

Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Ilkovičova 3, 842 16 Bratislava 4

---

# Virtuálna FIIT

Projektová dokumentácia

---

Vedúci projektu: Mgr. Alena Kovárová

Autori: Bc. Filip Hlaváček Bc. Michal Palček  
Bc. Ján Hudec Bc. Rastislav Pečík  
Bc. Pavol Mešťaník Bc. Ivan Polko  
Bc. Matúš Novotný

Ak. rok: 2010/2011

## Obsah

---

Zoznam obrázkov .....	vi
Zadanie .....	viii
1 Úvod.....	ix
2 Analýza predchádzajúceho riešenia.....	1
3 Analýza použitých technológií a nástrojov .....	4
3.1 Prehľad 3D technológií – O3D, WebGL .....	4
3.2 Knižnice pre JavaScript a PHP .....	6
3.3 Použité nástroje - prehliadače WebGL.....	7
3.4 Modelovanie.....	7
4 Analýza údajov v systéme .....	8
4.1 Prístup k údajom.....	8
4.2 Získavanie údajov z AIS.....	9
4.2.1 Programové prihlásenie do AIS .....	10
5 Analýza používateľského rozhrania aplikácie pre prehliadače nepodporujúce WebGL.....	12
5.1 Exteriérové aplikácie .....	12
5.2 Interiérové aplikácie .....	15
5.3 Záver .....	17
6 Špecifikácia riešenia.....	18
6.1 Špecifikácia požiadaviek .....	18
6.2 Charakteristika používateľov systému .....	19
6.2.1 Prípady použitia .....	19
6.2.2 Opis diagramu prípadov použitia .....	20
7 Návrh systému Virtuálnej budovy FIIT .....	26
7.1 Architektúra systému .....	26
7.1.1 Server.....	26
7.1.2 Klient.....	27
7.1.3 Optimalizácia mobilnej verzie aplikácie .....	27
7.2 Návrh GUI aplikácie .....	28
7.2.1 Grafické rozhranie pre 3D verziu.....	28
7.2.2 Grafické rozhranie pre 2D a mobilnú verziu.....	29

7.3	Dátový model .....	29
7.3.1	Opis dátových entít .....	30
8	Implementácia prototypu .....	34
8.1	Priority implementácie .....	34
8.2	3D klient .....	34
8.2.1	Používateľské rozhranie .....	35
8.2.2	Ovládanie aplikácie.....	35
8.2.3	Opis implementácie.....	36
8.2.4	Zhodnotenie výberu technológií .....	37
8.3	Mobilný klient.....	38
8.4	AIS parser.....	40
8.4.1	Popis metód.....	40
8.5	AIS importér .....	41
8.5.1	Opis knižnice.....	41
8.5.2	Zhodnotenie a plány do budúcnosti.....	42
8.6	Komunikačné rozhranie.....	42
8.6.1	Úloha servera pre komunikačné rozhranie .....	42
8.6.2	Zvolené technológie, architektúra návrhu a aktuálny stav prototypu .....	43
8.6.3	Zhodnotenie .....	43
8.7	Úpravy modelu budovy .....	44
8.7.1	Pôvodný stav modelu .....	44
8.7.2	Nevyhnutné úpravy modelu.....	44
8.7.3	Súčasný stav .....	47
8.8	Výškové mapy.....	47
8.9	Zhodnotenie prototypu .....	48
9	Model budovy FIIT .....	50
9.1	Postup vytvárania modelu.....	50
9.2	Opis súborov so skriptami .....	57
9.3	Štruktúra modelov.....	57
9.4	Aktuálny stav a porovnanie s minuloročným tímom .....	60
9.4.1	Možnosti ďalších vylepšení.....	60

10	Implementácia klientskej časti aplikácie .....	62
10.1	Navigácia .....	62
10.1.1	Navigačný algoritmus .....	64
10.2	Optimalizácia .....	65
10.2.1	Textúrovací atlas.....	66
10.2.2	Dvere .....	66
10.2.3	Garbage Collector.....	68
10.2.4	Kompresia modelov.....	69
10.3	Info server.....	70
10.4	GUI a jeho interaktívne prvky.....	70
10.5	Štruktúra používateľského rozhrania .....	73
10.6	Mobilný klient.....	74
11	Overenie funkcionality aplikácie .....	76
11.1	Testovanie 3D klienta .....	76
11.2	Testovanie info servera .....	76
12	Záver .....	78
12.1	Zhodnotenie .....	78
A.	Inštalačná príručka .....	1
A.1	Nainštalovanie súčastí Virtualfiit na webový server (quickstart).....	1
A.1.1	Nasadenie aplikácie server na webový server .....	1
A.1.2	Nasadenie aplikácie client na webový server.....	2
A.1.3	Príklad inštalácie na operačný systém GNU/Linux Debian 6.0 (tutoriál) .....	2
A.2	Rozšírené inštalačné možnosti .....	5
A.2.1	Inštalácia súčastí admin a client na jeden virtuálny web .....	5
A.2.2	Nastavenie prepisovania URL adries (mod_rewrite).....	6
A.2.3	Rozšírené nastavenie URL adries aplikácie.....	6
A.2.4	Nastavenie prístupu do databázy .....	7
A.3	Príprava databázy – používanie admin časti aplikácie .....	7
A.3.1	Nainštalovanie databázy a importovanie údajov z AIS.....	7
A.3.2	Odinštalovanie aplikácie Virtuálna FIIT .....	8
B.	Používateľská príručka.....	1

B.1	Rozloženie ovládacích prvkov.....	1
C.	Textúry.....	1
D.	Popis infoservera.....	1
D.1	Spoločné parametre služieb .....	1
D.1.1	Formát výstupu.....	1
D.2	Informácie o miestnosti.....	1
D.2.1	Základné informácie o miestnosti .....	1
D.3	Informácie o osobách .....	2
D.3.1	Lokalizácia osoby .....	2
D.3.2	Základné informácie o osobe .....	2
D.3.3	Predmety, ktoré daná osoba učí .....	3
D.3.4	Rozvrh danej osoby .....	3
D.4	Vyhľadávanie osôb, miestností, predmetov, alebo všetkého .....	3
D.5	Informácie o predmete.....	4
D.5.1	Rozvrh pre daný predmet.....	4
E.	Formulár pre používateľské testovanie.....	1

## Zoznam obrázkov

---

Obr. 2.1 Artefakt, ktorý vznikol pri vykresľovaní modelu .....	2
Obr. 5.1 Ukážka použitia rámca MapFish.....	13
Obr. 5.2 Rozhranie aplikácie Google Maps .....	13
Obr. 5.3 Rozhranie aplikácie Google Maps v prehliadači Safari na mobilnom zariadení iPod 2G .....	14
Obr. 5.4 Rozhranie aplikácie OpenLayers.....	15
Obr. 5.5 Špecializovaná aplikácie pre Mall of America .....	16
Obr. 5.6 Rozhranie aplikácie PoinInside .....	16
Obr. 5.7 Rozhranie aplikácie Micello.....	17
Obr. 6.1 Diagram prípadov použitia .....	20
Obr. 7.1 Schéma architektúry systému .....	26
Obr. 7.2 Logický dátový model.....	32
Obr. 7.3 Fyzický dátový model .....	33
Obr. 8.1 Rozhranie aplikácie.....	36
Obr. 8.2 Štartovacia stránka mobilnej verzie virtuálnej FIIT .....	38
Obr. 8.3 Zobrazenie plánu novej budovy FIIT v čiastočnom režime. ....	39
Obr. 8.4 Zobrazenie plánu novej budovy FIIT v úplnom režime.....	40
Obr. 8.5 Príklad zle nastaveného pivotu dverí .....	45
Obr. 9.1 Pôdorys poschodia .....	51
Obr. 9.2 Namodelovaná časť základu poschodia .....	51
Obr. 9.3 Výsledný 3D model po spustení skriptu .....	51
Obr. 9.4 Systém exportu modelu poschodia.....	52
Obr. 9.5 Príklad identifikácie miestnosti .....	53
Obr. 9.6 Výšková mapa – farby (steny a dvere) .....	54
Obr. 9.7 Výšková mapa - výška.....	54
Obr. 9.8 Určenie váh dverí .....	56
Obr. 9.9 Výsledná výšková mapa.....	56

Obr. 10.1 Navigačné šípky v modeli .....	62
Obr. 10.2 Doplnková navigačná šípka .....	63
Obr. 10.3 Google Chrome - Speed Tracer .....	65
Obr. 10.4 Textúrovací atlas. ....	66
Obr. 10.5 Príklad opakovania textúry.....	67
Obr. 10.6 Heap Profiler v Google Chrome.....	69
Obr. 10.7 Speed Tracer pre neoptimalizovanú aplikáciu.....	69
Obr. 10.8 Napovedanie pri vyhľadávaní.....	71
Obr. 10.9 Príklad šablóny na vypísanie informácií o osobe.....	71
Obr. 10.10 Šípka ukazujúca prichádzajúci výťah. ....	72
Obr. 10.11 Informačný panel .....	73
Obr. 10.12 Panel nástrojov.....	74
Obr. 0.1 Nástroj QUnit použitý pre testovanie výškových máp. ....	76
Obr. 11.1 Ukážka možnosti použitia avatarov.....	80

## Zadanie

---

Pamätáte si, ako vždy začína každý semester? Zistíte si, aký je váš rozvrh, no v ňom sú záhadne zakódované čísla miestností, v ktorých máte cvičenia. A ako dlho vám trvá, kým nájdete miestnosť, v ktorej sa nachádza váš vedúci? A ako zistíte, kedy má váš prednášajúci konzultačné hodiny? Alebo ktorý cvičiaci má v danej miestnosti cvičenie hneď po vás?

Toto je len pár z mnohých problémov. Ich riešenie momentálne spočíva v tom, že si musíte otvoriť ten správny informačný zdroj a v ňom informáciu nájsť a aj tak vám nakoniec nikto nepovie, kde je miestnosť BX04. Nebolo by to krásne, keby ste jednoducho počítaču zadali číslo miestnosti alebo meno človeka a on vás k nemu virtuálne zaviedol? A čo tak keby to všetko fungovalo, až budeme mať novú budovu FIIT?

Vašou úlohou bude:

- zanalyzovať danú oblasť (napr. aj O3D vs. WebGL)
- vytvoriť skutočný 3D model novej budovy FIIT resp. opraviť existujúci
- navrhnuť a zimplementovať jeho interaktívne časti ako je napríklad funkčný výťah, informačné tabule, možnosť zadania otázky
- to všetko by samozrejme nešlo bez databázy a možnosti importovania do nej z iných existujúcich systémov
- riešenie bude potrebné optimalizovať tak, aby ho používateľ mohol používať aj cez jednoduché webové rozhranie
- na záver nesmie chýbať testovanie a vyhodnotenie použiteľnosti



# 1 Úvod

---

Predkladaný dokument obsahuje projektovú dokumentáciu vytvorenú v rámci predmetu Tímový projekt.

Cieľom tejto dokumentácie je vysvetliť riešenie celého projektu s názvom Virtuálna FIIT. Systém bude slúžiť ako navigácia po novej budove fakulty a poskytovať informácie o jednotlivých miestnostiach.

## Riešenie minuloročného tímu

Naše riešenie vychádza z riešenia minuloročného tímu LOST<sup>1</sup>. Ich výsledkom sú namodelované poschodia 0 až 7 novej budovy FIIT. V ich aplikácii sa nachádzajú 3 režimy: prechádzanie, prehliadanie a navigácia. V režime prechádzania fungujú kolízie so stenami, otváranie dverí a chodenie po schodoch. V režime prehliadania funguje označovanie miestností kliknutím myšou. V režime navigácie je možné nechať si zobrazíť cestu medzi dvoma miestnosťami, ktoré sa dajú vyhľadať podľa názvu. Podľa ich projektovej dokumentácie<sup>2</sup> ďalej uvádzame nasledujúce nespĺnené, resp. nie úplne splnené, požiadavky:

- Nestihli namodelovať najspodnejšie poschodie, ktoré obsahuje prevažne skladové miestnosti.
- 3. a 4. poschodie nie je namodelované, ale iba skopírované 5.
- Nebol implementovaný výťah.
- Nezobrazuje sa aktuálna pozícia používateľa v 3D režime.
- V 3D režime je potrebné ošetriť prechádzania cez zatvorené dvere.
- Pri prechode medzi poschodiami sa niekedy vyskytne chyba.
- Databáza neobsahuje skoro žiadne údaje.

## Prehľad dokumentu

Dokument je rozdelený na niekoľko častí:

1. Úvod
2. Analýza
  - Analýza súčasného stavu riešenia, jeho funkcionality a nedostatkov
  - Analýza použitých technológií, knižníc a vývojových nástrojov
3. Špecifikácia
  - Špecifikácia funkcionálnych a nefunkcionálnych požiadaviek systému
4. Návrh
  - Návrh architektúry systému
  - Návrh dátového modelu
  - Návrh GUI systému
  - Návrh 2D a mobilnej verzie systému
5. Implementácia
  - Implementácia prototypu

---

<sup>1</sup> <http://labss2.fiit.stuba.sk/TeamProject/2009/team03is-si/>

<sup>2</sup> [http://labss2.fiit.stuba.sk/TeamProject/2009/team03is-si/docs/Projektova\\_dokumentacia\\_final.pdf](http://labss2.fiit.stuba.sk/TeamProject/2009/team03is-si/docs/Projektova_dokumentacia_final.pdf)

## 2 Analýza predchádzajúceho riešenia

---

### Detekcia kolízií

V analyzovanom riešení sú na detekciu kolízií použité výškové mapy. Princíp je taký, že na obrázku sú zakreslené červenou farbou miesta, kam sa používateľ pri prechádzaní modelu dostať nemôže a farbami od čiernej po bielu sú znázornené miesta, po ktorých sa pohybovať môže. Rôzne odtiene farby znázorňujú rôznu výšku podlahy. Dvere v mape vôbec nie sú zaznačené. Modrou farbou je znázornené miesto prechodu na vyššie poschodie.

Takýchto obrázkov je spolu 8. Problémom je ich veľkosť. Sú vo formáte PNG a spolu majú veľkosť 1,89 MB. Ich rozmery sú okolo 5000\*1000 pixelov, čo pri načítaní do pamäte, ktoré je nevyhnutné na zisťovanie hodnôt jednotlivých pixelov, zaberie okolo 220MB (testované na prehliadači Chrome, zobrazená stránka, ktorá obsahuje iba tieto obrázky). Vhodné by bolo vyskúšať zmenšiť rozmer kolíznych máp, aby sme optimalizovali pamäťovú náročnosť, ale tak, aby to používateľ nepostrehol pri pohybe po modeli, napr. zaseknutím sa o stenu.

Možnosti na zobrazovanie miestnosti, v ktorej sa používateľ nachádza v 3D režime, sú nasledovné:

- Zafarbiť každú miestnosť inou farbou, a v tabuľke si udržiavať záznamy, ktorá miestnosť patrí ku ktorej farbe. Vytváranie farebnej mapy pre všetky poschodia je však časovo zdĺhavý proces. Nevýhodou je tiež pamäťová náročnosť. Výhodou je jeho rýchlosť.
- Využiť funkciu „picking“, ktorú poskytuje aj O3D. Vieme takto zistiť, aký objekt sa nachádza na danom bode obrazovky. Vo všeobecnosti vráti prvý objekt, ktorého polygóny sa pretínajú s priamkou určenou bodom obrazovky. Ak by sme namiesto toho použili priamku, ktorá smeruje do podlahy, vedeli by sme určiť v ktorej miestnosti sa používateľ nachádza. Hľadanie priesečníka s polygónmi je však výpočtovo náročné, a muselo by prebiehať pri každom vykreslení, aby informácia o miestnosti bola aktuálna. V súčasnom riešení sa táto funkcia využíva na označovanie miestností v prehliadacom režime.

Z hľadiska rýchlosti navrhujeme teda využiť už existujúcu výškovú mapu, zafarbiť miestnosti a vytvoriť mapovaciu tabuľku. Aby sme vedeli zaznačiť rôznu výšku podlahy v rôzne zafarbených miestnostiach, za predpokladu súboru s formátom PNG s farbou určenou 24 bitmi, využili by sme 16 bitov na identifikáciu miestnosti a 8 bitov na identifikáciu výšky podlahy.

### Interaktivita – otváranie dverí

Otváranie dverí je v súčasnosti riešené pomocou funkcie „picking“. Je to výpočtovo náročné, preto je možnosť riešiť otváranie dverí pomocou klávesy, ako je to v mnohých počítačových hrách. Pri stlačení klávesy by sa zistili najbližšie dvere k používateľovi, ak by boli dostatočne blízko, tak by sa otvorili, resp. zatvorili. V blízkosti dverí by sa tiež zobrazila informácia pre používateľa, ktorým klávesom môže dvere otvoriť.

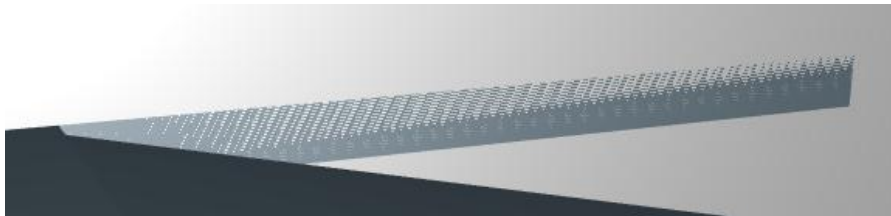
Aby nešlo prechádzať cez zatvorené dvere, bolo by vhodné zakomponovať ich do výškovej mapy, aby sme vedeli zistiť, že ideme prejsť cez dvere. Na zjednodušenie vytvárania výškovej mapy navrhujeme všetky dvere označiť jednou farbou. Ak pri prechádzaní modelom narazí používateľ na danú farbu, v zozname dverí budú nájdené dvere na danom poschodí, ktoré sú k nemu najbližšie. Podľa toho, či sú otvorené, bude dovolené používateľovi cez ne prejsť.

### 3D model

Model minuloročného tímu budeme musieť aktualizovať, tak aby zodpovedal najnovším dostupným plánom. Model je namodelovaný v programe 3ds max 2010 a je rozdelený do súborov podľa poschodí. Model poschodia sa skladá z jednotlivých miestností, dverí, skiel a ďalších komponentov.

Jednotlivé miestnosti musia byť v modeli vyčlenené ako samostatné pomenované objekty kvôli tomu, aby sa dala použiť funkcia picking, ktorá vráti objekt – miestnosť na ktorú používateľ klikol. Rozdelenie modelu do samostatných objektov však spomaľuje vykresľovanie. Súčasný riešenie ide plynulo, pri predbežných testoch s technológiou WebGL sme však zistili, že pri takomto rozdelení, je zobrazovanie modelu pomalšie. Pri spojení všetkých objektov do jedného, sa však model zobrazoval úplne plynulo. Pri spojení všetkých objektov, však strácame možnosť zvýrazňovať jednotlivé miestnosti. Vyriešiť by sa to dalo napr. zobrazovaním farebného priehľadného polygónu ako stropu miestnosti, čo by bol na pohľad rovnaký efekt ako zafarbenie celej miestnosti rovnakou farbou.

V modeli sú niektoré časti nesprávne namodelované, niektoré polygóny sa prekrývajú a vznikajú potom artefakty, ako je to ukázané na Obr. 2.1.



Obr. 2.1 Artefakt, ktorý vznikol pri vykresľovaní modelu

### Potreba 2D a mobilného rozhrania

V súčasnosti oficiálne podporujú technológiu WebGL len vývojové verzie troch prehliadačov. Naším cieľom je však umožniť čo najväčšej skupine ľudí využiť našu aplikáciu a preto sme sa rozhodli rozšíriť pôvodne len 3D verziu Virtuálnej FIIT aj o 2D verziu, ktorá by fungovala aj na prehliadačoch nepodporujúcich WebGL. Ďalej sme predpokladali, že každý novo nastupujúci poslucháč vlastní mobilný telefón a to nás viedlo k myšlienke rozšíriť 2D verziu aj pre mobilné zariadenia. Hlavnou myšlienkou je jednoduchá a nenáročná navigácia po novej budove FIIT prostredníctvom mobilného telefónu.

### Navigácia

Súčasťou analyzovaného riešenia je aj navigácia. Slúži na nájdenie najkratšej cesty medzi dvoma miestnosťami. Uskutočňuje sa to pomocou grafu, ktorého uzly reprezentujú miestnosti. Tie sú poprepájané ohodnotenými cestami, ktoré reprezentujú vzdialenosti medzi jednotlivými miestnosťami. Ďalej obsahuje pomocné uzly nachádzajúce sa na možných križovatkách v budove. Graf je uložený v XML súbore.

V dokumentácii minuloročného riešenia je uvedené, že váhy hrán medzi susednými miestnosťami, medzi ktorými existuje okrem štandardných priechodov na chodbu navyše aj ďalší priechod, boli zvýšené tak, aby nedochádzalo k nekorektnému skracovaniu cesty cez susednú miestnosť. Po otestovaní vyhľadania takejto cesty sme však zistili, že nie všetky váhy takýchto hrán, kde mohlo dochádzať k nájdeniu nesprávnej cesty, boli správne prehodnotené. Preto bude potrebné upraviť niektoré váhy ciest v XML súbore.

Hľadanie najkratšej cesty je realizované pomocou Dijkstrovho algoritmu. Jedná sa o štandardný známy algoritmus, ktorý je pre naše potreby dostatočne efektívny.

## 3 Analýza použitých technológií a nástrojov

---

### 3.1 Prehľad 3D technológií – O3D, WebGL

#### Technológia O3D

Minuloročný tím implementoval virtuálny model pomocou O3D zásuvný modul od spoločnosti Google, ktorý umožňuje zobrazovať 3D grafiku v internetovom prehliadači. S týmto riešením je spojených niekoľko problémov:

- O3D ako zásuvný modul je už ďalej nevyvíjaná technológia<sup>3</sup>. Namiesto toho Google prerobil technológiu O3D tak, aby fungovala ako knižnica nad WebGL.
- Nevyhnutnosť nainštalovať zásuvný modul do prehliadača. Používateľ nemá vždy možnosť nainštalovať zásuvný modul na počítači, za ktorým sa práve nachádza.

#### Technológia WebGL

Oproti minuloročnému tímu sme v inej situácii, pretože technológia WebGL je už použiteľná, čo dokazuje aj množstvo ukážok dostupných na internete. WebGL vychádza zo štandardu pre zobrazovanie 3D grafiky OpenGL ES 2.0<sup>4</sup>. V prehliadačoch, ktoré ho budú podporovať, nebude vyžadovať žiadny zásuvný modul, čo je veľkou výhodou. Zobrazená 3D scéna, je normálny HTML element a nad jeho plochu je možné umiestňovať ďalšie HTML elementy, slúžiace napr. na zobrazovanie ovládacích prvkov, alebo zobrazovanie informácií o označenej miestnosti.

Ešte nie je vydaný prehliadač s podporou WebGL, a musíme tak pracovať s vývojovými verziami Mozilla Firefox-u - verzia 4<sup>5</sup> a Google Chrome – verzia Canary<sup>6</sup>. Firefox 4 by však mal výjsť na začiatku roka 2011, teda pri dokončení tímového projektu by už mal byť dostupný aspoň jeden prehliadač, v ktorom bude WebGL fungovať.

Samotné WebGL pracuje na pomerne nízkej úrovni. Aby sme urýchlili a uľahčili vývoj, rozhodli sme sa, že použijeme niektorú z nasledujúcich knižníc, ktoré sme preskúmali.

#### GLGE

GLGE je knižnica založená na JavaScript-e zjednodušujúca prácu s WebGL. Hlavným cieľom tvorby GLGE je rýchla práca s WebGL, čím sa vývojár môže sústrediť na bohatší obsah svojich stránok. Daná knižnica v súčasnosti ponúka širokú funkcionálnosť. Súčasná verzia knižnice je v. 0.5.2, čo nám hovorí o jej neúplnosti, no neustále sa rozširuje a nadobúda nové funkcie. Poslednou pridanou funkciou danej knižnice bol „level of details“, pričom autor danej knižnice na svojej stránke<sup>7</sup> uvádza funkcie, ktoré plánuje v blízkej budúcnosti doplniť. Daná knižnica vie priamo pracovať s grafickým formátom Collada a podporuje animácie.

---

<sup>3</sup> <http://code.google.com/apis/o3d/>

<sup>4</sup> <https://cvs.khronos.org/svn/repos/registry/trunk/public/webgl/doc/spec/WebGL-spec.html>

<sup>5</sup> Ďalej len Firefox 4.

<sup>6</sup> Ďalej len Chrome.

<sup>7</sup> <http://www.glge.org/>

### **C3DL**

C3DL8 celým názvom The Canvas 3D JS Library, je ďalšou JavaScript knižnicou. Skoršie verzie využívali predchodcu WebGL (experimentálnu technológiu Canvas 3D, z ktorej sa neskôr vyvinulo WebGL), no od verzie 2.0 využíva technológiu WebGL. V súčasnosti sa web stránka danej knižnice prerába, a prístup k niektorým informáciám je neúplný. Ku knižnici je spravená jednoduchá webová aplikácia na prácu s 3D modelmi. Medzi klady danej knižnice patria súbory s matematikou na podporu 3D grafiky. C3DL je schopná importovať grafický formát Collada a samozrejmosťou sú animácie.

### **Copperlicht**

Knižnica Copperlicht<sup>9</sup> je založená na JavaScript-e. Ponúka veľké množstvo podporovaných grafických formátov. Má veľmi rýchle prekršľovanie 3D modelov. K danej knižnici je vytvorený aj editor (CopperCube), ktorý je potrebný na načítanie podporovaných grafických formátov. Na rozdiel od samotnej knižnice je spoplatnený. Táto knižnica už obsahuje pomerne rozsiahlu verziu 1.2.3 (detekcia kolízií, fyzika v modeloch), ktorú pravidelne rozširujú a fixujú staršie chyby. Základnou výhodou je dostatok informácií v podobe dobrej dokumentácie.

### **SceneJS**

SceneJS<sup>10</sup> je ďalšou „mladou“ knižnicou založenou na JavaScript-e poskytujúcou flexibilné API založené na JSON (*JavaScript Object Notation*). 3D modely vie zmeniť na JSON objekty, s ktorými je schopná pracovať. Súčasná verzia danej knižnice je v.0.7.8 beta. Daná knižnica zatiaľ nie je funkčnosťou porovnateľná s predchádzajúcimi. Dokumentácia k danej knižnici je zatiaľ veľmi skromná.

### **SpiderGL**

SpiderGL<sup>11</sup> je knižnica JavaScript-u, ktorá sľubujúca veľmi rýchle prekršľovanie. Súčasne dostupná verzia je len v.0.1.1.20100129. Táto verzia ponúka: matematické funkcie, nízku/vysokú úroveň lineárnej algebry funkcií a tried, geometrické štruktúry a triedy orientované na modelovanie miest, priame a nepriame prístupy k prekršľovaniu vo WebGL, používateľské rozhranie, správu interakcií a asynchrónne načítavanie obsahu. Dokumentácia je pomerne rozsiahla a kompletná.

### **GwtGL**

Technológia GwtGL<sup>12</sup> umožňuje vyvíjať web stránky a web aplikácie s 3D obsahom pomocou programovacieho jazyka Java, namiesto JavaScript-u. Nadväzuje na technológiu GWT, ktorá sa používa na budovanie komplexných web aplikácií. Na domovskej stránke tejto knižnice je uvedené, že GWT kompilátor generuje optimalizovaný JavaScript kód, čím prispieva k vyššej rýchlosti. Posledná vydaná verzia je z 15. februára 2010, vzhľadom na to, že odvtedy sa WebGL špecifikácia zmenila, a preto demá na stránke nie sú funkčné vo vývojovej verzii Firefox-u 4, je táto technológia pre nás nepoužiteľná.

### **O3D-WebGL**

Prepísaním technológie O3D, tak aby fungovala nad WebGL vznikla knižnica O3D-WebGL. Niektoré vlastnosti technológie O3D však neboli prenesené do O3D-WebGL:

---

<sup>8</sup> <http://www.c3dl.org/index.php/development-news/>

<sup>9</sup> <http://www.ambiera.com/copperlicht/>

<sup>10</sup> <http://scenejs.org/>

<sup>11</sup> <http://spidergl.org/>

<sup>12</sup> <http://code.google.com/p/gwtgl/>

- Animácie
- Vytváranie vlastných textúr, čo je užitočné, napr. pri dynamickom vytváraní nápisov v modeli.
- Modely nie sú prenášané v skomprimovanom tvare, a teda sú väčšie.

Na stránke O3D je uvedený návod<sup>13</sup>, podľa ktorého sme sa pokúšali prekonvertovať riešenie minuloročného tímu do O3D-WebGL, čo sa nám však nepodarilo.

### Výsledok porovnania

Knižnica CopperLicht vyšla z porovnania najlepšie. Má stabilnú verziu, kvalitné návody, kvalitnú dokumentáciu a pri prípadných problémoch je možné na jej fóre získať rýchle odpovede.

Ku knižnici CopperLicht však budeme musieť implementovať načítavanie 3D modelu, pretože priamo CopperLicht nepodporuje žiaden formát pre 3d modely (len cez platený editor). Z programu Autodesk 3ds max-u využijeme export do formátu Collada, pretože je to prehľadný XML súbor. Optimalizáciu formátu Collada nebudeme robiť, pretože efektívnejšie je nechať súbor s modelom skomprimovať web serverom kompresiou GZIP<sup>14</sup>. Jedinú konverziu, ktorú vykonávame, je prevod z Collady ako XML súboru do formátu JSON, čo je vlastne objekt v JavaScripte. Klient tak nemusí parsovať XML a načítanie súboru je tak jednoduchšie.

Porovnanie veľkostí súborov pre model 6. poschodia:

- Collada (XML) – 1415 KB
- Collada skonvertovaná do formátu JSON – 1177 KB
- JSON skomprimovaný cez kompresiu GZIP – 102 KB

## 3.2 Knižnice pre JavaScript a PHP

### JavaScript

Minuloročný tím využil knižnicu JQuery<sup>15</sup> na napovedanie pri vyhľadávaní miestností. Je to najpopulárnejšia knižnica<sup>16</sup> pre JavaScript. JQuery poskytuje prostriedky pre prácu s DOM (Document Object Model), umožňuje dynamicky meniť štruktúru stránky, ako aj meniť štýly použité na stránke. Veľmi dôležitý je tiež fakt, že knižnica JQuery skrýva odlišnosti a chyby v rôznych prehliadačoch, čím urýchľuje a zjednodušuje vývoj. Využijeme ju preto aj v našom projekte. Okrem napovedania JQuery použijeme aj na rôzne vizuálne efekty, ako sú vysúvacie panely alebo animované zobrazovanie informácií a pod. Tieto vizuálne efekty sa budú môcť odohrávať aj pred zobrazenou 3D scénou, čo je jedna z už spomenutých výhod WebGL.

### PHP

Pri súčasne plánovanom rozsahu rámec pre PHP ani nie je potrebný, keďže jediné, čo bude naprogramované v PHP, je výber z databázy a konverzia vybraných informácií do vhodného formátu pre stránku. Ak však budeme uvažovať do budúcnosti, že by mohlo pribudnúť administratívne rozhranie, vtedy by už bolo vhodné nejaký rámec využiť.

<sup>13</sup> <http://code.google.com/p/o3d/wiki/GettingStarted>

<sup>14</sup> [http://httpd.apache.org/docs/2.0/mod/mod\\_deflate.html](http://httpd.apache.org/docs/2.0/mod/mod_deflate.html)

<sup>15</sup> <http://jquery.com/>

<sup>16</sup> <http://trends.builtwith.com/javascript>

Uvažovali sme nasledujúce technológie (vylúčili sme CMS, ktoré nehľadáme, knižnicu ezComponents, ktorá nepodporuje AJAX a knižnicu Fusebox, ktorá už nie je vyvíjaná):<sup>17</sup>

- CakePHP
- CodeIgniter
- Qcodo
- Symfony
- Yii
- Zend Framework

Vybrali sme si rámec CodeIgniter<sup>18</sup>, pretože ako uvádzajú na stránke, je jednoduchý a rýchly, čo je na naše použitie ideálne, ďalej je dobre zdokumentovaný a jeden člen tímu už s ním má skúsenosti.

### 3.3 Použité nástroje - prehliadače WebGL

V tejto sekcii sú opísané nástroje, ktoré použijeme pri riešení projektu.

Prehliadače, ktoré podporujú WebGL, sú beta verzie Firefox-u 4, resp. jeho vývojové verzie s názvom Minefield, vývojové verzie Safari na Mac OS X a vývojové verzie Chrome - Chrome Canary<sup>19</sup>. Vzhľadom na to, že nikto z tímu nemá k dispozícii Mac, budeme 3D rozhranie skúšať na prehliadačoch Firefox 4 a Chrome Canary. Prednostne sa budeme orientovať na prehliadač Firefox 4, pretože očakávame, že sa bude najrýchlejšie rozširovať spomedzi prehliadačov podporujúcich WebGL.

#### Firefox 4

Vo Firefox-e 4 využijeme rovnako ako minuloročný tím plug-in Firebug, ktorý nám umožní ladiť kód v JavaScript-e, ako aj prehľadne zobrazovať jednotlivé HTTP požiadavky, čo určite využijeme pri optimalizácii.

#### Chrome Canary

Prehliadač Chrome Canary je vývojová verzia prehliadača Chrome. Poskytuje zabudované nástroje pre vývojárov, s podobnou funkcionalitou ako Firebug.

### 3.4 Modelovanie

Minuloročný tím vytvoril svoj model v nástroji Autodesk 3ds max 2010, preto ho budeme používať aj my. V súčasnosti je najnovšia verzia 2011, ktorá sa dá stiahnuť a vyskúšať po dobu 30 dní. Pre študentov však ponúkajú možnosť zaregistrovať sa a stiahnuť si plnú verziu softvéru zadarmo.<sup>20</sup>

---

<sup>17</sup> [http://en.wikipedia.org/wiki/Comparison\\_of\\_web\\_application\\_frameworks#PHP\\_2](http://en.wikipedia.org/wiki/Comparison_of_web_application_frameworks#PHP_2)

<sup>18</sup> <http://codeigniter.com/>

<sup>19</sup> [http://www.khronos.org/webgl/wiki/Getting\\_a\\_WebGL\\_Implementation](http://www.khronos.org/webgl/wiki/Getting_a_WebGL_Implementation)

<sup>20</sup> <http://students.autodesk.com/?nd=register&und=624>



## 4 Analýza údajov v systéme

---

V rámci projektu bude potrebné ukladať a zobrazovať údaje v rôznej forme. Okrem samotného zobrazovania 3D, alebo 2D modelu novej budovy FIIT, prípadne aj načítavania iných súborov s údajmi chceme, aby Virtuálna FIIT slúžila aj ako vhodný informačný zdroj. V univerzitnom prostredí môžeme uvažovať o dvoch skupinách používateľov. Prvou skupinou sú študenti a pracovníci fakulty. Pre týchto sú relevantné najmä informácie o mapovaní rozvrhov na miestnosti a pracovniach jednotlivých zamestnancov fakulty. Druhou kategóriou sú návštevníci fakulty, pre ktorých sú dôležité informácie, ako názvy miestností, ich pozícia a opäť pozície pracovní zamestnancov fakulty.

Z tohto je možné vidieť, že v rámci nášho projektu budeme potrebovať získavať a zobrazovať nasledujúce informácie.

- názvy a umiestnenia miestností
- zoznam predmetov
- rozvrhové akcie v jednotlivých miestnostiach
- údaje o zamestnancoch (pracovňa, telefón, email, ...)

Prístup k týmto informáciám opisujeme v nasledujúcej časti.

### 4.1 Prístup k údajom

Všetky relevantné informácie potrebné pre náš projekt sa nachádzajú v databáze Akademického informačného systému Slovenskej technickej univerzity skratene AIS STU<sup>21</sup>. Prístup k týmto informáciám je možný troma spôsobmi.

