanges();
62
63          // Zmena vybraneho elementu
64          changeFocusedElement($('.node' + nodeID));
65      }
66  }).click(function() {
67      // Docasne zakazani presouvani oblasti s poduzly
68      if (!$(this).draggable('option', 'disabled')) {
69          $(this).draggable({disabled: true});
70          focusTo($(this));
71      }
72  }).blur(function() {
73      // Opetovne povoleni presouvani oblasti s poduzly
74      $(this).draggable({disabled: false});
75  });
76 }
```

Zdrojový kód 1: Funkce pro vytvoření posluchače (listeneru) k přesouvání oblastí s poduzly

Jak je vidět ve zdrojovém kódu č. 1, vyvolanému uzlu je zaveden posluchač (listener), který hlídá, zda se daný uzel začíná táhnout, tedy zda je na uzlu stisknuto levé tlačítko myši (řádek č. 6). V tom případě danému uzlu vytvoří poloprůhledný duplikát a vytvořený duplikát vloží na původní pozici právě taženého uzlu (řádek č. 7 - 23). Tento duplikát slouží pro vizuální sjednocenost oblasti s poduzly.




Obrázek 11: Ukázka duplikátu uzlu z oblasti s poduzly

Dále je v kódu vidět, že při tažení zvoleného uzlu se překresluje cesta, která tento uzel spojuje s kořenovým uzlem (řádek č. 24 - 30).

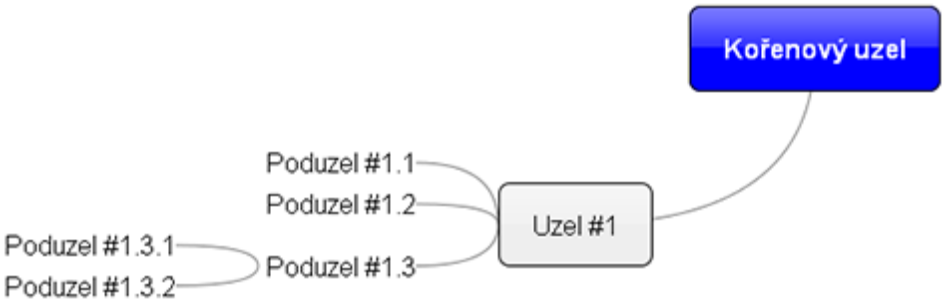
Při ukončení tažení, tedy uvolnění levého tlačítka myši, je zjištěna relativní pozice přetaženého uzlu od jeho původní pozice (řádek č. 33 a 34). Dále je z mapy odebrán

klon vytvořený při začátku tažení (řádek č. 37 a 38). Přetaženému uzlu jsou odebrány kaskádové vlastnosti, které určovaly jeho relativní posun od poduzlů (řádek č. 41 - 44). Následně jsou vyvolány funkce pro zavedení původní pozice do pole pro případné zpětné kroky (řádek č. 47 a 48). Dále je celá oblast s poduzly přesunuta na novou pozici (řádek č. 51) a hodnoty nové pozice jsou zavedeny do pole obsahujícího neuložené změny (řádek č. 54 a 55). Po přesunutí dané oblasti s poduzly je vyvolána funkce pro kolizní test (řádek č. 58), viz kapitola 7.1.2. Následně jsou data z pole pro neuložené změny odeslány ke zpracování php skriptu pomocí funkce „saveChanges();“ (řádek č. 61), viz kapitola 7.1.4. Na konci tohoto posluchače (listeneru) je uvedena funkce, která přesouvá pomyslný kurzor na právě přetažený uzel (řádek č. 64).

Na řádcích 66 - 72 je vidět další posluchač (listener), který zajišťuje, že při pouhém kliknutí na daný uzel, zakáže skriptu daný uzel přesouvat. Tento posluchač (listener) je zde kvůli možnosti přesouvání textového kurzoru po popisku tohoto uzlu. Řádek 72 - 76 zajišťuje, že pokud je kliknuto mimo tento uzel, přesun se znovu povolí.

**Orientace oblastí s poduzly**

Při použité volné manipulaci s uzly může nastat situace, že uzel z oblasti s poduzly bude přesunut relativně vlevo od kořenového uzlu. V této chvíli bylo kvůli přehlednosti důležité zajistit, aby se i poduzly od daného uzlu přesunuly doleva. Tuto část zajišťuje kaskádová vlastnost „direction“ s hodnotou „left“. Zmíněná vlastnost zajišťuje změnu direktivy textu, tedy změnu směru textu ze standardního směru zleva doprava na opačný směr. Jelikož mají všechny bloky z oblasti s poduzly kaskádovou vlastnost „display“ s hodnotou „inline-block“, jsou všechny bloky standardně považované za textové objekty, a v tom případě je i pro tyto bloky funkční změna směru textu.

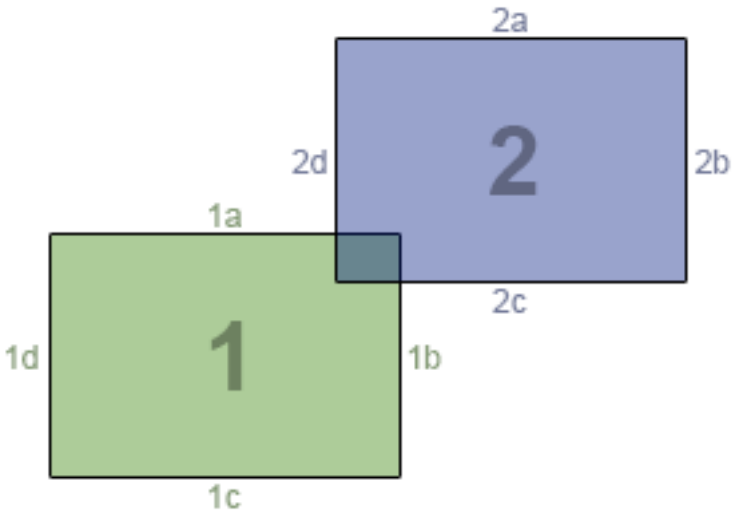



Obrázek 12: Ukázka doleva orientované oblasti s poduzly

V této fázi bylo důležité zajistit, aby nalevo orientované oblasti s poduzly neměly pozici X nastavenou z levého horního rohu, ale z pravého horního rohu. Důvod je jednoduchý, v případě, že by pozice X byla nastavena z levého horního rohu, oblast s poduzly by si při jakékoliv změně obsahu držela nastavenou pozici, a tak by se daná oblast s poduzly prodlužovala na opačnou stranu, než by bylo požadováno. Z tohoto důvodu jsou nalevo orientované oblasti s poduzly přepočítány a nastaveny na absolutní pozici z pravého horního rohu stránky.

### 7.1.2 Kolizní testy oblastí s poduzly

Při možnosti volné manipulace oblastí s poduzly, může nastat situace, kdy jedna oblast s poduzly překrývá druhou, a to ať z důvodu přesunutí nebo postupného zvětšování obsahu jedné oblasti přes druhou. Pokud by nám tato situace nevadila po vizuální stránce, určitě by nám vadila z hlediska úpravy popisků u překryté oblasti s poduzly. Z tohoto důvodu bylo potřeba vytvořit funkci, která by zkontrolovala dotek přesunuté oblasti s ostatními oblastmi na pracovní ploše.

