

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Just18

Dokumentácia k riadeniu projektu

Vedúci tímu: Ing. Peter Kapec, PhD.

Členovia tímu: Bc. Martin Gašpar

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Peter Marušin

Bc. Bence Ligárt

Bc. Miloslav Slížik

Bc. Marek Škriečka

Akademický rok: 2017/2018

Obsah

1	Úvod	5
2	Členovia tímu.....	6
2.1	Manažérské úlohy.....	7
2.2	Vývojárske úlohy.....	7
2.3	Podiel práce na častiach inžinierskeho diela.....	8
3	Aplikácie manažmentov	9
3.1	Manažment dokumentácie	9
3.2	Manažment komunikácie a plánovania úloh	10
3.3	Manažment verziovacieho systému a prehliadok kódu	11
3.4	Nástroj na manažment úloh v tíme	13
4	Sumarizácia šprintov	15
4.1	Úvodné stretnutie.....	15
4.2	Šprint č.1.....	15
4.3	Retrospektíva šprintu č. 1	15
4.4	Šprint č. 2.....	16
4.5	Retrospektíva šprintu č.2	16
4.6	Šprint č. 3.....	16
4.7	Retrospektíva šprintu č. 3	17
4.8	Šprint č. 4.....	17
4.9	Retrospektíva šprintu č. 4	17
4.10	Šprint č. 5.....	18
4.11	Retrospektíva šprintu č. 5	18
4.12	Šprint č. 6.....	18
4.13	Retrospektíva šprintu č. 6	19
4.14	Šprint č. 7.....	19
4.15	Retrospektíva šprintu č. 7	19
4.16	Šprint č. 8.....	19
4.17	Retrospektíva šprintu č. 8	20
5	Globálna retrospektíva za zimný semester.....	21
6	Big Picture.....	22

6.1	Motivácia pre vizualizáciu dát v podobe grafov	22
6.2	Ciele projektu 3DSoftviz	22
7	Prílohy	24
7.1	Export úloh TFS	24
7.1.1	Šprint č.1	24
7.1.2	Šprint č.2	24
7.1.3	Šprint č.3	25
7.1.4	Šprint č. 4.....	25
7.1.5	Šprint č. 5.....	26
7.1.6	Šprint č. 6.....	26
7.1.7	Šprint č. 7.....	27
7.1.8	Šprint č. 8.....	27
7.2	Zápisnice zo stretnutí.....	27
7.2.1	Zápisnica zo stretnutia č. 1.....	27
7.2.2	Zápisnica zo stretnutia č. 2.....	29
7.2.3	Zápisnica zo stretnutia č. 3.....	30
7.2.4	Zápisnica zo stretnutia č. 4.....	31
7.2.5	Zápisnica zo stretnutia č. 5.....	32
7.2.6	Zápisnica zo stretnutia č. 6.....	33
7.2.7	Zápisnica zo stretnutia č. 7	34
7.2.8	Zápisnica zo stretnutia č. 8.....	35
7.2.9	Zápisnica zo stretnutia č. 9.....	37
7.2.10	Zápisnica zo stretnutia č. 10.....	38
7.2.11	Zápisnica zo stretnutia č. 11.....	39
7.2.12	Zápisnica zo stretnutia č. 12.....	40
7.2.13	Zápisnica zo stretnutia č. 13	41
7.2.14	Zápisnica zo stretnutia č. 14.....	41
7.2.15	Zápisnica zo stretnutia č. 15	42
7.2.16	Zápisnica zo stretnutia č. 16.....	42
7.2.17	Zápisnica zo stretnutia č. 17	43
7.2.18	Zápisnica zo stretnutia č. 18.....	43

7.3	Motivačný dokument	44
7.4	Metodiky	46
7.4.1	Gitflow metodika	46
7.4.2	TFS metodika	48

1 Úvod

Predkladaná dokumentácia popisuje prácu na tímovom projekte s názvom Vizualizácia informácií v rozšírenej realite. Práca je náplňou predmetu Tímový projekt, ktorého hlavnou náplňou je naučiť absolventov pracovať v tíme na spoločnom projekte. Okrem programátorských zručností si teda máme na predmete osvojiť aj prácu s kolaboratívnymi nástrojmi, spôsoby deľby práce, manažment úloh či odhad ich náročnosti.

Vedúcim tímu je Ing. Peter Kapec PhD., ktorý je zároveň aj tvorcom a mentorom všetkých študentov pracujúcich na projekte 3DSoftviz. Na projekte sa snažíme pracovať podľa pravidiel agilnej metodiky Scrum. Vedúci tímu plní v rámci predmetu rolu product ownera a jeho hlavnou úlohou je sprostredkovať vývojom svoje požiadavky na funkcionality produktu. Dĺžka jedného šprintu je dva týždne.

Úlohy, ktoré sme si rozdelili v kontexte riadenia projektu či jeho vývoja, sú popísané v nasledujúcej kapitole. Dokument tiež obsahuje sumarizáciu jednotlivých šprintov, metodiky používané pri práci na projekte, aplikácie manažmentov v rámci tímovej práce a zápisnice zo spoločných stretnutí tímu.

Kapitola	Autor
Úvod, manažment dokumentácie, manažment verziovacieho systému a prehliadok kódu, zápisnice zo stretnutí, Big picture, Globálna retrospektíva za zimný semester	Peter Marušin
Členovia tímu, manažment komunikácie, nástroj na evidenciu úloh, export šprintov, formát, Sumarizácia šprintov (č. 1, 2, 3)	Marek Škriečka
Globálna retrospektíva za zimný semester	Bence Ligárt
Sumarizácia šprintov (č. 4, 5)	Martin Gašpar

2 Členovia tímu

V tejto kapitole sú popísané jednotlivé manažérské a vývojárske úlohy členov tímu. Na jednotlivých úlohách každého člena sme sa dohodli na 1. stretnutí a až po koniec 3. sprintu sme ich nemenili. Niektoré úlohy však počas semestra a práce na projekte pribudli.

Bc. Marek Škriečka

Marek bol prvé tri šprinty Scrum Master, staral sa o plnenie úloh, manažoval úlohy v TFS a facilitoval stretnutia. Vytvoril a spravuje webovú prezentáciu tímu. Z pohľadu vývoja produktu sa zaoberal spojazdnením projektu na platforme Windows Subsystem for Linux.

Bc. Peter Marušin

Poť je zodpovedný za dokumentáciu a tiež spravuje git repozitár projektu. Písal zápisnice počas stretnutí tímu. Udržiaval Slack, vytváral či revidoval dokumentáciu a snažil sa oboznámiť nielen s dokumentačnými nástrojmi využívanými v rámci projektu. Na projekte tiež plní úlohu macOS integrátora a spávca git repozitára, dohliada na správnosť commitov a údržbu branches.

Bc. Bence Ligárt

Bence je windows integrátor a tester. Z tímu len dvaja členovia: Bence a Marek pracujú na operačnom systéme windows, a keďže Marek sa rozhodol pracovať so systémom WSL, s ktorým staršie tímy sa doteraz nezaoberali, jedine Bence sa hodil na túto pozíciu. Úloha windows integrátora zahŕňa: testovanie kompatibility implementovaných častí na operačnom systéme windows, podpora ovládačov.

Bc. Martin Gašpar

Martin sa spolupodieľa na vývoji a testovaní pre platformu macOS. Pomáha pri vytváraní a manažovaní úloh cez nástroj TFS. Po Marekovi prevzal pozíciu Scrum Mastra na posledné dva šprinty.

Bc. Miloslav Slížik

Milo sa spolupodieľa na vývoji a testovaní pre platformu Linux a spravuje tímový server.

Bc. Tomáš Krupa

Tomáš sa podieľa na refaktORIZÁCII a automatizácii zostavovania zdrojového kódu. Primárnym zameraním je C++ a CMake na platforme Linux.

Bc. Michal Knapík

Michal je OS X integrátor, podieľa sa na správe projektu z pohľadu zostavovacieho nástroja cmake a na zjednodušení jeho používania vrámci projektu.

2.1 Manažérské úlohy

V nasledujúcej tabuľke sú uvedené jednotlivé manažérské úlohy s uvedením členov, ktorí ich zastávali:

Úloha	Člen
Web vývojár	Marek Škriečka, Martin Gašpar
Server administrátor	Miloslav Slížik
Scrum Master	Marek Škriečka (štart č. 1, 2, 3), Martin Gašpar (štart č. 4, 5)
Dokumentácia	Peter Marušin
Git Master	Peter Marušin
Slack administrátor	Peter Marušin

2.2 Vývojárske úlohy

V nasledujúcej tabuľke sú uvedené jednotlivé vývojárske úlohy s uvedením členov, ktorí ich zastávali:

Úloha	Člen/Členovia

MacOs integrátor	Peter Marušin, Martin Gašpar, Michal Knapík
Windows integrátor	Bence Ligárt
Windows Subsystem For Linux integrátor	Marek Škriečka
Linux integrátor	Tomáš Krupa, Miloslav Slížik

2.3 Podiel práce na častiach inžinierskeho diela

V tabuľke je uvedený podiel členov tímu na inžinierskom diele v zimnom semestri:

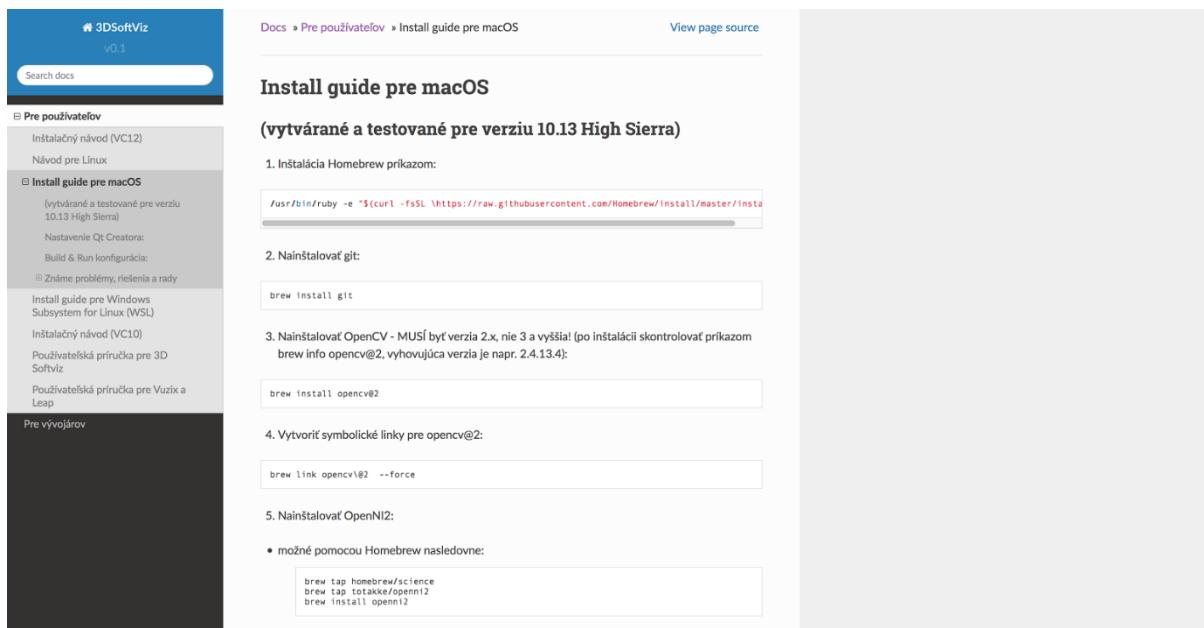
Časť diela	Autor/Autori
Build projektu na platforme WSL	Marek Škriečka
Webová prezentácia	Marek Škriečka
Aktualizácia použ. príručky	Peter Marušin, Marek Škriečka
Build projektu na platforme macOS	Martin Gašpar, Michal Knapík, Peter Marušin
Build projektu na platforme Linux	Tomáš Krupa, Miloslav Slížik
Revízia dokumentácie	Peter Marušin
Správa tímového servera	Miloslav Slížik
Vagrant	Miloslav Slížik
RefaktORIZÁCIA CMake	Michal Knapík, Tomáš Krupa
Build projektu na platforme Windows	Bence Ligárt

3 Aplikácie manažmentov

3.1 Manažment dokumentácie

Súčasťou práce na projekte bolo aj vytváranie, resp. údržba už existujúcej dokumentácie. Kedže sa jedná o roky existujúci projekt, snažili sme sa pokračovať v zabehnutých metódach dokumentovania, či už sa jedná o dokumentáciu pre vývojárov alebo používateľov 3DSoftvizu.

Používateľský manuál, metodiky, inštalačné manuály a ďalšia ručne písaná dokumentácia sú vytvárané pomocou nástroja Sphinx. Jedná sa nástroj napísaný v Pythone používaný Python komunitou, ktorý dokáže z RST dokumentov vygenerovať dokumentáciu napríklad vo formátoch HTML, PDF či LateX.



Obrázok 1: Ukážka vygenerovaného dokumentu pomocou nástroja Sphinx vo formáte HTML

Ďalším nástrojom používaným na generovanie UML diagramov je PlantUML. Ten vytvára z textových súborov so zdefinovanými entitami a vzťahmi medzi nimi vo vlastnom application-specific jazyku plnohodnotné UML diagramy vo formátoch .svg či .png. Ku generovaniu UML diagramov v projekte dochádza v rámci generovania Sphinx dokumentácie.