- priamy prístup do databázy AIS
- prístup do testovacej databázy AIS obsahujúcej vybrané pohľady na tabuľky
- parsovanie údajov z webovej stránky AIS

Najvhodnejší by bol prvý spôsob prístupu, teda priamy prístup do skutočnej databázy AIS, keďže by sme mohli pracovať rýchlo a efektívne s aktuálnymi údajmi. Nevýhodou a hlavným problémom je, že toto nie je z administratívneho dôvodu možné, keďže databáza obsahuje citlivé osobné údaje, ku ktorým nemôžeme mať prístup. Z tohto dôvodu by bolo nutné pre projekt virtuálnej FIIT vytvárať špeciálne pohľady, ktoré by obmedzovali prístup k informáciám, čo ale nie je možné. Teda prístup k skutočnej databáze AIS nie je pre tento projekt možný najmä z administratívnych dôvodov.

Druhou možnosťou je prístup k testovacej databáze AIS. V tejto testovacej databáze sa nachádzajú vybrané pohľady a kópie údajov zo skutočnej databázy a teda by mohla slúžiť ako vhodná náhrada za reálnu databázu. Nevýhodou je komplikovaný prístup, keďže táto testovacia databáza je v súčasnosti prístupná len z počítačov zamestnancov fakulty. Druhým problémom je nekompletnosť údajov. V tejto testovacej databáze sa nachádzajú len vybrané údaje a chýbajú v nej niektoré informácie, ktoré chceme v našom projekte zobrazovať. Tretím a jedným z najväznejších údajov je ale neaktuálnosť týchto údajov. V tejto testovacej databáze sa nachádzajú údaje staré jeden až dva roky. Tento prístup z praktických dôvodov nie je možný, keďže pre potreby projektu potrebujeme údaje, ktoré je možné aktualizovať minimálne v rozsahu mesiacov prípadne aj kratšom.

---

<sup>21</sup> <http://is.stuba.sk>

Poslednou možnosťou získania údajov je parsovanie údajov priamo zo stránky AIS. Tento spôsob získavania údajov síce nie je najrýchlejší, ani najvhodnejší, ale v súčasnosti sú v rámci AIS verejne prístupné všetky údaje potrebné pre náš projekt. Tieto údaje sú aktualizované okamžite pri každej zmene a sú prístupné z ľubovoľného miesta s pripojením k internetu. Preto sme sa rozhodli využiť tento spôsob prístupu. Podrobnejší opis prístupu k údajom sa nachádza v ďalšej časti.

## 4.2 Získavanie údajov z AIS

Základné informácie potrebné pre projekt sa dajú získať v podobe zdrojových kódov stránok z AIS STU. Ide konkrétne o stránku s informáciami o zamestnancoch fakulty<sup>22</sup> a stránku s informáciami o rozvrhových akciách<sup>23</sup>, kde je nutné ešte zvoliť výber zobrazenia zoznamu rozvrhových akcií. Obe stránky obsahujú informácie vo forme tabuliek.

Prvá z týchto stránok obsahuje údaje o zamestnancoch fakulty a externých zamestnancoch fakulty, konkrétne ide o tieto:

- meno
- pracovisko
- kancelária
- telefón
- e-mail

Druhá tabuľka obsahuje údaje o rozvrhových akciách:

- deň
- začiatok rozvrhovej akcie
- koniec rozvrhovej akcie
- predmet
- typ akcie
- miestnosť
- vyučujúceho
- obmedzenia
- poznámku k rozvrhovej akcii

Tieto dve tabuľky poskytujú väčšinu podstatných údajov. Ďalšie konkrétnejšie údaje o zamestnancoch by bolo síce možné získať, ale nebolo by praktické ich všetky ukladať opätovne do našej databázy, keď už sú verejne prístupné v AIS. Príkladom takýchto informácií, ktoré sú často potrebné sú konzultačné hodiny. Tie sa nachádzajú v profiloch jednotlivých zamestnancov, ktoré si však spravujú zamestnanci jednotlivo a nie sú preto v rovnakej forme a neobsahujú všetky informácie. Preto takéto údaje zobrazované v profiloch zamestnancov bude efektívnejšie nezobrazovať priamo, ale len poskytnúť odkaz do AIS.

Samotný proces získavania údajov z AIS je možné realizovať prostredníctvom parsovania zdrojového kódu stránok AIS. Toto parsovanie bude vykonávať program, respektíve skript, ktorý bude v pravidelných intervaloch aktualizovať všetky údaje v databáze. Ako efektívne riešenie bolo uvažované použitie programu v jazyku Java a použitie PHP skriptu. Oba tieto prístupy by boli

<sup>22</sup> <http://is.stuba.sk/pracoviste/zamestnanci.pl?id=70>

<sup>23</sup> [http://is.stuba.sk/katalog/rozvrhy\\_view.pl?konf=1;f=70](http://is.stuba.sk/katalog/rozvrhy_view.pl?konf=1;f=70)

efektívne a funkčné. Ako vhodnejšie sa ale javí použitie PHP najmä z dôvodu, že PHP bude potrebné pre realizáciu iných častí projektu a preto nebude nutná inštalácia ďalších komponentov. Avšak použitie programu v jazyku Java by si vyžadovalo inštaláciu Java Virtual Machine, čo by bola ďalšia dodatočná požiadavka nášho systému. Preto je teda vhodnejšie použiť PHP, pomocou ktorého je možné získať stránku z AIS s potrebnými údajmi. Zo zdrojového kódu tejto stránky je ďalej možné vyparsovať údaje a tieto následne uložiť do vybraného databázového systému

#### 4.2.1 Programové prihlásenie do AIS

Univerzitný informačný systém obsahuje informácie, ktoré by sme mali záujem ďalej spracovávať. Dostupnosť informácií je rozdelená na 2 skupiny:

- všeobecné voľne dostupné informácie,
- informácie dostupné len po prihlásení (overení identity).

Zo všeobecných informácií by sme mali záujem o:

- rozvrh fakulty,
- informácie o miestnostiach,
- informácie o ľuďoch na STU.

Z chránených informácií máme záujem o:

- rozvrh osoby.

Do univerzitného informačného systému sa prístupuje prostredníctvom internetu použitím štandardného internetového prehliadača. Zo znalosti HTTP protokolu predpokladáme, že server UIS by nám bol schopný odoslať nami požadované informácie po zaslaní vhodných HTTP hlavičiek. Odpoveď by nám vrátil vo forme HTML dokumentu, z ktorého by sme si museli získať údaje vyparsovať. Pre overenie nášho predpokladu bolo nutné vykonať analýzu spojenia s UIS.

Analýzu spojenia sme vykonali pomocou internetového prehliadača Mozilla Firefox s nainštalovaným doplnkom Firebug a Firecookie, ktoré umožňujú sledovať odosielanie požiadaviek a prijímanie odpovedí servera. Opakovane sme sa pokúšali o zobrazenie nami požadovaných informácií a o prihlásenie do UIS. Sledovali sme odosielené požiadavky prehliadača a vrátené odpovede servera. Tým sme si overili náš predpoklad a zistili sme, aké HTTP hlavičky žiada internetový prehliadač pre získanie nami potrebných informácií:

#### HTTP hlavičky pre vyžiadanie rozvrhu vo forme zoznamu

```
POST /katalog/rozvrhy_view.pl HTTP/1.1
Host: is.stuba.sk
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; sk; rv:1.9.2.12)
Gecko/20101026 Firefox/3.6.12
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: sk,cs;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Cache-Control: max-age=0
Content-Type: application/x-www-form-urlencoded
Content-Length: 167
```

z=0&k=0&f=70&studijni\_zpet=0&rozvrh=687&mistnost=0&garant=0&ucitel=0&predmet=0&ustav=0&den=0&stupen=0&program=0&obor=0&rocnik=0&skupina=0&format=list&zobraz=Zobraz%BB

### HTTP hlavičky pre zobrazenie informácie o miestnosti z rozvrhu

```
GET /mistnosti/?zobrazit_mistnost=<id_miestnossti> HTTP/1.1
Host: is.stuba.sk
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; sk; rv:1.9.2.12)
Gecko/20101026 Firefox/3.6.12
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: sk,cs;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Referer: http://is.stuba.sk/katalog/rozvrhy_view.pl
```

### HTTP hlavičky pre zobrazenie informácií o osobe na STU

```
GET /lide/clovek.pl?id=<id_osoby>; HTTP/1.1
Host: is.stuba.sk
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; sk; rv:1.9.2.12)
Gecko/20101026 Firefox/3.6.12
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: sk,cs;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
```

Pre získanie informácií, ktoré sú dostupné len po prihlásení sme zistili, že server pre overenie identity využíva metódu HTTP Authentication a nepoužíva cookies.

### HTTP hlavičky pre zobrazenie rozvrhu osoby

```
POST /auth/katalog/rozvrhy_view.pl HTTP/1.1
Host: is.stuba.sk
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; sk; rv:1.9.2.12)
Gecko/20101026 Firefox/3.6.12
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: sk,cs;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Referer: https://is.stuba.sk/auth/katalog/rozvrhy_view.pl?_m=117
Authorization: Basic eHBhbGNlazpxbzhlcDNNUA==
Cache-Control: max-age=0
Content-Type: application/x-www-form-urlencoded
Content-Length: 77
```

osobni=1&z=1&k=1&f=0&studijni\_zpet=0&rozvrh=<id\_z\_rozcestnika\_rozvrhov>&format=list&zobraz=Zobraz%BB

## 5 Analýza používateľského rozhrania aplikácie pre prehliadače nepodporujúce WebGL

---

Aj keď vývoj postupuje dopredu míľovými krokmi a WebGL by sa mal v skorej budúcnosti stať samozrejmou a byť dostupný v každom prehliadači, nie všetci držia krok s dobou. Naším cieľom je osloviť každého a ponúknuť možnosť dostať sa do cieľa komukoľvek, bez rozdielu aké zariadenie používa (samozrejme s obmedzeniami). Či už ide o 6 ročný kancelársky počítač, alebo mobilný telefón s možnosťou prehliadania internetu, ponúkneme možnosť aj týmto používateľom využívať naše služby. Pozrime sa teda na existujúce riešenia, ktoré by sme mohli využiť pri našom riešení.

Hlavnými kritériami sú kompatibilita so staršími prehliadačmi a možnosť spustenia bez potreby inštalácie rôznych prídavných zásuvných modulov. Naším cieľom je používateľom umožniť zorientovať sa v budove našej školy a preto nás zaujímajú existujúce riešenia, ktoré sa zaoberajú prezentáciou plánov, prípadne máp a navigáciou. Navigácia po budovách je momentálne „v plienkach“, v porovnaní s geografickými systémami riešiacimi prezentáciu máp, prípadne navigáciu. Nasleduje porovnanie najrelevantnejších riešení, ktoré môžeme rozdeliť na aplikácie zaoberajúce sa priamo navigáciou po budovách a na aplikácie riešiace navigáciu mimo budov. Druhé delenie, ktoré nás zaujíma je vhodnosť použitia pre počítače a mobilné zariadenia.

### 5.1 Exteriérové aplikácie

Táto oblasť je v dnešnej dobe veľmi rozšírená a dostupná takmer každému. Existuje množstvo riešení, ktoré ponúkajú zobrazenie geografických máp s množstvom rôznych detailov a rozširujúcich informácií. Množstvo z nich ponúka aj možnosť vyhľadávania a navigácie. Väčšina z nás sa už navštívila mapy na portáloch:

- <http://mapy.atlas.sk/>
- <http://mapy.zoznam.sk/>
- <http://maps.google.com/>

Obdobných aplikácií existujú kvantá. Všetky pracujú na obdobnom princípe a aj ich grafické používateľské rozhrania sú takmer identické. Aplikácie tejto skupiny nie sú stavané na navigáciu po budovách a použitie akejkoľvek pre náš projekt by znamenalo dôkladne preštudovanie existujúcej aplikácie a následne zásahy do zdrojového kódu.

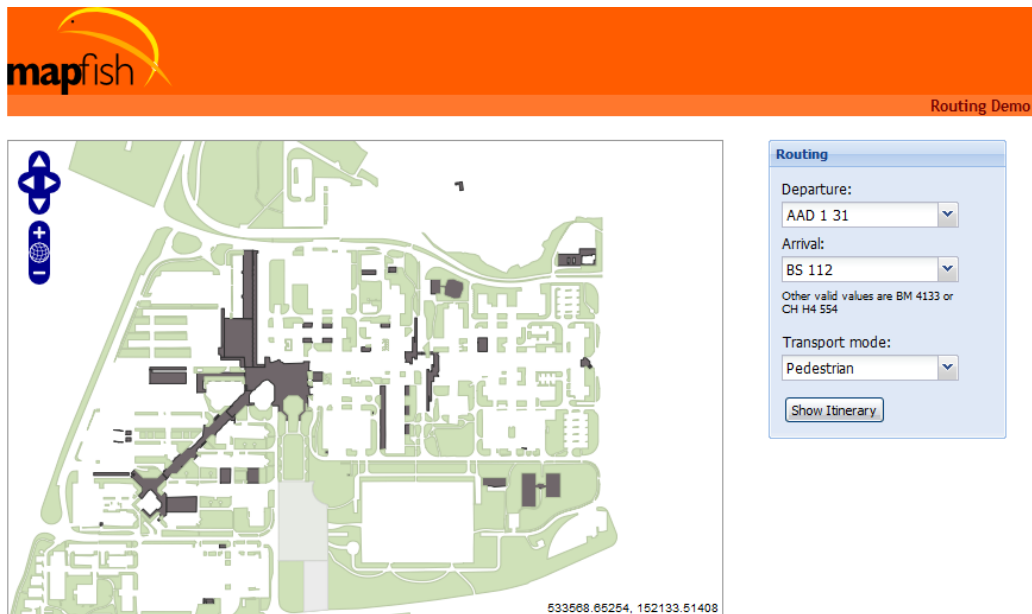
#### MapFish

MapFish<sup>24</sup> je flexibilný rámec na budovanie mapovacích webových aplikácií. Je budovaný na webovom rámci Pylons. Obohacuje ho o geo-priestorovú funkcionálnosť. V príkladoch uplatnenia tohto rámca je aj ukážka mapujúca vysokoškolské priestory. Pri skúmaní tejto aplikácie sme však narazili na problémy, napríklad kompatibility s webovými prehliadačmi. Na Obr. 5.1 je možné vidieť spomínanú ukážku.

Spomínaná ukážka by mala podporovať navigáciu (ako je možné vidieť z ponúkaného GUI), avšak tá nám vždy hlásila chybu, preto sme túto funkcionálnosť nevedeli jednoducho otestovať.

---

<sup>24</sup> <http://mapfish.org/>

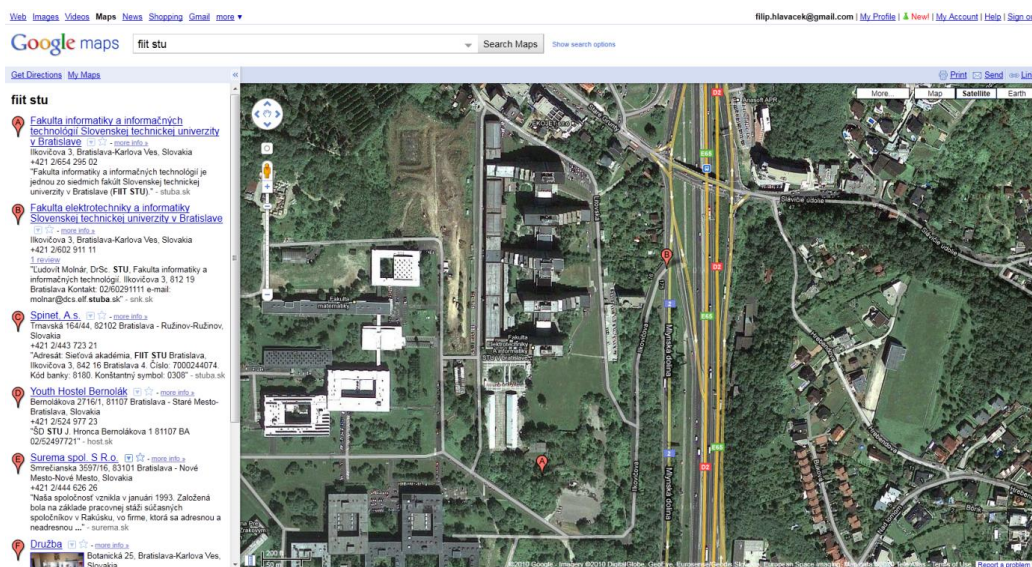


Obr. 5.1 Ukážka použitia rámca MapFish

## Google Maps

Spoločnosť Google poskytuje Google Maps<sup>25</sup> API (application programming interface) umožňujúce použitie máp na stránkach a v aplikáciách tretích strán. Google Maps sú takmer celé naprogramované použitím JavaScriptu a XML. Reverzným inžinierstvom vznikli rôzne klientské skripty umožňujúce používateľom prispôbovať ponúkanú funkcionálnosť.

Aplikácia Google Maps umožňuje prehliadanie máp v rôznych módoch a to klasických máp, leteckých snímok a zemegule. Google Maps, znázornená na Obr. 5.2, ponúka prepracovanú navigáciu a množstvo iných služieb.



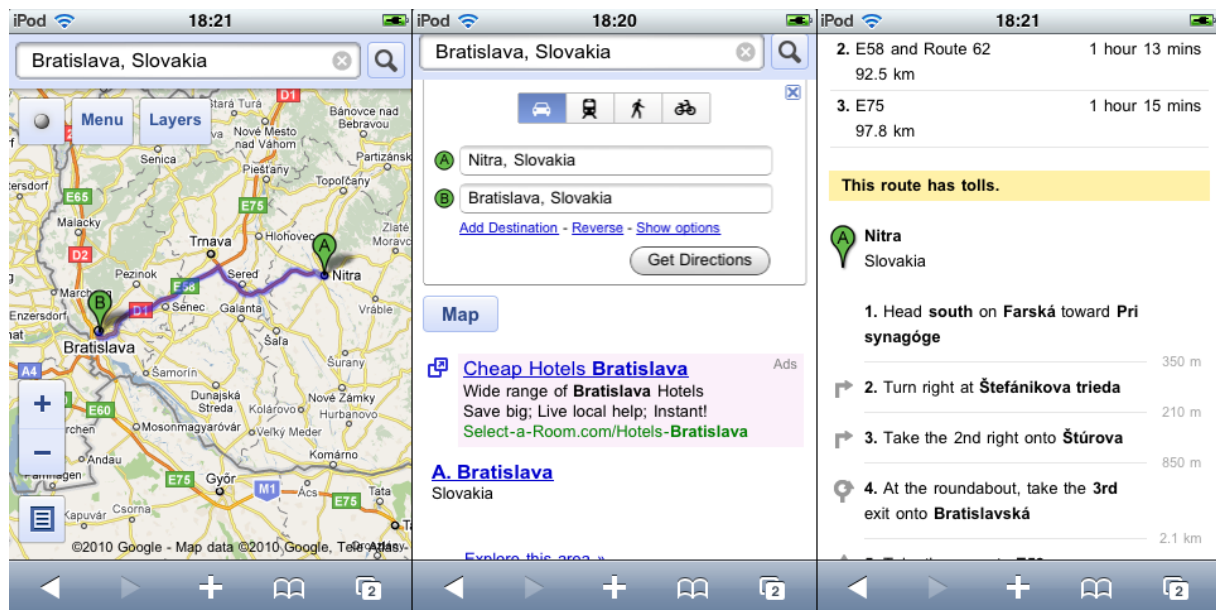
Obr. 5.2 Rozhranie aplikácie Google Maps

<sup>25</sup> <http://maps.google.sk/>

## Google Maps Mobile

Verzia Google Maps pre mobilné zariadenia. Ponúka rovnakú funkcionálnosť ako samotný Google Maps, len používateľské rozhranie je optimalizované pre mobilné zariadenia. V podstate je totožné s rozhraním Google Maps, ktoré je ale, tak ako je vidieť na Obr. 5.3, rozkúskované a súčasne sa zobrazujú len určité komponenty.

Samotná webstránka maps.google.com je optimalizovaná tak, aby sa pri načítavaní z mobilného zariadenia zobrazila mobilná verzia. Avšak existujú aplikácie priamo pre mobilné telefóny, ktoré ponúkajú túto funkcionálnosť (Google Earth, natívna aplikácia Maps pre iPhone, Android, atď.).



Obr. 5.3 Rozhranie aplikácie Google Maps v prehliadači Safari na mobilnom zariadení iPod 2G

## OpenLayers

OpenLayer<sup>26</sup> umožňuje umiestniť dynamické mapy na akúkoľvek stránku. Je napísaný v JavaScripte a je OpenSource aplikáciou. Ponúka veľa možností prispôbenia ako aj použitia dlaždíc z akéhokoľvek zdroja. OpenLayers je pomerne rozšírený a teda má väčšiu podporu a uplatnenie. Bola už aj snaha použiť OpenLayers na mobilných zariadeniach.

Používateľské rozhranie, znázornené na Obr. 5.4 je takmer identické ako Google Maps a ponúka rovnakú základnú funkcionálnosť. Chýba nám tu avšak navigácia a natívna podpora mobilných zariadení.

<sup>26</sup> <http://openlayers.org/>



Obr. 5.4 Rozhranie aplikácie OpenLayers

Za zmienku v tejto časti stojí MapBox<sup>27</sup>, ktorý slúži na tvorbu dlaždíc, ktoré sa následne dajú využiť vo vyššie spomínaných aplikáciách.

## 5.2 Interiérové aplikácie

Aplikácie tohto typu sú zriedkavé a stretávame sa s nimi vo výnimočných prípadoch. Ide prevažne o rôzne na zákazku tvorené, špecializované aplikácie. Vo väčšine prípadov ide o aplikácie v Adobe Flash a mapujú interiéry letísk, prípadne nákupných centier. Tieto aplikácie sú dostupné priamo na stránkach jednotlivých spoločností, ktoré si ich nechávajú vyrobiť. Takýmto príkladom je obchodné centrum Mall of America<sup>28</sup>, ktorého aplikáciu možno vidieť na Obr. 5.5. Cieľom týchto aplikácií je umožniť používateľovi rýchlo sa zorientovať v nákupnom centre.

V oblasti mapovania interiéru však nachádzajú viac uplatnenie práve mobilné zariadenia. Používateľovi umožňujú okrem hľadania pozície jednotlivých obchodov napríklad hľadať jednotlivé produkty, informujú používateľa o zľavách, kde sa nachádza najbližšia toaleta atď. Mobilné zariadenia sú zvýhodnené aj vďaka možnosti lokalizácie používateľa pomocou využitia GPS, WiFi, alebo Bluetooth technológii.

Samotnou problematikou sa zaoberajú už aj giganti ako Microsoft<sup>29</sup>, alebo Nokia<sup>30</sup>. Ide hlavne o desktopové / mobilné aplikácie fungujúce v sídlach týchto korporácií. Môžeme predpokladať je, že ak sa osvedčia v týchto spoločnostiach, budú sa snažiť preraziť aj na trh.

<sup>27</sup> <http://mapbox.com/>

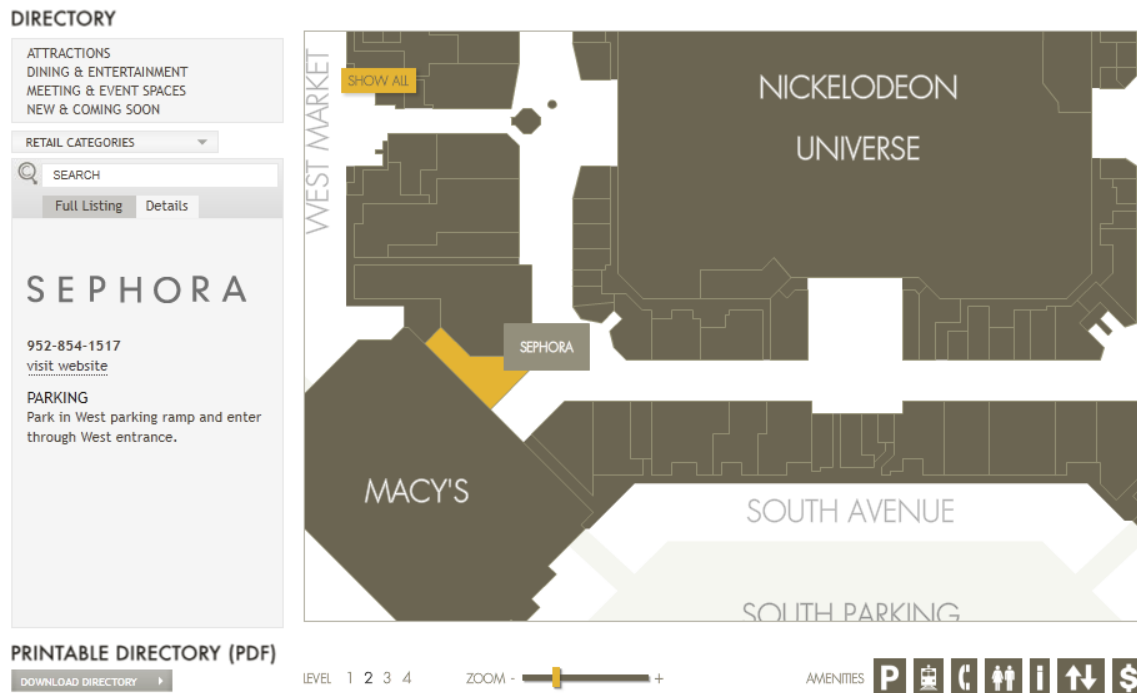
<sup>28</sup> <http://www.mallofamerica.com/>

<sup>29</sup> <http://blogs.msdn.com/b/lokeuei/archive/2009/08/25/gomaps-indoor-navigation.aspx>

<sup>30</sup> <http://research.nokia.com/news/9505> a

[http://www.nokia.com/NOKIA\\_COM\\_1/Press/Press\\_Events/The\\_Way\\_We\\_Live\\_Next\\_2008/presentations/TW\\_WLN08\\_Kimmo\\_Kalliola.pdf](http://www.nokia.com/NOKIA_COM_1/Press/Press_Events/The_Way_We_Live_Next_2008/presentations/TW_WLN08_Kimmo_Kalliola.pdf)

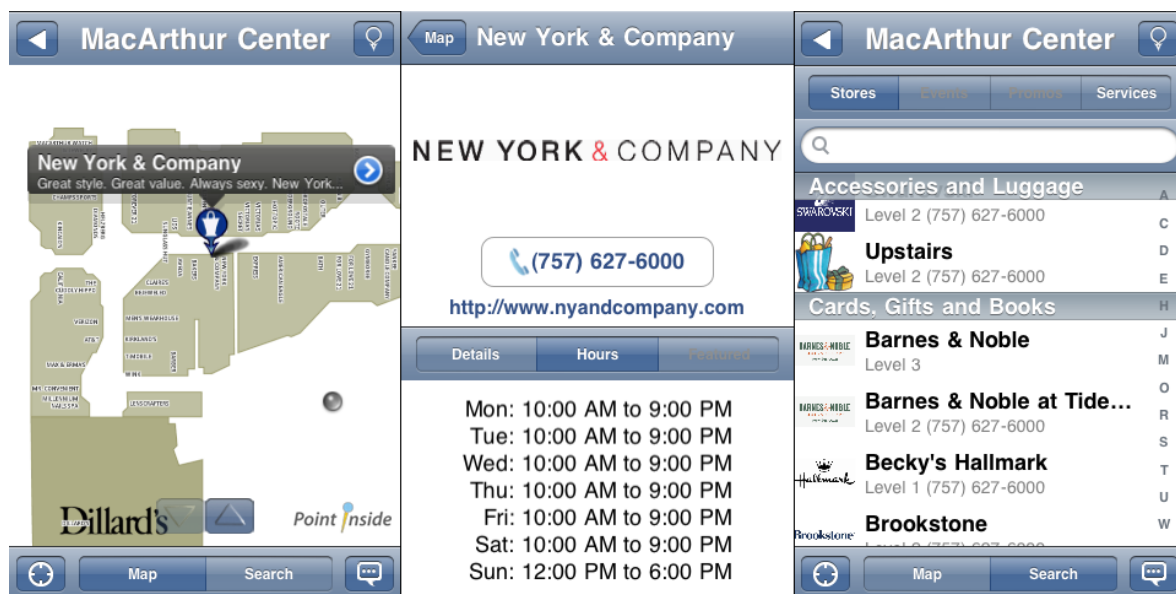




Obr. 5.5 Špecializovaná aplikácie pre Mall of America

### PointInside

PointInside<sup>31</sup> je aplikácia dostupná pre mobilné zariadenia Apple (iPhone, iPod, iPad) a mobilné telefóny s operačným systémom Android. Rozhranie, zobrazené na ponúka možnosť vyhľadávať napríklad obchody v obchodných centrách, pozrieť si ich otváracie hodiny a vidieť svoju polohu na mape.



Obr. 5.6 Rozhranie aplikácie PointInside

<sup>31</sup> <http://pointinside.com/>

## Micello

Micello<sup>32</sup> je taktiež aplikácia dostupná pre Apple aj Android. Táto aplikácia, zobrazená na Obr. 5.7, sa snaží na trh preraziť ako „Google Mapy pre Interiér“. Ponúka rovnakú funkcionality ako PointInside, navyše ponúka aj navigáciu z bodu A do bodu B.



Obr. 5.7 Rozhranie aplikácie Micello

Cieľom oboch vyššie spomínaných aplikácií je mapovať budovy a iné kryté priestory. Ich cieľom je okrem koncových používateľov osloviť aj samotné subjekty figurujúce na mapách (napríklad obchody). Spoplatnené sú služby pre obchody, ktoré môžu napríklad inzerovať rôzne zľavy pre používateľov prehliadajúcich si daný obchodný dom.

## Geopard

Geopard<sup>33</sup> je open-source projekt ktorého cieľom je mapovanie interiérových aj exteriérových priestorov s presnosťou na objekty o veľkosti človeka. Tento projekt je zaujímavý z hľadiska využitia WiFi ako technológie na určenie polohy používateľa.

## 5.3 Záver

Ako vidno z analýzy, oblasť mapovania interiérov je momentálne len v štádiu vývoja a experimentálneho nasadenia. Zatiaľ neexistuje, širokej verejnosti dostupná, aplikácia podobná Google maps, ktorá by ponúkala potrebnú funkcionality pre interiérovú navigáciu a bola by optimalizovaná súčasne pre prehliadače ako aj mobilné zariadenia. Preto rozhodli vytvoriť vlastné riešenie. To nám poskytuje voľnosť pri implementácii akejkoľvek funkcionality. Vzhľadom na rozsah tímového projektu si musíme určiť priority a tou je 3D verzia. Mobilná a 2D verziu budú ponúkať iba nevyhnutnú funkcionality na orientáciu sa po budove s jednoduchou formou navigácie.

<sup>32</sup> <http://www.micello.com/>

<sup>33</sup> <http://geopard.wordpress.com/> a

[http://wiki.openstreetmap.org/wiki/Geopard\\_Indoor\\_Outdoor\\_RTLS\\_%26\\_Mapping#Geopard\\_Client\\_.2F\\_Web](http://wiki.openstreetmap.org/wiki/Geopard_Indoor_Outdoor_RTLS_%26_Mapping#Geopard_Client_.2F_Web)

## 6 Špecifikácia riešenia

---

### 6.1 Špecifikácia požiadaviek

Naše riešenie vychádza z riešenia minuloročného tímu, uvádzame preto iba požiadavky, ktoré sú v našom riešení nové, resp. tie, ktoré neboli v riešení minuloročného tímu implementované.

#### Funkcionálne požiadavky

- Prerobenie zobrazovania modelu novej budovy FIIT do technológie WebGL
- Vylepšenie interaktivity
  - o Funkčný výťah
  - o Nemožnosť prechádzania cez zatvorené dvere
  - o Zobrazovanie názvu miestnosti, v ktorej sa používateľ nachádza
- Aktualizovanie modelu podľa najnovších plánov budovy
- Oprava nedostatkov navigácie medzi miestnosťami
- Vytvorenie 2D verzie modelu, ktorá bude funkčná na bežne používaných prehliadačoch
- Vytvorenie verzie pre mobilné zariadenia
- Získavanie a zobrazovanie verejných informácií z AIS
- Získanie osobného rozvrhu z AIS po prihlásení
- Pre 2D a mobilnú verziu sú spoločné nasledovné funkcionálne požiadavky:
  - o Navigácia
    - s manuálnym zadaním cieľovej miestnosti,
    - s automatickým zistením miestnosti podľa rozvrhu (vyžaduje sa prihlásenie).
  - o Zobrazenie informácií o miestnosti.
  - o Zobrazenie informácií o vyučujúcom.
  - o Zobrazenie plánu budovy po jednotlivých poschodiach.

#### Nefunkcionálne požiadavky

- Systém musí podporovať desiatky súčasne pripojených používateľov
- 3D verzia systému musí byť použiteľná do 20 sekúnd pri priemernej rýchlosti spojenia<sup>34</sup>, ktorá je na Slovensku 620kb/s (nemusia byť však načítané všetky poschodia, stačí iba aktuálne)
- 2D verzia systému musí byť použiteľná do 5 sekúnd pri priemernej rýchlosti spojenia.
- Mobilná verzia systému musí byť použiteľná do 10 sekúnd pri zvolení minimálnej úrovne detailov (v závislosti od vyťaženia a rýchlosti siete mobilného operátora).
- Všetky rozhrania by mali mať intuitívne ovládanie
- Aktualizácia uložených údajov v databáze by mala prebiehať každý deň v noci
- Prihlasovacie údaje užívateľa do AIS nesmú byť ukladané a ani logované.
- Prihlasovacie údaje musia byť zadávané cez zašifrované spojenie - HTTPS.
- Aktualizovanie údajov nesmie narušiť prácu používateľov so systémom
- Hardvérové a softvérové požiadavky na strane klienta:
  - o 3D rozhranie
    - Prehliadač s podporou WebGL
    - Samostatná grafická karta (pri integrovanej grafickej karte sa môžu objaviť problémy s plynulosťou prehliadania modelu)

---

<sup>34</sup> <http://www.dsl.sk/>

- 2D rozhranie
  - Štandardný prehliadač s podporou HTML, CSS a JavaScript.
- Mobilné rozhranie
  - Mobilné zariadenie s webovým prehliadačom a prístupom na internet.
  - Podpora JavaScript-u zariadením pre funkčnosť pokročilých interaktívnych možností.
- Hardvérové a softvérové požiadavky na strane serveru:
  - Webový server s podporou jazyka PHP (vyvíjané pod apache 2.2.9 s podporou PHP5)
  - Databázový systém MySQL 5.0
  - Operačný systém – každý, ktorý spĺňa predošlé požiadavky (vyvíjané pod OS GNU/Linux Debian Lenny s jadrom 2.6.26-2-686)

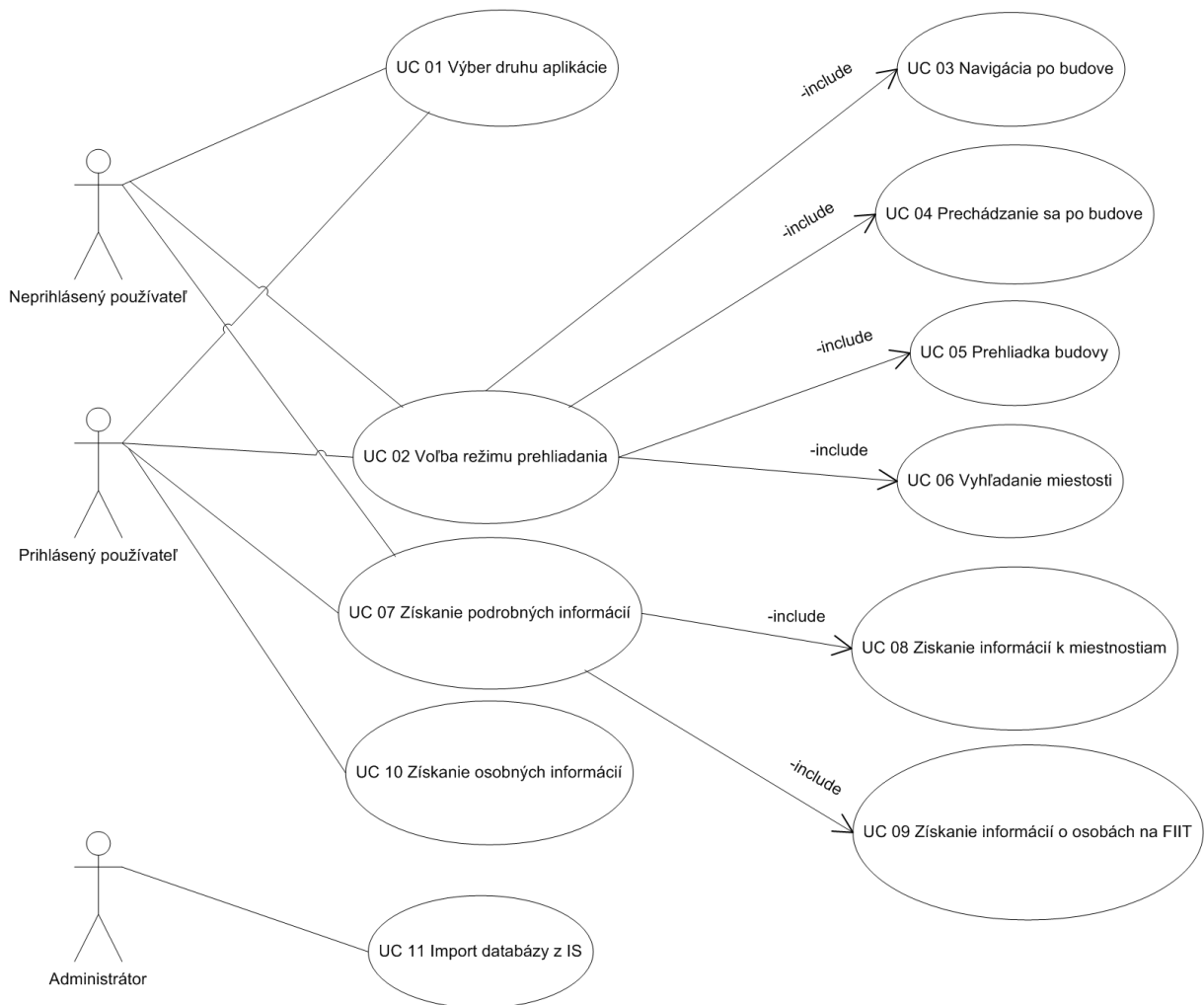
## 6.2 Charakteristika používateľov systému

Uvažujeme s tromi rolami používateľov:

- Administrátor – spravuje celý systém, vrátane modelu a údajov uložených v databáze.
- Používateľ – návštevník stránky, ktorý si chce prezrieť model budovy, alebo nájsť nejaké informácie.
- Prihlásený používateľ – návštevník stránky, ktorý zadal platné prihlasovacie údaje.