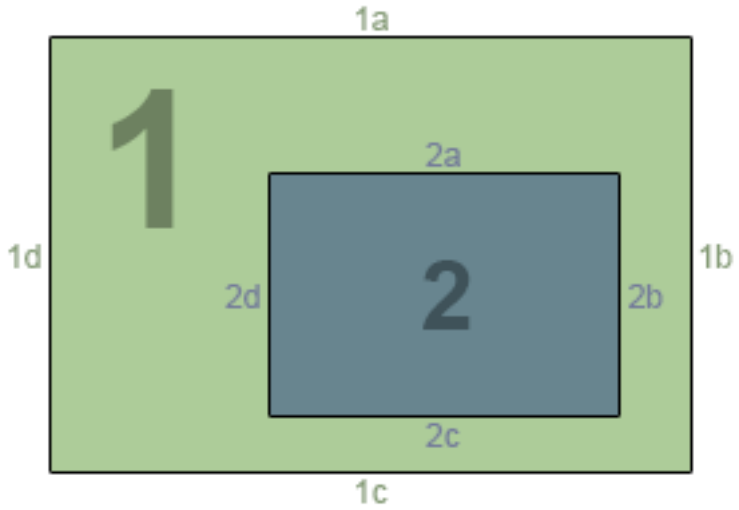



Obrázek 13: Dotek 2 objektů

Při pohledu na tento obrázek, který jsem si při řešení této problematiky načrtl, mě napadlo využít ke kontrole doteku mezních částí, neboli pomyslných hran, testovaných prvků.

**Příklad**
Podívejme se znovu na obrázek č. 13. Pokud přesuneme objekt 1 na objekt 2, tak aby vznikla situace zobrazená na obrázku, bude testováno, zda je některá z horizontálních hran posunutého objektu (objektu 1) v oblasti mezi hranami statického objektu (objektu 2). V tomto případě je tato podmínka platná, protože je jasně patrné, že hrana „1a“ je mezi hranami „2a“ a „2c“. Touto podmínkou bylo zkontrolováno, zda se objekty mohou (nikoliv musí) dotýkat ve vertikální ose. Pokud je splněna tato podmínka, je testováno, zda některá z vertikálních hran přesunutého objektu je mezi hranami statického objektu. V tomto případě je tato další podmínka také splněna, jelikož hrana „1b“ je umístěna mezi „2b“ a „2d“. Dotek tedy můžeme uznat za platný.

Při bližším pohledu na tuto problematiku je ovšem v tomto postupu viditelná trhlina. Jelikož není jisté, že všechny oblasti s poduzly jsou stejně velké, může nastat tato situace.




Obrázek 14: Plné překrytí objektu 2 objektem 1

Pokud přesuneme objekt 1 na pozici znázorněnou na tomto obrázku a budeme testovat objekty stejným způsobem jako minule, nebude splněna ani jedna podmínka. Ani jedna horizontální hrana posunutého objektu (objektu 1) není mezi horizontálními hranami statického objektu (objektu 2). Zároveň ani jedna vertikální hrana objektu 1 není mezi vertikálními hranami objektu 2. Dotek by tedy nebyl uznán platným. Ovšem při pouhém pohledu na obrázek, je jasně patrné, že se objekty navzájem dotýkají. Pokud se podívám na tento případ z jiné strany, konkrétně tedy tak, že objekt 2 přesuneme na pozici zobrazenou na obrázku č. 14, obě podmínky budou rázem splněny. Jak hrana „2a“, tak i „2c“ jsou mezi hranami „1a“ a „1c“ a zároveň obě hrany „2b“ a „2d“ jsou mezi hranami „1b“ a „1d“.

Postupnými testy možností, které mohou v této problematice nastat, jsem nakonec vymyslel jednoduché řešení tohoto problému. Při kontrole vertikálního doteku jsou vždy testovány hrany nižšího objektu a při kontrole horizontálního doteku jsou kontrolovány vždy užší objekty.

Pokud změní velikost oblasti s poduzly nebo je některá z oblastí s poduzly přesunuta na jiné místo, je vyvolána tato funkce.

```
1  /**
2   * Kontrola kolizi .nodesArea
3   * @param   int     nodeMove           id posunuteho uzlu (uzlu k
        otestovani)
4   * @param   array   nodesBlock         pole IDcek ktere se testovat nemaji
        (zamknute ID)
5   * @param   mixed   nodeMoveOrietation orientace posunu
6   */
7  function collisionTest(nodeMove, nodesBlock, nodeMoveOrietation) {
8     $.each(nodesProperties, function(nodeID, properties) {
9        // Prevedeni nodeID na String
10       nodeID = nodeID + '';
```

```
11          if ($.inArray(nodeID, nodesBlock) == -1) {
12              // Testovany node je korenovy [.node0]
13              if (nodeID == 0) {
14                  nodeID = nodeMove;
15                  nodeMove = 0;
16              }
17
18              // Ziskani oblasti s poduzly
19              var movingNodesArea = $(".node"+nodeMove).parent(),
20              testingNodesArea = $(".node"+nodeID).parent();
21
22              // Hodnoty pro kontrolu verikalniho doteku
23              if (movingNodesArea.height() < testingNodesArea.height()) {
24                  var smallerNodesArea = movingNodesArea,
25                  largerNodesArea = testingNodesArea;
26              } else {
27                  var smallerNodesArea = testingNodesArea,
28                  largerNodesArea = movingNodesArea;
29              }
30              var topSmallerNodesArea = smallerNodesArea.offset().top,
31              topLargerNodesArea = largerNodesArea.offset().top;
32              var bottomSmallerNodesArea = topSmallerNodesArea +
                    smallerNodesArea.height(),
33              bottomLargerNodesArea = topLargerNodesArea + largerNodesArea.
                    height();
34
35              // Kontrola vertikalniho doteku
36              if ((topSmallerNodesArea >= topLargerNodesArea &&
                    bottomLargerNodesArea >= topSmallerNodesArea) || (
                    bottomSmallerNodesArea >= topLargerNodesArea &&
                    bottomLargerNodesArea >= bottomSmallerNodesArea)) {
37                  // Hodnoty pro kontrolu horizontalniho doteku
38                  if (movingNodesArea.width() < testingNodesArea.width()) {
39                      var narrowerNodesArea = movingNodesArea,
40                      widerNodesArea = testingNodesArea;
41                  } else {
42                      var narrowerNodesArea = testingNodesArea,
43                      widerNodesArea = movingNodesArea;
44                  }
45                  var leftNarrowerNodesArea = narrowerNodesArea.offset().left,
46                  leftWiderNodesArea = widerNodesArea.offset().left;
47                  var rightNarrowerNodesArea = leftNarrowerNodesArea +
                        narrowerNodesArea.width(),
48                  rightWiderNodesArea = leftWiderNodesArea + widerNodesArea.