Ďalším nástrojom používaným v projekte je Doxygen, ktorý slúži na automatické generovanie dokumentácie priamo zo zdrojového kódu. Ten plánujeme viac používať v letnom semestri, keďže v zimnom sme sa venovali skôr analýze ako tvoreniu novej funkcionality.

Na začiatku semestra sme sa snažili analyzovať všetku existujúcu dokumentáciu. Mnoho materiálov si vyžadovalo, prípadne stále vyžaduje aktualizáciu a revíziu, keďže na projekte sa neustále pracovalo a pribúdala nová funkcia.

Na začiatku semestra sme sa rozhodli pre vytvorenie nových inštalačných manuálov osobitne pre každú platformu, keďže podľa už vytvorených sa nám často nepodarilo nakonfigurovať vývojárske prostredie. Nové inštalačné manuály sú súčasťou dokumentácie k produktu.

Nástroj PlantUML bol pridaný ako závislosť 3DSoftvizu a je distribuovaný ako súčasť repozitára (nie je potrebné jeho dodatočné získavanie). Po prejdení prác venovaných rozšíreniu 3DSoftvizu bol aktualizovaný používateľský manuál. Dokumentácia je a počas zvyšku tímového projektu plánuje byť priebežne revidovaná a aktualizovaná.

3.2 Manažment komunikácie a plánovania úloh

Najväčšia časť komunikácie prebieha cez stretnutia, ktoré mávame každý štvrtok od 10:00 do 13:00. Priebeh stretnutí facilituje Scrum Master.

V úvode je stretnutia je stand up, kedy každý člen tímu povie, čo od predchádzajúceho stretnutia spravil a s čím mal problém.

Ak je stretnutie v čase dokončenia šprintu na stretnutí sa ešte prezentuje retrospektíva, pričom každý člen tímu má možnosť vyjadriť sa k jej jednotlivým bodom.

Stretnutie pokračuje buď skupinovou alebo individuálnou konzultáciou problémov s vedúcim tímu, na ktoré každý člen pri plnení úloh narazil.

Ak stretnutie prebieha v čase, kedy sa začína nový šprint, v tíme diskutujeme a podľa priorít vyberáme úlohy z nástroja na manažment úloh, ohodnocujeme ich náročnosť a pridelujeme ich členom.

Retrospektívnu vytvárame deň pred stretnutím ukončenia šprintu, kedy sa stretneme. Diskutujeme, čo sme spravili za daný šprint dobre, v čom určite budeme pokračovať a naopak, kde sme spravili chybu a čoho sa v ďalšom šprinte vyvarovať. Výstup z retrospektívy má formu bodov 3 bodov: *start doing, stop doing, continue doing*, ktoré sú potom na stretnutí prezentované.

V tíme je potrebné komunikovať aj online. Na to používame nástroj Slack. Tento nástroj umožňuje komunikovať s členmi tímu buď individuálne alebo formou vlákien, ktoré sú vytvorené pre určitý okruh tém diskutovania. V tíme sú najviac využívané nasledujúce kanály:

- git - tu diskutujeme problémy pri verziovaní zdrojového kódu. V kanáli sú tiež integrované notifikácie z GitHub-u oznamujúce zmeny v repozitári projektu.
- build - na tomto kanáli komunikujeme chyby, ktoré sa vyskytli pri builde projektu
- tfs-scrum - tento kanál slúži na komunikáciu ohľadom nástroja tfs, v ktorom spravuje tok práce

3.3 Manažment verziovacieho systému a prehliadok kódu

Pri vývoji je používaný verziovací systém git a projekt je uložený vo vzdialenom repozitári na GitHub-e. Keďže sa jedná o projekt s už zabehnutými pravidlami vývoja z hľadiska vytvárania branches, označovania commitov či mergovania novovytvorenej funkcionality do spoločných branches, nás tím si osvojil už zaužívané praktiky s minimálnymi modifikáciami. Tie sú spísané v GitFlow metodike, ktorá je prílohou tohto dokumentu.

V projekte sú udržiavané dve hlavné branches:

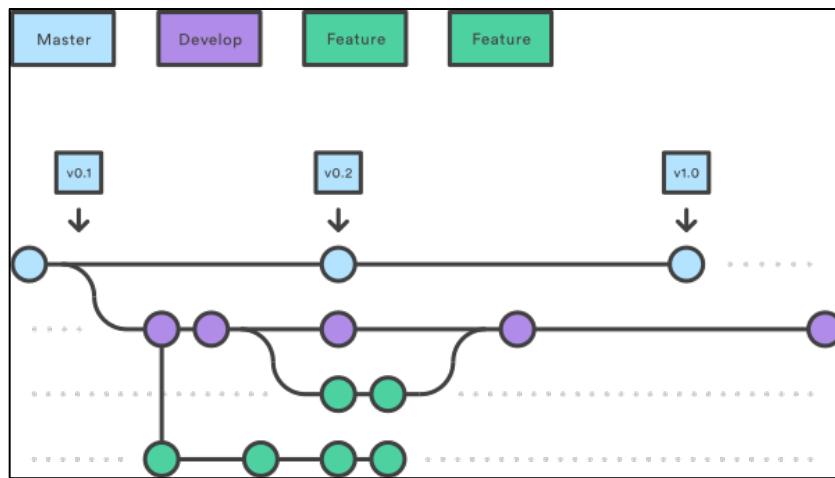
- **master** - hlavná branch obsahujúca otestovaný funkčný kód, v prípade release-u použiteľný. V čase nášho príchodu do projektu bol master už dlhšiu dobu neaktuálny pre množstvo úprav potrebných v existujúcom kóde na develop-e.
- **develop** - spoločná pracovná branch pre všetkých vývojárov. Do nej sa mergujú všetky zmeny vykonané vo vedľajších branches. Mergnutiu zmeny do develop branch musí ale predchádzať code review aspoň od jedného člena tímu (dôležité hlavne pri zmenách kódu či build procesu 3DSoftvizu). Zmeny sa mergujú prostredníctvom pull requestov. Ku prehliadkam kódu dochádzalo v zimnom semestri tiež formou pridelenia reviewers na daný pull request, čo je možnosť poskytovaná GitHub-om.

Okrem dvoch hlavných používame ešte vedľajšie branches, bližšie popísané v metodike:

- **feature** - nová funkcia
- **hotfix** - oprava chýb

Vedľajšie branches momentálne aj po mergnutí ostávajú v projekte, nemažú sa. Dôvodom je ľahšia možnosť návratu v prípade znefunkčnenia niektornej hlavnej branch a tiež ich malý počet (ich množstvo zatiaľ výraznejšie nestáruje orientáciu v projekte).

Minulé tímy vykonávali prehliadky kódu pomocou nástrojov Cppcheck, Cpplint a Astyle. My sme sa rozhodli pre pridelovanie reviewers na GitHub-e, keďže väčšina práce vykonanej v ZS sa netýkala priamo zásahov do kódu. No vyššie spomínané nástroje chceme určite používať v letnom semestri.



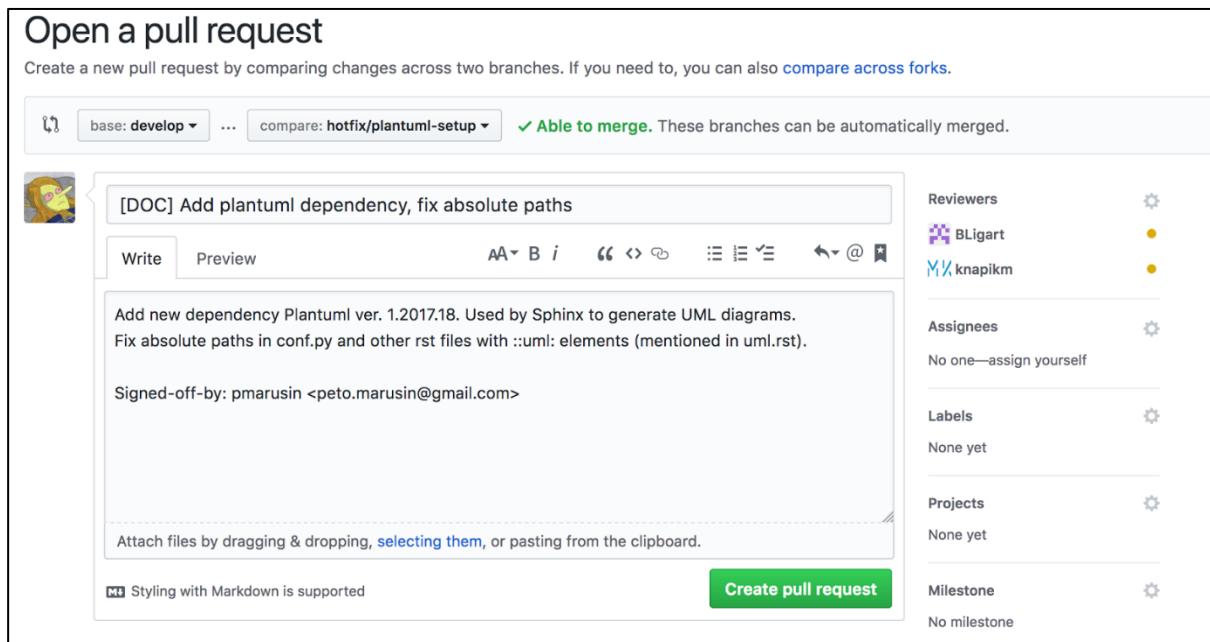
Obrázok 2: Ukážka vytvárania nových branch podľa Git metodiky¹

Workflow členov tímu pri práci bol nasledovný:

1. Je mi pridelený task z backlog-u, čo zistím v TFS
2. Pull-nem si z zo vzdialeného repozitára develop, čím sa uistím, že mám u seba aktuálnu vetvu.
3. Vytvorím si novú branch s názvom *hotfix/nazov_vetvy*, resp. *feature/nazov_vetvy* podľa toho, na čom idem v danej vetve pracovať.
4. Prepňem sa do danej vetvy
5. Vykonám zmeny, ktoré sú pripravené na commit.
6. Zmenené súbory pridám do staging area.
7. Commitnem zmeny, pričom pri písaní commit message sa pridržiam GitFlow metodiky - commit message je deskriptívna, stručná a obsahuje vhodný tag označujúci, čoho sa týka daný commit.
8. Zmeny pushnem do vedľajšej vetvy.
9. Vytvorím pull request požadujúci merge s develop-om. Pri vytváraní pull requestu mu zadám reviewers, ktorí musia mnou vykonné zmeny skontrolovať a otestovať ich funkčnosť. Snažím sa vytvárať osobitné pull requesty pre rôzne druhy úprav v projekte, napr. osobitný pull request pre modifikáciu dokumentácie, pre novú feature

¹ <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow>

čí hotfix. Ostatní členovia tímu sú notifikovaní na Slacku, prípadne mailom.



Obrázok 3: Vytváranie žiadosti o pull request s pridaními kontrolórmi

10. Ak je pull request schválený všetkými reviewers, môže byť vykonaný merge s develop-om. Ak nie, musím commitnúť zmeny, ktoré opravia prípadné nedostatky. Merge pritom môžem vykonať sám alebo ktokoľvek ďalší, kto zbadá, že pull request prešiel prehliadkou. Ostatní členovia tímu sú notifikovaní na Slacku, prípadne mailom.
11. V TFS označím task za splnený.

3.4 Nástroj na manažment úloh v tíme

Pre manažment úloh v tíme sme používali Microsoft Team Foundation Server (TFS). Tento nástroj umožňuje nastaviť rôzne šablóny pre spravovanie toku práce. Náš tím si vybral šablónu Scrum. V tejto šablóne je možné spravovať úlohy a vývoj celého produktu v nasledujúcej hierarchii:

- **EPICS** - táto časť združuje informácie o komplexnej funkciaľite časti produktu, ktorá je pre lepšiu organizovanosť rozdelená na niekoľko častí nazývaných FEATURES
- **FEATURES** - táto časť umožňuje zjednotiť viacero podobných BACKLOG ITEMS
- **BACKLOG ITEMS** - pozostávajú z úloh, ktoré sú podobné a súvisia s podobnou funkciaľitou. Na tejto úrovni tiež určujeme náročnosť kladenú na úspešné ukončenie všetkých podúloh, ktoré s daným BACKLOG ITEM-om súvisia.
- **TASKS** - na tejto úrovni sa nachádzajú jednotlivé úlohy, ktoré si určujeme pre úspešné beh celého vývoja produktu. Každá úloha má svojho autora, ktorý si ju spravuje a

nastavuje jej proces riešenia. Príklad hierarchie sa nachádza na nasledujúcom obrázku:

3	Epic	✓ 🏆 Support all platforms	● In Progress	Business
	Feature	✓ 🏆 Provide development environment on all supported platf...	● In Progress	Business
	Product Backl...	> [LINUX] Project setup	● Done	20
	Product Backl...	> [WINDOWS] Project setup	● Committed	20
+	Product Backl...	> [WSL] Project setup	... ● Done	20
	Task	☑ Installation of WSL	● Done	
	Task	☑ Installation of dependecies 3DSoftviz and IDE	... ● Done	
	Task	☑ Build and run project	● Done	
	Task	☑ Install guide	● Done	
	Product Backl...	> [macOS] Project setup	● Done	20
				Business

Obrázok 4: Ukážka hierarchie úloh v nástroji TFS

4 Sumarizácia šprintov

4.1 Úvodné stretnutie

Na úvodnom stretnutí sme diskutovali rozdelenie manažérskych úloh v tíme, voľby nástrojov pre manažovanie toku práce a tiež webovú prezentáciu tímu. Vytvorili sme nasledujúce úlohy – správa git reposítára, správa servera, osoba zodpovedná za dokumentáciu a reporty a správca webovej stránky.