### 6.2.1 Prípady použitia

Na Obr. 6.1 je znázornený diagram prípadov použitia, ktorý znázorňuje funkčnosť aplikácie. Diagram bol vytvorený pomocou MS Visio.



Obr. 6.1 Diagram prípadov použitia

## 6.2.2 Opis diagramu prípadov použitia

### UC1 Výber druhu aplikácie

<b>Názov prípadu použitia:</b>	<b>Výber druhu aplikácie</b>
<b>Identifikácia prípadu použitia:</b>	UC 01
<b>Vstupná podmienka:</b>	Korektné spustenie aplikácie v jednom z podporovaných zariadení
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"> <li>1. Používateľ spustí aplikáciu na podporovanom zariadení.</li> <li>2. Aplikácia v prípade podpory viacerých módov zobrazenia (textová vetva aplikácie / 2D vetva / 3D vetva vo WebGL) sprístupní možnosť voľby módu aplikácie. Pri podpore len jednej z možností, ju aplikácia volí automaticky.</li> <li>3. Používateľ si zvolí mód zobrazenia na základe možností použitého zariadenia.</li> <li>4. Aplikácia načíta potrebné dáta pre daný mód.</li> </ol>

<b>Výstupná podmienka:</b>	Aplikácia je spustená a zobrazená v móde podľa používateľovho výberu. V prípade zariadenia, ktoré podporovalo len jednu z módov zobrazenia, musí byť aplikácia korektne spustená v tomto móde.
----------------------------	--

**UC2 Výber režimu prehliadania**

<b>Názov prípadu použitia:</b>	<b>Výber režimu prehliadania</b>
<b>Identifikácia prípadu použitia:</b>	UC 02
<b>Vstupná podmienka:</b>	Aplikácia musí splniť UC 01.
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"> <li>1. Používateľ na základe voľby v UC 01 má k dispozícii niektoré, prípadne všetky, z režimov vizualizácie danej budovy (2D / 3D pohľad z vrchu / 3D pohľad s možnosťou lietania v modeli / 3D simulácia chôdze s interaktívnymi prvkami v budove FIIT). Používateľ klikne na ikonu predstavujúcu preferovaný režim náhľadu na scénu.</li> <li>2. Aplikácia nastaví potrebné parametre prehliadania na hodnoty zodpovedajúce výberu.</li> </ol>
<b>Výstupná podmienka:</b>	Režim podľa výberu je pripravený na prezeranie.

**UC 03 Navigácia po budove**

<b>Názov prípadu použitia:</b>	<b>Navigácia po budove</b>
<b>Identifikácia prípadu použitia:</b>	UC 03
<b>Vstupná podmienka:</b>	Aplikácia musí splniť UC 01.
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"> <li>1. Používateľ si zvolí možnosť navigácie v budove.</li> <li>2. Systém otvorí možnosť výberu miestností (kliknutím / vpísaním názvu).</li> <li>3. Používateľ si vyberie zo zoznamu miestností svoj počiatočný a cieľový bod. Možnosti je možné voliť aj kliknutím na danú miestnosť. Prípadne je možné cieľ a počiatočnú pozíciu načítať z iných prvkov v modeli.</li> <li>4. Systém zobrazí priechodnú cestu medzi dvoma miestnosťami.</li> </ol>
<b>Výstupná podmienka:</b>	Je zobrazená navigácia medzi dvoma miestnosťami, ak daná cesta existuje. Výsledné zobrazenie navigácie je závislé od módu zvoleného v UC 01.

**UC 04 Prechádzanie sa po budove**

<b>Názov prípadu použitia:</b>	<b>Prechádzanie sa po budove</b>
<b>Identifikácia prípadu použitia:</b>	UC 04
<b>Vstupná podmienka:</b>	Aplikácia musí splniť UC 01 a súčasne zariadenie, na ktorom je spustená, musí podporovať internetové prehliadače s podporou WebGL.
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"> <li>1. Používateľ si zvolí možnosť prechádzania sa po budove.</li> <li>2. Systém umiestni kameru na počiatočné miesto v modeli. V prípade prechodu z iného náhľadu systém vráti pozíciu v modeli, ktorá je najbližšie k predchádzajúcemu náhľadu.</li> <li>3. Používateľ má možnosť virtuálne sa prechádzať po budove FIIT. Otvárať dvere. Sledovať nástenky. Vozíť sa vo výťahu.</li> <li>4. Systém nesmie dovoliť prechod cez steny ani lietanie medzi poschodiami, ale musí zabezpečiť adekvátne správanie prostredia voči pohľadu používateľa (<i>first person camera</i>).</li> </ol>
<b>Výstupná podmienka:</b>	Používateľ sa môže virtuálne prechádzať po budove FIIT s interaktívnymi prvkami.

**UC 05 Prehliadka budovy**

<b>Názov prípadu použitia:</b>	<b>Prehliadka budovy</b>
<b>Identifikácia prípadu použitia:</b>	UC 05
<b>Vstupná podmienka:</b>	Aplikácia musí splniť UC 01.
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"> <li>1. Používateľ si zvolí možnosť prehliadky budovy.</li> <li>2. Systém ponúka používateľovi možnosť voľby prehliadky budovy bez akýchkoľvek obmedzení (možnosť prechodu cez steny, poschodia, okná a dvere + pohyb v priestore bez ohľadu na zemskú príťažlivosť).</li> <li>3. Používateľ po zvolení danej možnosti stráca spomínané obmedzenia a má možnosť prehliadky budovy.</li> </ol>
<b>Výstupná podmienka:</b>	Používateľ má možnosť si virtuálne prezerať model bez obmedzení.

**UC 06 Vyhľadanie miestnosti**

<b>Názov prípadu použitia:</b>	<b>Vyhľadanie miestnosti</b>
<b>Identifikácia prípadu použitia:</b>	UC 06
<b>Vstupná podmienka:</b>	1. Aplikácia musí splniť UC 01.
<b>Kroky prípadu použitia:</b>	2. Používateľ si zvolí možnosť vyhľadania miestnosti.

	<ol style="list-style-type: none"> <li>3. Systém dovoľuje vyhľadanie miestnosti podľa mena, kliknutím, prípadne prostredníctvom možnosti zobrazenia miestnosti, v ktorej sa daná osoba v súčasnosti nachádza.</li> <li>4. Používateľovi sa zobrazí daná miestnosť v modeli. Pokiaľ bola získaná pred vykonaním daného kroku súčasná pozícia používateľa je zobrazené aj cesta medzi danými dvoma bodmi.</li> </ol>
<b>Výstupná podmienka:</b>	Je zobrazená požadovaná miestnosť.

**UC 07 Získanie podrobných informácií**

<b>Názov prípadu použitia:</b>	<b>Získanie podrobných informácií</b>
<b>Identifikácia prípadu použitia:</b>	UC 07
<b>Vstupná podmienka:</b>	Aplikácia musí splniť UC 03 a súčasne je spustená na zariadení s prehliadačom podporujúcim technológiu WebGL.
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"> <li>1. Používateľ klikne na niektorý z bodov ponúkajúcich informácie v modeli (napr. nástenka, výveska pred miestnosťou, ...).</li> <li>2. Systém vyhľadá informácie vzťahujúce sa k danému objektu z databázy (export z informačného systému univerzity).</li> <li>3. Používateľovi sa zobrazia načítané informácie v prehľadnej forme.</li> </ol>
<b>Výstupná podmienka:</b>	Používateľovi sú zobrazené požadované informácie.

**UC 08 Získanie informácií o miestnostiach**

<b>Názov prípadu použitia:</b>	<b>Získanie informácií o miestnostiach</b>
<b>Identifikácia prípadu použitia:</b>	UC 08
<b>Vstupná podmienka:</b>	Aplikácia musí splniť UC 03 a súčasne je nutné aby bola spustená na zariadení s prehliadačom podporujúcim technológiu WebGL.
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"> <li>1. Používateľ klikne na interaktívnu nástenku v blízkosti vybranej miestnosti.</li> <li>2. Systém vyhľadá informácie o danej miestnosti (napr. rozvrh danej miestnosti, osoba vedúca výučbu v čase prehliadky, ...).</li> <li>3. Používateľovi sa zobrazia načítané informácie v prehľadnej forme.</li> </ol>
<b>Výstupná podmienka:</b>	Používateľ získal bližšie informácie pomocou interaktívnych



	násteniek vo virtuálnej budove.
--	---------------------------------

**UC 09 Získanie informácií o osobách na FIIT**

<b>Názov prípadu použitia:</b>	<b>Získanie informácií o osobách na FIIT</b>
<b>Identifikácia prípadu použitia:</b>	UC 09
<b>Vstupná podmienka:</b>	Aplikácia musí splniť UC 03 a súčasne je nutné, aby bola spustená na zariadení s prehliadačom podporujúcim technológiu WebGL.
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"> <li>1. Používateľ klikne na interaktívnu nástenku v blízkosti zvolenej miestnosti, prípadne vojde do miestnosti, kde momentálne prebieha výučba, alebo do kancelárie.</li> <li>2. Systém vyhľadá informácie o osobách nachádzajúcich sa v danej miestnosti ( napr. telefónne číslo na osobu, ktorá výučbu viedla, zoznam osôb v danej miestnosti, prípadne nájde miestnosť, kde by sa daná osoba mala nachádzať...).</li> <li>3. Používateľovi sa zobrazia načítané informácie v prehľadnej forme.</li> </ol>
<b>Výstupná podmienka:</b>	Používateľ získa bližšie informácií o osobách vo virtuálnej budove.

**UC 10 Získanie osobných informácií**

<b>Názov prípadu použitia:</b>	<b>Získanie osobných informácií</b>
<b>Identifikácia prípadu použitia:</b>	UC 10
<b>Vstupná podmienka:</b>	Aplikácia musí splniť UC 01.
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"> <li>1. Systém si vyžiada od používateľa prihlasovacie údaje do AIS-u.</li> <li>2. Systém vyhľadá informácie o štúdiu, miestnostiach a kolegoch, ktoré následne ponúkne používateľovi.</li> <li>3. Používateľ konkretizuje svoje vyhľadávanie v systéme (hľadá kolegov, miestnosti v osobnom rozvrhu, jedáleň, atď.).</li> <li>4. Systém prehľadne sprostredkuje používateľovi hľadané informácie.</li> </ol>
<b>Výstupná podmienka:</b>	Používateľ získa požadované informácie.

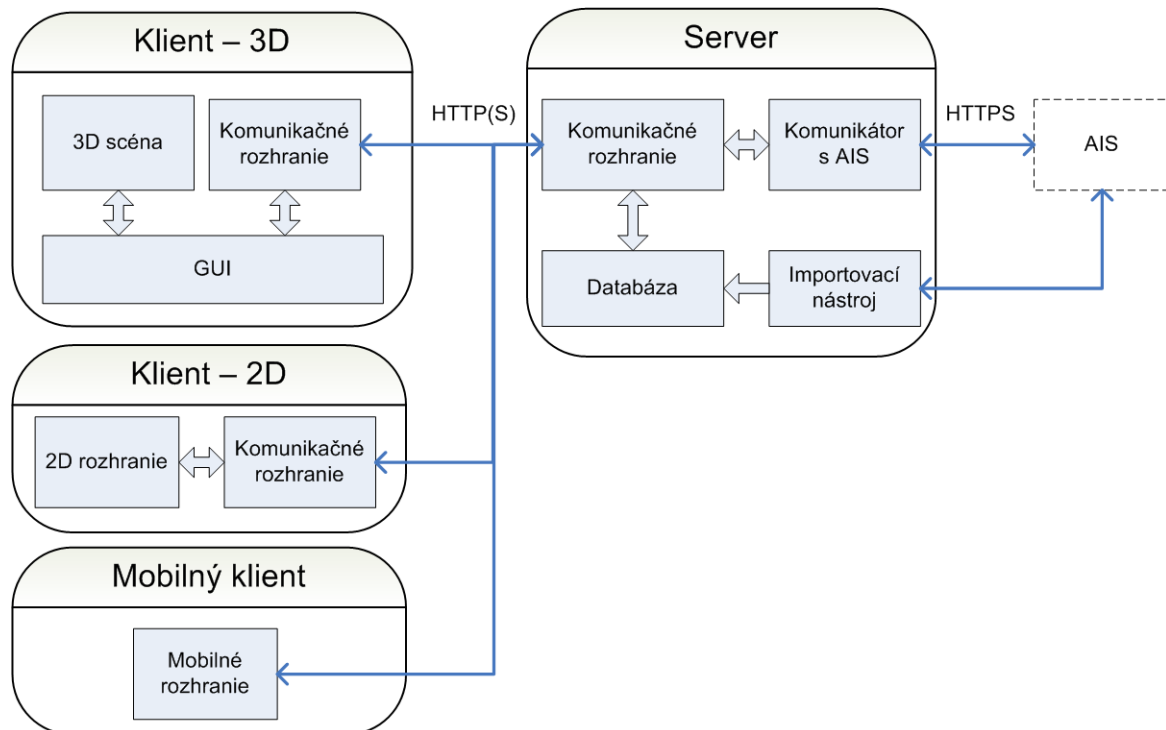
**UC 11 Import databázy z IS**

<b>Názov prípadu použitia:</b>	<b>Import databázy z IS</b>
<b>Identifikácia prípadu</b>	UC 11

<b>použitia:</b>	
<b>Vstupná podmienka:</b>	-
<b>Kroky prípadu použitia:</b>	<ol style="list-style-type: none"><li>1. Administrátor dá podnet na aktualizovanie údajov v databáze.</li><li>2. Aplikácia nahradí údaje v databáze exportovanými dátami z akademického informačného systému.</li><li>3. Aplikácia zobrazí kontrolné hlásenie o stave daného importu.</li></ol>
<b>Výstupná podmienka:</b>	Údaje v systéme budú zodpovedať aktuálnym údajom v databáze.

## 7 Návrh systému Virtuálnej budovy FIIT

### 7.1 Architektúra systému



Obr. 7.1 Schéma architektúry systému

Naše riešenie vychádza z klient-server architektúry, ktorá je znázornená na Obr. 7.1. Klientom je prehliadač, v ktorom je zobrazené niektoré z rozhraní – 3D, 2D, alebo mobilné. Nasleduje opis základných komponentov.

#### 7.1.1 Server

Implementačné technológie na strane servera sú PHP a MySQL.

##### Databáza

Uchováva všetky informácie, ktoré potrebujeme zobrazovať v modeli.

##### Importovací nástroj

Umožňuje načítavať údaje do databázy z AIS. Naprogramovaný bude v jazyku PHP.

##### Komunikačné rozhranie

Poskytuje rozhranie pre klientov na zisťovanie informácií. Komunikačné rozhranie dostane HTTP GET požiadavku, napr. na informáciu o určitej miestnosti. Požadované informácie vyberie z databázy a vráti klientovi vo formáte JSON, alebo v prípade mobilnej verzie vráti celú stránku s požadovanými informáciami. Naprogramované bude v jazyku PHP s rámcom CodeIgniter.

##### Komunikátor s AIS

Umožňuje programové prihlásenie do AIS a následné získanie osobného rozvrhu študenta.

### 7.1.2 Klient

Implementačné technológie na strane klienta sú JavaScript, HTML, CSS s využitím knižnice JQuery.

#### Komunikačné rozhranie

Poskytuje funkcie v JavaScript-e na zistenie informácií zo serveru. Technológiou AJAX zašle požiadavku v správnom tvare na server. Keď server odpovie, komunikačné rozhranie v klientovi oznámi, že prišla odpoveď. Tento proces prebieha asynchrónne, nečaká sa na odpoveď klienta. Komunikačné rozhranie je rovnaké pre 3D a 2D klienta.

#### GUI pre 3D rozhranie

Predstavuje používateľské rozhranie pre klienta, ktorý využíva 3D rozhranie. Obsahuje všetky prvky na stránke okrem samotného zobrazovania modelu a manipulácie s ním. Sú to prvky na vyhľadávanie informácií (miestností alebo osôb), na prepínanie medzi poschodiami, prepínanie medzi režimom prechádzania a prehliadania, zobrazovanie aktuálnej miestnosti (označenej, alebo tej, v ktorej sa používateľ nachádza) a prvky na navigáciu.

#### 3D scéna

Zabezpečuje zobrazovanie modelu, jeho prezeranie, prechádzanie a interaktivitu. Využíva technológiu WebGL s pomocou knižnice CopperLicht. Komunikácia s doplnkovým UI predstavuje:

- Vzájomné informovanie o označenej miestnosti, resp. miestnosti v ktorej sa používateľ nachádza
- Aktualizácia polohy prvkov používateľského rozhrania, ktoré sa viažu k určitej 3D pozícii (napr. nad miestnosťou bude zobrazená nejaká informácia, ktorá sa však musí posunúť podľa toho ako bude používateľ pohybovať s modelom)
- Požiadavka na zmenu poschodia vyvolaná z GUI, alebo informácia, že používateľ prešiel na iné poschodie a GUI sa musí aktualizovať

#### 2D a mobilné rozhranie

Zabezpečuje zobrazovanie pôdorysov poschodí v 2D režime, pre prehliadače bez podpory WebGL a mobilné zariadenia. Zabezpečuje prezeranie jednotlivých poschodí, zobrazovanie informácií o miestnostiach a navigáciu po budove.

### 7.1.3 Optimalizácia mobilnej verzie aplikácie

V 2D verzii určenej pre prehliadače môžeme počítať s vysokým rozlíšením monitora, (takmer) neobmedzeným množstvom prenášaných dát a vysokou pravdepodobnosťou dostupnosti technológií ktoré štandardne poskytujú internetové prehliadače. Mobilné zariadenia majú však obmedzené prostriedky, špecifické používateľské rozhranie a preto je mobilná verzia virtuálnej FIIT špecifická svojimi nasledujúcimi obmedzeniami:

- kompaktný (malý) displej,
- obmedzené množstvo pamäte,
- obmedzený procesorový výkon,
- obmedzenie množstva prenášaných dát,
- možná nedostupnosť technológie JavaScript,
- možná nedostupnosť technológie AJAX,
- nižšia prenosová rýchlosť.

Z obmedzení uvedených pre mobilnú verziu sme zostavili zoznam optimalizačných riešení, ktoré by mali pomôcť efektívne používať aplikáciu Virtuálna FIIT aj na starších modeloch mobilných zariadení:

- jednoduchý dizajn stránky,
- textový režim (minimalizácia použitia obrázkov)
- zvoliteľná veľkosť obrázku s plánom,
- rozkladanie obrázka s plánom budovy,
- textová navigácia,
- „server side“ navigácia (zostavenie cesty na serveri)

## 7.2 Návrh GUI aplikácie

Projekt virtuálnej FIIT si vyžaduje tri grafické používateľské rozhrania. Rozhrania pre 3D a 2D verziu sa budú mierne líšiť. Hlavným rozdielom je spôsob interakcie s aplikáciou, ktorý má následne vplyv na používateľské rozhranie. Hlavné rozdiely, princípy a komponenty opíšeme v nasledujúcich častiach pre jednotlivé verzie aplikácie.

### 7.2.1 Grafické rozhranie pre 3D verziu

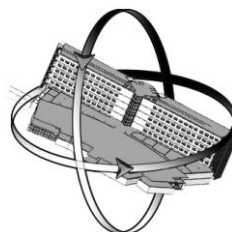
3D verzia ponúka používateľovi najviac interaktivity, preto bude GUI pre túto verziu ponúkať najviac možností. Rozhodli sme sa použiť pre GUI pre 3D verziu obdobný princíp ako využíva napríklad Google Maps. Jednotlivé komponenty ovládania sú vykresľované na neviditeľnú vrstvu nad samotným modelom budovy, prípadne budú vykresľované priamo do modelu. Týmto prístupom dosiahneme takmer nepozorovateľnú integráciu GUI so samotnou aplikáciou a vyhneme sa tak segmentácii aplikácie do časti s modelom, časti s vyhľadávaním a menu.

3D verzia našej aplikácie bude používateľovi poskytovať tri módy prehliadania modelu. Preto bude GUI obsahovať tri tlačidlá, ktoré budú meniť režimy zobrazenia:

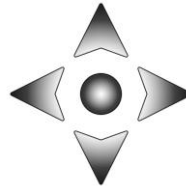
- tlačidlo na zobrazenie módu z pohľadu prvej osoby: model sa zobrazí v režime simulujúcom pohľad, ktorý by sa naskytol používateľovi ak by prechádzal chodbami budovy.
- tlačidlo na zobrazenie módu navigácie: mód navigácie zobrazí jedno, prípadne vrstvene viacero poschodí (podľa nastavenia), tak aby bolo možné prehľadne zobraziť vypočítanú trasu. Zo spomínaného dôvodu, bude rotácia do značnej miery obmedzená (používateľ nepotrebuje vidieť model napríklad zospodu)
- tlačidlo na zobrazenie módu takzvaného voľného letu: tento režim poskytuje používateľovi maximálnu voľnosť, používateľ môže lietať priestorom akýmkoľvek smerom a bez žiadnych zábran

Každý z týchto troch módov bude podporovať ovládanie myšou a klávesnicou, avšak bude možné ovládanie aj čisto len myšou, preto budú na najvyššej vrstve stále dostupné ovládacie prvky na manipuláciu s modelom (prehliadanie). Konkrétne ide o nasledovné tri elementy:

- rotácia modelu



- posun model



- zväčšenie / zmenšenie modelu



Samotný model bude podporovať interakciu s používateľom. Bude umožňovať kliknúť na konkrétnu miestnosť, kde sa následne zobrazí ponuka v jednoduchom a prehľadom dialógovom okne. Podobným spôsobom bude fungovať aj ostatná funkcionlita, ako je napríklad zobrazenie rozvrhu prihláseného používateľa.

### 7.2.2 Grafické rozhranie pre 2D a mobilnú verziu

Grafické rozhranie je pre 2D a pre mobilnú verziu aplikácie v podstate spoločné. Obe verzie sa od 3D verzie výrazne líšia. GUI je jednoznačne rozdelené do jednotlivých komponentov a to:

- 2D plán budovy: obrázok plánu jedného poschodia budovy
- textová navigácia: opisuje v krokoch ako sa používateľ dostane z bodu A do bodu B
- panel s nástrojmi: vyhľadávanie cesty, miestnosti, osoby, atď.

Komponent zobrazujúci 2D plán budovy bude tak, ako v 3D verzii obsahovať v najvyššej vrstve elementy na pracovanie s obrázkom:

- zväčšenie / zmenšenie plánu: bude poskytovať len pár úrovní, medzi ktorými sa bude môcť používateľ prepínať
- posun plánu

Komponent s textovou navigáciou bude obsahovať v bodoch opísaný postup ako má používateľ prechádzať budovou, tak aby trafil do hľadanej miestnosti. Panel s nástrojmi bude jednoduchý formulár, prípadne menu, ktoré poskytne používateľovi všetky špecifikované možnosti.

Normálna a mobilná verzia sa odlišujú sa len nasledovných bodoch:

- mobilná verzia musí byť dátovo nenáročná, preto bude kladený dôraz na prehľadnosť a jednoduchosť
- mobilná verzia bude zobrazovať naraz vždy len jeden komponent, kým normálna zobrazí súčasne všetky komponenty naraz

### 7.3 Dátový model

Ako databázový systém pre náš projekt sme vybrali databázu MySQL dôvodom je, že je dostupná bezplatne a najmä je tento systém potrebný aj pre iné časti projektu a bolo by zbytočné mať aktívne dva rôzne databázové systémy súčasne, najmä keď naše požiadavky na funkcionlitu spĺňajú prakticky všetky súčasné databázové systémy.

Z predchádzajúcej analýzy potrebných a dostupných dát sme vytvorili logický (Obr. 7.2) a fyzický (Obr. 7.3) dátový model. Oba sú vytvorené v IBM Rational Data Architect v7.

### 7.3.1 Opis dátových entít

Tu sa nachádza opis jednotlivých dátových entít. Všetky dátové entity majú ako primárny kľúč, stĺpec s menom ID typu Integer, preto ho nebudem pri jednotlivých entitách uvádzať. Dátové typy je možné vidieť v diagramoch preto ich opätovne nebudem uvádzať v prehľade dátových entít.

#### User

Táto entita predstavuje zamestnancov fakulty.

- First\_Name – predstavuje krstné meno zamestnanca
- Surname – predstavuje priezvisko zamestnanca
- Title\_Before\_Name – tituly zamestnanca pred menom
- Title\_After\_Name – tituly zamestnanca za menom
- Ais\_ID – ID číslo používané v AIS
- Room – pracovňa zamestnanca (ID cudzí kľúč na entitu Room)

#### Timetable

Táto entita predstavuje jednotlivé rozvrhové akcie.

- Type - typ rozvrhovej akcie (ID cudzí kľúč na entitu Type)
- Time\_From – čas odkedy trvá rozvrhová akcia
- Time\_To – čas dokedy trvá rozvrhová akcia
- Note – poznámka k rozvrhovej akcii
- Restriction – obmedzenia rozvrhovej akcie
- Day – deň v týždni (ID cudzí kľúč na entitu Day)
- Course – predmet, ktorého sa rozvrhová akcia týka (ID cudzí kľúč na entitu Course)
- Teacher – učiteľ, ktorý vedie danú rozvrhovú akciu (ID cudzí kľúč na entitu User)
- Room – miestnosť v ktorej sa rozvrhová akcia koná (ID cudzí kľúč na entitu Room)

#### Phone

Táto entita predstavuje telefón zamestnanca fakulty.

- Phone – telefónne číslo
- User – zamestnanec, ktorému patrí telefónne číslo (ID cudzí kľúč na entitu User)

#### Email

Táto entita predstavuje e-mail zamestnanca fakulty.

- Email – e-mail
- User – zamestnanec, ktorému patrí e-mail (ID cudzí kľúč na entitu User)

#### Course

Táto entita predstavuje predmet, ktorý má vytvorené nejaké rozvrhové akcie.

- Name – názov predmetu

#### Day

Táto entita predstavuje deň v týždni

- Name – názov dňa v týždni

- Abb – skratka dňa v týždni

### **Room**

Táto entita predstavuje miestnosti v univerzity.

- Name – názov miestnosti
- Building – budova v ktorej sa miestnosť nachádza (ID cudzí kľúč na entitu Building)

### **Building**

Táto entita predstavuje budovy univerzity.

- Name – názov budovy

Okrem týchto dátových entít sa bude nachádzať v databáze ešte dočasná entita Room\_FIIT. Ide o špeciálnu tabuľku na mapovanie existujúcich rozvrhov v reálnych miestnostiach budovy fakulty FEI na dosiaľ nedokončené miestnosti novej budovy fakulty FIIT. Po dokončení budovy a vytvorení skutočných rozvrhov v novej budove FIIT bude táto pridaná do tabuľky Building a miestnosti sa budú nachádzať v tabuľke Room. Táto entita nie je zobrazená v logickom ani fyzickom dátovom modeli, keďže ide len o dočasnú tabuľku, potrebnú, kým nebude dokončená budova FIIT a nebudú rozvrhové akcie prebiehať priamo v tejto budove.

### **Room\_FIIT**

Táto entita predstavuje dočasné mapovanie dosiaľ neexistujúcich miestností na reálne miestnosti fakulty FEI.

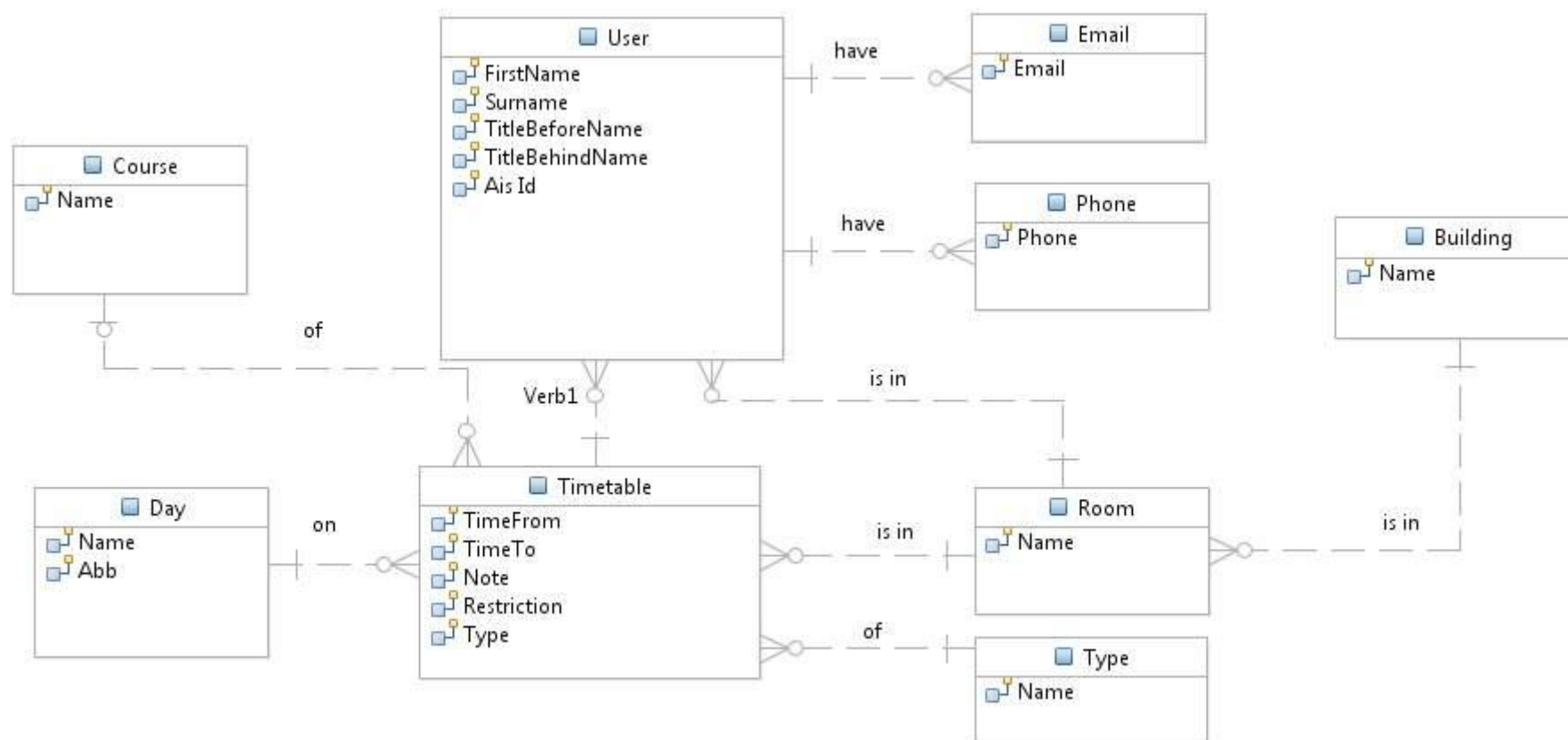
- Name – názov miestnosti v novej budove FIIT
- Room – miestnosť v existujúcej budove fakulty FEI (ID cudzí kľúč na entitu Room)

### **Type**

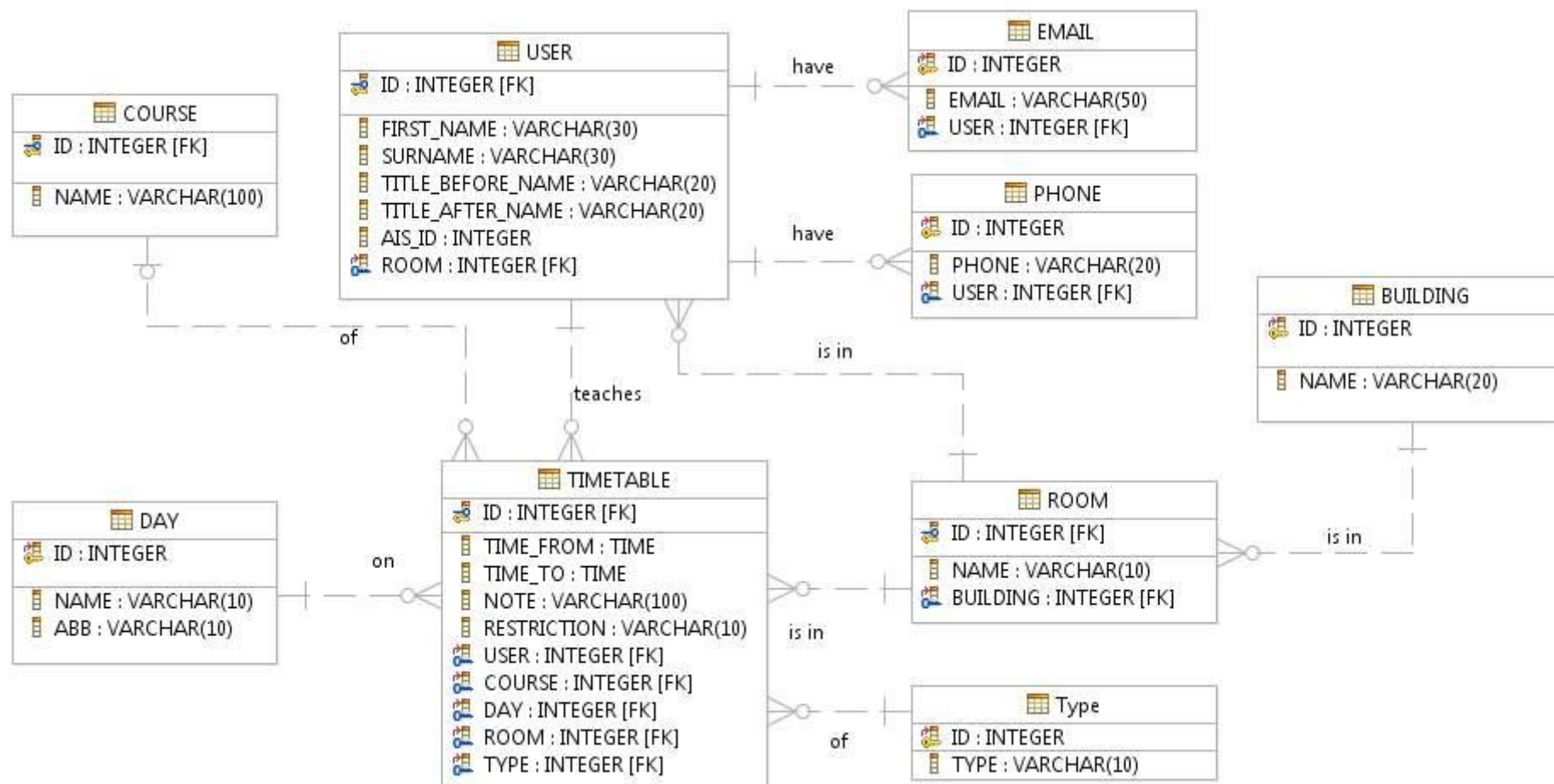
Táto entita predstavuje typy rozvrhových akcií.