                        width();
49
50                  // Kontrola horizontaliho doteku
51                  if ((leftNarrowerNodesArea >= leftWiderNodesArea &&
                        rightWiderNodesArea >= leftNarrowerNodesArea) || (
                        rightNarrowerNodesArea >= leftWiderNodesArea &&
                        rightWiderNodesArea >= rightNarrowerNodesArea)) {
52                      // V pripade, ze testovany node neni korenovy zjisti zda je
                            vhodna korekce pozice x
53                      if (nodeMove != 0) {
54                          if (testingNodesArea.hasClass("left")) {
55                              var testingNodesAreaRight = testingNodesArea.offset()
                                    .left + testingNodesArea.width(),
```

```
56                      movingNodesAreaRight = movingNodesArea.offset().left
                            + movingNodesArea.width(),
57                      nodeToBeAligned = ((movingNodesAreaRight - 50) <
                            testingNodesAreaRight && (movingNodesAreaRight +
                            50) > testingNodesAreaRight) ? true : false;
58                  } else {
59                      var testingNodesAreaLeft = testingNodesArea.offset().
                            left,
60                      movingNodesAreaLeft = movingNodesArea.offset().left,
61                      nodeToBeAligned = ((movingNodesAreaLeft - 50) <
                            testingNodesAreaLeft && (movingNodesAreaLeft +
                            50) > testingNodesAreaLeft) ? true : false;
62                  }
63              } else {
64                  nodeToBeAligned = false;
65              }
66
67              // Neni-li jiz definovano zjisti smer pohybu v ose Y
68              if ((testingNodesArea.offset().top < movingNodesArea.offset
                    ().top) || nodeMoveOrietation == "-") {
69                  var topMove = movingNodesArea.offset().top - (
                        testingNodesArea.offset().top + testingNodesArea.
                        height());
70                  moveOrientation = nodeMoveOrietation ?
                        nodeMoveOrietation : "-";
71              } else {
72                  var topMove = (movingNodesArea.offset().top +
                        movingNodesArea.height()) - testingNodesArea.offset
                        ().top;
73                  moveOrientation = nodeMoveOrietation ?
                        nodeMoveOrietation : "+";
74              }
75
76              // Presun testovane nodesArea
77              testingNodesArea.css({top: "+="+topMove+"px"});
78              // Prekresleni cesty k presunutemu nodu
79              redrawPathToNode(nodeID);
80              // Presun cesty k poduzlum
81              movePathsToSubnodes(nodeID, 0, topMove);
82              // Zavedeni puvodni pozice do pole pro zpetne kroky
83              addToBackwardSteps("UPDATE", "left", nodeID,
                    nodesProperties[nodeID]['position'][0], true),
84              addToBackwardSteps("UPDATE", "top", nodeID, nodesProperties
                    [nodeID]['position'][1], true);
85              // Zavedeni nove pozice do pomocneho pole
86              nodesProperties[nodeID]['position'][0] = $('.node'+nodeID).
                    offset().left,
87              nodesProperties[nodeID]['position'][1] = $('.node'+nodeID).
                    offset().top;
88              // Zavedeni nove pozice do pole pro ulozeni do DB
89              createRequestToStore("UPDATE", "left", nodeID,
                    nodesProperties[nodeID]['position'][0]),
90              createRequestToStore("UPDATE", "top", nodeID,
                    nodesProperties[nodeID]['position'][1]);
91              // Neni-li testovany uzel korenovy, zaved presunuty uzel do
                     zamknutych a vyvolej kolizni test pro presunuty uzel
92              if (nodeMove != 0) {
```

```
93                      collisionTest(nodeID, $.merge(nodesBlock, [nodeID]),
                            moveOrientation);
94                  }
95                  // Ma-li se pouzit korekce pozice X, pouzi ji
96                  if (nodeToBeAligned) {
97                      if (testingNodesArea.hasClass("left")) {
98                          testingNodesAreaRight = testingNodesArea.offset().
                                left + testingNodesArea.width();
99                          movingNodesArea.css('right', testingNodesArea.css('
                                right'));
100                         movePathsToSubnodes(nodeMove, (testingNodesAreaRight
                                - movingNodesAreaRight), 0);
101                     } else {
102                         testingNodesAreaLeft = testingNodesArea.offset().left
                                ;
103                         movingNodesArea.css('left', testingNodesAreaLeft);
104                         movePathsToSubnodes(nodeMove, (testingNodesAreaLeft -
                                movingNodesAreaLeft), 0);
105                     }
106                 }
107             }
108         }
109         if (nodeMove == 0) {
110             nodeMove = nodeID;
111         }
112     }
113 });
114 // Zaved potrebne hodnoty do vsech poli
115 if (nodesProperties[nodeMove]['position'][0] != $('.node'+nodeMove).
        offset().left) {
116     addToBackwardSteps("UPDATE", "left", nodeMove, nodesProperties[
        nodeMove]['position'][0], true, true);
117     nodesProperties[nodeMove]['position'][0] = $('.node'+nodeMove).
        offset().left;
118     createRequestToStore("UPDATE", "left", nodeMove, nodesProperties[
        nodeMove]['position'][0]);
119 }
120 if (nodesProperties[nodeMove]['position'][1] != $('.node'+nodeMove).
        offset().top) {
121     addToBackwardSteps("UPDATE", "top", nodeMove, nodesProperties[
        nodeMove]['position'][1], true, true);
122     nodesProperties[nodeMove]['position'][1] = $('.node'+nodeMove).
        offset().top;
123     createRequestToStore("UPDATE", "top", nodeMove, nodesProperties[
        nodeMove]['position'][1]);
124 }
125 }
```

Zdrojový kód 2: Funkce pro kontrolu kolizí mezi oblastmi s poduzly

Jak je vidět ve zdrojovém kódu č. 5, funkce pro kontrolu kolizí je vyvolána s několika parametry. Parametr „nodeMove“, který obsahuje identifikátor zvětšené či přesunuté oblasti s poduzly. Dále parametr „nodesBlock“ obsahující identifikátory oblastí s poduzly, které se již nemají testovat. Tento parametr zajišťuje, aby se funkce neopakovala vícekrát, než je opravdu nutné. Poslední parametr je „nodeMoveOrientation“, který dané funkci určuje, zda-li se má testovaná oblast s poduzly přesunout dolu či nahoru.

Případně, zda-li se má směr posunu určit výpočtem.