Na komunikácii sme sa zhodli na mailing liste a Slack a ako manažment pracovného toku sme zvolili Microsoft Team Foundation Server.

4.2 Šprint č.1

V šprinte č. 1 sme si stanovili za prioritný cieľ build projektu a jeho spustenie na s najnovšími knižnicami a dostupnými závislostami na platorme Windows, MacOs a Unix. Medzi úlohami tiež figurovalo vytvorenie webovej prezentácie tímu a nastavenie prideleného servera.

Rozbehanie projektu sa na niektorých platformách nepodarilo alebo podarilo len so starými knižnicami a závislosťami. Príčinou bolo nesprávne ohodnotenie úsilia potrebného pre dané úlohy spustenia projektu. V ďalšom šprinte č. 2 sme si preto tieto úlohy rozčlenili na menšie podúlohy typické pre každý operačný systém a prísnejšie ohodnotili úsilia rozbehnutia projektu na jednotlivých platformách. V šprinte č. 1 sa nám však podarilo vytvoriť webovú prezentáciu tímu.

4.3 Retrospektíva šprintu č. 1

Medzi diskutované vylepšenia, ktoré by sme mali začať dodržovať patrilo väčšia viditeľnosť procesov – veľa členom tímu unikali informácie, keďže sa šírili len individuálne a neboli nikde publikované. S tým súvisí aj ďalšie vylepšenie a to komunikácia o problémoch s projektom výhradne vo verejných vláknach, ktoré Slack poskytuje alebo cez mailing list. Uľahčenie manažmentu práce tiež môže vylepšiť správne značenie úloh v TFS podľa metodiky.

Medzi praktiky, ktoré by sme mali prestať robiť patrilo súkromná komunikácia o problémoch v projekte a celkovo tímu. Medzi zlé praktiky patrilo aj spomínané zle ohodnotené úlohy voči úsiliu a nedostatočne rozčlenenie na podúlohy.

V retrospektíve sme tiež usúdili, že by sme mali pokračovať v kladení otázok bud' vedúcemu projektu alebo aj jednotlivý členovia tímu, pretože nám to výrazne pomáha.

4.4 Šprint č. 2

V šprinte č. 2 bolo hlavným cieľom dokončiť prepracované úlohy zo šprintu č. 1, ktoré sa týkali spustenia projektu na každej platforme s dohodnutými knižnicami. Výstup bolo treba prezentovať formou aktualizovaného manuálu na inštaláciu projektu. Úlohou tiež bolo spraviť export a oprava chýb, ku ktorým sme počas spúšťaniu projektu prišli, tak aby ich bolo možné zlúčiť so zdrojovým kódom projektu. Samostatnou úlohu tiež bolo prejsť minuloročné diplomové a bakalárské práce, výstupy tímového projektu a aktualizovať podľa používateľského príručku.

Rozbehanie projektu sa úspešne podarilo na všetkých platformách okrem operačného systému Winodws, kde pretrvávali rôzne chyby. Všetky inštalačné príručky boli aktualizované a boli opravené chyby pri builde projektu. Aktualizovaná bola tiež používateľská príručka.

4.5 Retrospektíva šprintu č.2

Medzi praktiky, ktoré by sme mali zaviesť bola diskutovaná téme ohľadom práce na úlohách s pridanou hodnotou pre projekt – refaktORIZÁCIA, optimalizácia alebo úlohy, ktoré by pridávali novú funkcionality systému. Potrebné je tiež dávať si pozor pri písaní zdrojového kódu na white-spaces. Diskutované tiež bolo lepšia responzivita na pull requesty a potvrdzovanie funkčnosti zmien.

Medzi dobré praktiky, v ktorých chceme pokračovať patrí spolupráca – riešiť problém vo dvojiciach a lepšie značenie toku práce v TFS.

4.6 Šprint č. 3

V 3. šprinte bolo treba nutne dokončiť Windows build, keďže neboli v 2. šprinte úspešné. Pri súčasnej konfigurácii je build vykonaný úspešne avšak aplikácia padne pri spustení. V novom šprinte je potrebné spustiť projekt s aktualizovanými verziami debugger-a a 64-bitovými verziami závislostí. V šprinte bola tiež naplánovaná analýza knižnice OpenPose, ktorá sa zaoberá detekciou kľúčových bodov ľudského tela, refaktORIZÁCIA súboru Cmake, ktorý je používaný pri build-ovaní projektu a vytvorenie prostredia pre CI. Významnou úlohou v 3. šprinte bolo tiež vytvorenie dokumentácie ku kontrolnému bodu č.1.

Rozbehanie na platforme Windows sa nepodarilo, zistilo sa, že s novým debuggerom pravdepodobne nebude build možný a tak je treba skúsiť build s pôvodným na inej stanicí. Problém bol tiež s úlohami ohľadom vytvorenia prostredia pre CI, dôvodom bol nedostatok času. Ostatné úlohy sa podarilo splniť.

4.7 Retrospektíva šprintu č. 3

V šprinte č.3 boli zistené nedostatky ohľadom plnenie malých úloh , ktoré trvajú max. 2 minúty (napr. doplniť si niečo alebo zmeniť v TFS). Členovia si ich často nechávajú na neskôr a potom sa na ne zabúda. Vylepšenie bolo navrhnuté tak, že ak niekto dostane takýto typ úlohy musí ju bezodkladne vyriešiť. Chyba bola taktiež v komunikácii, bolo navrhnuté aby niektorí členovia dávali na správy dávali vždy odozvu. Navrhnuté bolo tiež začať pracovať na novej funkcialete projektu a menovať branch-e podľa čísla úloh z TFS. Na týchto zlepšeniach sme sa zhodli.

Medzi zlé praktiky patrilo miešanie slovenského a anglického jazyka v commit správach a tiež, že niektorí členovia plnili úlohy za iných, bez ich súhlasu alebo oznamenia.

Praktiky, ktoré sa nám overili bola git-flow metodika vo forme vetiev, ktorú sme začali v 3. šprinte naplno využívať.

4.8 Šprint č. 4

V tomto šprinte bol hlavným cieľom dokončiť build pre Windows, ktorý sa pre rôzne problémy predlžoval až doteraz. Hlavný problém bol s knižnicami Leap a Leap-Orion, keďže doteraz sme používali 32 bitové verzie a s Visual Studio 2017 sme prešli na 64 bitové komponenty. Aktualizácia týchto knižníc vyriešila náš najväčší problém s buildom pre Windows a ostali už len drobné bugy.

Medzi ďalšie naplánované úlohy patrilo pokračovanie v refaktORIZACII CMake súborov využívaných pri builde, odstraňovanie varovných hlášok, ktoré sa zobrazovali pri builde a tvorba big picture dokumentácie k projektu.

Hlavný cieľ sa nám nakoniec podarilo splniť a projekt pre platformu Windows sa nám podarilo zfunkčniť pomocou viacerých fixov v kóde, ktoré boli otestované aj na ostatných platformách.

4.9 Retrospektíva šprintu č. 4

Nedostatky zistené v šprinte 4 súviseli predovšetkým s navýšením počtu commitov a teda aj pull requestov čim sa zvýšilo množstvo potrebných code reviews, ktoré sa predlžovali bez informovania tvorca o dôvode tohto dlhého čakania, čo v určitom smere blokovalo ďalší vývoj. Boli navrhnuté riešenia v zmysle kontaktovania reviewera tvorcom v prípade dlhej odozvy a tiež sme navzájom dohodli, aby každý, kto bude pridelení na review upozornil tvorcu, že je v časovej tiesni a aby ten vybral niekoho iného.

Tiež sme vyhodnocovali možnosť pridelenia ďalšieho člena tímu na platformu Windows, aby sme znížili riziko veľkého časového sklzu pri vývoji na tejto platforme do budúcnia. Ako

riešenie sme po konzultácii rozhodli, že platforma Windows bude slúžiť len na testovanie a vývoj bude prebiehať na zvyšných dvoch platformách.

4.10 Šprint č. 5

V šprinte číslo 5 sme sa viac zamerali na refactoring kódu a z toho aj vyplynuli úlohy ako napr. odstraňovanie zbytočných includes, čím docielime rýchlejší build a zároveň to predstavuje úplne prvý krok v rámci snahy o modularizáciu celého systému. Ďalšou úlohou bolo doplnenie chýbajúcich a prepracovanie existujúcich logov, aby spĺňali základnú metodiku a obsahovali všetky potrebné informácie. Ďalšou úlohou bolo vytvorenie nového namespace pre vlastnú implementáciu časti LeapLib, aby nebola súčasťou namespace knižnice, ktorý poskytuje tretia strana a nakoniec úloha zameraná na automatické generovanie dokumentácie a zbuildovanej aplikácie využitím continues integration. Tiež sme pokračovali v niektorých nedokončených úlohách z predchádzajúceho sprintu.

4.11 Retrospektíva šprintu č. 5

V retrospektíve po šprinte 5 sa objavil dve pripomienky, z toho jedna veľmi podstatná a to je mazanie správ na Slacku. Niekedy niekoľko niečo napísal, následne si to ďalší člen prečítal, ale neskôr autor správu miesto aktualizácie zmazal, čím vznikali nejasnosti, nedorozumenia a pod. Ako riešenie sme navrhli zákaz mazania správ na slacku. Ak obsah správy nie je aktuálny je lepšie danú správu aktualizovať, poprípade napísať novú správu s doplňujúcim obsahom k tej predošej.

Druhým návrhom bolo začatie písania činností, ktoré jednotliví členovia vykonávajú predovšetkým v rámci písania dokumentu. Hoci využívame zdieľanie dokumentu, veľa z nás si to najskôr pripraví lokálne a až následne dáva do verejného dokumentu. Preto sa občas stane, že sa niektoré veci spravia dvakrát.

4.12 Šprint č. 6

V tomto šprinte sme pokračovali v refaktoringu. Ukázalo sa, že s IWYU sú väčšie problémy na platforme macOS. Pracovať na nej sa ale dá ďalej, len to ide pomalšie. RefaktORIZÁCIA ohľadom opravy warningov prebiehalo úspešne. Ďalej sa pracuje aj na prerábaní cmake. Boli zistené problémy s formátovaním kódu z dôvodu rôznych nastavení v qt creatora a nefungujúceho astyle na platforme windows. Úspešne sa podarilo dokončiť premiestnenie globálnych premenných a metód, ktoré neboli v namespacoch do namespacoch. V tejto šprinte sa urobila aj migrácia projektu z githubu na gitlab. Bolo treba vyriešiť niektoré problémy po migrácii. Tím už ďalej bude pracovať na gitlabe.

4.13 Retrospektíva šprintu č. 6

Témou tejto retrospektívy bola hlavne práca projekte na gitlabe. Keďže v tomto šprinte sa vykonalá migrácia projektu na gitlab, členovia mali rôzne menšie problémy spojené s touto platformou, ktoré sme museli vyriešiť. Tieto problémy sme mali len z dôvodu toho, že bola pre nás nová táto platforma. V budúcnosti už chceme využívať len túto platformu na zdieľanie projektu, okrem submodulov, ktoré sú stále na githubu.

Zaobrali sme sa aj s IWYU, s ktorým bolo viac problémov. Dohodli sme sa ale, že práca na ňom sa bude pokračovať.

4.14 Šprint č. 7

Pokračujúca refaktORIZÁCIA a dalšie úlohy z nej vyplývajúce boli náplňou šprintu 7.

Vyťahovanie špagiet so sebou prinieslo nutnosť zvýšenej komunikácie s vedúcim (ale aj medzi sebou), aby sme dosiahli nie len zamýšľaný, ale skutočne požadovaný stav. Relatívne dlhá dĺžka života daného projektu priniesla úlohy v podobe nahradenia resp. aktualizácie niektorých knižníc a prepracovanie časti kódu tak, aby boli kompatibilné v kombinácii s novými knižnicami. Pri testovaní sme zistili nefunkčnosť niektorých častí, čo doplnilo úlohy o nové. Nezastavila sa ani práca sa CI, kde boli nové úlohy zamerané na odstránenie nedostatkov resp. nesprávne pochopeného cieľa.

4.15 Retrospektíva šprintu č. 7

Pri retrospektíve šprintu číslo 7 sme identifikovali ako hlavný problém slabú komunikáciu. To spôsobovalo predĺžovanie riešenia úloh v dôsledku ich neustáleho dopracovávania resp. prerábania a tým odkladanie ďalších čakajúcich úloh. Ako riešenie sme navrhli, že každá nejasnosť bude prediskutovaná skôr než bude riešená.

Tiež zavedením nových povinností v rámci gitlabu bolo treba určiť zodpovedné osoby, aby nenastal stav, kedy sa každý spolieha na toho druhého. Aj napriek nevyhovujúcemu času, boli nové povinnosti určené ako úlohy git mastra, ak nesplnomocní niekoho iného.

Posledným nedostatkom bolo zistené zabúdanie na nástroj TFS, v dôsledku používania gitlab issues, čo viedlo k jeho neaktuálnosti a nahromadeniu informácií, ktoré v ňom nie sú.

Riešenie bolo navrhnuté vo forme synchronizovaného dopĺňania ako gitlab issues tak aj nástroja TFS.