- Name - Názov typu rozvrhovej akcie





Obr. 7.2 Logický dátový model



Obr. 7.3 Fyzický dátový model

## 8 Implementácia prototypu

---

### 8.1 Priority implementácie

V zimnom semestri budeme vyvíjať evolučný prototyp, z ktorého v letnom semestri vyvinieme výsledný produkt. V prototypy sú naše priority implementácie nasledovné:

- 3D klient
  - funkčnosť minuloročného riešenia - prehliadanie, prechádzanie, označovanie miestností
  - neprechádzanie cez dvere
  - výťah
  - zobrazovanie informácií nad miestnosťou
- 2D klient
  - vytvorený plán aspoň jedného poschodia
  - označovanie miestností
  - vyhľadávanie a zobrazovanie miestností
- Mobilný klient
  - vytvorený plán aspoň jedného poschodia
- Ku všetkým rozhraniam klienta
  - prototyp používateľského rozhrania
- Model
  - spísané potrebné aktualizácie modelu
  - aktualizované a upravené minimálne jedno poschodie
- Server
  - import údajov z AIS
  - komunikačné rozhranie na serverovej aj klientskej strane

Pre všetky uvedené časti platí, že v prototypy sa nezameriavame na používateľskú prívetivosť ani na kompletnosť. Stačí teda, keď v prototypy bude iba jedno poschodie, a nebudú importované a zobrazované všetky údaje z AIS, ale len vybrané.

V letnom semestri z hľadiska implementácie plánujeme:

- implementovať navigáciu
- domodelovať zvyšné poschodia
- vytvoriť zvyšné plány poschodí pre 2D a mobilného klienta
- implementovať funkcionality, ktorá bola v prototypy implementovaná iba čiastočne
- implementácia získavania osobných rozvrhov z AIS
- optimalizácia rýchlosti načítavania a množstva prenášaných údajov

### 8.2 3D klient

V tejto časti je opísaný prototyp z hľadiska 3D klienta.

### 8.2.1 Používateľské rozhranie

V 3D scéne je načítaný upravený model 6. poschodia minuloročného tímu s farbami a priehľadnými oknami. Používateľ má na výber 3 režimy pohybu v scéne:

- Lietanie – neobmedzený pohyb
- Prechádzanie sa po podlahe, kontrolujú sa kolízie so stenami
- Prehliadanie – kamera sa otáča okolo pevného bodu v strede modelu.

Pri prechádzaní sa funguje otváranie a zatváranie dverí, pričom cez zatvorené dvere nie je možné prejsť. Výťahové dvere sa správne otvárajú. Funguje tiež vychádzanie po schodoch. V používateľskom rozhraní je tiež vypísaný názov aktuálnej miestnosti, v ktorej sa používateľ práve nachádza. Ak používateľ vstúpi do novej miestnosti, načíta sa do ľavého panelu jej rozvrh.

V porovnaní s minuloročným tímom je plynulejšie vyriešené šmýkanie sa po stenách, ako je známe z mnohých počítačových hier. Pri kolízii so stenou sa používateľ totiž nemá zastaviť, ale šmyknúť sa popri nej.

V režime lietania a prehliadania je možné kliknutím označovať miestnosti. Pri kliknutí nad miestnosť sa zobrazí HTML element s názvom miestnosti. Tento element sa pri zmene polohy kamery presunie vždy tak, aby bol nad označenou miestnosťou. Zároveň sa pri označení miestnosti zo serveru načítajú informácie o rozvrhu vyučovania pre označenú miestnosť. Miestnosti zo súčasnej budovy sú na tento účel namapované na miestnosti 6. poschodia.

Pozadie 3D scény už nie je len farba, ale celá scéna sa nachádza v kocke s textúrou oblohy (tzv. skybox). Textúry pochádzajú z editoru CopperCube od tvorcov CopperLichtu.

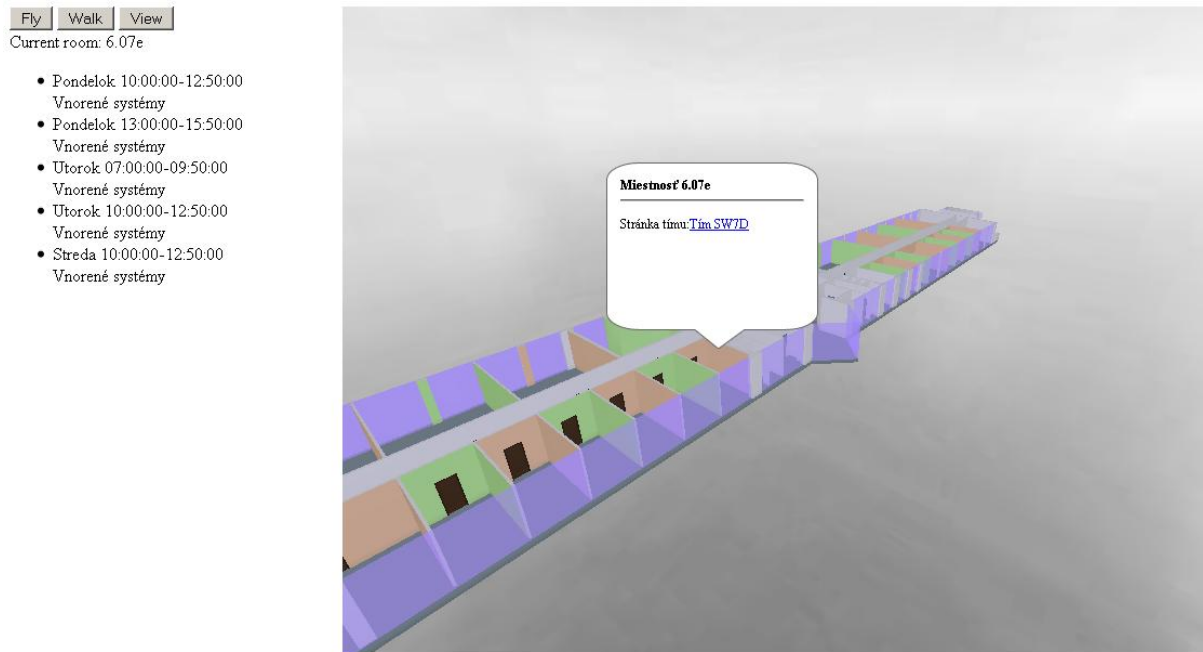
Pri otvorení a zatvorení dverí sa prehrá príslušný zvukový efekt. Takisto pri pohybe po modeli sa prehráva zvuk krokov.

Oproti minuloročnému tímu sme otváranie dverí vyriešili klávesou E, podobne ako v počítačových hrách. Dvere sa už otáčajú vždy na správnu stranu, pretože v minuloročnom tíme sa niektoré dvere otvárali do chodby, niektoré do miestnosti.

Používateľské rozhranie je len dočasné, pre účely prototypu. V letnom semestri bude nahradené plnohodnotným používateľským rozhraním.

### 8.2.2 Ovládanie aplikácie

V tejto časti je zhrnuté ovládanie jednotlivých režimov 3D klienta. Vyžaduje sa beta verzia prehliadača Firefox 4. Rozhranie aplikácie sa nachádza na Obr. 8.1.



Obr. 8.1 Rozhranie aplikácie

### Prepínanie režimov

Prepínanie režimov sa realizuje tlačidlami vľavo hore (Fly – režim lietania, Walk – režim prechádzania, View – režim prehliadania).

### Režim prechádzania

Pri štarte aplikácie sa používateľ nachádza v režime prechádzania modelom. Pohyb je riešený klávesmi W,S,A,D (dopredu, dozadu, doľava, doprava). Smer pohľadu sa mení stlačením ľavého tlačidla myši a ťahaním.

Dvere sa otvárajú a zatvárajú klávesom E, ak je používateľ dostatočne blízko.

### Režim lietania

Pohyb aj zmena smeru pohľadu sa vykonáva rovnako ako v režime prechádzania.

Pri kliknutí na miestnosť sa zobrazí element s jej názvom a na boku sa načítajú zo serveru údaje o rozvrhu v danej miestnosti (resp. miestnosti z AIS, ktorá je na ňu namapovaná).

### Režim prehliadania

Kamera sa otáča okolo stredu modelu stlačením ľavého tlačidla myši a ťahaním. Približovanie a vzdďalovanie kamery od stredu modelu prebieha kolieskom myši.

### 8.2.3 Opis implementácie

Identifikovanie miestnosti, v ktorej sa používateľ nachádza je možné vďaka tomu, že vo výškovej mape sú vo farbe pixel-ov zakódované čísla miestností. V každom pixel-i bajt pre červenú farbu určuje výšku podlahy a bajty pre zelenú a modrú farbu identifikujú miestnosť. Výšková mapa je tiež použitá na identifikáciu miestnosti, na ktorú používateľ klikol v režime lietania, alebo prehliadania.

Dvere sú určené vo výškovej mape špeciálnou farbou. Vždy, keď pri prechádzaní po modeli chce používateľ vstúpiť do priestoru dverí, zistia sa najbližšie dvere k používateľovi. Ak sú otvorené,

používateľovi je dovolené vstúpiť na novú pozíciu. Inak sa dvere správajú ako stena. Vo výškovej mape sme každé dvere neidentifikovali jednoznačne farbou, pretože vzhľadom na počet dverí, by to bolo dosť časovo náročné. Takýto prístup je jednoduchší a nikde v modeli nie sú dvere tak blízko seba, že by boli problémy zistiť, ktoré dvere sa majú otvoriť. Súčasnú implementáciu však bude treba upraviť, pretože sa zisťuje len vzdialenosť k pántom najbližších dverí a nekontroluje sa, či sa používateľ na nejaké dvere aj pozerá. Otvoriť sa tak dajú aj dvere, ktoré sú za chrbtom používateľa.

Šmýkanie sa po stenách je riešené tak, že pri pohybe na novú pozíciu sa skontroluje, či tam nie je stena, alebo zatvorené dvere. Ak áno, tak sa postupne skúša rovnaký pohyb, ale posunutý o určitý uhol viac vľavo, alebo viac vpravo. Tento uhol sa postupne zväčšuje. Ak sa nenájde taký ostrý uhol, pre ktorý by sa bolo možné posunúť, používateľ sa nepohne a zostáva na mieste. Takéto riešenie má za výsledok plynulejšie šmýkanie sa po stenách v porovnaní s minuloročným tímom, kde pri kolízii so stenami skúšali posunúť kameru len doľava, alebo doprava.

Pri počítačoch, na ktorých aplikácia bežala pomalšie sme riešili problém s prechádzaním cez steny. Pri pohybe sa kontroluje, či sa na cieľovej pozícii a okolo nej nenachádza stena, alebo zatvorené dvere. Ak nie, tak sa pohyb vykoná. Ak sa však aplikácia vykresľuje pomaly, cieľová pozícia je viac vzdialená od aktuálnej pozície a je tak možné prejsť cez stenu (alebo zatvorené dvere). Preto kolízie so stenou testujeme aj na viacerých pozíciách medzi aktuálnou a cieľovou pozíciou, čo tento problém vyriešilo.

Načítanie modelu je riešené cez formát Collada, ktorý exportujeme z modelovacieho programu 3ds max 2010. Model vo formáte Collada, čo je vlastne XML súbor, je jednoduchým konvertorom prevedený do formátu JSON. Vyhneme sa tak parsovaniu XML a práca s modelom ako objektom v JavaScript-e je tiež jednoduchšia.

Zvukové efekty využívajú technológiu HTML 5 audio. Samotné zvuky sú voľne dostupné na internete<sup>35</sup>.

Zdrojový kód v JavaScript-e je rozdelený do modulov, žiaden kód v JavaScript-e sa nenachádza v HTML kóde. Celý kód je umiestnený do menného priestoru vfiit. Zoznam modulov je nasledovný:

- vfiit.js – hlavná trieda aplikácie (Engine)
- vfiit.ui.js – konfigurácia aplikácie a prepojenie s používateľským rozhraním
- vfiit.cameras.js – kamery pre jednotlivé režimy
- vfiit.collisions.js – práca s výškovou mapou
- vfiit.doors.js – animácia dverí
- vfiit.elevatorDoors.js – animácia výťahových dverí
- vfiit.imports.js – načítanie Collada modelu (ako JSON)
- vfiit.materials.js – práca s materiálmi
- vfiit.rooms.js – zoznam miestností na druhom poschodí spolu s ich farbami v kolíznej mape
- vfiit.comInterface.js – komunikačné rozhranie pre spojenie so serverom

#### 8.2.4 Zhodnotenie výberu technológii

Výber knižnice CopperLicht bol dobrá voľba. Nachádza sa tam veľa funkcií, ktoré sme pri tvorbe prototypu využili. Naprogramované tam boli napríklad už rôzne typy kamier, ktoré stačilo pre naše

---

<sup>35</sup> [www.soundjay.com](http://www.soundjay.com)

potreby len upraviť. Počas práce na prototype bola odhalená chyba, ktorá bola nahlásená vývojárovi CopperLichtu. Táto chyba bola v nasledujúcej verzii CopperLichtu opravená.

Pri technológií WebGL sa ukazuje, že je už v použiteľnom stave. Vo Firefoxe 4 beta funguje aplikácia plynulo, aspoň na počítačoch, na ktorých sme aplikáciu testovali. Hlavným nedostatkom je neprítomnosť antialiasingu. V Chrome Canary je však vykresľovanie veľmi pomalé a aplikácia preto nie je v tomto prehliadači použiteľná.

### 8.3 Mobilný klient

V tejto časti je opísaný prototyp mobilného klienta.

#### Používateľské rozhranie

Používateľské rozhranie je prispôbené mobilným zariadeniam a ich obmedzeným prostriedkom, podľa zoznamu optimalizačných riešení popísaných v 10 kapitole projektovej dokumentácie.

Po otvorení štartovacej stránky sa zobrazí krátka úvodná správa a textové navigačné menu so štyrmi možnosťami:

- Fakulta - pre získanie informácií o fakulte.
- Ľudia na fakulte - pre získanie informácií o ľuďoch na fakulte.
- Miestnosti - pre získanie informácií o miestnostiach fakulty.
- Budova - pre získanie informácií o budove fakulty.

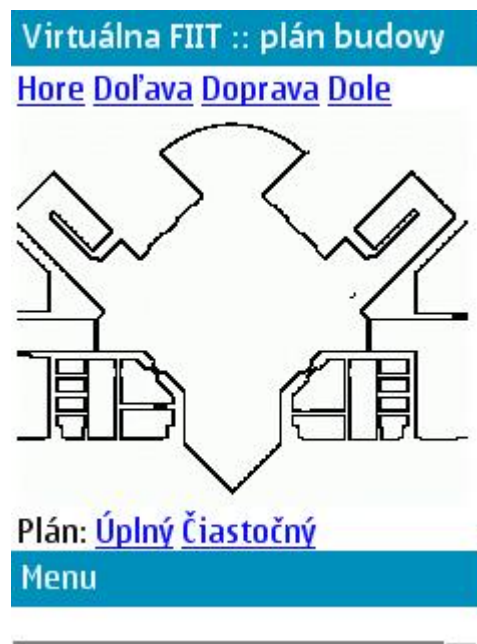
Štartovacia stránka je zobrazená na Obr. 8.2. Obrázok pochádza zo zobrazenia mobilného klienta na mobilnom telefóne Nokia E66.



Obr. 8.2 Štartovacia stránka mobilnej verzie virtuálnej FIIT

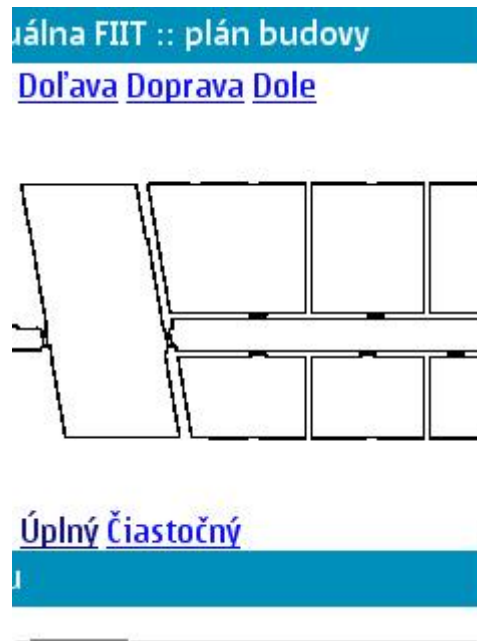
Časť Fakulta obsahuje všeobecné informácie o fakulte, zameraní, vzniku a kontaktné údaje. Informácie o ľuďoch na fakulte a miestnostiach v prototyp aplikácie neobsahujú žiadne informácie. V časti Budova je možné prehliadať plán (pôdorys) poschodia novej budovy FIIT v dvoch režimoch:

- Čiastočný – implicitne nastavený, zobrazuje len časť budovy. Posúvanie plánu sa realizuje kliknutím na jednu z možností: Hore, Doľava, Doprava, Dole. Šírka čiastočného plánu je v prototypu nastavená na 200 pixelov. Zobrazenie čiastočného plánu v mobilnom klientovi je zobrazené na Obr. 8.3.
- Úplný – zobrazí úplný plán budovy a posúvanie plánu je prenechané internetového prehliadača mobilného zariadenia. Možnosti pre posúvanie plánu v čiastočnom režime pri zobrazení úplného plánu spôsobia prechod do čiastočného režimu. Zobrazenie úplného plánu je zobrazené na Obr. 8.4.



Obr. 8.3 Zobrazenie plánu novej budovy FIIT v čiastočnom režime.





Obr. 8.4 Zobrazenie plánu novej budovy FIIT v úplnom režime.

### Ovládanie aplikácie

Ovládanie mobilného klienta je rovnaké ako pre iné internetové stránky. Pre zobrazenie a prácu s aplikáciou postačí bežný internetový prehliadač.

### Opis implementácie

Mobilný klient je vytvorený v jazyku PHP s rámcom CodeIgniter. Prezentačná vrstva je zložená z kombinácie technológií HTML a CSS. Plán poschodia budovy je obrázok vo formáte GIF, ktorý sa v úplnom režime zobrazí celý a v čiastočnom režime sa programovo generuje časť obrázka, ktorá sa má podľa výberu zobrazit'.

## 8.4 AIS parser

V tejto časti je opísaná trieda AISPredmety, ktorá slúži na získanie rozvrhu vyučovaných predmetov.

Trieda obsahuje nasledujúce metódy:

- private function odstranHTML(odpoved', začiatocnýZnak, konečnýZnak, caseSensitive= FALSE)
- public function nacistaj()

### 8.4.1 Popis metód

1. Metóda odstranHTML(odpoved', začiatocnýZnak, konečnýZnak, caseSensitive= FALSE) odstráni z reťazca „odpoved“ ľubovoľne dlhý podreťazec začínajúci reťazcom „začiatocnýZnak“ a končiaci reťazcom „konečnýZnak“. Metóda môže byť modifikovateľná na rozoznávanie malých a veľkých znakov nastavením príznaku „caseSensitive“ na TRUE.
2. Metóda nacistaj() vytvorí spojenie s akademickým informačným systémom a vyžiada si aktuálny rozvrh vyučovaných predmetov na Fakulte informatiky a informačných technológií.

Následne pomocou parsera `simple_html_dom`<sup>36</sup> vyparsuje rozvrh a uloží ho do poľa s nasledujúcou štruktúrou:

Rozvrh[Deň: {Po/Ut/St/Št/Pia}][Predmet: {0..\*}][Atribút: {0..7}]

kde:

- Deň: {{Po/Ut/St/Št/Pia}} je identifikátor dňa
- Predmet: {{0..\*}} vyučovaný predmet (v poradí akom bol načítaný z AIS)
- Atribút: {{0..7}} je identifikátor vlastnosti predmetu:
 

[0] => Po	- deň
[1] => 8.00	- začiatok vyučovania
[2] => 9.50	- koniec vyučovania
[3] => Elektronika	- názov predmetu
[4] => cvičení	- typ hodiny
[5] => e228	- miestnosť
[6] => V. Olah	- vyučujúci
[7] =>	- poznámka

Metóda vracia pole ako výsledok funkcie.

## 8.5 AIS importér

V tejto časti je popísaná knižnica `Importer`, ktorá slúži na uloženie získaných dát z informačného systému AIS do databázového systému MySQL. Jej implementačným jazykom je PHP a je implementovaná ako knižnica pre webový rámec `CodeIgniter`. V prototypy sa do databázy importujú iba dáta o rozvrhu, konkrétne:

- deň
- čas od
- čas do
- názov predmetu
- typ rozvrhovej akcie (prednáška, cvičenie)
- vyučujúci
- miestnosť

Neimportujú sa podrobnosti o miestnostiach a podrobnosti o ľuďoch (vyučujúcich).

### 8.5.1 Opis knižnice

Jedinou verejnou funkciou je metóda `import_array_into_database($array)`, ktorá na vstupe čaká pole hodnôt, ktoré je zhodné s výstupom metódy `nacitaj()` knižnice `AIS_parser`. Na výstupe vracia pole reťazcov (typ `String`), ktoré obsahuje správy logu o priebehu celého importu.

Ostatné metódy sú súkromné a rozdeľujú funkčnosť kódu do menších logických celkov nasledovne:

- pomocné metódy na vkladanie dát do databázy (metódy `empty_database()`, `generate_days()`, `insert_default_building()`)
  - ide o funkcie, ktoré neprijímajú žiaden parameter a vkladajú do databázy vopred definované údaje. Funkcia `empty_database()` je určená na zmazanie dát v databáze. Tieto

<sup>36</sup> <http://simplehtmldom.sourceforge.net/>

funkcie budú zväčša nahradené parametrizovanými funkciami a ich prítomnosť je len dočasná pre zaistenie funkčnosti prototypu.

- Metóda pre vkladanie záznamu o rozvrhu (`insert_single_schedule($schedule)`).
  - Jej úlohou je vložiť do tabuľky `Timetable` záznam o rozvrhovej akcii.
  - V prototypu nie je implementovaná kontrola správnosti záznamu a kontrola duplicitného záznamu v databáze
- Funkcie pre vkladanie údajov, alebo dopyt po údajoch z ostatných tabuliek databázy iných ako tabuľky `Timetable`. Úlohou metód je zistiť ID záznamu v databáze a vrátiť tento záznam. Ak sa takýto záznam nenachádza v tabuľke, vloží sa nový.
  - `get_course_id($course_name)`,
  - `get_day_id($day_abb)`,
  - `get_room_id($room_name)`,
  - `$get_type_id($type_name)`,
  - `$get_user_id($first_name,$surname)`
  - Pre zabránenie redundancii kódu sa niektoré rovnaké časti týchto metód zhromaždili v týchto metódach, ktoré môžeme považovať za ich zovšeobecnenie
    - `get_id($table_name, $column, $value)`
    - `get_id_where_array($table_name, $array)`
    - `get_first_column_in_row($result, $column_name)`
- Ostatné metódy, ktoré nezodpovedajú kategóriám určeným vyššie
  - `insert_single_day($array_of_day)` – rozdeľuje vstup na jednotlivé rozvrhy dňa
  - `time_format($input)` – upravuje formát času na formát databázového systému MySQL
  - `add_message($message)` – vkladá do poľa logovacích záznamov nový záznam

Podrobné popisy funkcií možno nájsť v zdrojovom kóde.

### 8.5.2 Zhodnotenie a plány do budúcnosti

Knižnica plní svoju úlohu korektne, avšak nemá implementované ochrany na kontrolu vstupov, preto sa pri jej funkcionalite spoľiehame na korektné vstupy. Ďalej obsahuje metódy, ktoré do databázy vkladajú „pevné hodnoty“. Tieto metódy však budú nahradené parametrizovanými metódami. Hlavnou úlohou na letný semester bude rozšíriť jej funkcionalitu aj o možnosť vkladania podrobných informácií o ľuďoch (vyučujúcich) a o miestnostiach.

## 8.6 Komunikačné rozhranie

V tejto časti je opísaný prototyp servera pre komunikačné rozhranie.

### 8.6.1 Úloha servera pre komunikačné rozhranie

Úlohou servera pre komunikačné rozhranie (ďalej len KR server) je sprostredkovať klientovi odpovede na http dotazy na konkrétnej URI adrese, ktoré sa týkajú záznamov uložených v databáze servera. Server je schopný vrátiť údaje vo viacerých formátoch (html stránka, formát json). Databáza servera obsahuje údaje o miestnostiach, rozvrhoch, vyučujúcich, ktoré boli exportované z informačného systému AIS. Pre potreby prototypu je server schopný poslať údaje o rozvrhových akciách vo formáte html a json.

### 8.6.2 Zvolené technológie, architektúra návrhu a aktuálny stav prototypu

Implementačný jazyk bol zvolený jazyk PHP najmä z dôvodu širokej podpory tohto jazyka na http serveroch. Rovnako sme vychádzali z predpokladu, že naša aplikácia bude nasadená na školskom serveri, ktorý teraz obsluhuje aplikáciu virtuálnej FIIT predošlého tímu, ktorá je rovnako programovaná jazykom PHP. Pre vývoj v jazyku PHP sme zvolili rámec CodeIgniter, ktorý je navrhnutý podľa architektúry Model-View-Controller, ktorá podmienila aj architektúru KR servera. Dáta sú uložené v databázovom systéme MySQL. Aplikácia je navrhnutá nasledovne:

- Controller – trieda Kr (súbor controllers/kr.php)
  - prijíma požiadavky na URI adresách
    - `http://<server>/<application>/index.php/kr/timetablehtml/<param>` – výstup ako html stránka
    - `http://<server>/<application>/index.php/kr/timetablejson/<param>/$callback` – výstup ako json formát, pričom parameter *\$callback* určuje názov funkcie JavaScript, ktorá sa použije na klientskej strane. Týmto spôsobom máme možnosť volať túto metódu aj z inej domény ako je táto. Je to vhodné predovšetkým pre testovacie účely.
    - `<param>` má nasledujúci formát:  
 „\$room\_name/\$day\_name/\$from\_time/\$limit/\$offset“. Ide o parametre, ktoré slúžia na zúženie výberu. Akonáhle je nasledovne:
      - \$room\_name - meno miestosti (napr. cd150 (BA-MD-FEI C-D))
      - \$day\_name - názov dňa po slovensky (napr. Štvrtok)
      - \$from\_time - čas od (napr. 8:00:00)
      - \$limit - počet záznamov, ktoré sa majú vrátiť (napr. 8)
      - \$offset - počet záznamov, ktoré sa majú preskočiť (napr. 20)
  - volá metódy triedy Timetable, ktoré reprezentujú požiadavky na databázový server
  - vráti výsledky pomocou html stránky (používa sa pri tom šablóna html stránky, predstavujúca view), alebo sa vráti výsledok vo formáte json.
- Model – trieda Timetable (súbor models/timetable.php)
  - je objektovým reprezentantom tabuľky Timetable v databázovom systéme
  - pri zavolaní jej funkcií posielajú dotazy viacerým tabuľkám súvisiacich s tabuľkou Timetable a vracia kompletne výsledky o rozvrhu
  - pomocou parametrov funkcií je možné filtrovať výsledky dopytov
- View – súbory timetable.php a no\_result.php
  - súbor import.php je šablóna stránky html, v ktorej sa zobrazí výsledok dopytu
  - súbor no\_result.php je stránka html, ktorá sa zobrazí v prípade, ak sa nenašiel žiaden výsledok pre zadaný dopyt

Pomocou prístupu, kedy posielame výstup ako formát JSON, sme schopní používať AJAX technológiu na strane klienta, ktorej hlavnou výhodou je možnosť obnoviť dáta na webovej stránke bez nutnosti načítať celú stránku, ale iba jej časť. Formát JSON je upravený tak, aby zodpovedal vstupu funkcie

### 8.6.3 Zhodnotenie

KR server v súčasnom stave pracuje korektne, vracia výsledky rozvrhových akcií a korektne ich filtruje na základe dodaných parametrov. Pre testovacie účely ponúkame formát výstupu aj ako htm stránku.

Prepojenie s klientskou stranou bolo bezproblémové a aplikácia reaguje na dotazy rýchlo. PHP rámec splnil, čo sľuboval na domovskej stránke a pomohol nám zamerať sa priamo na vývoj našej aplikácie. Počas vývoja sme narazili na problém ladenia aplikácie, kde nám chýbal jednak zo strany rámca, ako aj zo samotného jazyka PHP nástroj pre logovanie. Pre ladenie aplikácie sme preto používali vypisovanie sprav pomocou funkcie *print*, čo však malo za následok miešanie rozdielneho typu kódu. Na druhej strane však jazyk PHP obsahoval veľké množstvo funkcií pre prácu s textom a rovnako aj s konverziou premenných jazyka PHP na formát JSON.

V letnom semestri chceme zdokonaľiť KR server v prvom rade zlepšením terajšej implementácie najmä zapracovaním rôznych TODO sprav v kóde. Ďalej sa chceme venovať rozšíreniu schopností dotazov aj pre návrat informácií ohľadom učiteľov, prípadne prispôbiť dotazy pre konkrétne potreby klientskej aplikácie.

## 8.7 Úpravy modelu budovy

V tejto časti sú popísané úpravy modelov poschodia, ktoré boli získané od predchádzajúceho tímu.

Zmeny v modeloch sme uskutočnili v najnovšom Autodesk 3D Studio Max 2011, ktoré sme získali na základe študentskej licencie poskytovanej spoločnosťou Autodesk.

### 8.7.1 Pôvodný stav modelu

Ku každému poschodiu existujú tri .max (natívny formát Autodesk 3DS Max) súbory. V prvom Planes sa nachádza model rozdelený na jednotlivé obdĺžniky a jednoduché objekty, ktoré boli neskôr zlúčené, práve tento model je pravdepodobne pôvodcom veľkého množstva chýb, keďže jednotlivé roviny a jednoduchšie objekty k sebe pred spojením nepriliehali. Druhý model je Map, ktorý predstavuje všetky objekty zlúčené. Posledným je model Components v ktorom sú základné objekty zlúčené do miestností a zložitejších objektov. Zároveň sú v tomto objekte oddelené dvere. Model Components je použitý ako základ pre náš model poschodia, keďže obsahuje zlúčené objekty do takej miery, aby mohli byť použiteľné.

Pôvodný stav modelu budeme ilustrovať pomocou modelu 6. poschodia. Pôvodný model sa skladá zo 179 objektov obsahuje 9014 bodov a 4776 polygónov. Zároveň model obsahuje veľké množstvo rôznych chýb a nepresností ako sú:

- Nekonzistentne pomenované objekty (napríklad dvere)
- Prekrývajúce sa polygóny
- Prebytočné objekty, polygóny a body
- Chyby v modeli, ako sú napríklad nepriliehajúce steny

### 8.7.2 Nevyhnutné úpravy modelu

Na základe spomenutých nedostatkov modelu sme identifikovali potrebu nasledujúcich úprav:

1. Odstránenie hrán „Edges“ a odstránenie spodnej platne
2. Úprava a premenovanie dverí
3. Zlúčenie miestností do jedného objektu a nastavenie pivotu
4. Úprava modelu na základe aktuálnych plánov budovy
5. Odstránenie nepotrebných polygónov a bodov
6. Odstránenie drobných chýb a nepresností modelu

## Odstránenie hrán „Edges“ a odstránenie spodnej platne

V modeli sa nachádzala spodná biela platňa, ktorá je pre projekt nepotrebná, preto bola odstránená. Okrem toho sa v modeli nachádzali žlté hrany Edges, ktoré boli používané pre navigáciu, ale v našom modeli nie sú potrebné preto boli tiež odstránené.

## Úprava a premenovanie dverí

Dvere v modeli boli nekonzistentne pomenované. Príkladom môžu byť viaceré spôsoby pomenovania dvojitéch dverí (doordouble1\_02 a doordouble2), medzery v číslovaní dverí, iné typy pomenovania dverí (door520, door\_wc\_1004, doordouble04, door\_wc03), nejasné číslovanie (02, 1004, wc03, 520) Z toho dôvodu bol navrhnutý nový systém pomenovania:

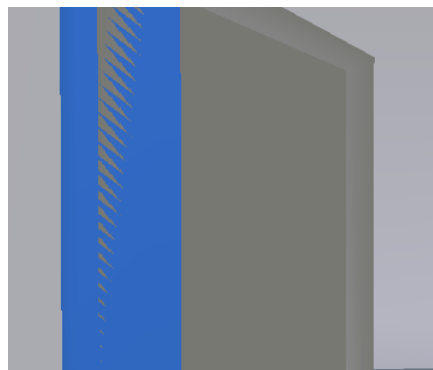
- Všetky dvere začínajú znakom *d*
- Druhý znak je *l* alebo *r* určujúci, či ide o ľavé, alebo pravá dvere
- Nasleduje trojčíselné označenie dverí, pričom prvá číslica určuje poschodie a zvyšné dve poradové číslo dverí. V prípade dvojkřídlových dverí majú obe křídla rovnaké číslo a líšia sa len v druhom znaku kde je *l* alebo *r*
- Za číslom môže nasledovať špeciálny príznak určujúci typ dverí pre prípady špeciálneho správania dverí. Dosiaľ použité príznaky sú *dd* – dvojkřídlové dvere, *wc* – záchodové dvere, *e* – výťahové dvere (výťahové dvere majú dve křídla, ale otvárajú sa iným spôsobom, preto majú iné označenie)

Príklady pomenovania:

- Obyčajné dvere – *dr633* a *dl619*
- Dvojkřídlové dvere – *dl617dd* a *dr617dd*
- Záchodové dvere – *dl634wc*
- Výťahové dvere – *dl650e* a *dr650e*

Druhým problémom bolo otváranie dverí, kde sa každé dvere otvárali rovnakým smerom a teda napríklad časť dverí sa otvárala do chodby a druhá časť do miestností. Toto bolo vyriešené pridaním príznaku *l* a *r* spolu s implementáciou otvárania, dverí, ktoré toto zohľadňuje.

Posledný problém spojený s dverami bol spôsobený zlým, nastavením pivotov dverí, čo spôsobovalo, že po otvorení dverí časť dverí prechádzala stenou. (Obr. 8.5)



Obr. 8.5 Príklad zle nastaveného pivotu dverí

Tento problém je spôsobený nastavením stredu otáčania, posunutím pivotu na správne miesto sa tento problém odstránil.