Funkce obsahuje cyklus „each“, který zajišťuje, aby se přesunutý uzel otestoval se všemi existujícími uzly na pracovní ploše (řádek č. 8). Na řádku č. 11 je uvedena podmínka, která filtruje testování zakázaných oblastí s poduzly z parametru „nodesBlock“. Má-li se daná oblast s poduzly testovat, je kontrolováno, zda testovaný uzel není kořenový. Pokud je tato podmínka splněna identifikátory určující přesunutý a testovaný uzel se přehodí (řádek č. 13 - 16). Kořenový uzel má statickou pozici, kterou není možné měnit. V případě kolize se však přesouvají testované oblasti s poduzly. Přehozením těchto proměnných je tedy zajištěno, že v případě kolize jakékoliv oblasti s kořenovým uzlem, je znovu přesunut právě přemístěný uzel a nikoliv kořenový uzel. Na řádcích 18 - 51 je uveden testovací postup, který jsem popisoval začátkem této kapitoly. Jsou-li splněny všechny podmínky, které určují zda se oblasti dotýkají, je dále testováno, zda se má přesouvaná oblast s poduzly vertikálně zarovnat s dotýkající se oblastí. Zarovnání bude provedeno až v další části kódu, ovšem pouze za předpokladu, že pozice X přesunuté oblasti je větší nebo menší o 50 pixelů od testované oblasti (řádek č. 53 - 65). Následně je vypočteno o kolik se má testovaná oblast posunout, aby se oblasti mezi sebou nepřekrývaly. Zároveň není-li určen směr pohybu, má být zjištěn (řádek č. 68 - 74). Dále je testovaná oblast přesunuta o právě vypočtenou hodnotu (řádek č. 77). Poté je překreslena cesta vedoucí k tomuto uzlu (řádek č. 81). Následně jsou zavedeny původní pozice do pole pro případné zpětné kroky (řádek č. 83 a 84). Hodnoty nové pozice jsou zavedeny do pomocného pole, které textově doprovází vizuální podobu objektů na pracovní ploše (řádek č. 86 a 87). Tyto hodnoty jsou také zavedeny do pole obsahujícího neuložené změny (řádek č. 89 a 90). Jelikož byl testovaný uzel přesunutý na jinou pozici, musí se i on otestovat na případné kolize (řádek č. 92 - 94). Bylo-li zjištěno, že má být přesunutý uzel vertikálně zarovnán, provede se tak nyní (řádek č. 96 - 113). Po ukončení testování ostatních oblastí s poduzly jsou zavedeny původní pozice přesouvaného uzlu do pole pro případné zpětné kroky, nové pozice do pomocného pole a do pole obsahujícího neuložené změny (řádek č. 115 - 124).

### 7.1.3 Dynamická velikost nástrojové lišty

Nejsou-li webové stránky cíleně navrženy pro dotykové zařízení, chovají se webové prohlížeče těchto zařízení trochu jinak, než klasické desktopové verze webových prohlížečů. Stránka se v základu zobrazí vždy tak, aby byly vidět všechny části webové aplikace. Tažnými pohyby prstů, lze následně regulovat hodnotu, a tak si případně zvětšit požadované informace. Tato vlastnost ovšem byla pro tento systém absolutně nevhodná. Při nulovém přiblížení nebyla nástrojová lišta téměř vidět, a tak samozřejmě nešly využívat ovládací prvky. Naopak při úplném zvětšení ovládací prvky zabíraly celé pracovní prostředí, a tak nešlo manipulovat s objekty v mapě.

Tento problém jsem se rozhodl vyřešit vytvořením skriptu, který by detekoval, zda je právě využívaný webový prohlížeč používán na dotykovém zařízení. Pokud by tato podmínka byla splněna, velikost nástrojové lišty by se postupně měnila v závislosti na maximální šířce, kterou by aktuálně zvolená hodnota přiblížení dovolovala zobrazit. Sepsán byl tedy následující skript, který je vyvolán při načtení systému.

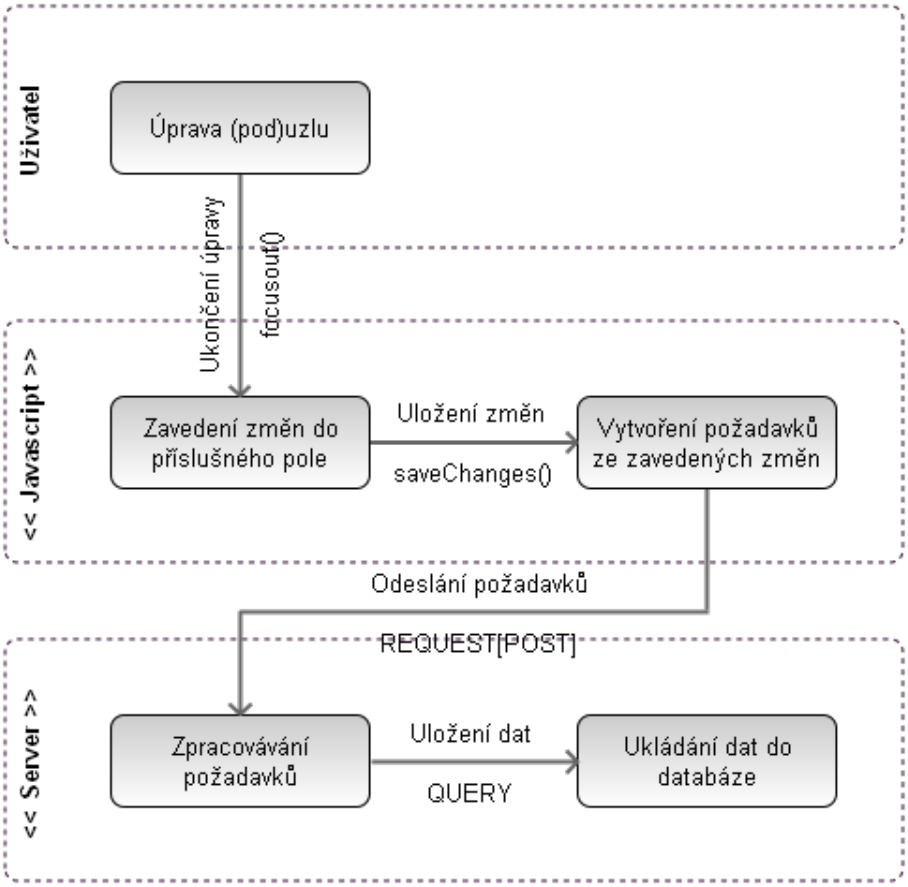
```
1  // Detekce dotykoveho zarizeni pro regulaci velikosti nastrojove listy
2  if (jQuery.support.touch) {
3
4      // Vytvoreni listeneru pro detekci zmeny velikosti okna ci prerolovani
           stranky
5      window.onresize = window.onscroll = function () {
6
7          // Nastaveni velikosti tlacitek z nastrojove listy
8          $('.toolbar button').css({
9              width: ((window.innerWidth / screen.width) * 40) + "px",
10             height: ((window.innerWidth / screen.width) * 40) + "px",
11             margin: ((window.innerWidth / screen.width) * 3) + "px"
12         });
13
14         // Zobrazeni tlacitka pro vysouvani doplnkoveho menu
15         if ($('.toolbar span').first().offset().top != $('.toolbar span').