4.16 Šprint č. 8

Šprint 8 priniesol nové tasky vyplývajúce z novo navrhnutej architektúry, ktorá je cieľovým bodom celej refaktORIZÁCIE. Jedna skupina týchto taskov je zameraná na refaktORIZÁCIU

modulu Data. Ich hlavným cieľom je odstrániť zbytočnú previazanosť tohto modulu na iné časti projektu. Ďalšou úlohou bude aj odstránenie nevyužitých častí modulu a vytvorenie hierarchickej štruktúry tried pre entitu Graph.

4.17 Retrospektíva šprintu č. 8

Ako nedostatky šprintu číslo 8 sme identifikovali veľkú zložitosť úloh a tým vznikajúce komplikácie pri riešení na prvý pohľad jednoduchých ale pomerne komplexných úloh. V spojení s previazaním s ďalšími taskami t. j. závislostí riešenia jedného od druhého sme sa dostali do časových problémov. Ako riešenie sme dohodli spôsob definovania úloh na takej úrovni, ktorej komplexnosť umožňuje okamžité a ľahko realizovateľné riešenie. Ďalším problém predstavovalo riešenie úloh zložitým spôsobom, ktorý ale oproti jednoduchšiemu spôsobu nepredstavoval žiadny prínos. Tento problém je však do veľkej miery závislý od človeka, ktorý na ňom pracuje a jeho spôsobe uvažovania. Riešenie sme preto nedefinovali.

5 Globálna retrospektíva za zimný semester

Na úplnom začiatku zimného semestra sme sa oboznamovali s projektom. Sprvu bolo potrebné sa oboznámiť s početnými závislosťami projektu a spojazdniť ho, aby bol možný paralelný vývoj na všetkých platformách. Prvý šprint sme mali problém splniť zadefinované úlohy: nepoznali sme projekt a potrebovali osvojiť so systémom TFS. Tiež v ňom bol problém určiť správnu granularitu úloh pre jednotlivých členov tímu.

Z prvého šprintu sme sa poučili a snažili sa lepšie odhadovať úsilie potrebné na jednotlivé úlohy (osvojili sme si planning poker).

Ďalšie šprinty sme zvládli o poznanie lepšie. Tím sa oboznámil s TFS, neskôr sme začali sme efektívnejšie komunikovať (verejné kanály) a lepšie definovať a rozbíjať backlog itemy na začiatkoch šprintov.

V neskorších šprintoch sa osvedčila práca na úlohách v pároch a chceme v nej pokračovať aj v letnom semestri.

Michal a Tomáš si zobrali na starosti build projektu, ktorý počas druhej polovice semestra refaktorovali s pomocou vedúceho tímu (product ownera). Peťo s Marekom pracovali na dokumentácii, tvorbe a prípadnom rozšírení metodík projektu. Milo bol sám vyčlenený na DevOps tasky, konkrétnie vytvorenie prostredia na vývoj vo Vagrante. Všetci sme zároveň boli integrátori a testeri na svojej platorme.

Počnúc 2 šprintom sme komunikovali výhradne prostredníctvom Slacku. Mailing list sa neosvedčil.

Od 4. šprintu sme začali pracovať na samostatnom forku 3DSoftvizu.

Na úplnom začiatku semestra sme s vedúcim tímu (product ownerom) diskutovali o globálnych cieľoch nášho snaženia. Sprvu sme sa zhodli na tom, že budeme pracovať na nových features 3DSoftvizu a rozšírime jeho funkciaalitu.

Neskôr sme začali prehodnocovať ciele našej práce na projekte, keďže v priebehu semestra sme na všetkých platformách zápasili s problémami so závislosťami, početnými bugmi a neošetrenými chybovými stavmi v už existujúcom kóde. Vývojové prostredia sú navyše často závislé na verziach závislostí, ktoré už nie sú v package manageroch dostupné.

Z dlhodobého hľadiska sa vývoj projektu ukázal pomerne náročný pri platforme Windows (s absenciou WSL). Na ňu bol dlhodobo vyčlenený len jeden integrátor a tester (Bence).

Podpora Windows-u ale je aj napriek tomu naďalej prioritou.

Naše snaženie sme teda pre vyššie spomenuté skutočnosti zamerali na refaktORIZÁCIU existujúceho kódu, čo bude s veľkou pravdepodobnosťou platiť až do konca tímového projektu.

6 Big Picture

V tejto kapitole opisujeme projekt 3DSoftviz, na ktorom sme sa podieľali v rámci tímového projektu a predstavuje náš produkt. Budeme sa snažiť opísať 3DSoftviz z hľadiska možného využitia, špecifikovať cieľového zákazníka a opísať zábery jeho tvorcov v čase vzniku projektu. Tiež sa budeme snažiť systém popísť z technického hľadiska - aktuálne poskytovaná funkcia, podpora viacerých periférnych zariadení či väčšie zmeny vykonané predošlými študentami v rámci svojich diplomových a bakalárských prác či tímových projektov.

6.1 Motivácia pre vizualizáciu dát v podobe grafov

Človek je v dnešnej dobe zahľtený čoraz väčším množstvom informácií a je pre neho obtiažne si dané informácie zapamätať. Z nedávnych poznatkov je zrejmé, že vizualizácia informácií nám pomáha si ich ľahšie zapamätať. Je to spôsobené zapojením oboch hemisfér mozgu narozen od zapojenia iba ľavej hemisféry pri spracovaní informácií vo forme čísel alebo písaného textu.

Ďalším dôležitým faktom je charakter dnešných dát. Tie bývajú často multidimenziomálne, pričom pod dimensiou chápeme počet atribútov, ktoré majú jednotlivé položky datasetu. Často nás zaujímajú iba niektoré dimenzie a tie ostatné vnímame ako šum, ktorý nám stáže vnímanie vlastností a vzťahov v datasete. Neprehľadnosť môže byť tiež umocnená veľkosťou analyzovaných dát. V takom prípade tiež výrazne pomôže vizualizácia dát vo forme grafu. Tá nám umožní vnímať vzťahy či vzory, ktoré predtým neboli viditeľné.

6.2 Ciele projektu 3DSoftviz

3DSoftviz je už niekoľko rokov vyvíjaný softvér na Fakulte informatiky a informačných technológií STU. Na začiatku bol vytvorený s úmyslom stať sa general-purpose nástrojom na vizualizáciu informácií pomocou grafových štruktúr. Neskôr sa zameral na vizualizáciu softvéru. Napriek neskoršiemu vymedzeniu sa je ho ale po pomerne malých modifikáciach možné prispôsobiť na vizualizáciu informácií aj z iných oblastí.

Aktuálne sa využíva na vizualizáciu statických aspektov softvéru vo forme orientovaných či neorientovačných grafov. Datasetsy s informáciami o uzloch a hranách dokáže načítať z XML súboru, databázy či git repozitára. Okrem iného je tiež možné pridávanie uzlov či hrán, viacero možností rozmiestnenia uzlov grafu, manipulácia s grafom pomocou myši, zhľukovanie uzlov či vizualizácia evolúcie softvéru - zmien softvéru v čase zachytených v trojrozmernom priestore.

Projekt tiež obsahuje moduly pre spoluprácu s nižšie uvedenými periférnymi zariadeniami:

- Kinect

- 3D myš 3DConnexion
- AR Okuliare Vuzix STAR 1200XL
- Okuliare Nvidia 3D Vision Pro
- Leap senzor

7 Prílohy

7.1 Export úloh TFS

7.1.1 Šprint č.1

Project: Just18 Server: tfs.fiit.stuba.sk\StudentsProjects Query: Just18 Team - Sprint 1 - Backlog_new List type: Tree						
ID	Work Item Type	Title 1	Title 2	State	Effort	Assigned To
6466	Product Backlog Item	Run and build project on all platforms		Done	10	
6437	Task		Run and build on macOS	Done		Bc. Peter Marusin
6359	Task		Run and build on UNIX	Done		Tomas Krupa
6491	Task		Run and build on windows	Done		Bc. Bence Ligart
6725	Task		Run and build on WSL	Done		Bc. Marek Skriecka
6671	Product Backlog Item	Team website		Done	3	
6660	Task		Creation of web presentation of tea Done			Bc. Marek Skriecka
6672	Task		Setup of server for web presentation Done			Bc. Miloslav Slizik
6727	Task		Add images of team members	Done		Bc. Martin Gaspar

Obrázok 5: Export úloh zo šprintu 1

7.1.2 Šprint č.2

Project: Just18 Server: tfs.fiit.stuba.sk\StudentsProjects Query: Just18 Team - Sprint 2 - Backlog List type: Tree						
ID	Work Item Type	Title 1	Title 2	State	Effort	Assigned To
6764	Product Backlog Item	Make export with all bugs		Done		
6765	Task		[WSL] Add errors	Done		Bc. Marek Skriecka
6766	Task		[macOS] Add errors to doc	Done		Bc. Peter Marusin
6767	Task		[LINUX] Add errors	Done		Bc. Miloslav Slizik
6763	Product Backlog Item	[LINUX] Project setup		Done	20	Bc. Miloslav Slizik
6775	Task		Installation of Dependencies and IDE	Done		Bc. Miloslav Slizik
6776	Task		Succesfully build and run project	Done		
6777	Task		Install guide	Done		
6762	Product Backlog Item	[WINDOWS] Project setup		Committed	20	Bc. Bence Ligart
6798	Task		Installation of Microsoft Visual Studio	Done		Bc. Bence Ligart
6799	Task		Installation of dependencies and IDE	Done		Bc. Bence Ligart
6741	Product Backlog Item	[WSL] Project setup		Done	20	
6742	Task		Installation of WSL	Done		Bc. Marek Skriecka
6743	Task		Installation of dependecies 3DSoftvi	Done		Bc. Marek Skriecka
6744	Task		Build and run project	Done		
6745	Task		Install guide	Done		Bc. Marek Skriecka
6746	Product Backlog Item	[macOS] Project setup		Done	20	
6747	Task		[macOS 10.13] Install dependencies	Done		Bc. Peter Marusin
6748	Task		[macOS 10.13] Set up build & run en	Done		Bc. Peter Marusin
6673	Product Backlog Item	Update user manual		Done	3	
6652	Task		Remake user guide - bachelor thesis	Done		Bc. Marek Skriecka
6674	Task		Remake user guide - diploma thesis	Done		Bc. Peter Marusin
6802	Bug		Build error: src/Mouse3d/LibMouse3d/Mac/Mou	Done		Bc. Peter Marusin

Obrázok 6: Export úloh zo šprintu 2

7.1.3 Šprint č.3

Project: Just18 Server: tfs.fiit.stuba.sk\StudentsProjects Query: Just18 Team - Sprint 3 - Backlog List type: Tree					
ID	Title 1	Title 2	State	Effort Iteration Path	Assigned To
6886	Preparation for CI		Committed	8 \Sprint 3	Bc. Miloslav Slizik
	Setup Vagrant machine with Ansible provisioning - Ubuntu 16.04		Done	\Sprint 3	Bc. Miloslav Slizik
6888	Create Docker container		In Progress	\Sprint 3	Bc. Miloslav Slizik
	Write documentation for Docker, Vagrant and Ansible		Done	\Sprint 3	Bc. Miloslav Slizik
6934	OpenPose library (analysis)		Done	20 \Sprint 3	
6933	OpenPose - report		Done	\Sprint 3	Bc. Martin Gaspar
6935	OpenPose - perform test		Removed	\Sprint 3	Bc. Bence Ligart
6932	Document for assignment n. 1 (AIS)		Done	5 \Sprint 3	Bc. Peter Marusin
6981	Product description		Done	\Sprint 3	Bc. Peter Marusin
7101	Management description		Done	\Sprint 3	Bc. Marek Skriecka
7102	Architecture description		Done	\Sprint 3	Bc. Bence Ligart
6762	[WINDOWS] Project setup		Committed	20 \Sprint 3	Bc. Bence Ligart
6798	Installation of Microsoft Visual Studio	Done		\Sprint 2	Bc. Bence Ligart
6799	Installation of dependencies and IDE	Done		\Sprint 2	Bc. Bence Ligart
6800	Build and run project		In Progress	\Sprint 3	Bc. Bence Ligart
6801	Install guide		In Progress	\Sprint 3	Bc. Bence Ligart
6666	Refactor CMakeLists.txt		Committed	34 \Sprint 3	
	Split main CMakeLists.txt (subdirectories)		Done	\Sprint 3	Bc. Michal Knapik
6667	Loops for repeated commands		Done	\Sprint 3	Bc. Peter Marusin
6913	Add options to CMake		Done	\Sprint 3	Tomas Krupa

Obrázok 7: Export úloh šprint 3

7.1.4 Šprint č. 4

Project: Just18 Server: tfs.fiit.stuba.sk\StudentsProjects Query: Just18 Team - Sprint 4 - Backlog List type: Tree							
ID	Work Item Type	Title 1	Title 2	State	Effort	Value Area	Assigned To
7203	Product Backlog Item	Big picture document		Done	5	Business	Bc. Marek Skriecka
7204	Task	Merge documentation from every te	Done				Bc. Marek Skriecka
7191	Product Backlog Item	Fixing build bugs/warnings		Done		Business	Tomas Krupa
7193	Task	cc1plus missing test_github_module	Done				Tomas Krupa
7143	Product Backlog Item	Split main CMakeLists.txt and big refactoring		Done		Business	Bc. Michal Knapik
7131	Task	Deploy CCACHE to Cmake	Done				Tomas Krupa
7195	Task	Put "Searching external dependencies	Done				Bc. Michal Knapik
7197	Task	Move "BDD Igloo" into tests	Done				Tomas Krupa
6762	Product Backlog Item	[WINDOWS] Project setup		Done	20	Business	Bc. Bence Ligart
6800	Task	Build and run project	Done				Bc. Bence Ligart
6801	Task	Install guide	Done				Bc. Bence Ligart