### **Zlúčenie miestností do jedného objektu a nastavenie pivotu**

V modeli boli jednotlivé miestnosti oddelené ako samostatné objekty a zároveň boli pridané špeciálne objekty Sector. Dôvodom pre takéhoto oddelenia miestností je, aby bolo možné určiť miestnosť v modeli v prehliadači. V našom modeli to už nie je potrebné, keďže na určovanie miestností sú použité výškové mapy a nie je preto potrebné oddeľovať miestnosti aj v modeloch. Preto boli miestnosti a objekty Sector zlúčené do jedného objektu floor6 (respektíve floorX podľa poschodia). Okrem miestností boli do jedného objektu zlúčené aj všetky ostatné objekty okrem dverí, ktoré sú predstavované samostatnými objektmi. Dvere zostali zoskupené v skupine Doors.

Okrem tejto zmeny bol upravený aj pivot takto zlúčeného objektu a pivot skupiny Doors. Keďže bol vytvorený nový zlúčený objekt bolo možné nastaviť pivot spoločne pre celý objekt a preto sme ho nastavili na súradnice bodu [0,0,0] zároveň sa, ale zistilo, že pivot je otočený vzhľadom na súradnicový systém modelu, preto bol pivot aj natočený tak, aby bol v súlade so súradnicovým systémom modelu. Rovnako bol upravený aj pivot skupiny Doors.

### **Úprava modelu na základe aktuálnych plánov budovy**

Na základe aktuálnych plánov budovy budú modely poschodí upravené tak, aby boli v súlade s týmito plánmi. Z dôvodu neskorého dodania plánov však nebolo možné tieto zmeny vykonať ešte v termíne odovzdania prototypu. Preto bude kompletný zoznam úprav a aj samotné úpravy vytvorený, až v ďalšej časti projektu. Vzhľadom na veľké množstvo rozdielov medzi aktuálnymi plánmi a modelom sa ako jedna z možností javí aj vytvorenie nového modelu, čo by mohlo byť menej časovo náročné, ako snaha upraviť existujúci model.

### **Odstránenie nepotrebných polygónov, hrán a bodov**

V pôvodnom modeli sa nachádzalo veľké množstvo prekrývajúcich sa bodov a polygónov, alebo nepotrebných hrán, tieto boli v rámci úprav zlúčené, respektíve nahradené iným polygónom pokrývajúcim celú plochu. Prípadne boli rozmery upravené tak, aby sa polygóny neprekrývali. Asi najpodstatnejšou zmenou bola úprava podlahy, ktorá bola tvorená desiatkami polygónov. Tieto polygóny boli nahradené jedným polygónom. Zlúčenie polygónov sa vykonalo zlúčením (Weld - zvarenie) bodov, alebo odstránením polygónov a ich nahradením iným polygónom pokrývajúcim rovnakú plochu.

### **Odstránenie drobných chýb a nepresností modelu**

Okrem všetkých už spomenutých chýb v modeli sa tu nachádzajú aj ťažšie spozorovateľné chyby ako sú napríklad:

- Nepriliehajúce steny
- Presahujúce steny (respektíve niektoré polygóny steny)
- Prekrývajúce sa steny, okná a iné

Hľadanie a oprava týchto chýb je veľmi náročná a nie je jednoduchá preto je možné, že sa časť týchto chýb dosiaľ nepodarilo opraviť

### 8.7.3 Súčasný stav

V novom modeli 6. poschodia sa nachádza už len 60 objektov z ktorých 58 tvoria samostatné dvere. Počet bodov bol zredukovaný na 6646 a počet polygónov na 3023. Napriek podstatnému zníženiu bodov a polygónov to žiadnym spôsobom neobmedzilo funkčnosť modelu. Napriek rozsiahlym úpravám sa v modeloch stále pravdepodobne nachádza niekoľko drobných chýb, keďže nájsť všetky chyby je veľmi zložité a aj časovo náročné.

## 8.8 Výškové mapy

Výšková mapa predstavuje farebné označenie prvkov modelu. Pričom farba určuje miestnosť a výšku v modeli vzhľadom na základnú výšku. Výškové mapy sú uložené vo formáte PNG a pre každé poschodie modelu existuje jedna výšková mapa. Jednotlivé časti mapy sú ofarbené špecifickou farbou. V mape sa rozlišujú dva základné prvky, priechodné a nepriechodné prvky. Priechodné prvky sú:

- miestnosť
- dvere
- chodba
- schodiská

Nepriechodné prvky sú steny, respektíve prázdne priestranstvo.

Každý z prvkov výškovej mapy má priradenú farbu. Farby sú priradené podľa zoznamu farieb uloženého v zoznam-farieb.xlsx. V zozname farieb sú v jednotlivých hárkoch uložené informácie pre špeciálne farby a pre jednotlivé poschodia 0 až 7.

Záznam v zozname je zložený z názvu a farby. Farba je vyjadrená hexadecimálne aj vo forme RGB zložiek v desiatkovej sústave. Na kódovanie farieb sa používa 24 bitov, čo predstavuje 6 hexadecimálnych hodnôt ľavé dve hodnoty určujú výšku vzhľadom na základnú výšku poschodia. Zvyšné 4 hodnoty kódujú prvok mapy, teda na určenie prvku mapy sa používa 16 bitov, čo umožňuje kódovať 65536 prvkov s hodnotami od 0 po 65535. Toto je zbytočne veľký počet, keďže počet prvkov je menší ako 512, preto bude stačiť krok farieb 128, teda sa bude dať kódovať práve 512 miestností. Na určenie výška sa používa 8 bitov, čo umožňuje kódovať 256 hodnôt výšky s hodnotami od 0 po 255.

Najnižšia farba kódujúca miestnosť v hexadecimálnom zápise je xx0000 a najvyššia xxxfff. Hodnoty farieb sa priradujú postupne prvkom od najvyššieho poschodia po najnižšie. Medzi jednotlivými po sebe idúcimi farbami prvkov je rozdiel 128 hodnôt, teda prvá miestnosť na 7. poschodí má číslo xx0000 a nasledujúca xx0080.

Špeciálne farby sa priradujú v opačnom poradí oproti bežným prvkom, teda od najvyššej hodnoty xxxfff až po xx0000, pre špeciálne farby neplatí pravidlo o priradovaní farieb prvkom, teda krok medzi farbami je 1 a nie 128.

Špeciálne farby sú nasledovné:

- steny a neprístupné oblasti – 00ffff
- dvere – 00feff
- vrch schodiska – 00fdff



- výťah 1 – 00fcff
- výťah 2 – 00fbff
- výťah 3 – 00faff
- výťah 4 – 00f9ff

Číslovanie výťahov je v smere hodinových ručičiek, výťah 1 je prvý vľavo.

## 8.9 Zhodnotenie prototypu

Podľa stanovených priorit implementácie v kapitole 8.1 uvádzame zhodnotenie ich plnenia:

- 3D klient
  - funkčnosť minuloročného riešenia je implementovaná (prehliadanie, prechádzanie, označovanie miestností, kolízie)
  - prechádzanie cez zatvorené dvere nie je dovolené
  - z výťahu je zatiaľ implementované otváranie dverí, ďalšiu funkčnosť budeme vyvíjať, až keď bude načítane aj ďalšie poschodie
  - zobrazovanie názvu miestnosti pri kliknutí
- 2D a mobilný klient
  - je vytvorený schematický plán jedného poschodia, bude sa však meniť, keďže sa zmenili plány budovy
  - nestihli sme označovanie miestností, vyhľadávanie a zobrazovanie miestností v 2D klientovi
- Ku všetkým rozhraniam klienta
  - je spravený základný prototyp používateľského rozhrania, v letnom semestri ho bude nutné vylepšiť
- Model
  - nové plány sme dostali až na koniec semestra, preto potrebné aktualizácie neboli spísané
  - upravené je jedno poschodie, ale len chyby pri modelovaní a rôzne úpravy pre potreby vykresľovania (napr. zmena označenia dverí)
- Server
  - pre rozvrhy v miestnosti funguje import údajov z AIS
  - rozvrhy v miestnosti je možné načítavať komunikačným rozhraním (serverová aj klientská časť)

Po zbežnom preskúmaní nových plánov budovy FIIT sme zistili, že v plánoch je oproti súčasnému modelu veľa zmien. Preto sme sa rozhodli, že model vytvoríme nanovo, pretože si myslíme, že to bude rýchlejšie, ako upravovať existujúci model podľa nových plánov a zároveň upravovať chyby v existujúcom modeli (napr. nedoliehajúce polygóny).

Do letného semestra teda pribúda vytvorenie modelu budovy podľa nových plánov. Aktualizovaný zoznam práce do letného semestra vyzerá nasledovne:

- implementovať navigáciu
- vytvoriť nové modely a plány poschodí pre 2D a mobilného klienta
- implementovať funkcionality, ktorá bola v prototypu implementovaná iba čiastočne

- import a zobrazovanie ostatných údajov z AIS
  - používateľské rozhrania pre všetkých klientov
- implementácia získavania osobných rozvrhov z AIS
- optimalizácia rýchlosti načítavania a množstva prenášaných údajov

## 9 Model budovy FIIT

---

### 9.1 Postup vytvárania modelu

Na základe plánov novej budovy FIIT, ktoré sme dostali na konci zimného semestra, sme sa rozhodli, že bude potrebné vytvoriť nový model budovy. Model minuloročného tímu sa totiž dosť líši od najnovších plánov, a preto sme sa rozhodli neupravovať ho, ale vytvoriť vlastný.

Namodelovanie celej budovy je však časovo veľmi náročná úloha. Jednotlivé poschodia sa ale skladajú z rovnakých prvkov: rôzne druhy stien, okien a dverí. Dostali sme teda nápad, či by nebolo možné namodelovať len základ poschodia a nejakým spôsobom určiť, kde bude stena, okno, či dvere. Následne by sa na základe vytvoreného pôdorysu programom vytvoril 3D model. Takýto prístup má niekoľko výhod:

- redukcia času potrebného na modelovanie
- väčšia kvalita modelu – keďže 3D model sa bude vytvárať programom, nebude dochádzať k chybám, ako sú napríklad neodliehajúce alebo prekrývajúce sa steny
- ľahšie úpravy modelu – stačí len upraviť pôdorys a nechať znovu vygenerovať 3D model

Nevýhodou je počiatočný čas potrebný na vytvorenie nástroja, ktorý bude model vytvárať.

Na modelovanie používame nástroj 3ds max 2011, ktorý obsahuje zabudovaný skriptovací jazyk MaxScript. Pomocou neho je možné ovládať väčšinu funkcií, ktoré tento nástroj poskytuje. Skripty na vytváranie modelu sme preto vytvorili práve v MaxScripte.

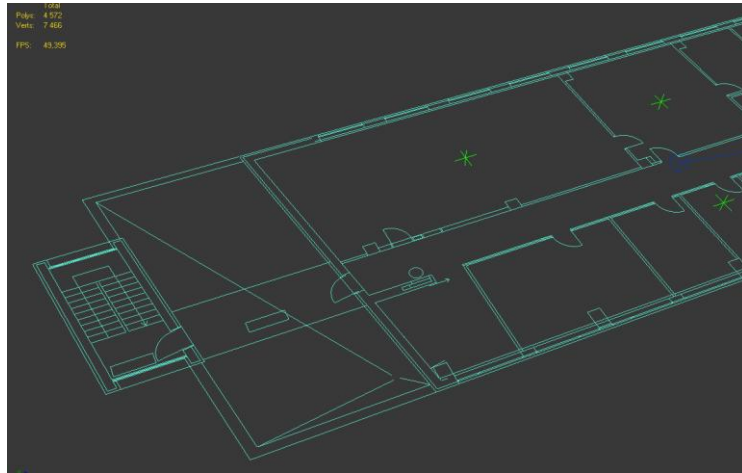
Vytvorené skripty z pohľadu modelovania umožňujú:

- z namodelovaného základu poschodia vytvoriť 3D model, konkrétne:
  - steny
  - okná
  - otvory pre dvere spolu so zárubňami
- do otvorov pre dvere umiestniť dvere, ktoré budú správne pomenované

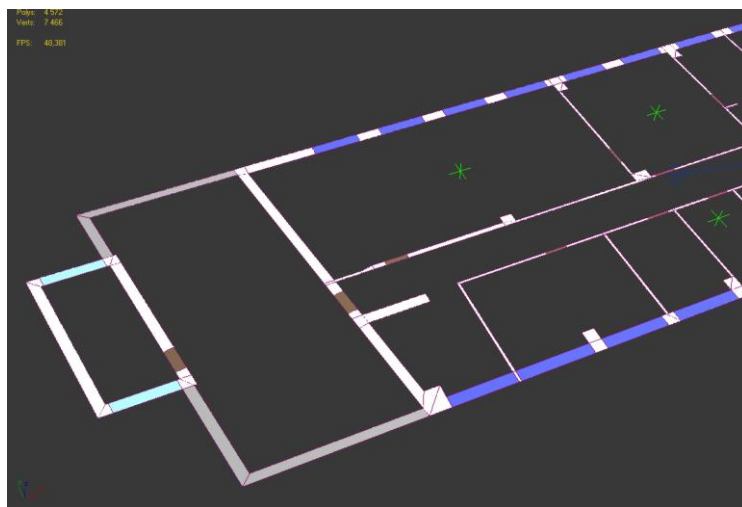
Príklad počiatočného pôdorysu poschodia je na Obr. 9.1. Na Obr. 9.2 je potom namodelovaný základ poschodia spolu s určenými oknami, dverami a stenami. Na Obr. 9.3 je výsledný model, ktorý vznikol použitím vytvorených skriptov. Ručne je potrebné ešte doplniť schodiská, podlahu a prípadne aj textúry.

Na základe fungujúcej automatizácie modelovania sme sa rozhodli zautomatizovať aj ďalšie časti práce a to vytváranie výškovej mapy a vytváranie zoznamu miestností a ich farieb, čo sme v zimnom semestri robili ručne.

Celý systém exportu modelu poschodia z 3ds max 2011 do podoby použiteľnej v internetovom prehliadači je znázornený na Obr. 9.4 (obdĺžniky predstavujú nejaký súbor, v zátvorke majú uvedený formát. Ovál predstavuje nejakú operáciu, ktorú vykoná program, alebo skript, v zátvorke je určený názov programu, resp. programovacieho jazyka). Jednotlivé operácie sú ďalej opísané. Príklady súborov sú uvádzané pre 6. nadzemné podlažie.



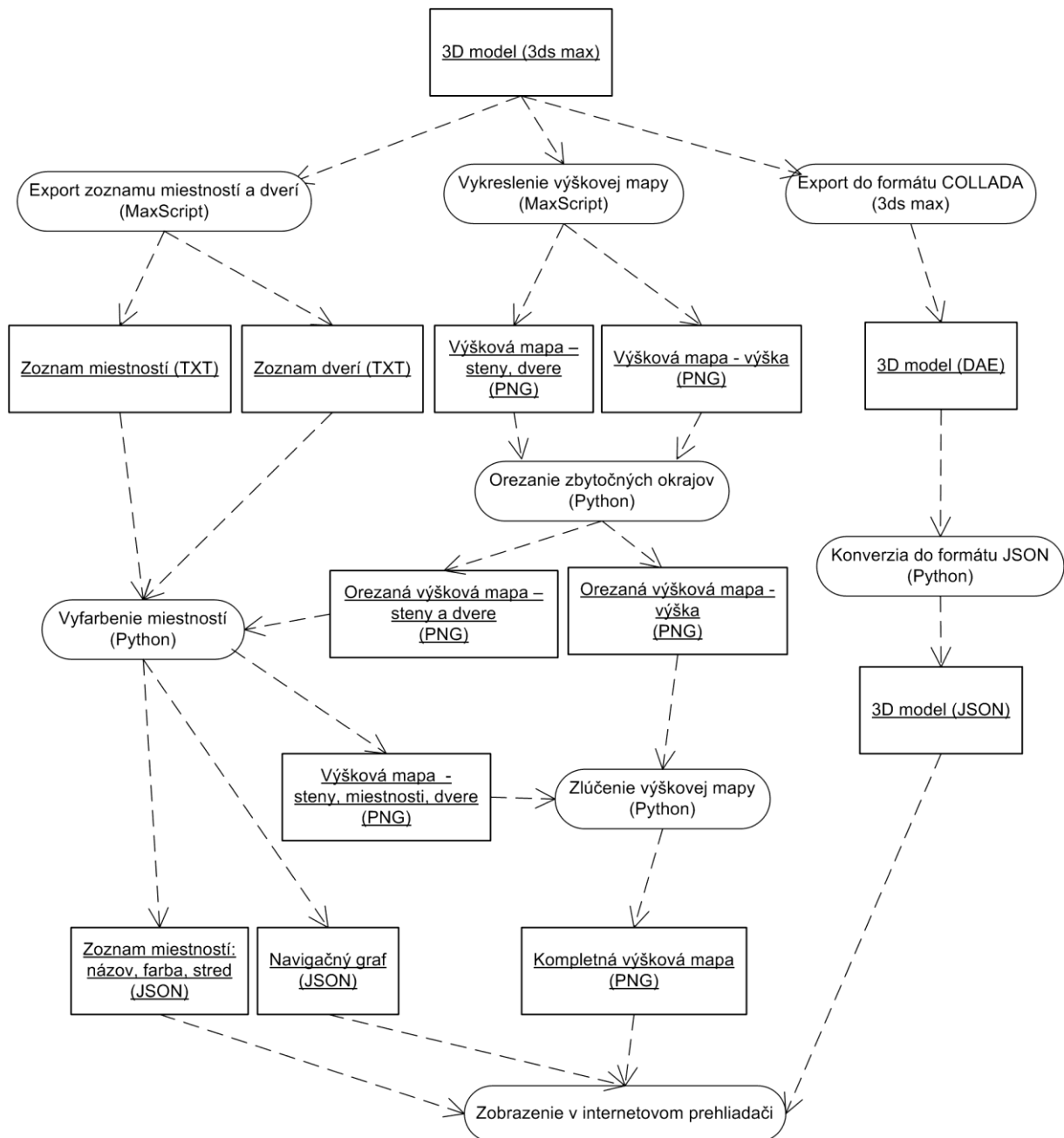
Obr. 9.1 Pôdorys poschodia



Obr. 9.2 Namodelovaná časť základu poschodia



Obr. 9.3 Výsledný 3D model po spustení skriptu



Obr. 9.4 Systém exportu modelu poschodia

### 3D model

Vstupom je 3D model poschodia, ktorý vznikol z namodelovaného základu poschodia. Ďalej v ňom boli identifikované miestnosti pomocou objektu typu Point, ktoré boli umiestnené do vnútra miestností a pomenované podľa názvu miestnosti, v ktorej sa nachádzajú. Príklad identifikácie miestnosti v aplikácii 3ds max je na Obr. 9.5. Ide o súbor fiit\_6NP.max.

### Export zoznamu miestností a dverí (MaxScript)

Všetky identifikované miestnosti sa vyexportujú do textového súboru, v ktorom bude názov miestnosti a jej pozícia v modeli. Tiež sa vyexportujú rozmery celého modelu. Ide o súbor fiit\_6NP\_room\_points.txt. Príklad z tohto súboru (minimálna súradnica celého modelu, maximálna súradnica celého modelu a identifikácia 2 miestností):

MIN;-50.6959;0.0;0.0

MAX;54.4488;16.5431;3.6

6.08

6.07 Terasa;-43.5901;8.2126

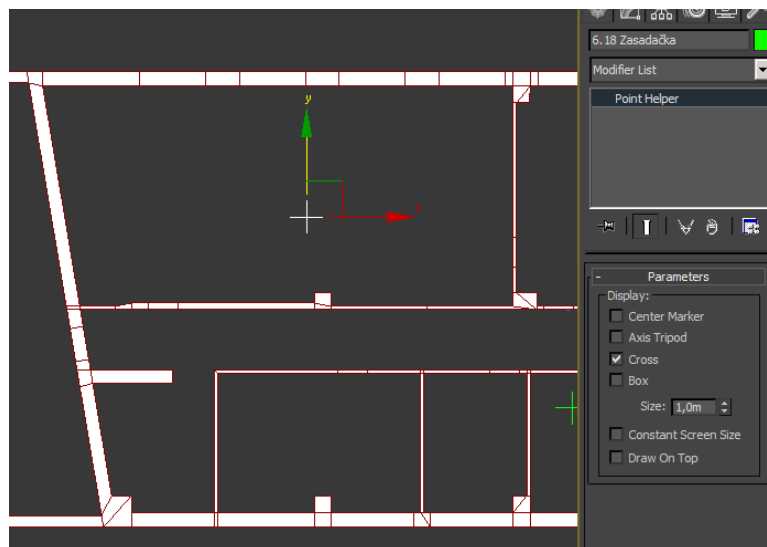
Schody;-48.1373;6.14001

Tiež sa vyexportuje aj zoznam dverí spolu s ich orientáciou do súboru fiit\_6NP\_doors.txt. Prvé tri riadky vyzerajú nasledovne (v každom riadku je pozícia a orientácie pre jednu dvere):

-7903.74,6587.5,0.0,0.0,-1.0,0.0

-11755.4,6600.0,0.0,0.0,1.0,0.0

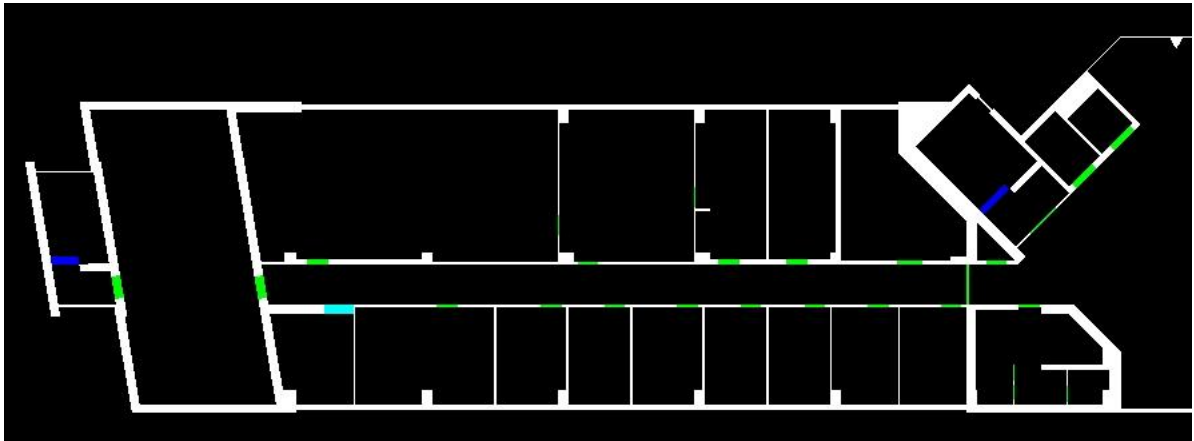
-16700.0,6625.0,0.0,0.0,1.0,0.0



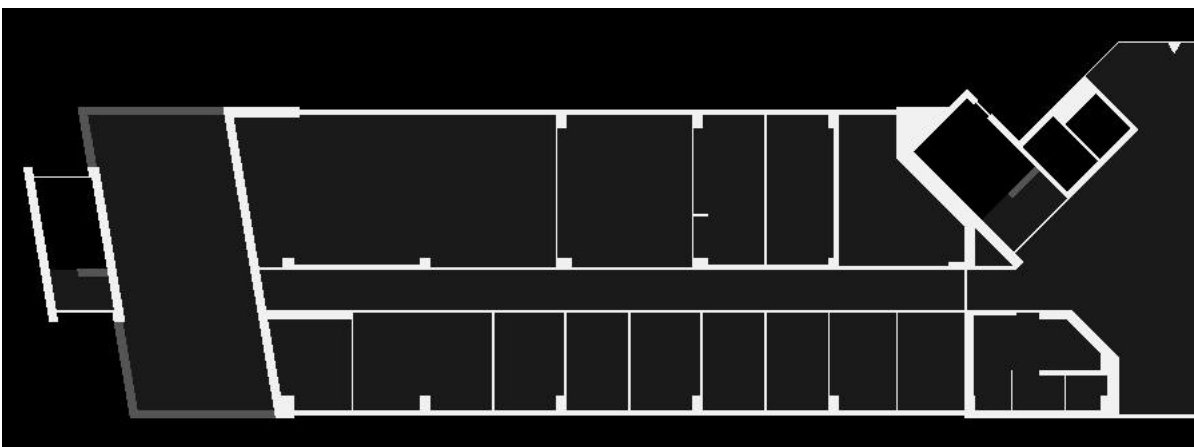
Obr. 9.5 Príklad identifikácie miestnosti

### Vykreslenie výškovej mapy (MaxScript)

Výstupom sú 2 obrázky. Najprv sa vykreslí základ modelu v pohľade zvrchu, v ktorom sú však farebne odlišené dvere (zelená farba), oblasti kde sa mení poschodie (modrá farba) a virtuálne dvere (tyrkysová farba), všetky ostatné prvky sú označené rovnakou bielou farbou. Výstupom je farebná časť výškovej mapy (Obr. 9.6) v súbore fiit\_6NP\_color.png. Potom sa vykreslí 3D model v pohľade zvrchu do čiernobielej výškovej mapy (súbor fiit\_6NP\_depth.png), kde čierna farba predstavuje minimálnu výšku a biela maximálnu výšku (Obr. 9.7).



Obr. 9.6 Výšková mapa – farby (steny a dvere)



Obr. 9.7 Výšková mapa - výška

### Orezanie zbytočných okrajov (Python)

Vo oboch častiach výškovkej mapy sú zbytočné okraje, ktoré sa orežú tak, aby mali obrázky minimálny potrebný rozmer. K názvu oboch obrázkov sa pripojí prípona „\_cropped“.

### Vyfarbenie miestností (Python)

Pre každú miestnosť zo zoznamu miestností sa určí, kde sa v obrázku nachádza. V danom mieste (pixeli) sa spustí vyfarbovanie novou farbou, ktorou sa vyplní celá plocha miestnosti. Výstupom je vyfarbený obrázok a tiež zoznam miestností vo formáte JSON (súbor fiit\_6NP\_rooms.js), kde ku každej miestnosti je určený jej názov a tiež farba, ktorou je v obrázku identifikovaná. Príklad :

```

vfiit.floors['6NP'].rooms = [
  {
    "center": {
      "x": 42.46999154691467,
      "y": 186.68977176669489
    },
    "color": 6500,
    "id": "6.08",
    "name": "Schody"
  },

```

Zároveň sa vytvorí aj navigačný graf, ktorý určuje prepojenie miestností dverami. Prepojenie sa určuje podľa susediacich farieb vo vyfarbenej výškovej mape. Príklad zo súboru fiit\_6NP\_navigation.js, kde vidieť uzol grafu predstavujúci miestnosť a uzol grafu predstavujúci dvere:

```
"6013": {
  "adjacent": [
    "6127"
  ],
  "type": "room"
},
"6100": {
  "adjacent": [
    "6500",
    "6501"
  ],
  "orientation": {
    "x": -0.9860189999999998,
    "y": -0.16663500000000001,
    "z": 0.0
  },
  "pos": {
    "x": -46.617800000000003,
    "y": 5.49716,
    "z": 0.0
  },
  "type": "door",
  "weight": 1
},
```

Pri dverách je pozícia a orientácia, ktorá sa načíta zo zoznamu dverí. Tieto údaje sú potrebné, aby bolo možné správne umiestniť navigačnú šípku pred dané dvere. Položka „weight“ je váha daných dverí, ktorá sa využije pri hľadaní cesty pre navigáciu. Pre potreby navigácie sme potrebovali iba niektorým dverám určiť vyššiu váhu. S ohľadom na jednoduchosť určovania váh sme to vyriešili tak, že do kópie súboru fiit\_6NP\_color\_cropped.png sme červenou farbou označili tie dvere, ktorým sa má priradiť vyššia váha. Súbor s váhami fiit\_6NP\_door\_weights.png vyzerá ako na Obr. 9.8.

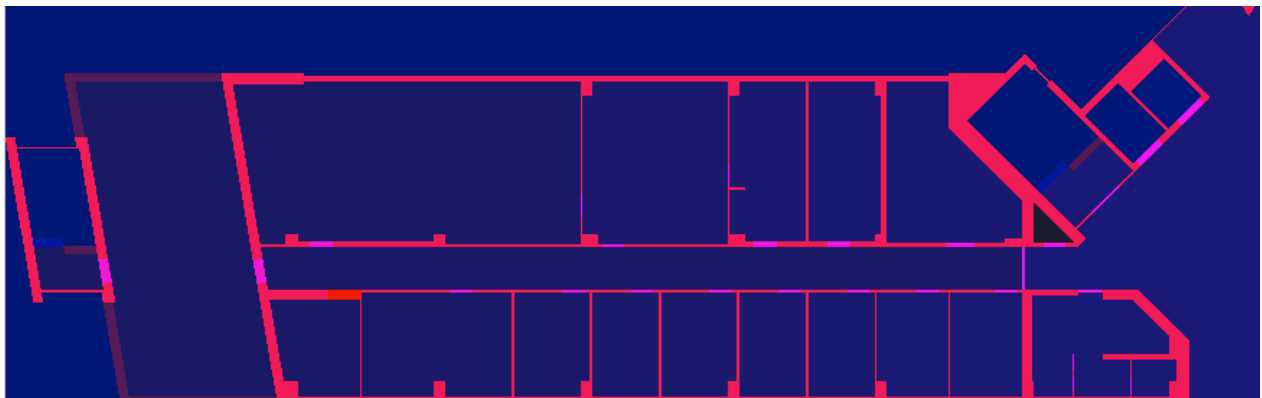




Obr. 9.8 Určenie váh dverí

### Zlúčenie oboch častí výškovej mapy (Python)

Do farebnej časti výškovej mapy sa pridá výška z čiernobielej výškovej mapy ako červený kanál. Vo výslednej výškovej mape je teda červený kanál vyhradený pre výšku a zelená farba spolu s modrou identifikujú miestnosti, steny a dvere. Príklad výslednej výškovej mapy je na Obr. 9.9. Farby miestností nasledujú tesne za sebou, preto tieto rozdiely na obrázku nevidieť.



Obr. 9.9 Výsledná výšková mapa

### Export do formátu COLLADA (3ds max)

Ide o export z 3ds max 2011 do formátu COLLADA (fiit\_6NP.DAE). Je to XML súbor opisujúci celý model.

### Konverzia do formátu JSON (Python)

Ide o jednoduchý skript, ktorý prevedie COLLADA formát ako XML do formátu JSON, s ktorým sa v JavaScripte jednoduchšie pracuje (fiit\_6NP.json).

### Zobrazenie v prehliadači

Na zobrazenie modelu v prehliadači spolu s funkčnosťou kolízií označovania miestností a navigácie sú teda potrebné tieto 4 súčasti:

- 3D model vo formáte JSON
- Zoznam miestností na danom poschodí spolu s ich farbami vo výškovej mape
- Výšková mapa, ktorá určuje miestnosti, steny, dvere a výšku podlahy
- Navigačný graf

## 9.2 Opis súborov so skriptami

Skripty v jazyku Python boli testované vo verzii 2.6.1. Vyžadujú knižnicu ElementTree (testované s verziou 1.2.6<sup>37</sup>) a Python Imaging Library (testované s verziou 1.1.7<sup>38</sup>). Súbory sú nasledovné:

- ColladaConverter.py - konverzia Collada formátu z XML do formátu JSON
- Crop.py – orezanie vykreslenej výškovej mapy na minimálnu veľkosť
- CreateHeightMap.py - vytvorenie výškovej mapy aj so zoznamom miestností, vytvorenie navigačného grafu

Nasledovné skripty v MaxScripte boli vytvorené v Autodesk 3ds max 2011:

- Config.ms – konfigurácia vytvárania modelu zo základu, sú tam určené rozmery všetkých prvkov (steny, dvere, okná) a ďalšie nastavenia
- ExportTools.ms – všetky veci týkajúce sa exportu, renderovanie výškovej mapy a volanie príslušných Python skriptov
- ExtrudeDoor.ms – vytvorenie otvorov pre dvere a umiestnenie dverí do vytvorených otvorov
- ExtrudeWindow.ms – vytvorenie okien
- Rollout.ms – používateľské rozhranie k poskytovanej funkcionalite
- SuperExtrude.ms – vytvorenie modelu zo základu
- Utils.ms – pomocné funkcie
- Checks.ms – kontroly modelu, či je správne pripravený na export (kontroly pomenovania, správneho umiestnenia pivotu, a pod.)

## 9.3 Štruktúra modelov

Po zhodnotení dodaných plánov sme sa rozhodli namiesto upravovania pôvodných modelov vytvoriť nové založené na týchto plánoch. Ako podklad sa použili najnovšie dostupné plány z leta 2010. Vytvorených bolo deväť modelov, jeden pre každé podlažie od druhého podzemného až po siedme nadzemné podlažie.

V rámci modelov sa nachádzajú rôzne objekty, ktoré sa dajú rozdeliť do niekoľkých kategórií:

1. Objekty, ktoré sa nachádzajú na každom poschodí.
2. Špecifické objekty pre niektoré poschodia.
3. Pomocné objekty, ktoré sú používané, len počas modelovania, ale nie vo výslednom vyexportovanom formáte.

V každom modeli bolo vytvorených niekoľko základných objektov:

- Base
- Model
- Floor
- Doors
- Helpers
- FloorBottom

Špecifické objekty pre niektoré poschodia:

- Ceiling

<sup>37</sup> <http://effbot.org/media/downloads/elementtree-1.2.6-20050316.win32.exe>

<sup>38</sup> <http://effbot.org/downloads/PIL-1.1.7.win32-py2.6.exe>

- Elevators
- Turniket

Pomocné objekty:

- Prototypy dverí
- Splines

## Base

Ide o základňu vytvorenú podľa objektu Splines. Base má priradený materiál BasePlan a podľa ID hodnoty v rámci tohto materiálu, ktorá je priradená jednotlivým polygónom objektu base, je určené, čo daný polygón predstavuje. Takto označené polygóny sa potom dajú použiť na automatické vytvorenie modelu z objektu Base.

**Tab. 1 Zoznam materiálov pre objekt Base**

Material ID	Názov	Popis
1	BWall	Obyčajná stena
2	BWindow	Štandardné okno na fasáde
3	BDoor	Jednokrídlové drevené dvere
4	-	Nepoužité
5	BSmallWall	Múrik okolo otvoreného priestoru pri bočných schodiskách.
6	BLargeWindow	Veľké okná na bočných schodiskách.
7	BWcWindow	Malé okno na záchodoch.
8	BGlassWall	Sklenená stena (napr. presklenie centrálného priestoru na poschodí, alebo stena s dverami pri hlavnom schodisku).
9	BWoodDoubleDoor	Drevené dvojkrídlové dvere.
10	BGlassDoubleDoor	Sklené dvojkrídlové dvere (napr. pri hlavných schodiskách)
11	BElevatorDoor	Výťahové dvere
12	-	Nepoužité
13	BWcDoor	Dvere na toalety
14	BFloorChange	Miesto, kde sa prechádza na ďalšie poschodie
15	BVirtualDoor	Vytvorí sa otvor pre dvere, ale samotné dvere do neho nebudú umiestnené

## Model

Model predstavuje už finálnu podobu stien, okien a iných častí budovy. Podstatná časť modelu je vytvorená pomocou skriptu, ktorý vytvorí model z objektu Base. Niektoré časti ale museli byť ručne upravené, ako napríklad prednáškové miestnosti, ktoré majú steny s rôznymi výškami a podobne. Toto sa vzťahuje najmä na poschodia 1PP a 1NP. Ostatné poschodia boli takmer celé (okrem drobných úprav) vytvorené pomocou skriptu. Tak ako Base, tak aj model má priradený materiál Model, kde sú tiež pomocou ID rozlíšené konkrétne materiály. Tento materiál je však špecifický pre každé poschodie, keďže na niektorých poschodiach sa nachádzajú špecifické materiály, ktoré na iných nie sú potrebné. Materiál Modelu je odvodený od materiálu v Base, napríklad pre všetky sklenené objekty s Base ID 2, 6, 7 a 8 je použitý materiál Model číslo 102 – Window.