               last().offset().top) {
16             // Zobrazeni tlacitka
17             $('#pressMenu').show();
18
19             // Zasunuti doplnkoveho menu
20             $('.toolbar').css({
21                 top: "-" + ((window.innerWidth / screen.width) * 40) - ((
                       window.innerWidth / screen.width) * 6) + "px"
22             });
23         } else {
24             // Skryti tlacika pro vysouvani doplnkoveho menu
25             $('#pressMenu').hide();
26
27             // Vysunuti toolbaru
28             $('.toolbar').css("top", "0px");
29         }
30     };
31 }
```

Zdrojový kód 3: Detekce dotykových zařízení pro regulaci velikosti nástrojové lišty

Pokud je využívané zařízení dotykové, je vytvořen posluchač (listener), který detekuje, zda bylo okno webového prohlížeče velikostně modifikováno či zda byla webová aplikace rolována (řádek č. 5). Některé mobilní verze webových prohlížečů detekují přiblížení jako změnu velikosti okna. Jiné prohlížeče zas tuto vlastnost neovládají, a tak bylo zároveň zabudováno i detekce rolování stránky. Dále je tlačítkům v nástrojové liště upravena velikost, která se vypočítává z velikosti šířky rozlišení daného zařízení a ze šířky, kterou je v aktuálně zvolené hodnotě přiblížení možno zobrazit. Tím se získá poměr velikosti aktuálního přiblížení, který je následně vynásoben standardní velikostí daných prvků (řádek č. 8 - 12). Dále je zjišťováno, zda se všechny ovládací prvky dokážou vměstnat do jedné řady. Pokud tomu tak není, zobrazí se v základu skryté tlačítko pro vysouvání zbytku ovládacích prvků (řádek č. 17) a nástrojová lišta se částečně skryje tak, aby byly vidět pouze ty nejdůležitější ovládací prvky (řádek č. 20 - 22). Pokud nastane situace, že se všechny ovládací prvky znovu dokážou vměstnat do jedné řady, je aplikován opačný postup (řádek č. 25 a 28).

### 7.1.4 Ukládání dat do databáze

Dá se říci, že jednou z nejdůležitějších částí systému je proces ukládání dat do databáze. Při řešení této problematiky jsem chtěl docílit toho, aby se data do databáze ukládala sama a uživatel nemusel při každé změně vyvolávat funkci pro ukládání manuálně. Vytvořil jsem si tedy následný model pro ukládání dat.




Obrázek 15: Model ukládání dat

Vytvořený model popisuje proces ukládání dat. Uživatel upraví popisek uzlu a následně klikne například na pracovní plochu. Tím opustí mód úpravy uzlu a javascript zavede nový popisek do pole obsahujícího neuložené změny. Javascript dále vyvolá funkci pro uložení nově zavedených dat do databáze, která vytvoří požadavek a odešle serveru data pomocí AJAXu. Server požadavky zpracuje a změny uloží do databáze. Z tohoto modelu byl následně vytvořen tento skript pro vytvoření a zavedení požadavku do pole obsahující neuložená data.

```
1  /**
2   * Vytvor pozadavek do pole pro neulozena data
3   * @param String           type       typ pozadavku (UPDATE|CHANGE-XPATH|
        INSERT|DELETE)
4   * @param String           property   co se ma zmenit (VALUE|XPATH|#
        Vlastnost uzlu z DB#)
5   * @param mixed             xpath      absolutni cesta k uzlu v ramci mapy
6   * @param String           value      nova hodnota
7   */
8  function createRequestToStore(type, property, xpath, value) {
9     // Je-li hodnota xpath odkaz na objekt zjisti jeho identifikator
10    if (typeof xpath != "object") {
11       var xpath = xpath;
12    } else {
13       if (xpath.hasClass("subnodesTitle")) {
14          var xpath = xpath.parent().attr('class').match(/node([0-9\-]+)/)
                [1];
15       } else {
16          var xpath = xpath.attr('class').match(/node([0-9\-]+)/)[1];
17       }
18    }
19
20    // Vytvor identifikator pozadavku
21    var timeRequest = $.now();
22    // Existuje-li uz dany identifikator pridej na konec 0
23    if (requestsToStore[timeRequest] !== undefined) {
24       timeRequest = parseInt(timeRequest + "0");
25    }
26
27    // Zaved pozadavek do pole pro neulozena data
28    console.log("Request created");
29    requestsToStore[timeRequest] = {
30       type: type,
31       property: property,
32       xpath: xpath,
33       value: value
34    };
35 }
```

Zdrojový kód 4: Funkce pro zavedení nového požadavku do pole pro neuložená data

Funkce je vyvolána se 4 parametry. Parametr „type“ reprezentuje typ požadavku. V rámci systému mohou nastat čtyři. Typ „UPDATE“ určuje modifikaci již zavedených data v databázi. Typ „CHANGE-XPATH“ je vyvoláván pro změnu určitých absolutních cest u (pod)uzlů. Dalším typem může být „INSERT“, který vkládá nová data do databáze a typ „DELETE“, který data z databáze odstraňuje. Parametr „property“ specifikuje, jaká část se má v databázi upravit. Parametr „xpath“ obsahuje absolutní cestu (pod)uzlu, pro který změna platí. Posledním parametrem je „value“, obsahující hodnotu, která má být vložena do databáze.

Skript tedy nejprve zjistí, zdali je zadaný parametr „xpath“ identifikátor elementu. V případě, že obsahuje odkaz na element, zjistí se identifikátor sám (řádek č. 10 - 18). Dále je vytvořen identifikátor požadavku formou aktuálního času (řádek č. 21). Pokud je vytvářeno více požadavků za sebou, může nastat situace, že by identifikátor byl pro

více požadavků stejný. Tento problém řeší řádek č. 23 - 25, který v případě duplicitního identifikátoru přidá 0. Nakonec se zavedou data do pole pro neuložená data (řádek č. 28 - 34).

Dále byl vytvořen tento skript pro odeslání požadavků serveru.

```
1  /**
2   * Odeslani zmen z pole pro neulozena data serveru ke zpracovani
3   */
4  function saveChanges() {
5     // Pole pro neulozena data neni prazdne
6     if (!$.isEmptyObject(requestsToStore)) {
7        // Odeslani pole pro neulozena data serveru
8        $.post("../saveChanges.php", {
9           requests: requestsToStore
10       })
11       // Pozadavky byly uspesne zpracovany
12       .done(function(success) {
13          // Server zpracoval pozadavky a odesila identifikatory uspesne
                ulozenych pozadavku
14          $.each(success, function(i, request) {
15             delete requestsToStore[request];
16          });
17          // Pokud nebyl nejaky pozadavek zpracovan odesli ho serveru jeste
                jednou
18          if (!$.isEmptyObject(requestsToStore)) {
19             // Aby nedoslo k zacykleni zkontroluj, zda se chyba jiz
                   neopakuje
20             if (save) {
21                save = false;
22                saveChanges();
23                return false;
24             } else {
25                saveFaild();
26             }
27          } else {
28             save = true;
29             return true;
30          }
31       })
32       // Nastala chyba na strane serveru
33       .fail(function() {
34          saveFailed();
35          setTimeout("saveChanges",3000);
36          return false;
37       });
38    } else {
39       return true;
40    }
41 }
```

Zdrojový kód 5: Funkce pro odeslání změn serveru

Při vyvolání funkce je zkontrolováno, zda není pole pro neuložená data prázdné (řádek č. 6). V případě, že prázdné není, jsou odeslány požadavky z tohoto pole AJAXem na server ke zpracování (řádek č. 8 - 10). Skript na serveru požadavky zpracuje

a odešle zpět identifikátory úspěšně zpracovaných požadavků. Přijaté identifikátory jsou následně zpracovávány tak, že data uložená pod těmito identifikátory v poli pro neuložená data, jsou postupně mazána (řádek č. 14 - 16). Tento krok zajišťuje, že pokud server z nějakého důvodu nezpracuje jakýkoliv požadavek, bude odeslán znovu ke zpracování a uživatel bude informován pomocí stavové lišty (řádek č. 18 - 30). Pokud nastane chyba na straně serveru, skript počká 50 sekund a odešle požadavky znovu (řádek č. 33 - 37).