Obrázok 8: Export úloh šprint 4

7.1.5 Šprint č. 5

Project: Just18 Server: tfs.fiit.stuba.sk\StudentsProjects Query: Just18 Team - Sprint 5 - Backlog List type: Tree						
ID	Work Item Type	Title 1	Title 2	State	Effort	Assigned To
7364	Product Backlog Item	Fix globals in namespaces		Committed	1	Bc. Bence Ligart
7365	Task		Remove globals from namespaces	In Progress		Bc. Bence Ligart
7191	Product Backlog Item	Fixing build bugs/warnings		Committed		Tomas Krupa
			Doxygen unable to generate temporary template			
7192	Task			In Progress		
		CMake undefined behaviour warning				
7194	Task			In Progress		
7143	Product Backlog Item	Split main CMakeLists.txt and big refactoring		Committed		Bc. Michal Knapik
7132	Task		add git info to build	Done		Tomas Krupa
7196	Task		Separate "Install" procedures to and	In Progress		
7198	Task		Separate "Packing" to another cmake	In Progress		Tomas Krupa
7215	Task		Separate sources and headers assign	Done		Bc. Michal Knapik
7324	Product Backlog Item	Analyze and optimise #includes using IWYU tool		Committed	5	Bc. Peter Marusin
7333	Task		Fix bad placed or redundant include	In Progress		Bc. Peter Marusin
7317	Product Backlog Item	Add and update logs in project		Committed	2	Bc. Martin Gaspar
7318	Task		Change different types of logs to on	In Progress		Bc. Marek Skrieka
7319	Task		Reformulation all logs messages	In Progress		Bc. Martin Gaspar
6886	Product Backlog Item	Preparation for CI		Done	8	Bc. Miloslav Slizik
7311	Task		pass SSH key to vagrant and docker	Done		Bc. Miloslav Slizik
6676	Product Backlog Item	GitLab Runner - CI		Committed	3	Bc. Miloslav Slizik
6677	Task		study documentation of gitlab-runner	Done		Bc. Miloslav Slizik
6678	Task		automatize sphinx , doxygen, cppli	In Progress		Bc. Miloslav Slizik
7312	Task		use gitlab-runner to build docker co	In Progress		Bc. Miloslav Slizik
7141	Product Backlog Item	Warning handling		Committed	2	Bc. Martin Gaspar
7142	Task		Warning handle in include and src d	In Progress		Bc. Martin Gaspar
7363	Product Backlog Item	Separate Leap namespaces		Committed	2	Bc. Michal Knapik
7376	Task		Create new namespaces for Leap ut	In Progress		Bc. Michal Knapik

Obrázok 9: Export šprintu č.5

7.1.6 Šprint č. 6

Project: Just18 Server: tfs.fiit.stuba.sk\StudentsProjects Query: Just18 Team - Sprint 6 - Backlog List type: Tree						
ID	Work Item Type	Title 1	Title 2	State	Effort	Assigned To
8216	Product Backlog Item	Make Astyle build and fix Diluculum for WIN		Done	2	Bc. Marek Skrieka
7913	Task		Fix Diluculum Win build	Done		Bc. Bence Ligart
8217	Task		Make Astyle build in file	Done		
8218	Task		Update CMake	Done		
8219	Task		Make tutorial for Astyle build	Done		
7923	Product Backlog Item	Analyze and optimise #includes using IWYU tool		Done		Bc. Peter Marusin
7915	Task		Fix includes using IWYU	Done		
7920	Product Backlog Item	Upgrade Diluculum to sol2		Committed	2	Bc. Martin Gaspar
7921	Task		Analyse upgrade Diluculum to sol2	Done		Bc. Martin Gaspar
7917	Product Backlog Item	Submodule for luagraf a luainterface		Committed	2	
7918	Task		IWYU for luagraph and luainterface	Done		Bc. Martin Gaspar
7919	Task		Remove QT dependencies from luag	To Do		
7364	Product Backlog Item	Fix globals outside namespaces		Done	1	Bc. Bence Ligart
7365	Task		Move globals outside namespaces ir	Done		Bc. Bence Ligart
7317	Product Backlog Item	Add and update logs in project		Done	2	Bc. Martin Gaspar
7911	Task		Build GitLib and LeapLib as shared libraries	To Do		Tomas Krupa
7912	Task		Fix installers in Install.cmake	To Do		Tomas Krupa
7916	Task		CMake: move GitLib and LeapLib low in the hiera	Done		Bc. Michal Knapik
8223	Task		Add CI support for cppcheck,cpplint,doxygen,sph	Done		Bc. Miloslav Slizik

Obrázok 10: Export šprintu č.6

7.1.7 Šprint č. 7

Project: Just18 Server: tfs.fiit.stuba.sk\StudentsProjects Query: Just18 Team - Sprint 7 - Backlog List type: Tree						
ID	Work Item Type	Title 1	Title 2	State	Effort	Assigned To
8320	Product Backlog Item	CMake: move 3rd party sources into dependency	Done	Done	Bc. Michal Knapík	
8224	Product Backlog Item	Additional CI features	Done	Done	Bc. Miloslav Slížik	
8225	Task	add IWYU support for CI	Done	Done	Bc. Miloslav Slížik	
8226	Task	optimize CI image provisioning	Done	Done	Bc. Miloslav Slížik	
8216	Product Backlog Item	Make Astyle build and fix Diluculum for WIN	Done	2	Bc. Marek Skrieka	
8207	Product Backlog Item	Analyze and optimise #includes using IWYU tool	Done	Done	Bc. Peter Marusin	
8155	Product Backlog Item	Build GitLib and LeapLib as shared libraries	Committed	5	Tomas Krupa	
8157	Product Backlog Item	Folder Data as library	Committed	5		
8156	Product Backlog Item	Refactor minimize include of ApplicationConfig	Committed	2		
8148	Product Backlog Item	Refactor module Data	Committed	5	Bc. Bence Ligart	
8149	Task	[REF] Data::Graph	In Progress	In Progress	Bc. Bence Ligart	
8150	Task	[REF] Data::Type	In Progress	In Progress	Bc. Bence Ligart	
8151	Task	[Ref] Data::GraphSerializer	In Progress	In Progress	Bc. Bence Ligart	
8318	Task	[Ref] Remove using DAO classes from	In Progress	In Progress	Bc. Bence Ligart	
8153	Product Backlog Item	Update modules Aruco and OpenCV	Committed	2	Bc. Martin Gaspar	
8334	Task	[Fix] Fix errors makes by updated lib	Done	Done	Bc. Martin Gaspar	
8335	Task	[Ref] Test marker detection with new	Done	Done	Bc. Martin Gaspar	

Obrázok 11: Export šprintu č.7

7.1.8 Šprint č. 8

Project: Just18 Server: tfs.fiit.stuba.sk\StudentsProjects Query: Just18 Team - Sprint 8 - Backlog List type: Tree						
ID	Work Item Type	Title 1	Title 2	State	Effort	Assigned To
8232	Bug	Fix webcam Bug		Committed		Bc. Martin Gaspar
8319	Product Backlog Item	Split Importer::Parsing into separate	Done	Done		Bc. Michal Knapík

Obrázok 12: Export šprintu č.8

7.2 Zápisnice zo stretnutí

7.2.1 Zápisnica zo stretnutia č. 1

Téma stretnutia: Úvodné stretnutie, rozdelenie úloh v tíme, voľba nástrojov, webstránka, vytvorenie spoločnej fotografie na plagát tímu

Dátum: 28. 10. 2017

Čas: 10:00-13:00

Miestnosť: 4.46

Prítomní:

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Diskusia o vzhľade a obsahu webstránky tímu. Marek a Martin si vzali na starosti tvorbu webstránky. Marek si zapisoval pripomienky členov a pokyny zo stránky predmetu a vyžiadal si fotografie členov.
- V tíme sme si rozdelili nižšie vypísané roly. Na každú z nich sa dotyčný člen prihlásil sám. Správcu git repozitára sme zvolili, lebo predpokladáme, že pri vývoji je použitý git. Zistenie konkrétnych detailov bude náplňou ďalšieho stretnutia.
- správa git repozitára: Martin
- správa servera: Milo
- dokumentácia, reporty: Peťo
- správa webstránky: Marek
- Zhodli sme sa komunikácií prostredníctvom mailing listu, ako možnosť padol aj Slack. Je potrebné zistiť, prečo sa minulým tímom neosvedčil. Mailing list sme na stretnutí nevytvorili, konfiguráciu si zobraľ na starosti Mišo.
- Tvorbu tímového plagátu si zobraľ na starosti Bence. Počas stretnutia sme vytvorili spoločnú fotografiu tímu
- Zhodli sme sa na používaní systému systému TFS na SCRUM management. Oboznamovali sme sa spoločne so systémom.
- Reporty budú dostupné pre všetkých členov tímu na GDrive tímu a na jeho webe.

Úlohy:

Do ďalšieho stretnutia bude potrebné vyriešiť nižšie uvedené úlohy:

Č. úlohy	Popis	Zodpovedný
1	Oboznámenie sa s TFS	všetci
2	Konfigurácia mailing listu	Mišo
3	Tvorba tímového plagátu	Bence

4	Oboznámenie sa s hostingom	Milo
5	Poslať fotografiu a bio správcovi websránky	všetci
6	Začať s tvorbou webstránky	Marek, Martin

7.2.2 Zápisnica zo stretnutia č. 2

Téma stretnutia: Stretnutie s vedúcim tímu, definovanie cieľov projektu a motivácie tímu

Dátum: 5. 10. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Vedúci tímu nás oboznámil s prvotnými organizačnými pokynmi. Ako tím sme si stanovili ciele a odpovedali na 3 otázky:
 - a. Ako sme spokojní s projektom?
 - b. Čím by sme chceli my prispieť do projektu?
 - c. Čo očakávame, že sa ako tím naučíme?
- Migrácia na GitLab je v stave riešenia.
- Organizačné pokyny a rady ohľadom fungovania SCRUM-u a agilného vývoja. Prehodnotenie rozsahu dokumentácie. Dokumentácia by mala byť preusporiadaná a prečistená.

- Bude potrebné si definovať user stories/system stories (ohľadom podpory a rozbehnutia produktu na rôznych platformách).
- Projekt by mal byť funkčný na 3 podporovaných platformách (Windows, Linux, macOS).
- Možná system story - aktualizácia modulov a knižníc na najnovšie verzie (hlavne QT) a prispôsobenie inštalačnej príručky. Taktiež aktualizované OpenCV má iné API ako nami používané (možný task pre budúcnosť).
- Zvážiť Vagrant & Ansible a ich konfiguráciu ako task pre prvý sprint. Doxygen - automatické generovanie dokumentácie.
- Prioritou pre začiatok by malo byť vytvorenie backlog-u.
- Dĺžka sprintu 2 týždne, stand-up každý týždeň.
- Retrospektíva - čo nám nešlo, čo nám šlo a s čím by sme vôbec mali začínať.

Úlohy:

Do ďalšieho stretnutia bude potrebné vyriešiť nižšie uvedené úlohy:

Č. úlohy	Popis	Zodpovedný
1	Testovanie Linux subsystému (report)	Marek
2	Oboznámenie sa s Vagrantom (report)	Milo (a hocikto ďalší, kto má záujem a čas)
3	Oboznámienie sa s gitflow metodikou	všetci
4	Oboznámenie sa s Doxygen, Sphinx, štruktúrou dokumentácie (report)	Pető (a hocikto ďalší, kto má záujem a čas)

7.2.3 Zápisnica zo stretnutia č. 3

Téma stretnutia: Vzájomná diskusia o problémoch s buildom projektu, spoločná práca

Dátum: 12. 10. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Snažili sme sa vzájomne si pomôcť s builnutím projektu 3dsoftviz.
- Vedúci projektu nám radil ohľadom opráv v CMakeLists.txt
- Úspešný build sa počas stretnutia podaril viacerým (Milo a Peťo), no ako problém sa ukázal cotire, ktorý nie je pod stálym vývojom a neustále aktualizácie viacerých knižníc (aktualizácia knižnice Qt bola vydaná dve hodiny pred začiatkom stretnutia)
- Do budúceho stretnutia by bolo najlepšie, keby každý vedel projekt buildnúť a pustiť
- Pripravujeme sa na buducotýžňovú retrospektívnu
- Bol zriadený Slack na urýchlenie komunikácie medzi členmi tímu
- Tasky sa budú nahadzovať už iba do TFS

7.2.4 Zápisnica zo stretnutia č. 4

Téma stretnutia: Retrospektíva, stand-up, vytvorenie nového backlogu

Dátum: 19. 10. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Stretnutie začalo retrospektívou, scrum-master (Marek) menežoval priebeh retrospektívy.
- Marek splnil task zostavenia a spustenia projektu na na platforme Windows s Windows Subsystem for Linux. Projekt beží s QT5, testy fungujú.
- Peťo projekt zostavil a spustil na platfotme macOS, verzii 10.13 (High Sierra). Vizualizácie grafov však nemajú hrany.
- Milo zostavil predpripravené vývojové prostredie pre 3dsoftviz vo Vagrante, projekt beží s QT 4.
- Tomáš projekt rozbehal vo virtuálnej mašine na Linuxe.
- Hlavnou náplňou stretnutia bolo vytváranie backlogu, ktorý je obsahom priloženého textového súboru. Počas brainstromingu sme si v ňom zadefinovali množinu úloh, ktorých rozsah pravdepodobne vystačí na viacero budúcich sprintov.
- Backlog a rozpracované tasky je k priložený v samostatnom textovom súbore.
- Vyskúšali sme si planning poker, praktika sa osvedčila.