## Floor

Objekt Floor predstavuje podlahu podlažia. Vzhľadom na komplikovanosť automatického generovania bola podlaha vytváraná ručne s použitím objektu Base ako podkladu. Súčasťou podlahy sú aj všetky schodiská na príslušnom podlaží. Ručné vytváranie podlahy zjednodušil fakt, že podlahy na vyšších podlažiach (2. až 6. nadzemné podlažie) sú takmer totožné a bolo možné použiť opakovane len s minimom úprav. Pre ostatné podlažia (2. a 1. podzemné podlažie, 1. a 7. nadzemné podlažie) museli byť vytvorené vlastné podlahy. Na podlahu je použitý materiál Model s ID 122 – Podlaha.

## Doors

Doors predstavuje skupinu objektov dverí. Tie sú všetky generované automaticky pomocou skriptu, ktorý na základe hodnôt ID materiálu a špeciálneho bodu určujúceho orientáciu dverí, vytvorí dvere spolu so zárubňou určeného typu vrátane ich špeciálneho natočenia. Dvere sú vytvárané na základe prototypov dverí, ktoré sa nachádzajú v každom modeli ako pomocné objekty. Vytvorí sa vlastne kópia tohto prototypu, pričom sa upraví veľkosť a umiestnenie na správne miesto v modeli.

Samotné dvere sú tiež automaticky pomenovávané, podľa novej konvencie, ktorá sa mierne líši oproti konvencie použitej v prototypu. Začína sa názvom *door*, nasleduje znak „\_“ a za ním písmeno *l*, alebo *r* určujúce orientáciu a teda otváranie dverí na ľavú alebo pravú stranu. V prípade, že ide o jednu časť dvojkřídlových dverí nasleduje písmeno *d* a nakoniec trojciferné označenie čísla dverí. Obe časti dvojkřídlových dverí majú toto číslo rovnaké. Špeciálnym prípadom sú výťahové dvere, ktoré sú všetky zložené z dvoch častí, pričom tieto časti sa zasúvajú na tú istú stranu. Kvôli tomu je označenie mierne upravené a za písmenom určujúcim orientáciu zasúvania nasleduje číslo časti 1 alebo 2, pričom číslo 1 má priradená časť bližšie k stene ku ktorej sa budú dvere zasúvať, za týmto číslom je písmeno *e* určujúce, že ide o výťahové dvere a nakoniec opäť trojciferné číslo dverí.

## Helpers

Ide o skupinu objektov z kategórie Helpers typu Point. Tie sú použité na označovanie miestností, kde názov objektu slúži ako názov miestnosti. Tieto objekty sa využívajú na označenie miestnosti pri generovaní navigačných grafov a výškových máp, ale vo finálnom modeli sa nevyskytujú. Názvy miestností boli získané z plánov budovy.

## FloorBottom

FloorBottom je pomocný objekt, ktorý slúži, pri generovaní výškovej mapy, ako podklad určujúci priestor, ktorý nie je prístupný. Po vygenerovaní výškovej mapy sa ďalej nepoužíva, preto sa ani nevyskytuje vo vyexportovanom modeli.

## Ceiling

Ide o strop miestností, iný názov by mohol byť aj strecha (Roof). V modeloch sa tento objekt nachádza iba na 2. a 7. nadzemnom podlaží, keďže na všetkých ostatných podlažiach je stropom pre dané podlažie podlaha podlažia, ktoré sa nachádza nad ním. Strop je podobný podlahe, a tiež ho nebolo možné vygenerovať, ale bolo nutné vytvárať ho ručne. Strop na 7. nadzemnom podlaží používa špeciálnu textúru napodobujúcu kameň, čím sa dotvára realistickejší obraz. Strop na 2. nadzemnom podlaží obsahuje aj strešné okná a bol vytváraný na základe vizualizácií a plánov budovy.

## **Elevators**

Ide o skupinu štyroch objektov na 1. nadzemnom podlaží, ktoré predstavujú výťahové kabíny. Tie sa v konečnej podobe pohybujú vo výťahovej šachte, ktorá prechádza každým poschodím, preto sú oddelené ako samostatné objekty.

## **Turniket**

V modeloch 1. nadzemného a 1. podzemného podlažia sa nachádzajú turnikety, ktoré boli vytvorené ručne ako samostatné objekty, aby s nimi bolo možné v aplikácii otáčať podobne ako s výťahovou kabínou.

## **Prototypy dverí**

Ide o špeciálne pomocné objekty predstavujúce vzory dverí, ktoré sa používajú pri generovaní dverí pomocou skriptu.

## **Splines**

Ide o podklad na vytváranie objektu Base. Pre každé podlažie je tento objekt špecifický a predstavuje pôdorys tohto podlažia. V podstate je to zjednodušený plán importovaný z formátu DWG do 3ds max 2011, pričom pri importovaní boli z plánov vynechané mnohé, pre nás nepotrebné, vrstvy. Tieto objekty však budú z modelov, ktoré budú sprístupnené odstránené z dôvodu ochrany práv autorov plánov budovy FIIT a prístupné budú teda len odvodené objekty vytvorené naším tímom.

## **9.4 Aktuálny stav a porovnanie s minuloročným tímom**

V súčasnosti sú namodelované všetky podlažia novej budovy FIIT od 2. podzemného až po 7. nadzemné podlažie – spolu 9 podlaží podľa posledných dostupných plánov. Na každom podlaží boli namodelované všetky miestnosti, dvere, okná, podlaha, schodiská a strop.

Oproti modelom predchádzajúceho tímu boli doplnené, alebo vylepšené niektoré časti modelov. Zoznam vylepšení, alebo nových častí modelov:

- Pridanie 2. podzemného podlažia – v minuloročných modeloch sa toto podlažie vôbec nenachádzalo
- Namodelovanie strechy
- Pridanie textúr – namiesto obyčajného nafarbenia stien sme na vybrané povrchy pridali textúry
- Namodelovanie výťahov a turniketov
- Úpravy podľa nových plánov reálnej budovy a vizualizácií - napríklad prednáškové miestnosti majú klesajúci strop
- Veľkosť modelu bola prispôbená realite – model predchádzajúceho tímu používal ako jednotku 1 palec náš model používa ako jednotku 1 milimeter (asi 1/25 palca), teda celý náš model je asi 25 krát väčší (v prostredí 3ds max) ako model predchádzajúceho tímu, čo umožňuje aj detailnejšie úpravy modelov.

### **9.4.1 Možnosti ďalších vylepšení**

Aj napriek vylepšenému a zrýchlenému postupu modelovania stále zostáva ďalšia práca do budúcnosti. Možné ďalšie úpravy, ktoré nebolo možné zrealizovať z časových alebo technických dôvodov:

- Upraviť model podľa reálnej budovy – Plány podľa ktorých sa modelovalo sú z leta 2010 a je možné, že boli odvtedy upravené. Rovnako plány neboli schopné zachytiť všetky podrobnosti, a preto sa model môže líšiť aj v iných bodoch. Preto by bolo vhodné po dostavaní budovy upraviť model podľa reálnej dokončenej budovy FIIT.
- Doplniť parkovisko a dva zadné vchody – Z medziposchodí hlavných schodísk 1. a 2. podzemného podlažia vedú podľa plánov dva zadné vstupy do budovy, ktoré by mali byť napojené na parkovisko za budovou.
- Upraviť textúry podľa reálnej budovy (farby stien, materiály, podlaha, schodiská, ...)
- Doplniť zábradlia na schodiskách
- Pridať interaktívne aj neinteraktívne prvky (stoly, stoličky, ostatné vybavenie miestností, lavice do prednáškových miestností, ...)
- Odstrániť drobné chyby modelov – Napriek vylepšenému a poloautomatickému systému modelovania stále na niektorých miestach mohli vzniknúť nepresnosti ako môžu byť nedoliehajúce steny. Tieto je veľmi ťažké nájsť a napriek našej snahe sa nám pravdepodobne nepodarilo nájsť všetky takéto nepresnosti.

## 10 Implementácia klientskej časti aplikácie

---

### 10.1 Navigácia

Navigáciu na rozdiel od minuloročného tímu, ktorý používal súvislú navigačnú čiaru, riešime pomocou výrazných šípok, ktoré umiestňujeme pred dvere a na iné vhodné miesta, aby navigovali používateľa na cieľové miesto. Ukážka je na Obr. 10.1.



Obr. 10.1 Navigačné šípky v modeli

Takýto spôsob sme si zvolili hlavne z dôvodu jednoduchosti a rýchlosti implementácie. Vytvorenie navigačných čiar by totiž bolo pomerne náročné automatizovať. Pre zobrazenie šípok však potrebujeme vedieť v podstate iba pozície a orientáciu dverí, ktoré sa dajú automaticky vyexportovať z modelu.

Základom pre navigáciu sú navigačné grafy, ktoré sa vytvoria skriptom z výškovej mapy. V navigačnom grafe je určené prepojenie miestností a dverí. Navigačné grafy pre jednotlivé poschodia je však potrebné zlúčiť do jedného spoločného grafu, aby navigácia fungovala medzi ľubovoľnými dvoma miestnosťami na rôznych poschodiach.

Navigačné grafy sa prepájajú tým spôsobom, že sa vytvorí ďalší uzol grafu predstavujúci schody. Tento uzol prepája schodiskové miestnosti medzi podlažiami. Prepojenia boli vytvorené manuálne v súbore vfiit.conf.js:

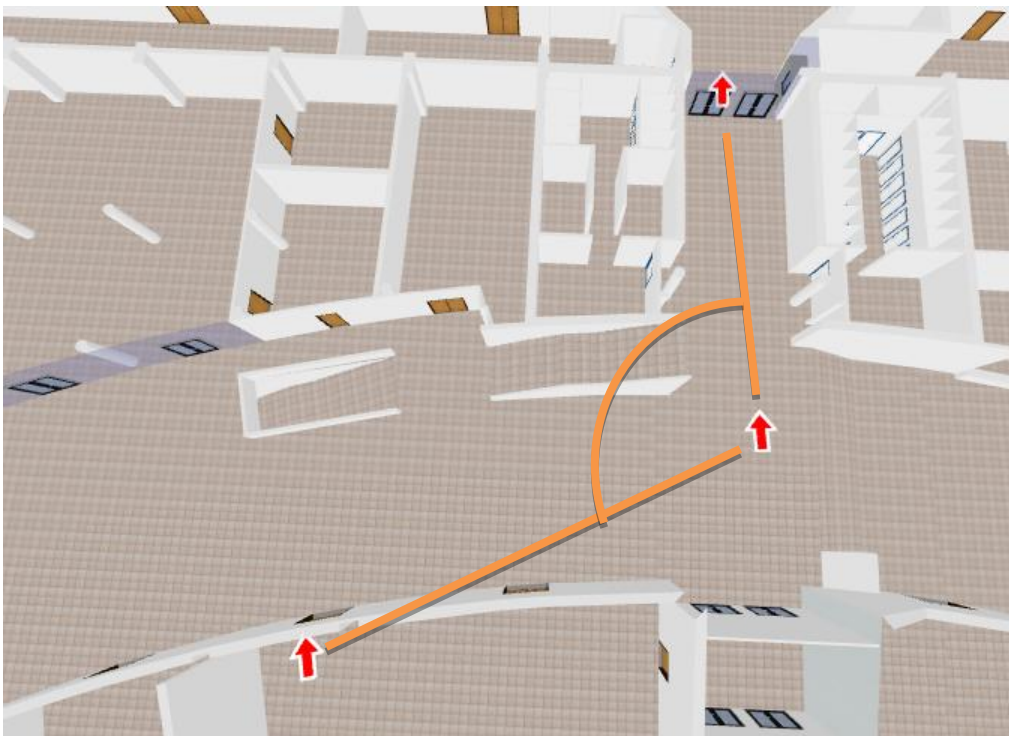
```
Navigation : {  
  stairs : [  
    {from : { floor : '2PP', x : -42.7, z : -17.1 },  
      to : { floor : '1PP', x : -42.7, z : -12.0 }},  
    {from : { floor : '2PP', x : -22.5, z : -15.0 },  
      to : { floor : '1PP', x : -15.0, z : -13.4 }}  ]  
}
```

V uvedenom príklade vidieť definíciu 2 schodísk vedúcich z -2.poschodia na -1. poschodie . Uvedené pozície určujú schodiskové miestnosti (z výškovej mapy sa zistí, aká miestnosť sa na danom mieste nachádza) a tiež sú dôležité pre umiestňovanie navigačnej šípky.

Navigačné šípky sa umiestňujú:

- Pred každé dvere, cez ktoré musí používateľ prejsť
- Na začiatok schodov, cez ktoré musí používateľ prejsť. Šípka v tomto prípade buď smeruje hore, alebo dole.

Umiestňovanie len na uvedené miesta však nestačí , pretože používateľ nemusí vidieť nasledujúcu šípku, ak sa nachádza za rohom. Preto sme dodali možnosť zdefinovať doplnkové pozície v miestnostiach, na ktoré sa môže pridať ďalšia navigačná šípka. Podmienkou zobrazenia šípky v takejto doplnkovej pozícii je, že uhol definovaný predchádzajúcou šípkou, doplnkovou pozíciou a nasledujúcou šípkou musí byť väčší ako stanovená hodnota ( $60^\circ$ ). Takáto situácia je znázornená na Obr. 10.2. Predpokladáme, že ak je daný uhol menší, nasledujúcu šípku používateľ priamo vidí a ďalšia šípka na doplnkovej pozícii by ho iba pomýlila.



Obr. 10.2 Doplnková navigačná šípka

Doplnkové pozície sa nastavujú tiež v súbore vfiit.conf.js:

```
midpoints : {
    '1PP' : [{x: 0, z: -16}]
}
```

Doplnková pozícia je určená názvom poschodia a pozíciou, kde má byť šípka umiestnená. V jednej miestnosti sa môže nachádzať aj viac takýchto pozícií a pre každú z nich sa určí, či má byť zobrazená, alebo nie.



Pri spustení navigácie v režime prechádzania môže nastať situácia, že sa používateľ pozerá iným smerom, ako sa nachádza prvá šípka. Preto sa kamera v takomto prípade automaticky natočí tak, aby pohľad smeroval na prvú šípku.

V režime prechádzania sú šípky označujúce prechod cez dvere umiestňované na podlahu. V režime prehliadania sa zdvihnú na úroveň stropu, kvôli lepšej viditeľnosti.

Navigáciu v režime prechádzania automaticky prepočítame, ak používateľ vstúpi do miestnosti, ktorá je mimo nájdenú trasu do cieľovej miestnosti.

### 10.1.1 Navigačný algoritmus

Základom navigácie je Dijkstrov algoritmus, ktorého implementácia sa nachádza v súbore `vfiit.navigation.js`. Jedná sa o známy algoritmus na hľadanie najkratšej cesty v grafe. Graf pozostáva z množiny vrcholov a hrán. V našom prípade sú vrcholmi miestnosti a dvere, hranami rozumieme prepojenia medzi miestnosťou a dverami, ktoré k nej prislúchajú, respektíve prepojenia medzi dverami a miestnosťami, ktoré prepájajú. Pôvodne sme plánovali použiť rovnakú metódu, ako minuloročný tím, ktorý generoval graf do xml súboru a v ňom následne vykonával manuálne úpravy (nastavovanie váh a pod.). Nakoniec sme sa však rozhodli pre efektívnejšie riešenie, ktorým bolo vygenerovanie grafu pomocou skriptu z výškovej mapy priamo do štruktúry v jazyku JavaScript, použitého aj na implementáciu navigácie. Tu už nebolo potrebné vykonávať žiadne manuálne úpravy. Prehľadávanie grafu s takouto štruktúrou bolo taktiež veľmi jednoduché.

V štandardnej implementácii algoritmu sú všetky hrany ohodnotené váhami, ktoré reprezentujú ich dĺžku, potrebnú pri hľadaní najkratšej cesty. V našom grafe je parameter obsahujúci váhu, udaný pri uzloch. K niektorým uzlom (v tomto prípade ide o niektoré dvere) je pridaná vysoká váha, pri ostatných uzloch parameter váhy nie je definovaný. Pri výpočte najkratšej cesty je teda možné prechádzať cez uzly, ktoré nemajú priradenú žiadnu váhu a cez uzly, ktoré majú priradenú vysokú váhu. Pri výpočte a porovnávaní dĺžky cesty je pre hranu, ktorá vchádza do vrcholu s parametrom váhy, pripočítaná pre danú hranu táto váha. Pri hranách, ktoré vchádzajú do vrcholov bez parametru váhy sa uvažuje pre danú cestu veľkosť váhy rovná jednej.

Váhy sú určené len pre uzly, ktoré boli farebne vyznačené vo výškovej mape. Označenie vrcholu znamená, že by sa cez daný uzol (dvere) pri bežnej ceste po budove nemalo prechádzať. Jedná sa o prípady, keď sa cesta nemala skracovať cez priechodnú kanceláriu alebo prednáškovú miestnosť s dvoma vchodmi. Taktiež malo byť uprednostňované hlavné schodisko pred bočnými.

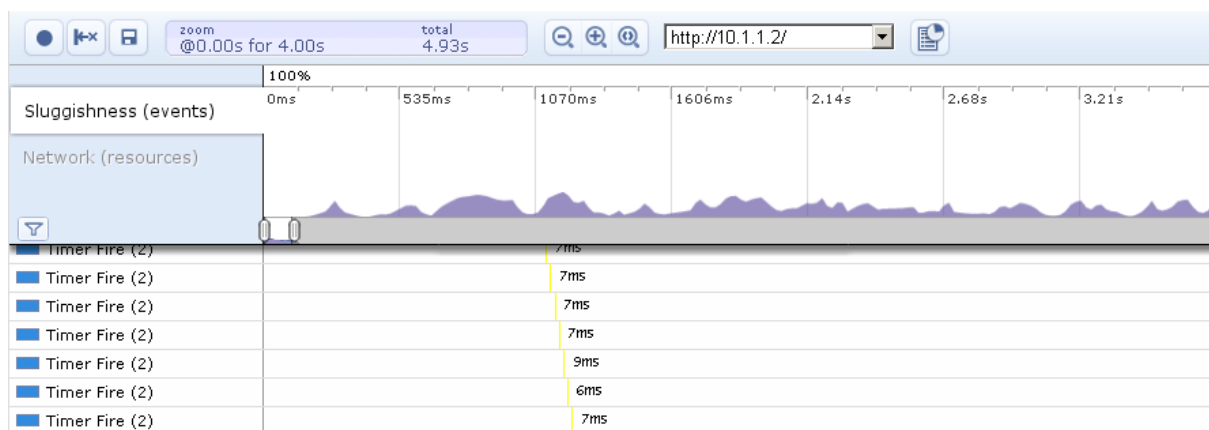
Mohla však nastať aj situácia, keď by bolo zadané vyhľadávanie cesty medzi susednými miestnosťami a bolo potrebné použiť dvere medzi miestnosťami, ktoré sa pri dlhšej ceste nepoužívali. Teda aby algoritmus napríklad nenašiel ako najkratšiu cestu medzi susediacimi prepojenými kancelármi cestu, ktorá sa ich priamemu priechodu vyhýba a obchádza ho napríklad cez chodbu ale aby bol použitý daný priechod medzi miestnosťami. Preto je ešte pred spustením prehľadávania pomocou Dijkstrovho algoritmu vykonaná kontrola, či sa nejedná o susedné miestnosti. Ak áno, tak váhy neboli brané do úvahy a bola použitá nájdená priama najkratšia cesta. Ak nie, tak sa pokračuje ďalej vykonaním jednotlivých krokov navigačného algoritmu.

## 10.2 Optimalizácia

V priebehu implementácie sme riešili problémy týkajúce sa rýchlosti vykresľovania 3D scény, ktorá bola spočiatku nedostatočná. V tejto kapitole sú preto uvedené rôzne implementované spôsoby optimalizácie.

Samotné vykreslenie polygónov na obrazovku (volaním WebGL metódy) je záležitosť grafickej karty, čo je, vzhľadom na to, že náš model nemá veľmi veľa polygónov (cca 50 tisíc), rýchle. Vykresľovanie však spomaľujú operácie v JavaScripte, ktoré sa vykonávajú medzi volaniami WebGL. Ide napríklad o výpočet rôznych matic potrebných pre vykreslenie (násobenie matic, či výpočet inverznej matice, sú výpočtovo pomerne náročné úlohy). Namiesto vykresľovania 10 objektov po 1000 polygónoch je teda výhodnejšie vykresliť 1 objekt s 10000 polygónmi, čím sa zmenší počet náročných operácií vykonávaných v JavaScripte.<sup>39</sup>

Na sledovanie rýchlosti vykresľovania sme používali rozšírenie Google Chrome – Speed Tracer<sup>40</sup>. Na Obr. 10.3 vidieť výstup, ktorý nám tento nástroj poskytuje. „Timer Fire“ je udalosť, ktorá je vyvolaná vždy, keď je potrebné vykresliť scénu. Vidieť, že vykreslenie trvá okolo 6 až 9 milisekúnd. V hornom grafe vidieť priebeh dĺžky vykreslenia v závislosti od času. Uvedený obrázok pochádza z optimalizovanej verzie aplikácie.



Obr. 10.3 Google Chrome - Speed Tracer

Hlavným cieľom teda bolo zmenšenie počtu vykresľovaných objektov. V CopperLicht-e je objekt, ktorý sa skladá z polygónov, reprezentovaný triedou CL3D.MeshSceneNode. Pre každý takýto objekt sa pred vykreslením musia vypočítať príslušné transformačné matice. Každý takýto objekt sa však skladá z viacerých objektov triedy CL3D.MeshBuffer, ktoré obsahujú polygóny s rovnakým materiálom. Pred vykreslením každého objektu MeshBuffer sa musí nastaviť vo WebGL textúra, ktorá bude aplikovaná na polygóny. Potom sa už môžu jedným volaním vykresliť všetky polygóny, ktoré MeshBuffer obsahuje.

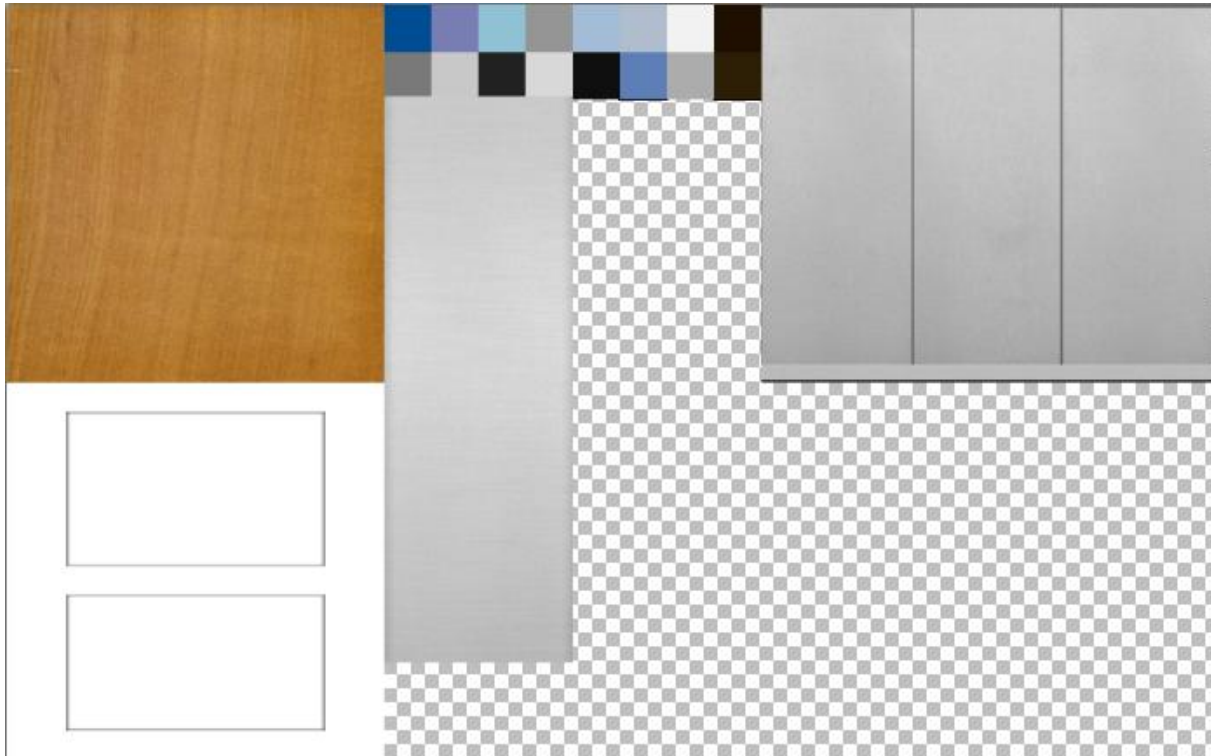
Chceme teda aby sa celá scéna skladala z čo najmenej objektov MeshSceneNode (aby sa zmenšil počet maticových výpočtov) a zároveň to, aby sa tieto objekty skladali z čo najmenej objektov MeshBuffer (aby sa zmenšil počet zmien textúr).

<sup>39</sup> Skúsenosti s vykresľovaním Quake3 máp vo WebGL <http://blog.tojicode.com/2010/08/rendering-quake-3-maps-with-webgl-tech.html>

<sup>40</sup> <http://code.google.com/webtoolkit/speedtracer/>

### 10.2.1 Textúrovací atlas

Objekty, ktoré používajú rôzne textúry sa za istých podmienok môžu zlúčiť do jedného objektu, ktorý bude mať priradenú špeciálnu textúru, tzv. textúrovací atlas<sup>41</sup>. Mierne upravený textúrovací atlas, ktorý používame v aplikácii sa nachádza na Obr. 10.4. Vidieť v ňom textúry drevených dverí, dverí na záchody, textúry použité vo výťahu a nakoniec všetky farebné textúry (len čistá farba, žiaden vzor).



Obr. 10.4 Textúrovací atlas.

V textúrovacom atlase sa nemôžu z technických dôvodov nachádzať textúry, ktoré majú nejaký vzor a sú aplikované na väčšie plochy, na ktorých sa daný vzor opakuje (platí to napr. na textúru podlahy, alebo stropu). Textúry, ktoré sa skladajú iba z jednej farby, nemajú žiaden vzor, takže sa môžu umiestniť do textúrovacieho atlasu a zaberajú v ňom málo miesta, teda z tohto hľadiska sú dosť efektívne.

Použitím textúrovacieho atlasu tak môžeme zlúčiť všetky MeshBuffer objekty s textúrami, ktoré sa dajú umiestniť do textúrovacieho atlasu, do jedného spoločného MeshBuffer objektu. Zrýchli sa tak vykresľovanie.

### 10.2.2 Dvere

Keď boli dvere samostatné objekty (cca. 50 dverí na poschodie), vykresľovanie bolo dosť pomalé. Jednotlivé dvere však potrebujeme samostatne ovládať (otvárať/zatvárať), preto ich nie je možné jednoducho zlúčiť dokopy. Pritom väčšina dverí je stále zatvorená a nemuseli by existovať ako samostatné objekty.

Naše riešenie umožňuje zlúčiť všetky dvere do jedného objektu, zároveň však umožňuje otvárať a zatvárať jednotlivé dvere. Dvere sa načítajú z modelov poschodí ako samostatné objekty (objekty

<sup>41</sup> [http://developer.download.nvidia.com/assets/tools/files/Texture\\_Atlas\\_Whitepaper.pdf](http://developer.download.nvidia.com/assets/tools/files/Texture_Atlas_Whitepaper.pdf)

začínajúce reťazcom „door\_“). Pri načítavaní stránky sa všetky tieto dvere zlúčia do jedného statického objektu (objekt MergedDoors). Samostatné dvere sa skryjú, zobrazuje sa iba objekt MergedDoors. Keď používateľ ide otvoriť nejaké dvere, časť objektu MergedDoors, ktorá predstavuje dané dvere sa skryje a na jej mieste sa zobrazia pôvodné samostatné dvere, ktoré je potom možné animovať, teda otvoriť, alebo zatvoriť. Spomínané objekty vidieť v štruktúre scény pre najvyššie poschodie (takýto výpis je možné získať zadaním príkazu

```
vfiit.dumpNode(engine.scene.getRootSceneNode())
```

do vývojárskej konzoly v prehliadači:

- FloorNode (none)
  - Floor (mesh) m:4 p:0 b:4
  - doors (none)
    - door\_l001 (mesh) m:2 p:24 b:2 invisible
    - door\_r001 (mesh) m:2 p:24 b:2 invisible
    - door\_ld001 (mesh) m:2 p:24 b:2 invisible
    - door\_rd001 (mesh) m:2 p:24 b:2 invisible
    - door\_rd002 (mesh) m:2 p:24 b:2 invisible
    - door\_ld002 (mesh) m:2 p:24 b:2 invisible
  - Ceiling (mesh) m:2 p:0 b:2
  - Model (mesh) m:7 p:0 b:7
  - MergedDoors (mesh) m:2 p:0 b:2

Na vysvetlenie spôsobu skrývania časti objektu MergeDoors je nutné vysvetliť textúrovacie súradnice. Každý obrázok, ktorý slúži ako textúra, sa nanáša na povrch objektov pomocou dvojice textúrovacích súradníc, ktoré má priradený každý vrchol v objekte. Celému obrázku zodpovedajú textúrovacie súradnice od [0,0] po [1,1]. Textúrovacie súradnice sú vždy mapované do rozsahu 0 až 1 tak, že obrázok sa „opakuje“. Vidieť to na Obr. 10.5, kde je zobrazené, ako vyzerá štvorec, ktorý má priradené textúrovacie súradnice v rozsahu [0,0] až [3,3].



Obr. 10.5 Príklad opakovania textúry.<sup>42</sup>

Základná myšlienka skrývania časti objektu MergeDoors je tá, že každým dverám sa textúrovacie súradnice posunú o celé číslo, ktoré zodpovedá číslu daných dverí (každé dvere majú priradené unikátne číslo). Keďže textúra sa opakuje, vykreslené budú rovnako, používateľ nevidí zmenu. Vertex shader (kód pre grafickú kartu, ktorý určuje ako sa vykresľujú objekty) potom na základe textúrovacej

<sup>42</sup> <http://msdn.microsoft.com/en-us/library/aa916107.aspx>

súradnice dokáže skryť polygóny, ktoré prislúchajú konkrétnym dverám (uvedený kód je z client\_index.php):

```
float delta = 0.0;

if (doorId == int(vTexCoord1.s) || doorId2 == int(vTexCoord1.s)) {
    delta = 999999.0;
}

gl_Position = worldviewproj * vec4( vPosition.x,
                                   vPosition.y + delta,
                                   vPosition.z,
                                   1.0);
```

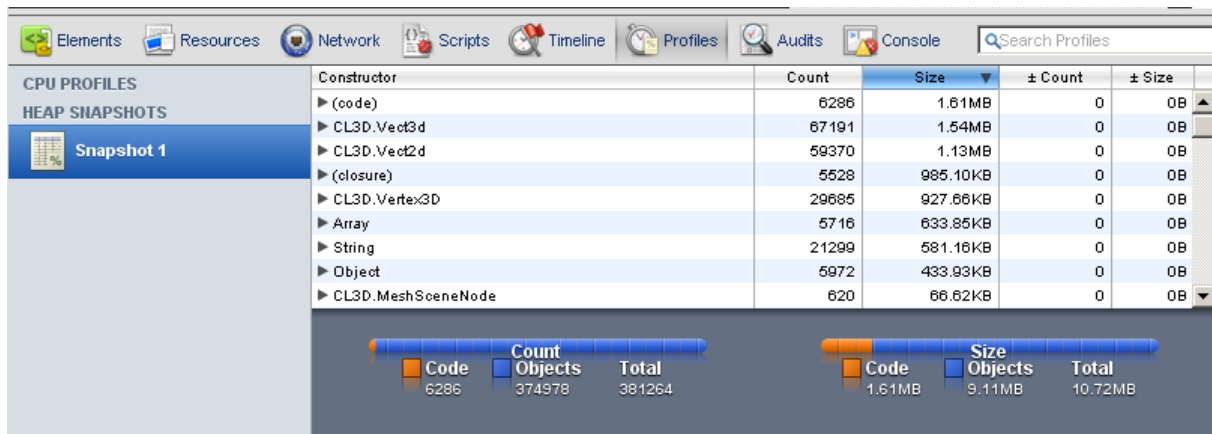
Uvedený kód sa spúšťa na grafickej karte pre každý jeden vykresľovaný vrchol. V premenných doorId a doorId2 sú nastavené čísla dverí, ktoré sú práve aktivované (sú to tie dvere, ktoré sú zobrazené ako samostatný objekt, aby ich bolo možné otvoriť, resp. zatvoriť). Vertex shader pri vykresľovaní objektu MergeDoors z textúrovacej súradnice určí, ktorým dverám daný vrchol prináleží. Ak prináleží aktivovaným dverám, daný vrchol sa zdvihne do výšky (vPosition.y + delta). Tým že zdvihnem všetky vrcholy pre dané dvere do veľkej výšky, zmizne daná časť objektu MergeDoors. Namiesto zmiznutej časti sa zobrazia samostatné dvere, ktoré vyzerajú rovnako. Používateľ tak nevidí žiadnu zmenu.

Výhodou je zásadne zmenšenie počtu vykresľovaných objektov, namiesto približne 50 samostatných dverí na poschodie sa vykreslí len jeden objekt MergedDoors a maximálne 2 objekty samostatných dverí. Ako vidieť v kóde, nastavujú sa len dve čísla aktívnych dverí, preto môžu byť naraz v modeli otvorené len jedny dvere (ak sú niekde otvorené druhé dvere, tak tie sa automaticky zatvoria).

### 10.2.3 Garbage Collector

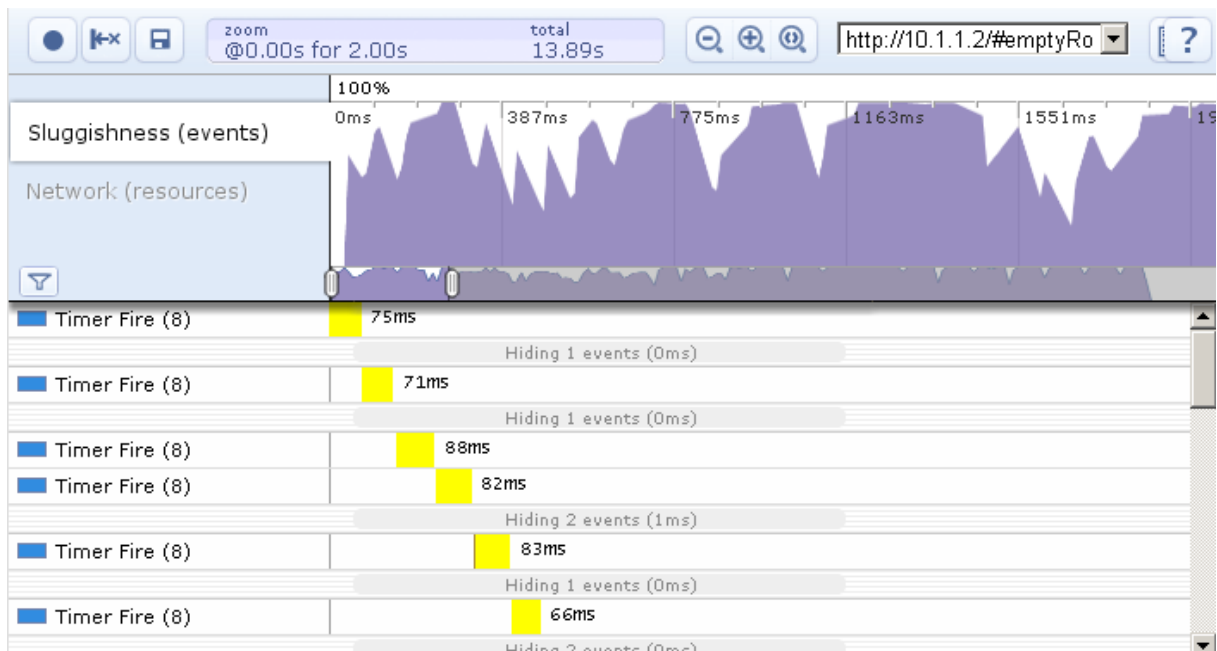
Po uvedenej optimalizácii fungovala aplikácia rýchlejšie, ale stále nie dostatočne plynulo. Vo vývojovej verzii Firefoxu 4 sa stávalo, že vždy po určitom čase (okolo 10 až 20 sekúnd) sa aplikácia na chvíľu zasekla. Uvažovali sme preto, či to nemôže spôsobiť Garbage Collector, ktorý uvoľňuje z pamäte nepoužívané objekty.