## 7.2 Použité programovací jazyky a frameworky

### 7.2.1 PHP

PHP je programovací jazyk často používaný pro psaní skriptů na straně serveru (server-side script). Využívá se pro generování dynamického obsahu typicky HTML stránek v internetových aplikacích. PHP ovšem není omezen pouze na generování HTML. Může být použit pro generování jakéhokoli typu obsahu, např. obrázků, dynamických grafů, generovaných reportů v různých formátech atd.[15]

### 7.2.2 MySQL

MySQL je relační databázový systém typu DBMS (Database Management System) vlastněný společností Oracle. Každá databáze v MySQL je tvořena z jedné nebo více tabulek, které mají řádky a sloupce. V řádcích rozeznáváme jednotlivé záznamy. Sloupce mají jméno a uvozují datový typ jednotlivých polí záznamu. Práce s databázemi, tabulkami a daty se provádí pomocí příkazů, respektive dotazů. Dotazy vycházejí z deklarativního programovacího jazyka SQL (Structured Query Language).[16]

### 7.2.3 jQuery

jQuery je rychlá, malá a na funkce bohatá knihovna javascriptu. Díky této knihovně je možné procházet a manipulovat s HTML dokumenty, zpracovávat javascriptové události, vytvářet animace a používat technologii AJAX s velmi jednoduchým API, které je funkční v celé řadě prohlížečů.[17]

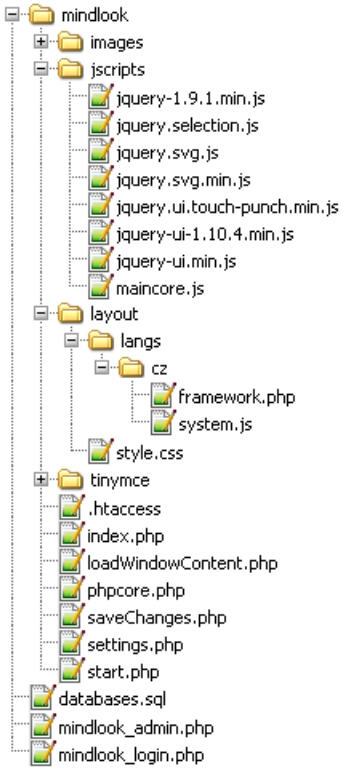
**AJAX**
AJAX (asynchronous javascript and XML) je javascriptová technologie, díky které je možné vytvářet vysoce interaktivní webové aplikace, které je možné měnit bez nutnosti jejich celkového znovunačtení. Autorem této technologie je James Garrett, který tuto technologii představil roku 2005 v publikovaném článku s názvem „Ajax: A New Approach to Web Applications“, což v překladu znamená „Ajax: Nový přístup k webovým aplikacím“.

**SVG plug-in pro jQuery**
Tento plug-in umožňuje manipulovat s SVG plátnem pomocí jazyku jQuery. Autorem tohoto plug-inu je Keith Wood.

**TinyMCE plug-in pro jQuery**
TinyMCE je nezávislá HTML platforma WYSIWYG editoru založená na javascriptu. Uvolněna je jako Open Source pod licencí LGPL.[18]

## 7.3 Adresářová struktura systému

V této kapitole jsou popsány adresáře a soubory, které byly vytvořeny pro tento systém.




Obrázek 16: Adresářová struktura

V komprimovaném souboru, který je ke stažení na oficiálních stránkách projektu mindlook je **adresář „mindlook/“**, který obsahuje následující adresáře a soubory potřebné pro chod systému:

**Adresář „images/“** obsahuje obrázky využívané v samotném prostředí systému, jako jsou například ikony pro nástrojovou lištu, apod.

**Adresář „jscripts/“** obsahuje javascriptové knihovny pro jQuery, jQuery UI, jQuery SVG a soubor nezbytný pro chod systému, konkrétně „maincore.js“, který obsahuje javascriptové jádro celého systému.

**Adresář „layout/“** obsahuje soubor „style.css“ s kaskádovými styly formátující vzhled systému. Dále je zde adresář „langs/“, který obsahuje lokalizační adresáře a soubory pro případný jazykový překlad systému do cizích jazyků.

**Adresář „tinymce/“**, kde jsou soubory potřebné pro chod WYSIWYG editoru TinyMCE.

**Soubor „.htaccess“**, který obsahuje nastavení pro server Apache.

**Soubor „index.php“**, který obsahuje základní HTML strukturu systému.

**Soubor „loadWindowContent.php“**, který obsahuje PHP skript generující obsah pro doplňkové okno (například pro vlastnosti uzlu a grafických odkazů).

**Soubor „phpcore.php“**, který obsahuje definice základních PHP funkcí, proměnných využívající PHP skripty celého systému a přihlášení do databáze MySQL.

**Soubor „saveChanges.php“**, který obsahuje PHP skript vyvolávaný AJAXem pro zavedení online změn do databáze MySQL.

**Soubor „settings.php“**, který obsahuje rozšiřující definice a proměnné využívající PHP skripty v systému. Data v tomto souboru je možné upravit a nastavit tak požadované chování systému.

**Soubor „start.php“**, který obsahuje PHP skript vytvářející strukturu mapy ve formě pole JSON uložené v databázi MySQL.

V komprimovaném souboru jsou dále umístěny tyto soubory:

**Soubor „databases.sql“**, který obsahuje skript pro vytvoření struktury MySQL databáze.

**Soubor „mindlook_admin.php“**, pomocí tohoto souboru si může přihlášený uživatel spravovat své mapy uložené v databázi.

**Soubor „mindlook_login.php“**, pomocí tohoto souboru se uživatel může přihlásit či zaregistrovat do systému mindlook.

## 7.4 Možnost implementování systému

Systém Mindlook je navržen tak, aby jej bylo možno implementovat do již existujících systémů, jako jsou například různé redakční systémy či edukační systém Moodle. Implementovat systém mindlook je možné ve dvou variantách. Popis těchto variant s postupem jejich implementace je uveden níže.

### 7.4.1 Základní implementace

Základní implementace je velmi jednoduchá a zvládne jí i mírně pokročilý uživatel. Touto verzí implementace je možné získat plnohodnotný systém, který bude nezávislý na ostatních webových aplikacích umístěných na stejném webovém serveru.