7.2.5 Zápisnica zo stretnutia č. 5

Téma stretnutia: Diskusia o migrácii projektu na GitLab, diskusia o SCRUM-e, prvé live zadávanie taskov do TFS

Dátum: 26. 10. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Hlavnou náplňou stretnutia po úvodnom stand-upe bola diskusia ohľadom správnej granularity taskov v TFS a celkovo o manažmente nášho projektu v rámci SCRUM-u.
- Zhodli sme sa na adaptáciu gitflow metodiky dostupnej v remote repozitári, pričom sme ju rozšírili o pravidlo v rámci commit messages používať angličtinu.
- Diskutovalo sa taktiež o bugoch nájdených počas uplynulého týždňa.
- Hlavný CMakeLists.txt bude optimálne “rozbit” do viacerých menších v subadresároch projektu, možný task na ďalší sprint. K tasku sa možno prihlási Michal
- Vyskúšali sme si vytvorenie backlog itemov priamo v TFS, pričom to chceme robiť aj nadalej už iba takto.
- Platforma Windows je stále problémová, Bencemu sa nepodarilo 3dsoftviz spustiť.

7.2.6 Zápisnica zo stretnutia č. 6

Téma stretnutia: Retrospektíva druhého šprintu, zadefinovanie epicov a iných náležitostí pre 1. odovzdanie, plánovanie 3. šprintu

Dátum: 2. 11. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Hlavnou náplňou stretnutia bola retrospektíva druhého šprintu a príprava na tretí.
- Radi by sme od tretieho šprintu pokračovali v manažmente TFS načrtnutom na minulom stretnutí.
- Nastavenie notifikácií na GitHub-e pre lepší prehľad o zmenách
- Správna granularita commitov a pull requesty pre navzájom súvisiace commity (osobitný pull request pre úpravy dokumentácie, osobitný pre fixy apod).
- Pokračuje analýza a úpravy dokumentácie projektu, aktualizovaný user manual.
- Zadefinovanie nového epicu (Interaktivna manipulacia s ulzami grafov v AR) a viacerých features s ním súvisiacich (priamo do TFS).
- Zadefinovanie zákazníka pre dokumentáciu k produktu. V odovzdanom dokumente treba rozobrať projekt podrobnejšie, spomenúť jeho chystanú migráciu a zavedenie CI/CD.
- súčasťou odovzdáneho dokumentu môže byť aj analýza knižnice OpenPose na real-time detekciu kľúčových bodov postavy (popis knižnice, aktuálny stav vývoja, prípadne či by bola možná jej integrácia do projektu). Report si vzal na starosti Martin.

7.2.7 Zápisnica zo stretnutia č. 7

Téma stretnutia: Rozvrhnutie práce na dokumente, zmena scrum-mastera, diskusia ohľadom migrácie

Dátum: 9. 11. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapc

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Začali sme stand-upom. So začiatkom 4. šprintu sme sa zhodli na zmene scrum-mastera.
- Martin spracováva analýzu OpenPose.
- Vzhľadom k zvolenému template-u systému TFS sme sa zhodli, že pravdepodobne nebude pre nás smerodajným ukazovateľom burn-down chart. Bolo by totiž potrebné odznova nakonfigurovať TFS.
- Snažili sme sa rozbehnúť nástroj PlantUML zavedený predchádzajúcim tímovým projektom, na generovanie UML diagramov v rámci generovania Sphinx dokumentácie. PlantUML bude pridaný ako závislosť projektu.
- Bence sa nadálej snažil projekt rozbehať na platforme Windows.
- Počas uplynulého týždňa došlo k vykonaniu tasku Peťom napriek tomu, že ho mal pridelený Mišo. K podobným omylem už nesmie dochádzať.
- Nadálej sa pracuje na dokumente k riadeniu a produktu na prvý kontrolný bod(Peťo a Marek)
- Tomáš s Michalom refaktorujú CMake.
- Všetci sme začali pracovať na Peťovom forku ako BergiSK, na Slack sa zavesila dočasná metodika na code review využívajúca reviews pull requestov na GitHub-e. V letom semestri chceme používať už aj Astyle a cppcheck.

7.2.8 Zápisnica zo stretnutia č. 8

Téma stretnutia: Retrospektíva tretieho šprintu, planning poker pre 4. šprint, diskusia o estimácii storypointov, GitLab runneri, snaha o rozbehanie Windowsu

Dátum: 16. 11. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Po úvodnom stand-upe sme začali retrospektívou 3. šprintu. Ukázalo sa, že nevieme ako tým efektívne komunikovať. Čas odozvy niektorých členov tímu je dlhší ako 24 hodín, čo brzdí dianie v tíme a rozhorčuje scrum-mastera.
- Mišo navrhol do označenia commitov pridávať aj číslo tasku z TFS. Tiež bude mierne aktualizovaná gitflow metodika
- Windows setup stále neúspešný. Pre budúcnosť projektu je však veľmi dôležitý, keďže na ňom budú prebiehať ďalšie bakalárskie a diplomové práce.
- Znova sa prizvukovalo, že situácia, keď Peťo urobil Michalov task, sa už nesmie zopakovať.
- Planning poker pre 4. šprint, tasky v TFS (tu spomíname iba niektoré):
 - pokračovanie v refaktore CMake-u (Mišo a Tomáš).
 - zapracovanie “big picture” opisov funkcionality 3DSoftvizu z bakalárskych a diplomových prác a zapracovanie do Sphinx dokumentácie
 - ošetrenie warningov (Martin)
 - GitLab CI Runner a Docker (Milo)
- Dobrý spôsob na uľahčenie odhadu zložitosti taskov pre budúcnosť je vytvorenie zdieľaného dokumentu, kde by sme si značili doteraz odhadnuté úlohy. Tie poslúžia ako odrazový mostík pri ďalšom odhadovaní.

7.2.9 Zápisnica zo stretnutia č. 9

Téma stretnutia: diskusia (hlavne spojazdenie Windowsu), priebežná práca.

Dátum: 23.11. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Tomášovi s Michalom sa veľmi darí pri refactoringu a optimalizácii CMake-u a buildu projektu.
- Peťo s Marekom ďalej pracujú na big-picture dokumente pre projekt, pokračujú s rešeršom všetkých relevantných dokumentov minulých tímových projektov a diplomových prác (za relevantné sú považované tie, kde sa pracovalo na funkcionálnych požiadavkach).
- Bence tiež uspel pri rozbehaní projektu na Windows platforme. Veľký úspech, keďže je na platfomu sám, je vo veľkej nevýhode v porovnaní s ostatnými členmi tímu. Po predvedení zmien a pushnutí do repa sa môžme všetci pomaly sústrediť na kód projektu.
- Vedúci predviedol niektoré nelogické umiestnenia premenných či volanie funkcionality z nesúvisiacich namespaceov. Bude potrebné identifikovať podobné nezmysly a oddeliť ich tak, aby boli naše moduly použiteľné aj v iných projektoch, pokial' možno. Na správnu modulárnu dekompozíciu mnohí, ktorí pracovali na projekte, nedbali a postupom času vznikli sektory, ktoré bude veľmi obtiažne refaktorovať.
- Martinovi sa darí pri odstraňovaní warningov v projekte, no po diskusii s vedúcim prišiel na to, že niektoré sú nevyhnutné a ich odstránenie je horšie riešenie.
- Ako task do budúcnosti sa náuka zistiť, prečo sa 3DSoftviz na macOS platforme púšťa so starým OpenGL. Problém sa ale týka iba jednej platformy.

7.2.10 Zápisnica zo stretnutia č. 10

Téma stretnutia: Retrospektíva 4. šprintu, planning poker pre piaty šprint, ukážky chýb v architektúre projektu

Dátum: 30. 11. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Venovali sme sa retrospektíve štvrtého šprintu a tiež planning pokeru pre piaty šprint. Nižšie je uvedený zoznam BI, spolu s ohodnoteným a členom tímu, ktorý si daný BI vzal/bol naň pridelený:
 - refaktoring NULL (C-like) na nullptr - Martin (1)
 - vyčlenenie globálnych premenných mimo namespace-ov - Bence (1)
 - Prečistenie namespace-u LeapLib, lepšie rozdelenie Leap-related častí
 - Nahradenie qDebug EasyLoggingom, tiež preformulovanie výpisov - Marek, Martin (2)
 - Include What You Use - oprava nepotrebncích alebo zle umiestnených includov pre urýchlenie komplilácie - Tomáš, Peťo (5)
 - pokračovanie na odstraňovanie warningov - Martin (1)
 - CI - Milo (2)

BI sú zaevidované v TFS

- Vedúci nám predviedol pomocou nástroja Sourcetrail chyby na úrovni kódu v jednotlivých moduloch či headeroch (nezmyselné includy, cyklické závislosti apod.)

- Vedúci taktiež predviedol nástroj Include What You Use na odhalenie nepotrebných či zle umiestnených include-och. Pre použitie nástroja je potrebné vypnúť cotire, avšak predpokladá sa, že na dlhšie sa použije len raz a pri ďalšom vývoji sa bude dbať na správnosť includovania. Pri projekte našeho rozsahu to môže výrazne urýchliť kompliaciu a zvýši udržateľnosť kódu. Prácu s nástrojom si vzali na starosti Peťo s Tomášom.
- Marek s Martinom si vzali na starost upratanie loggingu v projekte
- Bence sa bude snažiť skonsolidovať globálne premenné v projekte, keďže na veľa miestach je ich použitie zbytočné a nevyhovujúce.
- Po vzájomnej dohode sme sa zhodli, že na úlohách pre tento šprint a veľký rozsah niektorých z nich bude možné pracovať aj počas druhej polovice decembra, keďže momentálne finišujeme na ostatných predmetoch a niektorým už začínajú skúšky.

7.2.11 Zápisnica zo stretnutia č. 11

Téma stretnutia: Prezentácia v rámci predmetu MTS

Dátum: 7. 12. 2017

Čas: 10:00 - 13:00

Miestnosť: 4.46 (10:00-12:00) a 4.20 (12:00-13:00)

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Hlavnou náplňou dnešného stretnutia bolo stretnutie s Ing. Jakubom Šimkom, PhD., ktorý mal zhodnotiť riadenie nášho tímu v rámci zimného semestra. To ale bolo naplánované až o 12:00, dovtedy sme diskutovali o priebehu našich taskov v aktuálnom sprinte.
- Tomáš s Bencem počas stretnutia osobne prediskutovali Tomášov pull request, ktorý dostal na code review Tomáš. Nakoniec došlo k mergu do developu.

- Diskutovalo sa tiež o ďalších otvorených pull requestoch, ktoré čakali na posúdenie od product ownera. Problémy s pomalou odozvou pri code review sme dobehli na stretnutí veľmi dobre.
- Martinovi sa ďalej darí odstraňovať warningy v projekte.
- Peťo má problém s nástrojom Include What You Use (IWYU), pravdepodobne vznikol na strane vývojárov nástroja, keďže ten nie je kompatibilný s verziou Clangu prítomnou na macOS High Sierra.
- Na stretnutí s cvičiacim MTS prezentovali 4 členovia tímu: Martin, Marek, Bence a Peťo. Cvičiaci chcel vidieť TFS, kládol otázky na prípadné vzniknuté problémy v riadení počas semestra, diskutoval o metodikách a v neposlednom rade o samotnom projekte.
- Záverečné hodnotenie bolo pozitívne, jediná väčšia výčitka sa týkala výkyvom vo velocity tímu naprieč jednotlivými šprintami.

7.2.12 Zápisnica zo stretnutia č. 12

Téma stretnutia: Stav úloh po zimnom semestri, IWYU, ďalšie smerovanie

Dátum: 14. 2. 2018

Čas: 11:00 - 14:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušín

Bc. Marek Škriečka

Priebeh stretnutia:

- hlavnou tému stretnutia bolo zhodnotenie rozbehnutých taskov z predchádzajúceho šprintu v zimnom semestri.
- s IWYU sú problémy na platforme macOS (Clang). Dá sa s ním pracovať, no práca je veľmi pomalá
- Martin sa postupne blíži k dokončeniu refactoringu ohľadom warningov v projekte.
- Mišo s Tomášom pokračujú v prerábkach CMake-u
- Témou tiež je správne formátovanie kódu (stále prítomné zmeny bielych znakov v diffoch commitov).

- Peťo je s IWYU pozadu, práca trvá dlhšie, ako sa zdalo.

7.2.13 Zápisnica zo stretnutia č. 13

Téma stretnutia: Migrácia projektu na GitLab počas stretnutia

Dátum: 21. 2. 2017

Čas: 11:00 - 14:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Počas stretnutia prebiehala migrácia projektu na GitLab.
- Členovia tímu, ktorí nemali vytvorené účty na GitLab-e, sa zaregistrovali priamo na stretnutí.
- Prítomný na stretnutí bol aj študent, ktorý na projekte 3DSoftviz pracuje v rámci svojej bakalárskej práce.
- Boli vytvorené separátne repozitáre pre bakalárov, diplomantov aj tímový projekt.
- GitLab umožňuje nasadenie CI, podľa Milových slov už čoskoro.
- Bude potrebné domyslieť synchronizáciu zmien medzi repozitárimi (v štádiu riešenia).