Pomocou nástroja Heap Profiler (Obr. 10.6) v Google Chrome sme zistili, že CopperLicht zbytočne vytvára veľké množstvo objektov, pri každom vykreslení snímku. Garbage Collector teda musel bežať často a dlho aby vytvorené objekty uvoľnil z pamäte. Spomaľoval tým vykresľovanie a spôsoboval aj pravidelné zasekávania sa aplikácie. Kód CopperLichtu sme predefinovali tak, aby sa nové objekty nevytvárali a radšej sa využili už raz vytvorené inštancie. Pred úpravami sa napr. vytváralo okolo 1000 objektov matíc, pri každom vykreslení snímku. Po úprave to bolo okolo 10. Rôzne takéto optimalizácie sa aktivujú vo funkcii \_applyOptimization vo vfiit.js.



Obr. 10.6 Heap Profiler v Google Chrome.

Rýchlosť vykresľovania a plynulosť viditeľne vzrástla. Pre porovnanie s Obr. 10.3 je vidieť na Obr. 10.7 výsledok behu nástroja Speed Tracer pri vypnutých opísaných optimalizáciách. Vidieť že vykreslenie jedného snímku trvá rádovo dlhšie.



Obr. 10.7 Speed Tracer pre nezoptimalizovanú aplikáciu.

#### 10.2.4 Kompresia modelov

Modely poschodí vo formáte JSON zaberajú okolo 10MB, čo je pomerne dosť na prenášanie pri načítaní stránky. Ide pritom o textový formát, ktorý sa dá dobre skomprimovať. V súbore `server/resources/client/assets/.htaccess` sme preto nastavili, nech JSON súbory modelov posielajú Apache web server skomprimované:

```
AddOutputFilterByType DEFLATE application/json
```

Množstvo prenášaných údajov pre modely sa tak zmenšilo približne na desatinu, čím sa zrýchlilo celé načítavanie aplikácie.

### 10.3 Info server

Zdrojový kód Info servera (predtým uvádzaný ako KR server) je napísaný v PHP frameworku CodeIgniter, ktorého úlohou je sprístupniť informácie ohľadom používateľov, miestností, predmetov a rozvrhu klientskej strane. Sprístupňujú sa informácie, ktoré boli importované z AIS pomocou importéra. Info server pristupuje k dátam v databáze pomocou readonly prístupu, teda aplikácia nemení žiadne dáta v databáze. Aplikácia odpovedá na http dopyty na konkrétnej URL adrese. Klientom teda môže byť akákoľvek aplikácia, ktorá je schopná spracovávať http protokol.

Aplikácia na http dopyty odpovedá http odpoveďami, pričom v tele http odpovede sa nachádzajú dáta zakódované vo formáte json. Formát json je predvoleným formátom návratových informácií. Je možné vrátiť informácie aj vo formáte html kódu, čo je výhodné hlavne pre vývojára, ktorý potrebuje vidieť dáta v prehľadnom formáte.

Súčasťou info servera sú aj testovacie procedúry, kedy je nutné do databázy nahráť testovaciu množinu údajov a zavolať príslušnú URL adresu, následkom čoho bude zoznam testovacích procedúr spolu s ich výsledkom.

V súčasnej verzii poskytuje info server nasledujúce informácie:

- Základné informácie o miestnostiach
- Lokalizáciu osoby
- Základné informácie o konkrétnej osobe
- Výpis predmetov, ktoré daná osoba učí
- Rozvrh danej osoby
- Vyhľadávanie osôb, miestností, predmetov alebo všetkého
- Vypísanie rozvrhu pre daný predmet
- Testovanie funkčnosti

Info server môžeme tiež považovať za časť kompletnej serverovej aplikácie pre virtuálnu FIIT. Jeho triedy sa nachádzajú v štandardných adresároch frameworku CodeIgniter (*models, views, controllers*), pričom samotné triedy controller sa nachádzajú v samostatnom priečinku *info*. Súčasťou celej serverovej aplikácie je aj mobilný klient, ktorý pre svoju funkcionality využíva triedy modelov pôvodne napísane pre info server.

### 10.4 GUI a jeho interaktívne prvky

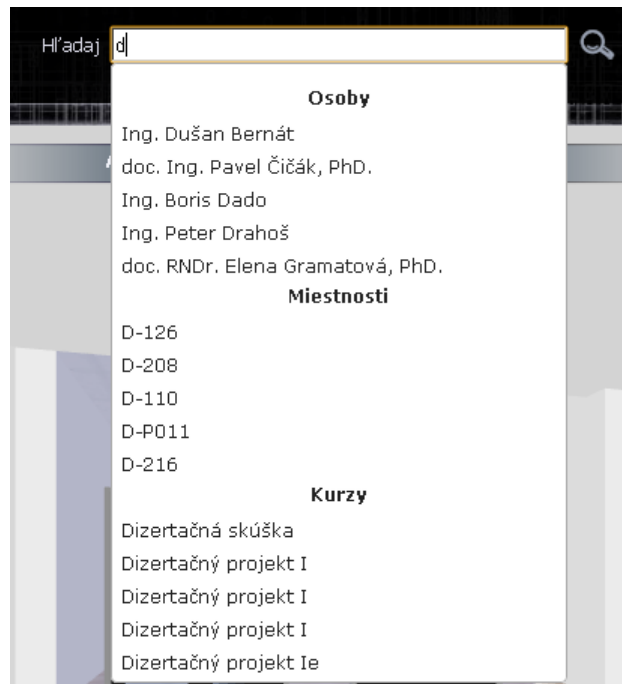
V tejto kapitole sú opísané záležitosti týkajúce sa funkcionality používateľského rozhrania a interaktivity v 3D scéne.

#### Vyhľadávanie a zobrazovanie informácií

Na implementáciu funkcionality vyhľadávania sme použili knižnicu JQuery UI<sup>43</sup> – konkrétne zásuvné moduly Autocomplete a Accordion. Autocomplete poskytuje napovedanie pri vyhľadávaní, zobrazuje miestnosti, osoby a predmety, ktoré vyhovujú zadanému textu (Obr. 10.8) . Používateľ tak nemusí napísať celý reťazec, ale často si už po niekoľkých znakoch môže zvoliť zo zoznamu správnu možnosť. Pri napovedaní sa zobrazujú iba tie výsledky, v ktorých sa zadaný reťazec nachádza na začiatku slova; vyhľadávať je tiež možné aj bez diakritiky.

---

<sup>43</sup> <http://jqueryui.com/>



Obr. 10.8 Napovedanie pri vyhľadávaní.

Na formátovanie informácií v ľavom paneli sme použili knižnicu pre prácu so šablónami JQuery templates<sup>44</sup>. Zo vstupného JSON objektu, ktorý sme získali zo serveru, tak s pomocou šablóny vznikol HTML s naformátovanými údajmi. Tento nástroj poskytoval všetko, čo sme potrebovali (odkazy na iné šablóny, podmienky, ...). Príklad šablóny, kde vidieť podmienky, cyklus a odkazy na inú šablónu vidieť na Obr. 10.9 (uvedený kód pochádza zo súboru client\_index.php).

```

Kancelária:
{{if room}}
  <a class="room" href="#">${room}</a>
  <a class="showInModel" href="#" data-room="${room}">${email}</a><br>
Telefón: ${phone}<br><br>
<b>Predmety:</b><br>
{{each courses}}<a href="#" class="course">${$value.course}</a><br>{{/each}}
<br>
<b>Rozvrh:</b>
{{tmpl "#timetableTemplate"}}

```

Obr. 10.9 Príklad šablóny na vypísanie informácií o osobe.

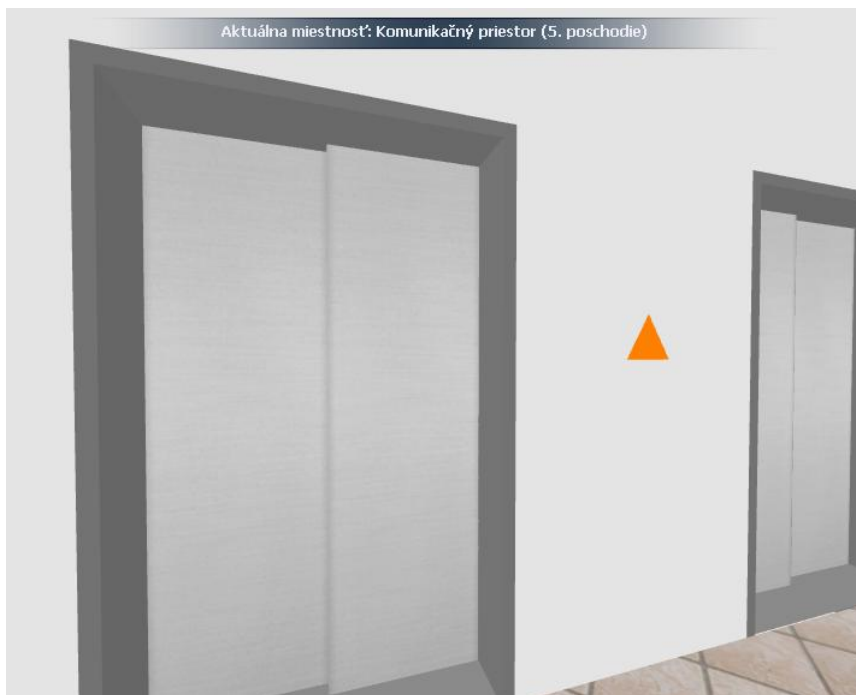
## Výťahy

V modeli sú plne funkčné všetky výťahy. Výťahové dvere sa otvárajú rovnako ako iné dvere, ak sa však výťahová kabína nachádza na inom poschodí, je nutné počkať, kým príde na aktuálne poschodie.

<sup>44</sup> <http://api.jquery.com/jquery.template/>



Výťahy sa teda v tomto ohľade správajú ako v reálnom prostredí. Čakanie na výťahovú kabínu je znázornené šípkou medzi výťahmi (Obr. 10.10). Pohyb výťahov, otváranie dverí a príchod na cieľové poschodie sú sprevádzané zvukovými efektmi.



**Obr. 10.10 Šípka ukazujúca prichádzajúci výťah.**

Do výťahu sme nedorobili vnútorné dvere preto, aby bolo viditeľné, že výťah sa naozaj hýbe. Ak by sme pridali aj vnútorné výťahové dvere, výťah by sa vôbec nemusel plynulo presunúť z jedného poschodia na druhé, ale stačilo by len posunúť kabínu na správne poschodie a chvíľu počkať.

### **Prechody kamier**

Pri prepínaní medzi režimami prechádzania, lietania a prezerania sa pohľad neprepne skokom, ale plynulou animáciou. Používateľovi sa teda nestane, že sa mu prepne pohľad a zrazu nevie kde sa nachádza.

Z používateľských testov vyplynulo, že keď sa po načítaní aplikácie používateľ ocitol pred dverami v strede budovy, tak nevedel, kde vlastne je. Preto sme ako začiatkový pohľad zvolili pohľad na celú budovu, ktorý sa plynulou animáciou zmení na pôvodný pohľad v strede budovy. Používateľ tak získa prehľad o tom, kde sa približne nachádza.

### **Ovládanie pohľadu**

Na začiatku sme mali ovládanie v režime prechádzania riešené pomocou kláves W, S, A, D a šípok, pričom bočné klávesy (A, D, šípka vľavo, šípka vpravo) boli úkrok, nie otáčanie. Otáčanie pohľadu prebiehalo tak, že bolo treba stlačiť ľavé tlačidlo a ťahať myšou. Takýto spôsob ovládania už bol implementovaný v CopperLichte.

Spôsob otáčania bez potreby stlačenia tlačidla myši, ako ho poznáme z počítačových hier, nie je možné implementovať, pretože WebGL, resp. internetový prehliadač všeobecne (ak to nie je riešené zásuvným modulom) nepodporuje možnosť „zachytiť“ kurzor myši. Kurzor tak vždy dôjde na okraj

obrazovky a ďalej sa už otáčať nepôjde. Počítačové hry kurzor „zachytia“, a teda kurzor sa reálne nehýbe, ale aplikácia len prijíma informácie o pohybe myši.

Skúšali sme ešte inú možnosť a to otáčanie bez držania tlačidla. Bol to však neštandardný spôsob v tom, že rýchlosť otáčania bola určená vzdialenosťou kurzoru od stredu 3D scény. Keď bol teda kurzor naľavo od stredu scény, pohľad sa automaticky otáčal vľavo. Na zastavenie otáčania pohľadu bolo potrebné trafiť myšou presne do stredu obrazu. Ďalší problém nastal, keď používateľ chcel ísť niečo zadať do vyhľadávacieho poľa. Kurzorom začal prechádzať cez scénu a pohľad sa tak začal otáčať, aj keď to vlastne používateľ nechcel. Tento spôsob ovládania sme preto po používateľských testoch zavrhlí. Opisovaný spôsob ovládania možno vyskúšať na stránke CopperLichtu<sup>45</sup>.

Používateľské testy tiež odhalili, že veľa ľudí intuitívne používa šípky a očakáva, že šípky vľavo a vpravo spôsobia otáčanie a nie úkrok. Upravili sme teda šípky na takéto ovládanie a pridali sme klávesové ovládanie zdvíhania a znižovania pohľadu, aby bolo možné ovládať pohľad kompletne iba klávesmi. Ďalej klávesu E na ovládanie dverí sme doplnili aj klávesom Enter, čo tiež vyplynulo z používateľských testov.

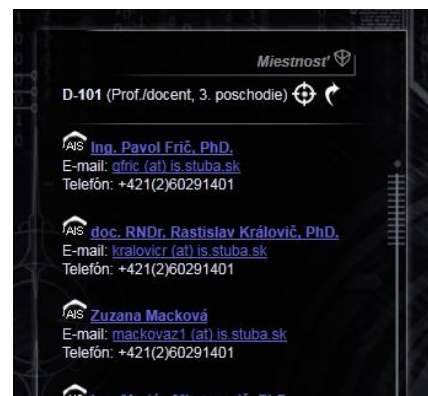
Skončili sme teda s dvoma rôznymi možnosťami ovládania (pre režim prechádzania, resp. lietania):

- Základne ovládanie pomocou klávesnice
  - šípky vľavo a vpravo otáčajú pohľad, kláves Enter otvára dvere, pohľad sa dá zdvíhať cez klávesy PageUp a PageDown
- Ovládanie pre hráčov so skúsenosťami s počítačovými hrami
  - klávesy W, S, A, D na pohyb, pričom A a D sú úkrok, kláves E otvára dvere, stlačením tlačidla myši a ťahaním sa otáča pohľad

## 10.5 Štruktúra používateľského rozhrania

Od navrhovaného GUI sme upustili z dôvodu malého priestoru, ktorým disponoval panel na webovej stránke. Pri jeho veľkosti 800x600 pixlov by navrhované prvky GUI zobrazené nad samotným panelom zaberali príliš veľa miesta. S väčším rozmerom sa spočiatku nepočítalo, keďže už pri tomto rozmere mala aplikácia miestami veľmi nízke FPS (počet snímok za sekundu). Preto sme navrhli stránku nasledovne, do troch panelov:

- informačný panel (Obr. 10.11)
  - umiestnený naľavo od WebGL panelu
  - v tomto paneli sa prezentujú všetky relevantné informácie z ostatných informačných systémov (informácie o osobách, predmetoch, miestnostiach a rozvrhoch)



Obr. 10.11 Informačný panel

<sup>45</sup> [http://www.ambiera.at/copperlicht/demos/demo\\_quakelevel\\_external.html](http://www.ambiera.at/copperlicht/demos/demo_quakelevel_external.html)

- panel nástrojov (Obr. 10.12)
  - tento panel umiestnený nad WebGL panelom obsahuje nasledovné ovládacie prvky:
    - voľba režimu zobrazenia (režim náhľadu, chôdze, voľného letu)
    - vyhľadávacie pole
    - tlačidlá režimu zobrazenia na celú obrazovku, ovládanie zvuku a tlačidlo pomocníka



Obr. 10.12 Panel nástrojov

- WebGL panel
  - tento panel o veľkosti 800x600 pixlov obsahuje 3D interaktívny model budovy
  - okrem 3D scény, obsahuje tento panel aj niekoľko plávajúcich prvkov:
    - popis označenej alebo momentálne navštívene miestnosti
    - šípka s popisom miestnosti (zobrazí sa len pri kliknutí na miestnosť v režime náhľadu)
    - zmena poschodia (zobrazí sa pri vstupe do výťahu a v režime náhľadu)

Počas vývoja prešla naša aplikácia optimalizáciou, čo nám umožnilo spustiť aj režim zobrazenia WebGL panelu na celú plochu prehliadača. Kôli nedostatku času sme však nestihli do tohto režimu implementovať všetky vyššie spomínané panely.

## 10.6 Mobilný klient

V súčasnom stave mobilný klient poskytuje informácie o fakulte, ktoré sú verejne dostupné na jej stránke (informácie o študijnom oddelení, telefónne čísla, informácie o možnostiach štúdia), umožňuje vyhľadávanie osôb a miestností a umožňuje zobrazenie plánu budovy v dvoch režimoch.

Po vyhľadaní osoby je možné zobraziť si jej dostupné informácie (telefónne číslo, email, číslo miestnosti) a v prípade, že osoba aj učí nejaké predmety, zobrazí sa aj jej rozvrh. Zo zobrazenia profilu v mobilnom klientovi je možné sa prekliknúť na profil v AIS-e.

Po vyhľadaní miestnosti je možné si zobraziť profily osôb, ktoré majú v miestnosti kanceláriu. Zobrazenie rozvrhu miestnosti v súčasnosti nie je implementované.

Plán budovy je možné zobraziť v dvoch režimoch:

- V režime rozkladania: plán budovy je generovaný podľa šírky zobraziteľnej plochy.
- V plnom režime: plán budovy je zobrazený celý a prehliadanie je prenechané prehliadaču zariadenia.

Po kliknutí na plán server dokáže identifikovať miestnosť, na ktorú používateľ klikol (je jedno v ktorom režime) a zobrazí informácie o miestnosti.

Možnosti rozšírenia:

- vymeniť plány,
- implementovať lokalizáciu miestností,

- implementovať zobrazenie rozvrhu pre miestnosť
- vyhľadávanie predmetov

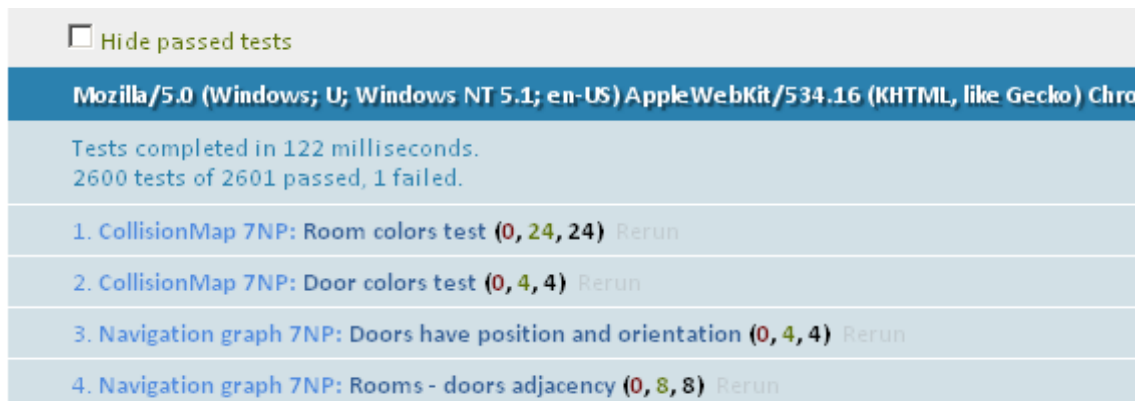
## 11 Overenie funkcionality aplikácie

### 11.1 Testovanie 3D klienta

Pre 3D klienta sme použili knižnicu QUnit<sup>46</sup> na automatizované testovanie výškových máp (časť z testov je ukázaná na Obr. 11.1). Testovalo sa to, či miestnosti majú priradené správnu farbu, či sa na pozícií stredu miestnosti nachádza farba danej miestnosti, správne očíslovanie dverí a podobne. Takéto testovanie veľmi pomohlo pri odhaľovaní chýb v modeli, napríklad:

- nesúvislé steny (pri vyfarbovaní miestností vo výškovej mape sa potom miestnosti prefarbovali)
- dvere v navigačnom grafe, ktoré nemali priradenú pozíciu (nedala sa potom umiestniť navigačná šípka)

Po každom exporte sa tak dalo skontrolovať, či je všetko v poriadku. Tieto testy sú prístupné pripojením nasledujúcej cesty k adrese hlavnej stránky: `/index.php/tests/collision_maps`.



Obr. 11.1 Nástroj QUnit použitý pre testovanie výškových máp.

### 11.2 Testovanie info servera

V rámci testovaní sme sa rozhodli testovať aj Info server. Vychádzali sme z odporúčaní, ktoré je možné nájsť na domovskej stránke CodeIgniteru pre použitie Unit testov. Php framework CodeIgniter poskytuje vlastnú funkcionality pre písanie Unit testov v tzv. Unit Testing Class knižnici.

Samotné testovanie spočíva v dvoch krokoch a to v definovaní očakávaných výstupov a v porovnaní očakávaných výstupov s reálnymi výstupmi. Samotný test sa spustí metódou:

- `$this->unit->run(test, expected result, 'test name', 'notes');`
  - test sú testovacie dáta (v našom prípade výstup z metódy)
  - expected results sú dáta, ktoré očakávame
  - test name je názov testu (v našom prípade názov testu je testovacia metóda)
  - notes sú poznámky ku testu

Po spustení testovacích prípadov je možné vygenerovať výstup testovania v HTML formáte pomocou metódy:

<sup>46</sup> <http://docs.jquery.com/Qunit>

- `echo $this->unit->report();`

Vzhľadom na to, že pomocou tejto funkcionality sa dajú testovať výstupy metód, rozhodli sme sa testovať metódy v triedach modelu. Pred samotným testovaním je nutné nahráť do databázy modelové údaje, ktoré a nachádzajú v svn repozitári v súbore `/trunk/databaza/virtualfiit_test_data.sql`. Samotný test sa spustí zavolaním URL adresy:

- `http://<serverName>/<virtualfiitAppName>/test`

## 12 Záver

---

### WebGL

WebGL podporujú vydané prehliadače Chrome (aktuálne verzia 11) a Mozilla Firefox 4. Prehliadač Opera podporuje WebGL zatiaľ len v testovacích verziách<sup>47</sup>. Menšou nepríjemnosťou je, že pri niektorých kombináciách operačných systémov a grafických kariet je WebGL v uvedených prehliadačoch zablokované a je potrebné ho zapnúť v konfigurácii (Firefox), alebo parametrom pri spúšťaní (Chrome). Chrome má oproti Firefox-u výhodu v zapnutom vyhladzovaní (antialiasing), teda vykresľovaný obraz vyzerá lepšie.

Ako konkurencia pre WebGL by sa dalo považovať ohlásené Flash 3D API s názvom Molehill, ktoré pridá do Flashu možnosť využiť grafickú kartu pri vykresľovaní 3D obsahu<sup>48</sup>. Flash je asi najrozšírenejší plugin, ktorý majú používatelia nainštalovaný, takže by používateľ nemusel nič nové inštalovať.

Pre WebGL sa stále objavujú nové ukážky, hry, knižnice<sup>49</sup>, takže podľa nás má táto technológia budúcnosť. Jedinú prekážku pre rozšírenie WebGL vidíme v Internet Exploreri, keďže Microsoft zatiaľ neohlásil zámer implementovať WebGL.

### 12.1 Zhodnotenie

Počas dvoch semestrov tímovej práce a pravidelnej prípravy každého jednotlivca z tímu sa nám podarilo vytvoriť jedinečnú aplikáciu. Jej základným a najcennejším prínosom je šetrenie toho, čo sa nedá oceniť v peniazoch a ani predať, čo sa nedá vrátiť a ani stornovať. Daná aplikácia Vám umožní šetriť čas. Čas je v dnešnej dobe jedinečný element, ktorého šetrenie sa vzťahuje na každého jedného obyvateľa tejto planéty, bez ohľadu na národnosť, vieru, rasu či sociálne postavenia.

Ako je možné šetriť niečo tak cenné? V našej aplikácii nájdete všetky potrebné informácie do 5 sekúnd. Nepotrebuje žiadne zásuvne moduly. Nepotrebuje dokonca ani počítač. Celá aplikácia je tvorená s myšlienkou: „použi čo máš a my ti dáme najlepšie čo unesieš“.

Ako je to možné? Vytvorili sme aplikáciu spájajúcu informácie z informačného systému s navigáciou vo virtuálnej budove. Pri tvorbe sme mysleli na optimalizáciu prenesených dát a súčasne sme použili najmodernejšie technológie na trhu. Aplikácia dosahuje vynikajúce hodnotenia použiteľnosti v testovaní, ktoré prebieha online širokou verejnosťou. Jej atraktivnosť pripomína moderné hry, pričom aj na monitoroch s vysokým rozlíšením aplikácia nepôsobí ako internetová aplikácia. Zároveň funkcionality danej aplikácie môžete využiť na mobilnej verzii kompatibilnej so všetkými mobilnými telefónmi, ktoré podporujú prehliadanie webu (okrem už zastaraného WAP).

Systém spájajúci aktuálny čas, skutočný priestor a reálne osoby vo virtuálnej realite je neuveriteľne silná kombinácia pre kvalitnú a funkčne využiteľnú aplikáciu vysokej hodnoty. Projekt sme zamerali na budovu Fakulty informatiky a informačných technológií, ktorá je v súčasnosti vo výstavbe. Predpokladáme, že náš projekt svojou funkcionality a dizajnom upúta vedenie fakulty natoľko, že budú ochotní vytvoriť informačné kiosky pre danú aplikáciu.

---

<sup>47</sup> <http://labs.opera.com/news/2011/02/28/>

<sup>48</sup> <http://labs.adobe.com/technologies/flashplatformruntimes/incubator/features/molehill.html>

<sup>49</sup> <http://learningwebgl.com/blog/>

Zhrnutie faktov o finálnej aplikácii ju predurčuje na využitie v školách, nákupných strediskách, zdravotných zariadeniach, úradoch a každej zložitejšej stavbe, ktorá má záujem na zvyšovaní návštevnosti. Jej reálna hodnota je vysoká, nakoľko je veľmi jednoducho modifikovateľná a obsahuje precízne spracovaný importér z informačného systému, skripty pre rýchlejšie modelovanie na ceste z formátu dwg (AutoCAD) do formátu max (Autodesk 3ds Max), zaujímavé grafické prevedenie, jednoduchú obsluhu a skutočne rýchle a kvalitné vyhľadávanie.

Vízia do budúcnosti je postavená na reprezentácii prihlásených používateľov vo forme avatarov (beta verzia danej aplikácie je už na svete (Obr. 12.1)). Každá osoba prihlásená na stránke by mala možnosť predstavovať dynamický objekt v aplikácii. Aplikácia by sa doplnila o interaktívnu komunikáciu a špeciálnu formu mailu „papierovú poštu na stole, alebo v schránke“. Ďalším stupňom vo vývoji by bolo mapovanie fotografií z informačného systému na hlavy avatarov, čím by sme dosiahli väčšiu personifikáciu. Ďalšou podporovanou vlastnosťou by mohol byť video-hovor s mapovaním tváre na postavičku v modeli, čo by ocenili hlavne konzultanti (konzultácia z pohodlia domova) alebo kolegovia, prípadne spolužiaci, ktorí sa v modeli stretnú. Virtuálne navštívení majú veľkú výhodu v zachovaní súkromia (nezverejňujete svoj status, nezverejňujete súkromné účty v komunikátoroch, ani žiadne osobné údaje). Viditeľnosť v modeli by samozrejme ostala iba ako doplnková a voliteľná služba. Vhodným doplnkom by bola možnosť záznamu poznámok a skíc na tabuliach v učebniach a rozdelenie používateľských práv.

S naším systémom zabudovaným v obchode ponúkajúcim elektronický predaj by ste tak mali možnosť získať interaktívnu odozvu na otázky spojené s nákupom. Možnosť prehrať svoje rozhodnutie s ostatnými nakupujúcimi nadšencami, alebo si len obzrieť nový tovar vo viacerých obchodoch rýchlejšie ako kedykoľvek pred tým (skrytá možnosť pre dámy: „možnosť spoločného nakupovania aj s kamarátkami vzdialenými míle ďaleko“).





Obr. 12.1 Ukážka možnosti použitia avatarov

## A. Inštalčná príručka

### Požiadavky

- operačný systém GNU/Linux Debian 6.0 (alebo Windows XP, alebo vyššia verzia, prípadne iná distribúcia Linuxu, prípadne BSD)
- Databázový systém MySQL 5.1 alebo vyšší
- PHP 5.1.6 alebo vyššia verzia s knižnicou GD 5.3.3 alebo vyššia verzia
- webový server Apache 2.2.16 alebo vyššia verzia s nasledujúcimi modulmi
  - mod\_rewrite (nie je nevyhnutný – viz. časť A.2.2)
  - mod\_php (podporovaná verzia php 5.1.6 alebo vyššia)
  - mod\_auth\_digest (nie je nevyhnutný)
  - mod\_ssl (nie je nevyhnutný)
  - mod\_deflate

### Popis aplikácií

VirtualFIIT sa skladá z dvoch samostatných webových aplikácií admin a client, ktoré je odporúčané nasaďiť na dva samostatné virtuálne weby v rámci servera Apache (prípadne na dva samostatné servery).

- Admin – je aplikácia určená pre správcu systému a umožňuje vykonávať import zo systému AIS. Pomocou tejto aplikácie je možné importovať aktuálny rozvrh, popis miestností a zamestnancov, ktorí sa budú zobrazovať v aplikácii client pri prechádzaní virtuálnym modelom. Aplikácia tiež obsahuje funkcionality pre vytvorenie a vymazanie tabuliek z databázy.
- Client – je aplikácia určená verejnosti. Aplikácia obsahuje model budovy FIIT a komunikuje s databázou (predtým naplnenou pomocou aplikácie admin), odkiaľ získava dáta o rozvrhoch, miestnostiach a zamestnancoch. Táto aplikácia nerobí žiadne úpravy v databáze.

## A.1 Nainštalovanie súčastí Virtualfiit na webový server (quickstart)

### A.1.1 Nasadenie aplikácie server na webový server

1. Vytvorte samostatné virtuálne weby v serveri Apache a povoľte všetky potrebné moduly
2. Vytvorte databázu a databázové konto na MySQL serveri, ktoré má všetky práva v danej databáze.
3. Rozbaľte súboru admin.zip do koreňového adresára daného virtuálneho webu
  - a) upravte súbor /application/admin/config/database.php
    - `$db['default']['hostname'] = '<nazov MySQL servera>' (napr. localhost)`
    - `$db['default']['username'] = '<meno pouzivatela>';`
    - `$db['default']['password'] = '<heslo pouzivatela>';`
    - `$db['default']['database'] = '<nazov databazy>'; (napr virtualfiit)`
  - b) upravte súbor /application/admin/config/config.php
    - `$config['base_url'] = 'http(s)://<adresa_servera>/admin/'; (napr: https://virtualfiit.sk/admin/)`
    - `$config['base_res_url'] = 'http(s)://<adresa_servera>/admin/resources/'; (napr: http://virtualfiit.sk/admin/resources)`
4. Spustite aplikáciu na `http(s)://<nazov_servera>/admin`

Aplikácia admin neobsahuje internú schopnosť autentifikácie, preto odporúčame vytvoriť tzv. digest autentifikáciu pomocou Apache modulu `mod_auth_digest` a zabezpečiť pripojenie pomocou `mod_ssl`.

### A.1.2 Nasadenie aplikácie client na webový server

1. Vytvorte samostatné virtuálne weby v serveri Apache a povoľte všetky potrebné moduly
2. Vytvorte databázu a databázové konto na MySQL serveri, ktoré má všetky práva v danej databáze a naplňte databázu údajmi cez admin aplikáciu
3. Rozbaľte súbor `client.zip` do koreňového adresára daného virtuálneho webu
  - a) upravte súbor `/application/client/config/database.php`
    - `$db['default']['hostname'] = '<nazov MySQL servera>' (napr. localhost)`
    - `$db['default']['username'] = '<meno pouzivatela>';`
    - `$db['default']['password'] = '<heslo pouzivatela>';`
    - `$db['default']['database'] = '<nazov databazy>'; (napr virtualfiit)`
  - b) upravte súbor `/application/client/config/config.php`
    - `$config['base_url'] = 'http(s)://<adresa_servera>'; (napr: http://virtualfiit.sk/)`
    - `$config['base_res_url'] = 'http(s)://<adresa_servera>/resources/'; (napr: http://virtualfiit.sk/admin/resources)`
4. Spustite aplikáciu na `http(s)://<nazov_servera>/`

### A.1.3 Príklad inštalácie na operačný systém GNU/Linux Debian 6.0 (tutoriál)

Nasledujúca časť popisuje príklad inštalácie aplikácie na operačný systém Debian 6.0. Naším cieľom je mať nainštalovaný admin aj client na dvoch virtuálnych weboch na jednom operačnom systéme. Virtuálny web pre aplikáciu admin bude navyše chránený zabezpečeným prenosom pomocou http over ssl (https) a prístup bude podmienený používateľským menom a heslom pomocou módu `mod_auth_digest`. Aplikácia client bude bežať na nezabezpečenom pripojení, ktoré je pre túto aplikáciu postačujúce. Predpokladáme, že doména, na ktorú bude aplikácia nasadená je `virtualfiit.sk` a že všetky operácie sa dejú pod používateľom `root`.

Postup:

1. Nainštalujte webový server Apache a všetky módy na základe požiadaviek a nainštalujte MySQL na základe požiadaviek. Inštaláciu odporúčame vykonať z oficiálnych repozitárov pomocou programu `apt` (prípadne `aptitude`, alebo `Synaptic`) a povolenie modulov pomocou nasledujúcich príkazov:
  - `$ a2enmod ssl`
  - `$ a2enmod rewrite`
  - `$ a2enmod deflate`
  - `$ a2enmod auth_digest`
2. Vytvorte nasledujúce adresáre s týmito právami `owner:root; group:root; permissions:755`:
  - `$ mkdir /var/www/admin`
  - `$ chmod 755 /var/www/admin`
  - `$ chown root:root /var/www/admin`
  - `$ mkdir /var/www/client`
  - `$ chmod 755 /var/www/client`
  - `$ chown root:root /var/www/client`

3. Vytvorte súbor `/etc/apache2/adminUsers.md5`, ktorý bude obsahovať používateľov pre aplikáciu Admin. Pridanie používateľa `user` a súčasné vytvorenie súboru vykonajte nasledujúcim príkazom, pričom počas vykonávania dvakrát zadajte heslo používateľa:
  - `$ htpasswd -c /etc/apache2/adminUsers.md5 „Virtualfiit admin“ user`
 Poznámka: ostatných používateľov pridajte pomocou toho istého príkazu bez parametra `-c`.
4. Vytvorte dva virtuálne servery. Modifikujte pôvodné nastavenia pre virtuálne weby v adresári `/etc/apache2/sites-available/`.