**Postup**
Implementace systému Mindlook je možná v následujících krocích:

1. Stažení archivu z webových stránek projektu Mindlook (http://www.mindlook.ic.cz/).
2. Rozbalení staženého archivu na disk.
3. Vytvoření databázové struktury pomocí skriptu „/databaze.sql“.
4. Doplnění přihlašovacích údajů na databázový server v souboru „/mindlook/-settings.php“ (řádky 6 - 9)

```
1 # PRIHLASOVACI UDAJE DO DATABAZE
2 $DB_HOST = ""; //Adresa serveru
3 $DB_USER = ""; //Login
4 $DB_PASS = ""; //Heslo
5 $DB_NAME = ""; //Databaze
```

Zdrojový kód 6: Přihlašovací údaje do databáze

5. Nakopírování obsahu ze staženého archivu na webový server.

### 7.4.2 Rozšířená implementace

Rozšířená implementace je podstatně složitější a měl by jí provádět velmi zkušený uživatel, nejlépe však webmaster či programátor. Touto verzí implementace je možné získat plnohodnotný systém, který je ovšem možné navázat na jiné webové aplikace umístěné na stejném webovém serveru.

Databázi systému Mindlook je možné navázat na existující databázovou tabulku s uživateli. Výhody jsou v tomto případě patrné. Pokud se uživatel přihlásil do již existujícího systému, do kterého byl implementován systém Mindlook, bude mít uživatel k dispozici své mapy bez nutnosti přihlašovat se či dokonce registrovat se do implementovaného systému Mindlook.

Před touto variantou implementace je důležité znát, případně umět dohledat tyto informace:

- Název existující tabulky s uživateli,
- proměnnou, případně kód pro získání ID právě přihlášeného uživatele.

**Postup**

1. Stažení archivu z webových stránek projektu Mindlook (http://www.mindlook.ic.cz/).
2. Rozbalení staženého archivu na disk.

3. Úprava skriptu ze souboru „/databaze.sql“ dle následujících pokynů:

   Je vhodné odstranit řádky 1 - 6, které standardně vytvářejí tabulku pro registrované uživatele.

```
1 CREATE TABLE IF NOT EXISTS `users` (
2   `user_id` int(11) NOT NULL AUTO_INCREMENT,
3   `user_mail` varchar(255) DEFAULT NULL,
4   `user_pwd` varchar(32) DEFAULT NULL,
5   PRIMARY KEY (`user_id`)
6 ) ENGINE=InnoDB  DEFAULT CHARSET=utf8;
```

Zdrojový kód 7: Řádky k odstranění

   Dále je důležité změnit kód na řádkách 87 a 102, do následujícího formátu.

```
1 ADD CONSTRAINT `/*TUTO CAST NEMENIT*/` FOREIGN KEY (`user_id`)
    REFERENCES `/*NAZEV TABULKY S REGISTROVANYMI UZIVATELI*/` (`/*
    NAZEV SLOUPCE S ID Z TABULKY REGISTROVANYCH UZIVATELU*/`) ON
    DELETE CASCADE ON UPDATE CASCADE;
```

Zdrojový kód 8: Formát úpravy

4. Vytvoření databázové struktury pomocí upraveného skriptu „/databaze.sql“.

5. Úprava údajů v souboru „/mindlook/settings.php“ dle následujících pokynů:

   Doplnění přihlašovacích údajů na databázový server (řádky 6 - 9).

```
1 # PRIHLASOVACI UDAJE DO DATABAZE
2 $DB_HOST = ""; //Adresa serveru
3 $DB_USER = ""; //Login
4 $DB_PASS = ""; //Heslo
5 $DB_NAME = ""; //Databaze
```

Zdrojový kód 9: Přihlašovací údaje do databáze

   Změna funkce pro získání ID právě přihlášeného uživatele na řádku 16.

```
1 # ID UZIVATELE
2 function USER_ID() {
3     $USER_ID = $_SESSION['USER']; //$_SESSION['USER'] nahradit
          promennou, kterou vyuziva vas system
4     if (isset($USER_ID)) {
5         return $USER_ID;
6     } else return false;
7 }
```

Zdrojový kód 10: Funkce pro získání identifikátoru uživatele

Změna definice názvu tabulky pro registrované uživatele na řádku 34.

```
1 # DEFINICE PRO TABULKY Z DATABAZE
2 define('DB_USERS', 'users'); //users nahradime za nazev tabulky s
      uzivateli
```

Zdrojový kód 11: Definice názvu tabulky s registrovanými uživateli

6. Nakopírování složky „mindlook“ na webový server.
7. Navázání existující aplikace na systém Mindlook. Například vytvořením modulu pro příslušný systém. K vytvoření modulu, lze využít kódu ze souboru „mindlook_admin.php“, který vypisuje mapy právě přihlášeného uživatele uložené v databázi.

### 7.4.3 Minimální požadavky na webhosting

Minimální požadavky na webhosting pro správný chod systému jsou:

- PHP ve verzi 5.0 a vyšší,
- databázový server MySQL,
- cca 5 MB prostoru na webovém serveru.

# 8 Implementace a hodnocení systému

Za účelem ověření funkčnosti implementace systému byl vybrán veřejný webový server Internet Centrum (ic.cz), který nabízí využívání webhostingu zdarma. Na zmíněném webhostingu byl založen účet pro webovou adresu www.mindlook.ic.cz. Pro účely prezentace realizovaného systému byly vytvořeny webové stránky, které na zmíněné adrese slouží nejen jako systémová dokumentace, ale jsou zde i zdrojové kódy ke stažení, formulář pro hlášení chyb apod. Implementace systému Mindlook na zmíněný webhosting proběhla rychle a bez jakýchkoliv problémů.

Z důvodu vysoké časové náročnosti na samotný vývoj tohoto systému, byla implementace provedena až v době zpracovávání této písemné části. Proto nemohu v současné době popsat úplné výsledky z tohoto testování. Ovšem již v současnou chvíli byla nalezena a opravena jedna vážnější chyba i několik chyb poměrně zanedbatelného charakteru.

Za vážnější pochybení při vývoji považuji chybu, která byla popsána takto: „Po úpravě formátu textu v uzlu pomocí okna pro úpravu vlastností, se text v zobrazené mapě upravil správně. Při znovunačtení webové stránky se ale formát upraveného uzlu vrátil do původního nastavení.“. Při odhalování původu této chyby jsem zjistil, že server na kterém byl spuštěn systém Mindlook obsahuje vlastnost v nastavení PHP, která má předcházet možnosti útoků „SQL injection“ nebo „XSS“ (konkrétně je to zapnutá vlastnost „magic_quotes_gpc“). Toto nastavení zajišťuje, že všechny speciální znaky ze vstupních proměnných jsou převedeny na HTML entity. Ovšem sám systém obsahuje také vlastní ochranu před možností zmíněných útoků, a to PHP funkci „mysql_real_escape_string()“. Tato dvojitá ochrana ovšem zapříčinila, že data odeslaná pomocí TinyMCE byla 2 krát převedena na HTML entity. Ty se následně již nedokázali při načtení přetransformovat zpět do původních znaků, a tak nebyly nastavené vlastnosti správně načteny. Tuto chybu jsem vyřešil kontrolou nastavení PHP vlastnosti „magic_quotes_gpc“ a v případě jeho vypnutí je použita PHP funkce „mysql_real_escape_string()“.