7.2.14 Zápisnica zo stretnutia č. 14

Téma stretnutia: Prioritizácia úloh, zamýšľané zmeny v namespace-i Lua, diskusia o synchronizácii

Dátum: 28. 2. 2018

Čas: 11:00 - 14:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Budú potrebné zmeny v namespace-i Lua projektu.

- Warningy sú hotové, to isté platí o regaktoringu logovania programu, na ktorý boli pridelení Marek a Martin
- IWYU sa rieši, Peťo potrebuje kolegu. Od vedúceho dostal možnosť si ho zvoliť, vybral si Martina.
- CI sa rieši.
- Synchronizácia bude pravdepodobne prebiehať týždenne. Každé repo bude mať synchronizačnú vetvu.
- Diskutuje sa o možnom termíne. Pre prvú synchronizáciu bol na skúšku zvolený piatok. dovtedy musia byť v našom repe uzavreté všetky MR.

7.2.15 Zápisnica zo stretnutia č. 15

Téma stretnutia: Koniec šprintu, úlohy pre ďalší, znova zváženie TFS, Sol 2

Dátum: 7. 3. 2018

Čas: 11:00 - 14:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Tým znova zvažuje prechod na TFS, od ktorého sa upustilo pri predchode na GitLab. Vtedy sme skúšali fungovať s GitLab-ovským derivátom KanBan-u, kde sa evidovali issues.
- Koniec šprintu, sumarizácia, v akom stave sú úlohy.
- Marek si berie ďalej na starosti rozbehanie Astyle na Windows
- Mišo s Tomášom sa držia namespace-ov, Martin začne s Peťom na IWYU tento týždeň.
- Piatková synchronizácia sa osvedčila
- Do Slacku budú pridaní aj diplomanti a bakalári
- Veľká téma Sol 2 namiesto Dilluculum. Možný task pre budúcnosť.

7.2.16 Zápisnica zo stretnutia č. 16

Téma stretnutia: Pokračovanie šprintu, všeobecná diskusia

Dátum: 14. 3. 2018

Čas: 11:00 - 14:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar
Bc. Bence Ligárt
Bc. Michal Knapík
Bc. Tomáš Krupa
Bc. Miloslav Slížik
Bc. Peter Marušin
Bc. Marek Škriečka

Priebeh stretnutia:

- Stále sa rieši Sol2 ako náhrada za Diluculum
- Bence zatial opraví bug build bug dillucula na platforme Windows (buildovať ako .dll). Do CMake-u bude treba pridať automatický export symbolov pre MSVC platformu (podobne ako pri libnoise alebo aruco)
- IWYU v štádiu riešenia, CI v štádiu riešenia
- Marek sa pokúsi spojazdniť AStyle pre Windows.
- TFS sa nahadzuje z GitLab-u pre účely dokumentácie

7.2.17 Zápisnica zo stretnutia č. 17

Téma stretnutia: Koniec šprintu, retrospektíva,

Dátum: 28. 3. 2018

Čas: 11:00 - 14:00

Miestnosť: 4.20

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar
Bc. Bence Ligárt
Bc. Michal Knapík
Bc. Tomáš Krupa
Bc. Miloslav Slížik
Bc. Peter Marušin
Bc. Marek Škriečka

Priebeh stretnutia:

- Nový model architektúry (diagram dostupný na <https://docs.google.com/document/d/137Rp6xPFG8UN4pCW5dEdVhCjAzd1JxohmMFj2o5VCb4>)
- Každý má za úlohu identifikovať jeden prípad, kt. porušuje vyššie navrhnutý model a treba refaktor (namespace, závislosti etc).
- IWYU sa konečne dokončilo, čaká sa na review vedúceho tímu. Konflikty sa resolvli, je potrebné ešte otestovať na každej platforme.
- CI sa rieši
- Diskusia o prezentovanom návrhu

7.2.18 Zápisnica zo stretnutia č. 18

Téma stretnutia: Aktualizácia Aruca & OpenCV, kontrola IWYU, CI, diskusia o architektúre

Dátum: 4. 4. 2018

Čas: 11:00 - 14:00

Miestnosť: 4.46

Prítomní: Ing. Peter Kapec

Bc. Martin Gašpar

Bc. Bence Ligárt

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Miloslav Slížik

Bc. Peter Marušin

Bc. Marek Škriečka

Priebeh stretnutia:

- Martin sa pokúsil na stretnutí rozbehať 3DSoftviz s aktualizovanou verziou Aruca a OpenCV 3. Sme blízko úspechu a držíme mu všetci prsty.
- IWYU sa mergne pri najblišom syncu, zmeny fungujú pre každého. Na Windowse bolo ale potrebných zopár fixov
- Martin a Marek diskutujú o návrhu prezentovanom na minulom stretnutí
- Sol2 stále v štádiu riešenia.
- Peťovi s Martinom sa podarilo "hotfix"-núť bug s nefungujúcou webkamerou, zmeny sa zapracujú čoskoro
- Tomáš rieši linkovanie, diskusia s vedúcim
- CI sa rieši

7.3 Motivačný dokument

Predkladaným motivačným listom sa budeme snažiť vyzdvihnúť hlavné kvality nášho tímu, ktoré hrali úlohu pri výbere tému, no tiež nám pomôžu pri práci na projekte.

Tím obsahuje jednotlivcov so skúsenosťami s tvorbou webových aplikácií (Marek, Martin, Bence). Niektorí sa zaujímame o oblasť dátovej analýzy (Milo, Peťo), iní o vnorené systémy a siete (Mišo a Tomáš) či o architektúru a softvérové inžinierstvo (Marek). Máme aj členov so skúsenosťami s prácou v tíme ktoré získali v rámci brigád, resp. stáži vo firmách (Martin, Mišo, Tomáš).

Práve spomínaná rôznorodosť je našou najväčšou devízou. Schopnosť adaptovať sa na nové doteraz nepoznané technológie je ale vlastnosť, ktorá nás všetkých spája.

Aj pri voľbe rozvrhu sme sa každý zariadili podľa vlastných preferencií, no kombinácia predmetov hrala úlohu aj pri výbere tému. Niektoré nami zvolené predmety sú: Aspektovo-orientovaný vývoj softvéru, Pokročilé databázové technológie, Spracovanie obrazu, grafika a multimediami, Vizualizácia dát a Neurónové siete.

Takmer všetci sa medzi sebou dlho poznáme a vieme navzájom odhadnúť svoje kvality.

Všetci bývame na internáte alebo v okolí Bratislavы, čiže ani osobné stretnutia tímu kedykoľvek v prípade potreby nepredstavujú žiadny problém. Veľkou výhodou sú pestré

záujmy a zameranie jednotlivých členov. Traja z nás sa na inžinierskom stupni rozhodli pre štud. program Internetové technológie a môžu tak prispieť ďalšimi znalosťami či iným uhlom pohľadu.

Motivácia pre projekt Inteligentný bazár [IBazar]

Projekt IBazar nás oslovil najviac. Viacerí sme sa v rámci svojich bakalárskych projektov už venovali práci s väčšími množinami rôzne štruktúrovaných dát (Mišo, Peťo a Tomáš) a viacerí by sa chceli „veľkým dátam“ a ich spracovaniu venovať aj v ďalšom štúdiu a následne v práci. Pre Peťa je motiváciou tiež získať skúsenosť s jazykom Python či konečne si poriadne „ohmatať“ svet webu a projekt sa mu javí ako ideálna voľba pre oba spomínané ciele. Mišo a Tomáš už majú s Pythonom skúsenosti. Všetkých nás zaujali odporúčané technológie. Kedže všetci denne používame bazáry a weby ponúkajúce „tovar z druhej ruky“, vieme, akými neduhmi trpia najznámejšie portály. Hlavne na portáli bazos.sk je situácia miestami naozaj zúfalá, čo ale vyplýva aj z jeho všeobecného zamerania. Umožňuje predaj širokého spektra druhov tovaru a nie je možné jeho zaradenie až do toľkých podkategórií ako pri iných portáloch špecializujúcich sa napríklad iba na oblečenie. Na bazos.sk navyše musia byť všetky kľúčové slová súčasťou inzerátu. Dohľadateľnosť inzerátu preto závisí od jeho sformulovania a často aj od faktorov ako preklepy či znalosť gramatiky zadávateľa. Chceme zlepšiť možnosti indexovania položiek v novom modeli bazáru a zvýšiť relevanciu inzerátov. Prieskum podobných riešení ako aplikácia LetGo či český portál vinted.cz už máme za sebou a poučení ich nedostatkami by sme radi navrhli vlastné riešenie.

Príloha A – Zoradenie všetkých tém podľa priority

4. Inteligentný bazár [IBazar]
9. Odporúčanie pre e-biznis (Recommendation for eCommerce) [reCommers]
18. Rozpoznávanie clouдовých služieb [OntoSEC]
20. Behaviorálna biometria na mobilných zariadeniach [Behametrics]
17. Vnorený systém monitorovania osôb [Bleyslet 2.0]
26. Inteligentné parkovanie [SmartParking]
15. DeepSearch, alebo nájdeme to, čo práve potrebujete [DeepSearch]
24. Investment Portal [Invest]
25. Artificial Intelligence: Voice Channel [VirtualAsist]
3. Otvorené zmluvy: Budovanie prepojení vo verejných dátach [Zmluvy]
2. Group de'Cider [Group]
13. Analýza správania sa používateľa v mobilných aplikáciách [Mob-UX]
14. 3D UML, optimized version [3D-UML]
11. Databanka otázok a úloh [FIIT-DU]
5. Monitorovanie a vyhodnocovanie fyziologických procesov človeka [StresMonitor]

- 19. Pohlcujúci Web [iWeb]
- 12. Kolaboratívne prototypovanie používateľských rozhraní [Collab-UI]
- 7. Vizualizácia informácií v rozšírenej realite [VizReal]
- 8. Vzdelávanie vo Virtuálnej realite [EduVirtual]
- 10. Podpora diagnostiky [Look-Inside-Me]
- 1. Importér verejných datasetov [PubDatasets]
- 23. Návrh systému MOD [Future MOD]
- 16. Softvérovo definované siete pre budúci Internet [SDN4Futl]
- 22. Softvérovo riadené siete rozšírené o WiFi štandard [SDWN]
- 21. 3D simulovaný robotický futbal [3D futbal]
- 6. Internet vecí v našich životoch [IoT]

Príloha B



	ŠKOLA
	VOL'NO

Obrázok 13: Rozvrh tímu s farebne odlišenými voľnými a obsadenými časovými úsekmi v rámci jednotlivých dní

7.4 Metodiky

Nižšie uvedené metodiky sme po analýze revidovali a uplatnili do praxe, aj keď pôvodné verzie boli vytvorené týmami pred nami. Snažili sme sa zachovať zaužívaný workflow, no zároveň si prispôsobiť časti, ktoré sa pri práci na projekte v našom tíme neosvedčili.

7.4.1 Gitflow metodika

Forkovanie na GitHub-e

Fork na GitHub-e neprenesie tag-y do forkнутeho repozitara, treba ich rucne preniest, v novom repozitari:

- git remote add povodny-repozitar git@github.com:povodny-repozitar/nazov_repo.git
- git fetch povodny-repozitar 'refs/tags/:refs/tags/'
- git push --tags

Vetvy

- Master - hlavný projekt

- Develop - branchnuta z mastra, kazdy sprint ma vlastnu Develop vetvu, na konci sprintu sa merge späť do mastra,
- **!!! pred mergom treba spravit komplet build (nie len unity)**
- Feature - branchnuta z developu, kazdy novy kus funkcionality (task v issue tracking nastroji), ktorý sa kodi musí mať vlastnu
 - Feature vetvu... po dokončení a validácii kódu sa merge späť do Developu, NEINTERAGUJE S MASTER VETVOU
- Hotfix - vetva na rýchly fix priamo z mastera, merge sa do mastera AJO developu, navysuje aktuálnu verziu

Vždy mergejeme cez Shell a s prepínacom **--no-ff**

Cheat sheet so všetkymi základnými commandmi:

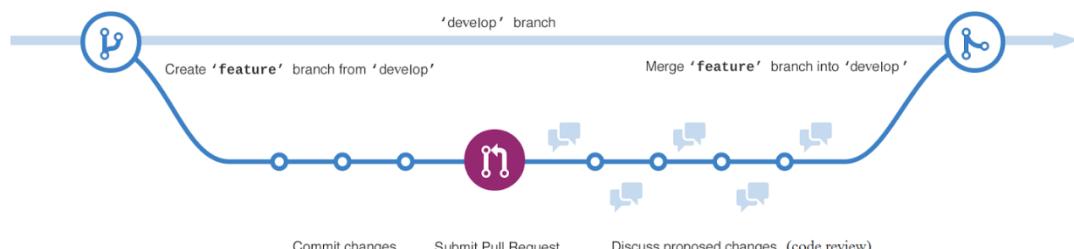
<https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf>

Pull requesty

Po dokončení práce, keď sme ready to review sa dava pull request na vetvu, do ktorej sa bude mergeovať.

Pull request sa robi z GUI GitHubu (pravý horný roh), alebo \$ git request-pull {meno_commitu} {URL} (doporučujem robiť cez GUI)

Po odsúhlasení Pull requestu sa potom pristupí k merge.