**Tabuľka 1. Príklad konfigurácie pre aplikácie admin a client (zvýraznené sú vykonané zmeny)**

<b>Admin</b>	<b>Client</b>
Súbor: <code>/etc/apache2/sites-available/default-ssl</code>	Súbor: <code>/etc/apache2/sites-available/default</code>
<pre>&lt;IfModule mod_ssl.c&gt; &lt;VirtualHost *:443&gt;   ServerAdmin webmaster@localhost    <b>DocumentRoot /var/www/admin</b>   &lt;Directory /&gt;     Options FollowSymLinks     AllowOverride None   &lt;/Directory&gt;   &lt;Directory /var/www/admin&gt;     <b>Options Indexes FollowSymLinks</b>     <b>AllowOverride All</b>     Order allow,deny     allow from all   &lt;/Directory&gt;    &lt;Location /admin/&gt;     <b>AuthType Digest</b>     <b>AuthName "Virtualfiit admin"</b>     <b>AuthDigestDomain /admin/</b>     <b>AuthDigestProvider file</b>     <b>AuthUserFile /etc/apache2/adminUsers.md5</b>     <b>Require valid-user</b>   &lt;/Location&gt;    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/   &lt;Directory "/usr/lib/cgi-bin"&gt;     AllowOverride None     Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch     Order allow,deny     Allow from all   &lt;/Directory&gt;    ErrorLog \${APACHE_LOG_DIR}/error.log</pre>	<pre>&lt;VirtualHost *:80&gt;   ServerAdmin webmaster@localhost    <b>DocumentRoot /var/www/client</b>   &lt;Directory /&gt;     Options FollowSymLinks     AllowOverride None   &lt;/Directory&gt;   &lt;Directory /var/www/client&gt;     <b>Options Indexes FollowSymLinks</b>     <b>AllowOverride All</b>     Order allow,deny     allow from all   &lt;/Directory&gt;    ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/   &lt;Directory "/usr/lib/cgi-bin"&gt;     AllowOverride None     Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch     Order allow,deny     Allow from all   &lt;/Directory&gt;    ErrorLog \${APACHE_LOG_DIR}/error.log    LogLevel warn    CustomLog \${APACHE_LOG_DIR}/access.log combined    Alias /doc/ "/usr/share/doc/"   &lt;Directory "/usr/share/doc/"&gt;     Options Indexes MultiViews FollowSymLinks     AllowOverride None     Order deny,allow</pre>

<pre> LogLevel warn      CustomLog \${APACHE_LOG_DIR}/ssl_access.log combined      Alias /doc/ "/usr/share/doc/"     &lt;Directory "/usr/share/doc/"&gt;         Options Indexes MultiViews FollowSymLinks         AllowOverride None         Order deny,allow         Deny from all         Allow from 127.0.0.0/255.0.0.0 ::1/128     &lt;/Directory&gt;      SSLEngine on     SSLCertificateFile /etc/ssl/certs/ssl-cert-snakeoil.pem     SSLCertificateKeyFile /etc/ssl/private/ssl-cert- snakeoil.key  &lt;FilesMatch "\.(cgi shtml phtml php)\$"&gt;     SSLOptions +StdEnvVars &lt;/FilesMatch&gt; &lt;Directory /usr/lib/cgi-bin&gt;     SSLOptions +StdEnvVars &lt;/Directory&gt;  BrowserMatch "MSIE [2-6]" \     nokeepalive ssl-unclean-shutdown \     downgrade-1.0 force-response-1.0     BrowserMatch "MSIE [17-9]" ssl-unclean-shutdown  &lt;/VirtualHost&gt; &lt;/IfModule&gt; </pre>	<pre> Deny from all Allow from 127.0.0.0/255.0.0.0 ::1/128 &lt;/Directory&gt;  &lt;/VirtualHost&gt; </pre>
--	--

5. Reštartujte Apache pomocou príkazu:

- \$ /etc/init.d/apache restart

6. Nainštalujte databázu MySQL pomocou nasledujúceho SQL príkazu (namiesto xxx vložte heslo používateľa databázy):

```
CREATE USER 'virtualfiit'@'%' IDENTIFIED BY '***';
```

```
GRANT USAGE ON * . * TO 'virtualfiit'@'%' IDENTIFIED BY '***' WITH
MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0
MAX_UPDATES_PER_HOUR 0 MAX_USER_CONNECTIONS 0 ;
```

```
CREATE DATABASE IF NOT EXISTS `virtualfiit` ;
```

```
GRANT ALL PRIVILEGES ON `user` . * TO 'virtualfiit'@'%' ;
```

### 7. Nainštalujte aplikáciu admin:

- a) rozbaľte zip súboru **admin.zip** do adresára */var/www/admin*
- b) v súbore */var/www/admin/application/admin/config/config.php*
  - `$config['base_url'] = 'https://virtualfiit.sk/admin/';`
  - `$config['base_res_url'] = 'https://virtualfiit.sk/admin/resources/';`
- c) v súbore */var/www/admin/application/admin/config/database.php* zmeňte riadky:
  - `$db['default']['hostname'] = 'localhost';`
  - `$db['default']['username'] = 'virtualfiit';`
  - `$db['default']['password'] = 'xxx';` (miesto xxx naše heslo)
  - `$db['default']['database'] = ' virtualfiit';`

### 8. Nainštalujte aplikáciu client:

- a) rozbaľte zip súbor *client.zip* do adresára */var/www/client*
- b) v súbore */var/www/admin/application/admin/config/config.php*
  - `$config['base_url'] = 'https://virtualfiit.sk/';`
  - `$config['base_res_url'] = 'https://virtualfiit.sk/resources/';`
- c) v súbore */var/www/admin/application/admin/config/database.php* zmeňte riadky:
  - `$db['default']['hostname'] = 'localhost';`
  - `$db['default']['username'] = 'virtualfiit';`
  - `$db['default']['password'] = 'xxx';` (miesto xxx naše heslo)
  - `$db['default']['database'] = ' virtualfiit';`

Ak všetko prebehlo v poriadku, tak po týchto krokoch bude možné spustiť aplikácia Virtualfiit na nasledujúcich URL adresách:

Admin aplikácia

- <https://virtualfiit.sk/admin>

Client aplikácia

- <https://virtualfiit.sk>

Po úspešnom nainštalovaní servera a všetkých jeho súčastí, môžete pokračovať inštaláciou samotnej aplikácie Virtuálna FIIT.

## A.2 Rozšírené inštalačné možnosti

Táto časť inštalačnej príručky pojednáva o rozšírených inštalačných možnostiach aplikácie VirtuálnaFIIT. Ide o možnosti, ktoré sa dajú aplikovať najmä vtedy, ak nie je možné splniť niektoré požiadavky (ide o požiadavku nemožnosti použitia `mod_rewrite` a umiestnenia aplikácie do rôznych virtuálnych serverov, prípadne ich umiestnenie do koreňového adresára virtuálneho webu).

### A.2.1 Inštalácia súčastí admin a client na jeden virtuálny web

V tomto prípade budeme používať inštalačný súbor **complete.zip**. Celý obsah `complete.zip` nahrajte na server do koreňového alebo vami zvoleného podadresára na serveri. Umiestnenie v inom ako koreňovom adresári nemá na funkčnosť aplikácie po správnom nastavení URL adresy vplyv.

Inštalčný priečinok aplikácie Virtuálna FIIT musí obsahovať:

- priečinok **admin**,
- priečinok **application**,
- priečinok **resources**,
- priečinok **system**,
- súbor **index.php**,
- súbor **.htaccess**.

Ďalšie nastavenia sa nachádzajú v nasledujúcich častiach:

### A.2.2 Nastavenie prepisovania URL adries (mod\_rewrite)

Aplikácia má štandardne zapnutý režim prepisovania URL adries. Ak váš server s touto funkciou nepracuje správne, alebo si nepracujete používať túto funkciu, vypnúť ju môžete upravením súborov .htaccess, ktoré sa nachádzajú v koreňovom adresári aplikácie a v adresári admin, nasledovne:

V oboch súboroch prepíšete prvý riadok z: RewriteEngine on na: RewriteEngine off.

### A.2.3 Rozšírené nastavenie URL adries aplikácie

Jedným z najdôležitejších nastavení je správne nastavenie URL adries aplikácie, preto mu venujte dostatočnú pozornosť! Nastavenie URL adries je závislé od funkcie prepisovania URL adries!

#### Administračné rozhranie

1. Otvorte súbor „application/admin/config/config.php“.
2. Do premennej \$config['base\_url'] vložte URL adresu administračného rozhrania. Ak máte vypnutú funkciu prepisovania URL adries, v adrese musí byť definovaný aj súbor „index.php“. Adresa musí končiť znakom „/“!  
Príklad so zapnutým režimom mod\_rewrite: „<https://virtualnafit.sk/admin/>“  
Príklad s vypnutým režimom mod\_rewrite: „<https://virtualnafit.sk/admin/index.php/>“
3. Do premennej \$config['base\_res\_url'] vložte URL adresu, kde sú umiestnené zdroje administračného rozhrania. Štandardne sú zdroje umiestnené v priečinku „admin/resources“. Adresa zdrojov nie je závislá na režime prepisovania URL adries. Adresa zdrojov tiež musí končiť znakom „/“!  
Príklad adresy zdrojov admin. rozhrania: „<https://virtualnafit.sk/admin/resources/>“

#### Rozhranie klienta

1. Otvorte súbor „application/client/config/config.php“.
2. Do premennej \$config['base\_url'] vložte URL adresu rozhrania klienta. Ak máte vypnutú funkciu prepisovania URL adries, v adrese musí byť definovaný aj súbor „index.php“. Adresa musí končiť lomkou!  
Príklad so zapnutým režimom mod\_rewrite: „<http://virtualnafit.sk/>“  
Príklad s vypnutým režimom mod\_rewrite: „<http://virtualnafit.sk/index.php/>“
3. Do premennej \$config['base\_res\_url'] vložte URL adresu, kde sú umiestnené zdroje rozhrania klienta. Štandardne sú zdroje umiestnené v priečinku „resources“. Adresa zdrojov nie je závislá na režime prepisovania URL adries. Adresa zdrojov taktiež musí končiť znakom „/“!

Príklad adresy zdrojov rozhrania klienta: „<http://virtualnafiiit.sk/resources/>“

## A.2.4 Nastavenie prístupu do databázy

Administračné rozhranie a aj rozhranie klienta musia používať rovnakú databázu, ale je nutné zadať prístupové údaje do databázy na dvoch miestach, samostatne pre administračné rozhranie a samostatne pre rozhranie klienta.

Otvorte súbory:

- „application/admin/config/database.php“ a
- „application/client/config/database.php“,

a do oboch zadajte rovnaké prístupové údaje. Odporúčame zadať len prístupové údaje a ostatné polia nemeniť.

Nastavenie prístupu do databázy:

```
$db['default']['hostname'] = 'localhost';  
$db['default']['username'] = 'prihlasovacie_meno';  
$db['default']['password'] = 'heslo';  
$db['default']['database'] = 'schema';
```

## A.3 Príprava databázy – používanie admin časti aplikácie

### A.3.1 Nainštalovanie databázy a importovanie údajov z AIS

Po úspešnom nastavení URL adres aplikácie a prístupu do databázy je nutné vytvoriť tabuľky v databáze. Aplikácia admin obsahuje samo-inštalačný skript, ktorý vytvorí všetky požadované tabuľky a naplní ich preddefinovanými hodnotami. V internetovom prehliadači otvorte administračné rozhranie (URL adresa administračného rozhrania musí byť zhodná s adresou zadanou v sekcii Nastavenie URL adres aplikácie pre administračné rozhranie) a kliknite na odkaz *Nainštalovať databázu (vykonať CREATE tabuliek)*.

#### Import údajov

Po vytvorení tabuliek kliknite v administračnom rozhraní na odkaz *Importovať údaje (zahŕňa import zamestnancov, miestností a predmetov)* a postupujte podľa pokynov zobrazených v jednotlivých krokoch importu údajov z AIS.

#### Import zamestnancov

- Označte pracoviská, ktorých zamestnancov si prajete importovať a kliknite na odkaz *Importuj zamestnancov*.

#### Import miestností

- Označte všetky budovy, ktorých miestnosti si prajete importovať a kliknite na odkaz *Importuj miestnosti*.
- Ak si želáte importovať miestnosti z budovy, ktorá nie je uvedená v zozname, postupujte podľa návodu zobrazeného nižšie



**Import predmetov**

- Podľa návodu zobrazeného na stránke importu predmetov vložte všetky požadované informácie pre vykonanie importu.

Po importe údajov z AIS, je pre správne fungovanie aplikácie nutné importovať aj údaje z rozvrhu fakulty. Import údajov z rozvrhu nie je kompatibilný s internetovým prehliadačom Google Chrome! Kliknite na odkaz *Importovať osoby z rozvrhu fakulty* a podľa pokynov zobrazených na stránke vložte požadované údaje. Po dokončení importu osôb z rozvrhu následne kliknite na odkazy:

- *Importovať miestnosti z rovnakého rozvrhu*
- *Importovať predmety z rovnakého rozvrhu*
- *Importovať rovnaký rozvrh*

Po dokončení importu údajov z rozvrhu, je aplikácia Virtuálna FIIT pripravená na použitie.

**A.3.2 Odinštalovanie aplikácie Virtuálna FIIT**

Ak si prajete odinštalovať aplikáciu Virtuálna FIIT, otvorte si v internetovom prehliadači administračné rozhranie a kliknite na odkaz *Odinštalovať databázu (vykonať DROP tabuliek)*. Tým sa odstránia všetky údaje a tabuľky, ktoré aplikácia používala. Následne môžete aplikáciu vymazať zo servera.

## B. Používateľská príručka

Daná používateľská príručka obsahuje všetky potrebné informácie pre používateľov aplikácie Virtuálna FIIT. Používateľská príručka je rozdelená do troch základných častí.

V prvej časti používateľskej príručky je zobrazené umiestnenie základných ovládacích prvkov spolu s ich popisom a popisom výstupných prvkov. Kapitola Ovládanie obsahuje opis spôsobu ovládania v každom z režimov, ktoré daná aplikácia podporuje. Otázky týkajúce sa navigácie nájdete v poslednej časti tejto príručky s názvom Navigácia. Nachádzajú sa tu popísané základné scenáre pre vyhľadanie objektov v interaktívnom modeli.

### B.1 Rozloženie ovládacích prvkov

Rozloženie ovládacích prvkov na úvodnej obrazovke je znázornené na Obr. 1. Základnú obrazovku aplikácie môžeme rozdeliť do piatich aktívnych častí. Pre ich jednoduchšiu identifikáciu sú priradené aj číselné označenia.



Obr. B.1 Úvodná obrazovka aplikácie s doplneným rozložením

## Interaktívna tabuľa

Pod číselným označením 1 je vyznačená pravá časť obrazovky. Daný ovládací prvok je interaktívna tabuľa, ktorá je zobrazená aj na Obr. B.2. Poskytuje:

- informácie o aktuálnom umiestnení,
- výsledky vyhľadávania z 3. časti úvodnej obrazovky (vyhľadávacieho poľa)
- v prípade prednáškovej sály jej využitie v priebehu týždňa,
- priame prepojenie na vyhľadané objekty do AIS,
- v prípade kancelárie viacerých osôb, zobrazí informácie o každej osobe s možnosťou prejsť na podrobnejšie informácie jedným kliknutím,
- v prípade vstupu do kancelárie, v ktorej sídli iba jedna osoba, alebo jej vyhľadaním v 3.časti úvodnej obrazovky sa zobrazí okrem základných informácií o danej osobe aj jej týždenný rozvrh,
- navigačné tlačidlo (tlačidlo spúšťajúce navigáciu k danej miestnosti),
- tlačidlo „Ukáž na mape“ spúšťa režim náhľadu so zameraním na aktuálny obsah interaktívnej tabule (viac v časti Ovládanie)



Obr. B.2 Interaktívna tabuľa

## Výber režimu

Aktívna časť aplikácie, určená pre výber režimu, je na Obr. B.1 označená číslom 2. Nachádza sa v pravej hornej časti obrazovky na ľavo od loga projektu (VirtualFIIT) a na pravo od vyhľadávacieho okna.

Pre výber režimu aplikácia obsahuje graficky spracované tlačidlá v tvare oka (režim prehliadky), kráčajúcej postavičky (režim chôdze) a lietadla (režim voľného letu). Ako je vidieť na Obr. B.3, aktuálne vybraná možnosť je vždy zvýraznená jasnejšou farbou.



Obr. B.3 Panel pre voľbu režimu prehliadania

## Vyhľadávacie okno

Vyhľadávacie okno je jasný veľmi dobre viditeľný ovládací prvok. Na Obr. B.1 je označený číslom 3.

Pre lepšiu ilustráciu je doplnený Obr. B.4, uvedený nižšie. Obsahuje iba jedno tlačidlo slúžiace na potvrdenie vyhľadávania (rovnakú funkcionálnosť má stlačenie tlačidla Enter na klávesnici).



Obr. B.1 Vyhľadávacie okno

## Nastavenia

Pre danú aplikáciu nie je potrebné veľké množstvo nastavení, ku ktorým sa je možné dostať z úvodnej obrazovky. Na Obr. B.1 sú nastavenia označené číslicou 4.

Na Obr. B.3 sú zobrazené tlačidlá prístupu k nastaveniam. Ich umiestenie je v pravom hornom rohu aplikácie, nakoľko sa predpokladá len minimálne využívanie týchto prvkov pri pravidelnom používaní danej aplikácie.



Obr. B.5 Nastavenia aplikácie


## 3D panel

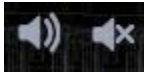
Najdôležitejší interaktívny prvok danej aplikácie je 3D panel, ktorý zobrazuje všetky prijaté podnety aplikácie na modeli budovy FIIT.


Jeho umiestenie je v strede obrazovky a na Obr. B.1 je označený najvyšším číslom 5. Rozmery daného prvku sa môžu meniť na základe nastavení (celá obrazovka / normálne zobrazenie). V prípade, že nie je možné pohybovať sa v danom paneli a preštudovali ste spôsoby pohybu pomocou Pomocníka (časť 4 Nastavenia – ikona otáznik), skontrolujte prosím svoj internetový prehliadač a jeho kompatibilitu s technológiou WebGL.

## Ovládanie


V prvej časti si vysvetlíme spôsoby ovládania samotnej aplikácie a v druhej časti ovládanie pohybov. Ovládanie pohybov v danej aplikácii je závislé na zvolenom režime prehliadania.

Stlačením tlačidla  prepnete režim prehliadania na celú obrazovku (zväčší sa 3D panel danej aplikácie).

Tlačidlo s jedným so zobrazením:  slúži na zapínanie a vypínanie zvuku, pričom vždy v aplikácii máte možnosť pozorovať práve jednu z týchto možností,

Tlačidlo  slúži na vyvolanie pomocníka, v ktorom je zrozumiteľne a jasne popísaný spôsob pohybu po virtuálnom modeli FIIT.



Lišta:  s označením poschodia, slúži na prepínanie poschodí v režime náhľadu. Pri režime prechádzok plní funkciu tlačidiel vo výťahu (daná lišta je vtedy skrátaná o možnosti: All, 6, -2, ktoré nemajú pokrytie danými výťahom).

### Ovládanie v režime náhľadu



- rotáciu kamery dosiahneme držaním ľavého tlačidla myšky a súčasného pohybu do zvolenej strany,



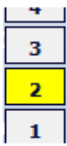
- pre posunutie stredu otáčania kamery, držíme stlačené pravé tlačidlo myšky a súčasne ňou pohybujeme v smere žiadaneho pohybu,



- stredným kolieskom myši sa v danom režime môžete posúvať bližšie, prípadne ďalej od stredu otáčania,



- stlačením ľavého tlačidla myšky nad miestnosťou vyvoláte zobrazenie detailov o danej miestnosti (jej obsadenie, účel, rozvrh),



- bočná lišta slúži na prepínanie zobrazovaných poschodí.

### Ovládanie v režime chôdze



- otáčanie pozorovateľa prebieha za stlačeného pravého tlačidla a pohybom myšky do strán (pri posune dopredu sa zrak pozorovateľa dvíha hore a pri posune dozadu sa zrak pozorovateľa posúva k zemi),



- pohyb po priestoroch budovy môžeme realizovať buď pomocou šípok na klávesnici, kde šípky do strán realizujú otáčanie a šípka hore a dole realizuje chôdzu dopredu a dozadu,



- pre „hráčov“ je tu pripravená aj sekundárna možnosť ovládania danej aplikácie a to pomocou kláves „w“, „s“, „a“, „d“. Tu je rozdiel v tlačidlách „a“ a „d“, ktoré na rozdiel od

bočných šípok nerealizujú otáčanie do strán, ale chôdzu do strán. Klávesy „w“ a „s“ realizujú chôdzu dopredu a dozadu,



- zdvíhanie a sklápanie zraku je realizované pomocou kláves *Page up* (zdvihnutie pohľadu) a *Page down* (sklopenie pohľadu),



- pre rýchlejší pohyb je pre Vás pripravená nielen možnosť chôdze, ale aj behu (aktivujte držaním tlačidla *Shift*),



- na otvorenie dverí je pripravená hneď dvojica tlačidiel, pre optimalizovanie pohodlia pri prezeraní budovy: tlačidlo „e“ je vhodnejšie v prípade využitia ovládania WSAD a tlačidlo *Enter* v prípade používania šípok.

### Ovládanie v režime voľného letu



- smer pohybu pozorovateľa prebieha za stlačeného pravého tlačidla a pohybom myši do strán (pri posune dopredu sa smer pozorovateľa dvíha smerom k oblohe a pri posune dozadu sa smer pozorovateľa posúva k zemi, posun do strán mení smer pozorovateľa do strán),



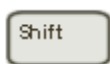
- voľný let môžete realizovať buď pomocou šípok na klávesnici, kde šípky do strán realizujú otáčanie a šípka hore a dole realizuje voľný let dopredu a dozadu,



- pre „hráčov“ je tu opäť pripravená aj sekundárna možnosť ovládania danej aplikácie, a to pomocou kláves „w“, „s“, „a“, „d“ (význam tlačidiel je rovnaký ako v režime chôdze),



- zdvíhanie a sklápanie zraku je realizované pomocou kláves *Page up* (zdvihnutie pohľadu) a *Page down* (sklopenie pohľadu),



- pre dlhé lety je pre Vás pripravená možnosť zrýchleného letu realizovaná tlačidlom *Shift*,



- v danom režime pribudli tlačidlá na vertikálny posun (tlačidlo *Space* je určené na stúpanie a tlačidlo *Ctrl* je určené na klesanie v modeli).

### **Navigácia**

Aplikácia podporuje hneď niekoľko spôsobov, ktorými sa používateľ môže dostať do vyhľadávaného miesta (prípadne miestnosti, kde sa nachádza určitá osoba).

1. Využite vyhľadávacie okno popísané vyššie.
2. K dispozícii sú na výber dve možnosti:




- po stlačení daného tlačidla sa zmení režim na režim náhľadu a zobrazí sa miestnosť v modeli,



- do modelu pribudnú šípky ukazujúce cestu k vyhľadanému miestu,

3. Presuňte sa do cieľa (popísané v časti Ovládanie), teraz môžete zrušiť navigačné šípky, pomocou nápisu „zrušiť“ v hornej časti interaktívnej tabule.

A screenshot of a dark grey navigation bar with a white border. It contains the text 'Navigácia do [D-323 Zrušiť](#)' in white font.

Navigácia do [D-323 Zrušiť](#)

## C. Textúry

Textúra oblohy - Coppercube editor - <http://www.ambiera.com/coppercube/index.html>

Textúra podlahy - [http://texturebits.blogspot.com/2008/02/tileable\\_27.html](http://texturebits.blogspot.com/2008/02/tileable_27.html)

Textúra drevených dverí (<http://www.defcon-x.de/c4d/textures>)

Textúra stropu - <http://machinekode.com/src/tag/ceiling-tile-texture>

Textúry trávy (grass.jpg), štrku na streche (gravel.jpg), betónu na bočných schodiskách (concrete.jpg), kov použitý pri výrobe textúr do výťahoch (brushed\_metal.jpg), drevená textúra pre lavice - Patrick Hoesley <http://www.flickr.com/photos/zooboing/collections/72157622668797999/>

Textúry tabule a kriedou napísaných písmen

<http://www.chrislanephoto.com/blog/2010/08/20/blackboard-and-full-lettersymbol-free-high-res-texture-set/>

Látková textúra pre lavice v prednáškových miestnostiach

<http://webtreats.mysitemyway.com/8-tileable-fabric-texture-patterns/comment-page-1/>

Zvuky pre zatváranie a otváranie dverí a zvuk krokov pochádzajú zo stránky

<http://www.soundjay.com>

Zvuk pri príchode výťahu na poschodie

<http://www.freesound.org/samplesViewSingle.php?id=91926>

Zvuk otvárania /zatvárania výťahových dverí:

<http://www.freesound.org/samplesViewSingle.php?id=77789>

Zvuk behu výťahu:

<http://www.freesound.org/samplesViewSingle.php?id=33873>



## D. Popis infoservera

### D.1 Spoločné parametre služieb

Každá služba má v URL adrese nejaké parametre, ktoré sú spoločné.

#### D.1.1 Formát výstupu

URL adresa: `http://<server>/info/.../<json|html>`

`<json|html>` - určuje formát výstupu, pre testovacie účely bol zavedený aj formát `html`, ak sa formát nezadá, automaticky sa berie `json`.

#### Testovanie vstupov

Pomocou nasledujúcej metódy je možné overiť správnosť údajov, ktoré info server poskytuje na testovacej množine údajov, ktorú je však nutné najprv nahráť do databázy. Testovacie údaje sú uložené ako `sqldump` na CD (alebo v `svn` v adresári `trunk`) v adresári `databáza` pod názvom `virtualfiit_test_data.sql`.

URL adresa : `http://<server>/test`

### D.2 Informácie o miestnosti

#### D.2.1 Základné informácie o miestnosti

Služba vráti rozvrh a zoznam osôb, ktoré majú v zadanej miestnosti kanceláriu. To znamená, že služba vráti buď rozvrh alebo zoznam ľudí.

URL adresa : `http://<server>/info/room/about/<nazovMiestnosti>/<json|html>`

- `<nazovMiestnosti>` - celý názov miestnosti. Napr. `de300, D-202`
- `<json|html>` - formát výstupu

Výstupný json objekt:

```
{
  "room_name": "<nazovMiestnosti>",
  "result": {
    timetable: [ {
      time_from: čas začiatku
      time_to: čas konca
      day: názov dňa
      course: názov predmetu
      course_url: url adresa predmetu do AIS
      teacher: meno učiteľa
      teacher_url: url adresa učiteľa do AIS
      room: názov miestnosti (rovnaké ako room_name)
      room_url: url adresa miestnosti do AIS
      note: text poznámky pod čiarou
      type: typ predmetu (Prednáška, cvičenie)
      restriction: text obmedzenia
    }, { ... }, { ... }, ... ],
    person: [ {
      name: meno osoby
      ais_url: url adresa učiteľa do AIS
      email: email osoby
    }
  ]
}
```

```

        phone:      telefón osoby
        room:       názov miestnosti (rovnaké ako room_name)
    }. {...}, {...}, ...] }
}

```

## D.3 Informácie o osobách

### D.3.1 Lokalizácia osoby

Služba vráti názov miestnosti, v ktorej sa osoba nachádza v zadanom čase a dni:

URL adresa :

http://<server>/info/person/localise/<menoČloveka>/<day>/<time>/<json|html>

URL adresa : http://<server>/info/person/now/<menoČloveka>/<json|html>

- <MenoČloveka> - kompletne meno človeka aj tituly
- <day> - názov dňa podľa databázy (Pondelok, Utorok, Streda, Štvrtok, Piatok, Sobota, Nedeľa), alebo číslo dňa: 1- pondelok, 7- nedeľa
- <time> - čas vo formáte *hh:mm*
- <json|html> - formát výstupu

Variant now lokalizuje človeka v aktuálnom dni a čase servera.

Výstupný json objekt:

```

{
    „person_name“:   meno osoby
    „day“:           deň (číselne alebo slovne) v ktorom bolo vyk. hľadanie
    „time“:          čas vo formáte hh:mm, v ktorom bolo vykonané hľadanie
    „room“:         miestnosť v ktorej sa dotyčná osoba v danom čase nachádza
}

```

### D.3.2 Základné informácie o osobe

Služba vráti základné informácie o osobe:

URL adresa : http://<server>/info/person/about/<menoČloveka>/<json|html>

- <MenoČloveka> - kompletne meno človeka aj tituly
- <json|html> - formát výstupu

Výstupný json objekt:

```

{
    „person_name“:   meno osoby
    „result“ {
        „name“:      meno osoby
        „ais_url“:   url adresa osoby do AIS
        „email“:     email osoby
        „phone“:     telef. číslo
        „room“:      miestnosť, kde má daná osoba kanceláriu
    }
}

```

### D.3.3 Predmety, ktoré daná osoba učí

Jedná sa o predmety, ktoré daná osoba učí

URL adresa : `http://<server>/info/person/courses/<menoČloveka>/<json|html>`

- `<MenoČloveka>` - kompletne meno človeka aj tituly
- `<json|html>` - formát výstupu

Výstupný json objekt:

```
{
  „person_name“:      meno osoby
  „result“ {
    „course“:         názov predmetu
    „course_url“:     URL adresa predmetu do AIS
  }
}
```

### D.3.4 Rozvrh danej osoby

Služba vráti rozvrh danej osoby v daný deň. Ak chceme zobrazíť rozvrh pre všetky dni, tak do parametra day vložíme číslo nula „0“.

URL adresa : `http://<server>/info/person/timetable/<menoČloveka>/<day>/<json|html>`

- `<MenoČloveka>` - kompletne meno človeka aj tituly
- `<day>` - názov dňa podľa databázy (Pondelok, Utorok, Streda, Štvrtok, Piatok, Sobota, Nedeľa), alebo číslo dňa: 1- pondelok, 7- nedeľa; 0-všetky dni
- `<json|html>` - formát výstupu

Výstupný json objekt:

```
{
  „person_name“:      meno osoby
  „day“:              deň (číselne, alebo slovne, záleží od vstupu) v ktorom bolo vyk.
```

hľadanie

```
  „result“: [{
    „course“:         názov predmetu
    „course_url“:     URL adresa predmetu do AIS
    „teacher“:        Meno učiteľa – rovnaké ako meno osoby
    „teacher_url“:    URL adresa osoby do AIS
    „room“:           Názov miestnosti
    „room_url“:       URL adresa miestnosti do AIS
    „note“:           Poznámka pod čiarou k danému predmetu
    „type“:           typ predmetu (Prednáška, cvičenie)
    „restrition“:     text obmedzenia
  }...{...}]
}
```

## D.4 Vyhľadávanie osôb, miestností, predmetov, alebo všetkého

Služba vráti výsledky voči zadanému vyhľadávaciemu reťazcu:

URL adresa : `http://<server>/info/search/all/<SearchString>/<json|html>`

URL adresa : `http://<server>/info/search/person/<SearchString>/<json|html>`

URL adresa : `http://<server>/info/search/room/<SearchString>/<json|html>`

URL adresa : `http://<server>/info/search/course/<SearchString>/<json|html>`

- <SearchString> - Vyhľadávaný reťazec
- <json|html> - formát výstupu

Výstupný json objekt:

```
{
  „search_string“:      Vyhľadávaný reťazec
  „person“: [ { „name“ :   Meno osoby
                „ais_url“:  Url adresa osoby do AIS
                }, {...}, {...}, ... ]
  „room“: [ { „name“:      Názov miestnosti
              „ais_url“:   Url adresa miestnosti do AIS
              },{...}, {...}, ... ]
  „course“:[ { „name“:     Názov predmetu
              „ais_url“:   Url adresa predmetu do AIS
              },{...},{...}, ... ]
}
```

## D.5 Informácie o predmete

### D.5.1 Rozvrh pre daný predmet

Služba vráti rozvrh pre zadaný predmet v konkrétny deň. Ak chceme zobrazit' rozvrh pre všetky dni, tak do parametra day vložíme číslo „0“.

URL adresa : <http://<server>/info/course/timetable/<názovPredmetu>/<day>/<json|html>>

- <názovPredmetu> - kompletný názov predmetu
- <day> - názov dňa podľa databázy (Pondelok, Utorok, Streda, Štvrtok, Piatok, Sobota, Nedeľa), alebo číslo dňa: 1- pondelok, 7- nedeľa; 0-všetky dni
- <json|html> - formát výstupu

Výstupný json objekt:

```
{
  „course_name“:      názov predmetu
  „day“:              deň (číselne, alebo slovne) v ktorom bolo vyk. hľadanie
  „result“: [ { „course“:  názov predmetu
                „course_url“: URL adresa predmetu do AIS
                „teacher“:   Meno učiteľa
                „teacher_url“: URL adresa osoby do AIS
                „room“:      Názov miestnosti
                „room_url“:  URL adresa miestnosti do AIS
                „note“:      Poznámka pod čiarou k danému predmetu
                „type“:      typ predmetu (Prednáška, cvičenie)
                „restrition“: text obmedzenia
                }...{...}... ]
}
```

## E. Formulár pre používateľské testovanie

Test obsahuje 10 otázok týkajúcich sa aplikácie Virtuálna FIIT. Zameraný je hlavne na:

- Intuitívne využívanie tlačidiel
- Orientácia v modeli
- Využívanie interaktívnych prvkov

Zodpovedajte prosím nasledujúcich 10 odpovedí spojených s úlohami v aplikácii Virtuálna FIIT.

1. Zobrazte legendu (help / pomocníka) aplikácie Virtuálna FIIT.

Podarilo sa: Áno<sup>50</sup> / Nie      Čas: .....s

2. Prepnete režim chôdze na režim voľného letu a zobrazenie aplikácie prepnete na celú obrazovku.

Podarilo sa: Áno / Nie      Čas: .....s

3. Vypnite režim zobrazenia na celú obrazovku. Do vyhľadávania vložte meno miestnosti T-PC B presuňte sa nad danú miestnosť. Aké označenie (meno) má daná miestnosť v novej budove (vložte názov z interaktívnej nápovedi, nie názov T-PC B)?

Podarilo sa (napíšte názov): ..... / Nie      Čas: .....s

4. Zmeňte režim náhľadu na režim chôdze a pokúste sa nájsť výťah bez pomoci funkcionality vyhľadávania (metódou pokus omyl / a hľadaním v budove).

Podarilo sa: Áno / Nie      Čas: .....s

5. Privolajte výťah (E / Enter) a vyvezte sa na najvyššie poschodie. Ste na streche? Viete ako by ste sa tam mohli dostať?

Podarilo sa: Áno / Nie      Čas: .....s

6. Zistite kto má cvičenie z predmetu Princípy softvérového inžinierstva v stredu v čase 19:00 – 20:50 a zároveň sa presuňte do miestnosti, kde sa konáva dané cvičenie.

Podarilo sa: Áno / Nie      Čas: .....s

7. Vyhľadajte medzi osobami osobu prof. Ing. Máriu Bielikovú PhD., ďalej si zapnite navigáciu do jej kancelárie a prejdite sa do nej.

Podarilo sa: Áno / Nie      Čas: .....s

8. Je možné získať zjednodušený prístup do systému AIS pomocou danej aplikácie (priamy odkaz na časť systému AIS s hľadanou informáciou)?

Áno / Nevieť čo myslíte / Nie.

---

<sup>50</sup> Správnu odpoveď vyznačte jej škrtnutím, podčiarknutím, alebo zakrúžkovaním.

9. Páčila sa Vám daná aplikácia?

Jedinečná a úžasná / Úžasná / Dobrá / Priemerná / Slabá / Hrozná aplikácia

10. Čo Vám v danej aplikácii chýbalo?

Nič / .....

.....

.....