Další nahlášené chyby byly prozatím pouze lokalizačního či stylistického původu. Ovšem testování v současnou dobu stále probíhá, a tak předpokládám další nalezené chyby, které bude potřeba vyřešit. Ohlasy jsou ze strany uživatelů na tento systém prozatím kladné, ovšem až po úplném vyladění systému se ukáže, zda systém v praxi plně obstojí.

# 9 Závěr

Tato bakalářská práce se zabývá návrhem a realizací nového systému pro tvorbu pojmových a myšlenkových map. Při realizaci této práce bylo využito vývojového modelu ADDIE, který výrazně přispěl správné metodice při vývoji.

Analyzováno bylo celkem 21 konkurenčních řešení. Z výsledků této analýzy vyplývá, že autoři těchto systému optimalizují své webové řešení pro dotykové zařízení jen sporadicky. Prostředky investují spíše do vytváření aplikací pro mobilní operační systémy. Implementace systému do jiných webových aplikací není v mnohých případech vůbec nabízena. Proto byl při návrhu nového systému kladen důraz na vytvoření volně šiřitelného řešení, které je možné ovládat pomocí většiny současných webových prohlížečů, a to i na dotykových zařízeních. Dále byl kladen důraz na to, aby bylo možné tento systém jednoduše implementovat do jiných webových aplikací.

Systém vytvořený v rámci této bakalářské práce splňuje všechny stanovené cíle. Po úplném odladění systému je plánován další vývoj rozšiřujících nástrojů a funkcionalit, které uživatelům zpříjemní a zjednoduší případnou práci v systému.

Za částečný neúspěch se dá považovat nefunkčnost systému na dosud používaném operačním systému Windows XP za použití webového prohlížeče Internet Explorer. Bohužel jak je popsáno v kapitole 6.2, na operační systém Windows XP je možné nainstalovat Internet Explorer v nejvyšší verzi 8. Tato verze ovšem nepodporuje určité prvky HTML5 využívané v navrženém systému, a tak je mapa v tomto webovém prohlížeči neúplná. Ovšem za použití neustále vyvíjených webových prohlížečů, jako jsou například Google Chrome, Mozilla Firefox, Opera apod., je funkčnost 100% i na operačním systému Windows XP. Tento problém by teoreticky mohlo vyřešit konečné schválení specifikace HTML 5.0, které by mělo proběhnout v posledním čtvrtletí tohoto roku. V této souvislosti by mohlo dojít k hojnějšímu využívání této technologie na nových webových aplikacích, čímž by uživatelé Internet Exploreru byli donuceni k přechodu na jiný webový prohlížeč či k aktualizaci operačního systému.

I přes všechny jednotlivé problémy, které při vývoji vznikly, se podařilo realizovat systém, který jistě bude po úplném odladění plně konkurenčně schopný a zajisté bude dobře sloužit účelům bakalářské práce.

# Reference

[1] VAŇKOVÁ, Petra. *POJMOVÉ MAPY VE VZDĚLÁVÁNÍ.* Praha: Karolinum, 2014. ISBN 978-80-7290-650-5.

[2] Myšlenková mapa. *Inkluzivní škola* [online]. 2013 [cit. 2014-04-08]. Dostupné z: http://www.inkluzivniskola.cz/pedagogicka-prace-s-diverzitou/myslenkova-mapaTypografie.

[3] What are mind maps?. *Grammarmindmaps.com* [online]. 2012 [cit. 2014-04-08]. Dostupné z: http://www.grammarmindmaps.com/about-mind-maps/

[4] PRICE, Geraldine a Pat MAIER. *Efektivní studijní dovednosti: odemkněte svůj potenciál.* Vyd. 1. Praha: Grada, 2010, 361 s. Psychologie pro každého. ISBN 978-80-247-2527-7.

[5] Adobe - Flash Player: Nápověda. *Adobe* [online]. 2014 [cit. 2014-04-08]. Dostupné z: https://www.adobe.com/support/documentation/cz/flashplayer/help/help10.html

[6] Free, open-source framework | Adobe Flex. *Adobe* [online]. 2014 [cit. 2014-04-08]. Dostupné z: http://www.adobe.com/cz/products/flex.html

[7] What is Silverlight?. *Webopedia* [online]. 2014 [cit. 2014-04-08]. Dostupné z: http://www.webopedia.com/TERM/S/Silverlight.html

[8] PILGRIM, Mark. *HTML5: up and running.* 1st ed. Sebastopol, CA: O'Reilly, 2010. ISBN 05-968-0602-7.

[9] Canvas tutorial. *MDN* [online]. 2014 [cit. 2014-04-11]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Canvas_tutorial

[10] Co je SVG. *SVG tvůrce* [online]. 2013 [cit. 2014-04-11]. Dostupné z: http://svg.jkoweb.cz/coje.xhtml

[11] BUCHALCEVOVÁ, Alena; PAVLÍČKOVÁ, Jarmila; PAVLÍČEK, Luboš. *Základy softwarového inženýrství - materiály ke cvičení.* 1.vyd. Praha : Vysoká škola ekonomická, 2007. 222 s. ISBN 987-80-245-1270-9.

[12] FOWLER, Martin. *UML Distilled: A Brief Guide to the Standard Object Modeling Language.* 3. edition. Addison Wesley, 2003. 208 s. ISBN 0- 321-19368-7.

[13] ŠARMANOVÁ, Jana. *RELAČNÍ DATOVÝ MODEL.* Ostrava, 2007. Dostupné z: http://wiki.cs.vsb.cz/images/2/2a/Inz2.pdf

[14] Wireframes webových stránek. *MediaCentrik* [online]. 2014 [cit. 2014-04-16]. Dostupné z: http://www.mediacentrik.cz/tvorba-webu-na-miru/nabidka-sluzeb/wireframes-webovych-stranek.aspx

[15] Pojem: PHP. *TOVARNA.CZ* [online]. 2013 [cit. 2014-04-13]. Dostupné z: http://www.tovarna.cz/cz/slovnik-pojmu/35-php/

[16] MySQL databáze - český manuál. *Junext* [online]. 2014 [cit. 2014-04-08]. Dostupné z: http://www.junext.net/mysql/

[17] Home. *JQuery* [online]. 2014 [cit. 2014-04-17]. Dostupné z: http://jquery.com/

[18] Home. *TinyMCE* [online]. 2014 [cit. 2014-04-16]. Dostupné z: http://www.tinymce.com/

# Seznam obrázků

# Seznam tabulek

# Seznam příloh

1. CD-ROM: Přiložené médium obsahuje bakalářskou práci v plném znění pod názvem souboru online_system_pro_tvorbu_pojmovych_map.pdf, dále obsahuje tabulku konkurenčních řešení ve formátu XLS pod názvem souboru tabulka_konkurencnich_reseni.xls, součástí jsou také obrázky použité v této písemné části a zdrojové kódy systému Mindlook.

2. Webová stránka: Na adrese www.mindlook.ic.cz se nachází veškeré zdrojové kódy ke stažení.