Obrázok 14: Práca na novej funkcionality z pohľadu verziovacieho systému podľa Git metodiky

Obrazok ilustruje vytvorenie feature branch z develop vetvy, implementáciu rozdelenú do znázornených commitov, nasledný pull request predstavujúci žiadosť o code review a finalný merge do develop vetvy daného sprintu.

Commit messages

v Commit messages používame tagy a ID úlohy na záčiatok:

- [FIX] - fixili sme nejakú chybu z minula, bugfix, hotfix a podobne
- [ADD] - pridali sme novú funkcionality, súbor, ...
- [DOC] - pridali sme dokumentáciu, komenty...
- [REF] - pre refactoring
- [FMT] - formátovanie textu, uprava
- [TEST] - pre testy

- [BUILD] - aktualizacia CMake build systemu, modulov

Za tym velmi strucne (a vystizne) opiseme, ake zmeny sme spravili. Message by mali byt kratke, no pokryvat vsetko, co sme v committe spravili. !!! **vseobecny tvar: "[tag] #taskId Popis vykonanej zmeny"** Napr. *[DOC] #3654 Pridanie uvadzania ID ulohy do gitflow metodiky*

Useful commands

- \$ git submodule update --init --recursive
 - update submodulov (dependencies)
- \$ git checkout -f meno_branch
 - checkout branche aj napriek lokalnym zmenam, budu zahodene
- \$ git status
 - vypise vsetky vykonane zmeny
- \$ git stash / \$ git stash pop
 - ulozi stav projektu do stashu, z ktoreho sa da potom tento stav pop-nut, dobre na prenos zmien medzi vettami

Tvorba feature branch-u:

- \$ git checkout -b "feature/meno-feature" develop //Switched to a new branch "feature/meno-feature"

Mergovanie hotoveho feature:

- \$ git checkout develop //Switched to branch 'develop'
- \$ git merge --no-ff meno-feature
- \$ git push origin develop

Tvorba hotfix branch-u:

- \$ git checkout -b "hotfix/nazov-co-fixujem" master //Switched to a new branch "hotfix-{cislo_verzie}"
- \$ git commit -m "sprava, co som spravil"

Uzatvorenie Hotfix branchu:

- \$ git checkout develop //Switched to branch 'develop'
- \$ git merge --no-ff "hotfix/nazov-co-fixujem"

7.4.2 TFS metodika

Všeobecná metodika na manažment úloh v tíme

Pridanie novej úlohy

- Pri každej úlohe je potrebné uviesť opis. Opis si k úlohe zadáva ten, kto si ju vyberie.

- Uviestť odhadovaný čas dokončenia.
- Opis musí byť podrobný, aby každému členovi bolo jasné, čo ma vykonať po pridelení úlohy.

Rozdeľovanie úloh

- Každý si vyberie (potiahne) úlohu/úlohy, ktorá/é majú najvyššiu prioritu.
- Ak ostanú nepridelené úlohy, študentský vedúci tímu pridelí členom zvyšné úlohy.

Kedy je úloha hotová

- Dokumentácia: Keď je znova vygenerovaná.
- Kód: Potrebné spraviť code review a vykonať pull request do vetvy, ktorá sa bude mergovať.
- Testy: Keď je spravený report z testu.
- Zápisnica: Keď je nahratá vo formáte pdf na stránke tímu.

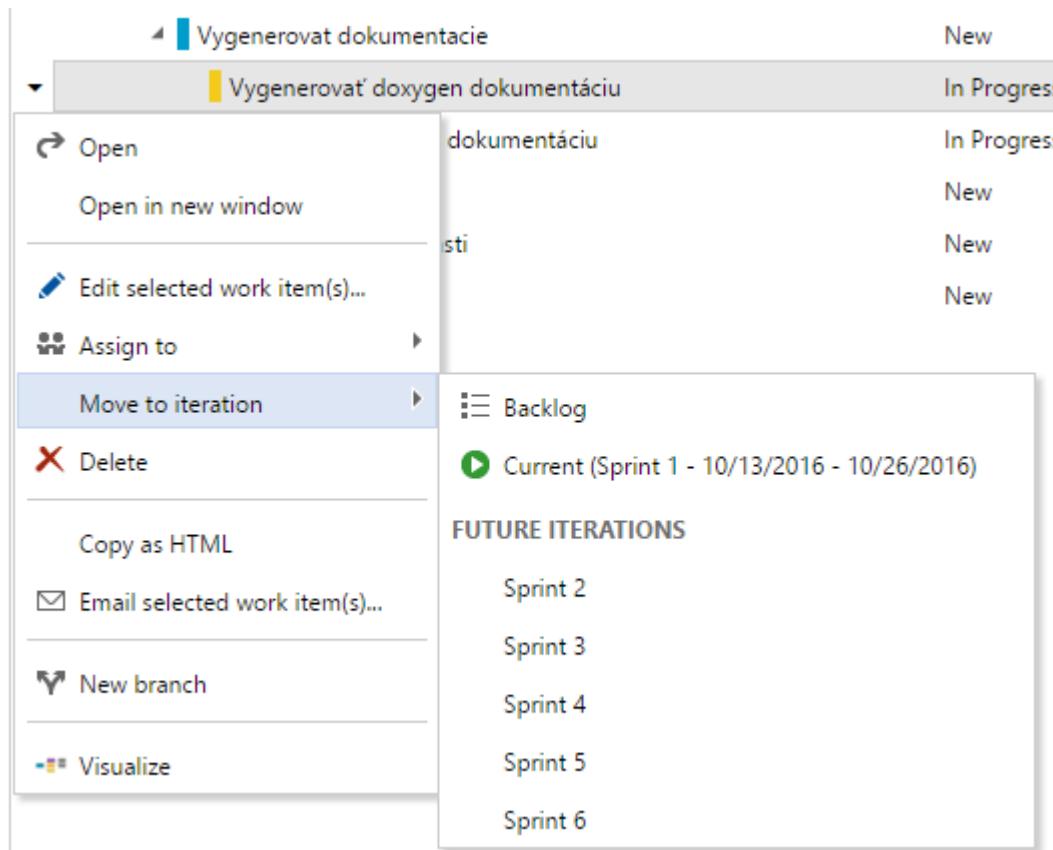
TFS metodika

- Adresa TFS: <https://tfs.fiiit.stuba.sk:8443/tfs/> (Potrebné sa lognúť 2x)
- Projekt: Just18

Pridávanie úloh

- Úlohy podobného charakteru priradíme do spoločného backlog itemu.
- Pri pridávaní úlohy sa automaticky nastaví stav 'To Do'.
- V prípade objavenia chyby, je potrebné vytvoriť novú úlohu typu Bug (Chyba)
- Odhadovaný čas dávame na backlog itemy, rovnako ako aj Acceptance Criteria (predpripravený field pri rozkliknutí backlog itemu v TFS).
- POZOR: TFS podporuje estimáciu času v IBA v hodinách.

Úlohy (tasky) sa môžu nachádzať v troch stavoch: * To Do * In Progress * Done Nesplnené úlohy, ktoré sa nestihli dokončiť v danom šprinte, presunieme do nasledujúceho šprintu.



Obrázok 15: Priradovanie úlohy do prislúchajúceho šprintu

Vytváranie exportov z TFS

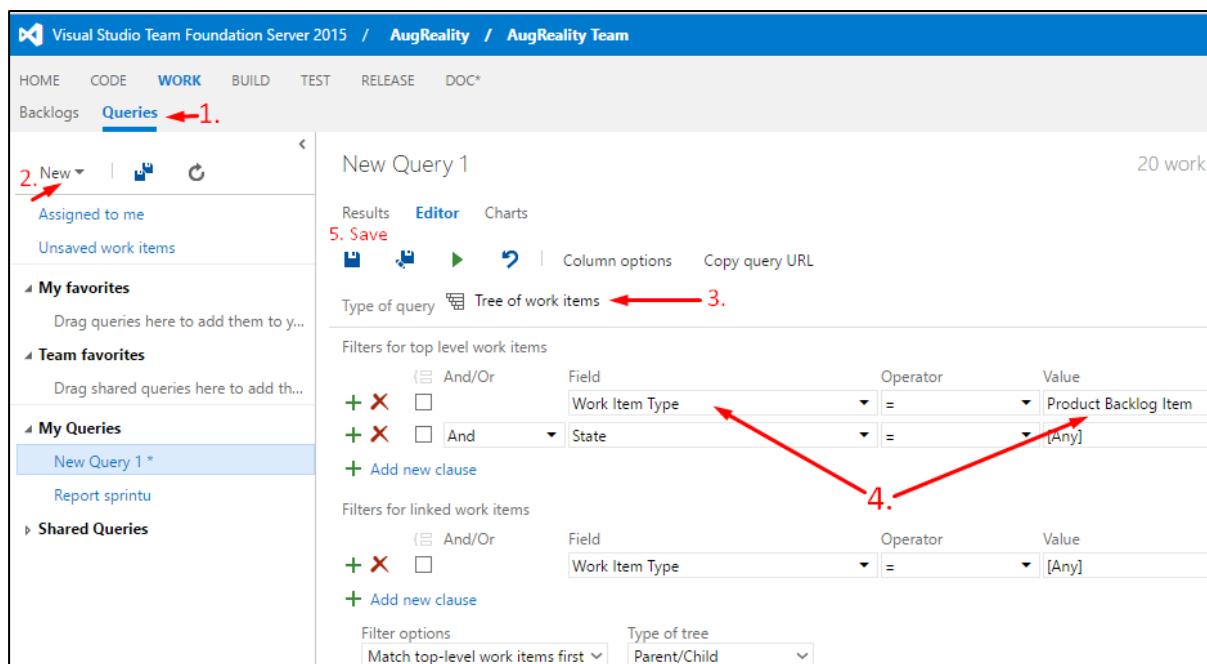
Navod na stranke: [https://msdn.microsoft.com/en-us/library/dd286627\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/dd286627(v=vs.110).aspx)

URL: <https://tfs.fiit.stuba.sk:8443/tfs/>

Login: ako do AIS-u

Pred prvým exportom je potrebné si vytvoriť query, ktoré vráti stav úloh v danom šprinte.

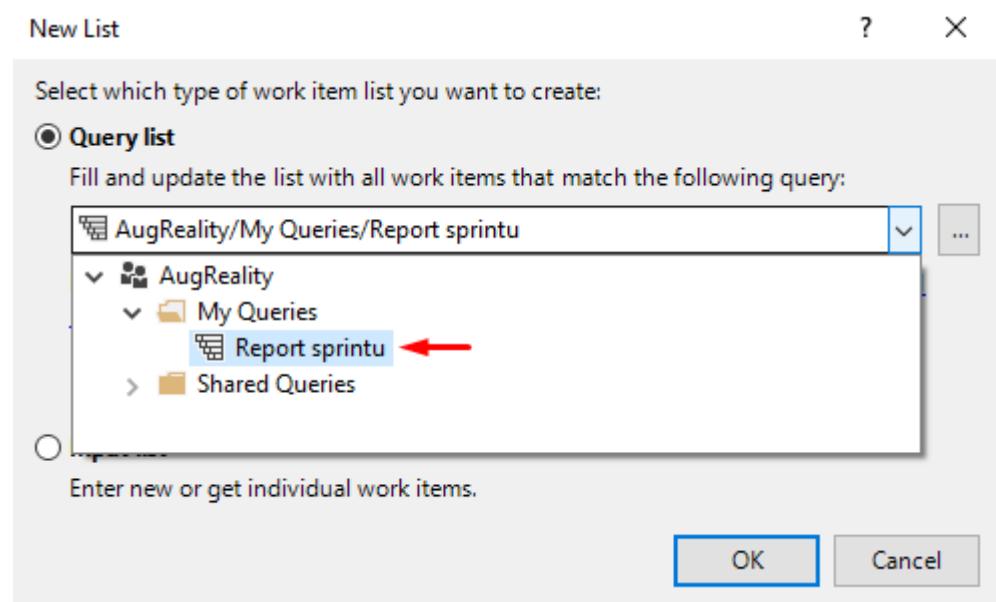
Návod na vytvorenie query:



Obrázok 16: Vytváranie query za účelom exportu úloh zo systému TFS

Excel

- V hlavnom menu vybrať záložku TEAM (mala by byť vpravo hore).
- Klik na New List (umiestnená na ľavo pod záložkou File/Súbor).
- Vybrať novo vytvorené query:



Obrázok 17: Výber query pre export úloh zo systému TFS

- Hotovo

Metodika tvorby a údržby UML diagramov prostredníctvom PlantUML

PlantUML

[PlantUML](#) je jednoduchý program na tvorbu UML diagramov prostredníctvom ich textového opisu. K samotnému programu prislúcha aj rozsiahla [dokumentácia](#).

PlantUML je voľne dostupný na [stiahnutie](#) z oficiálnej stránky, prípadne je možné na otestovanie použiť aj jednoduchú [web aplikáciu](#).

Pre plnohodnotné využitie je potrebné mať taktiež nainštalovaný [Graphviz](#).

Tiež ponúka možnosť [integrácie](#) s viacerými textovými editormi a wiki stránkami.

Pravidlá pre tvorbu súborov

1. Každý diagram sa nachádza v samostatnom textovom súbore (koncovka .txt, resp .wsd pri použití integrácie so sublime text).
2. Vygenerovaný diagram má identický názov ako prislúchajúci textový súbor (koncovka .png).
3. Názvy súborov sú po anglicky.

Užitočné príkazy a postupy

Odstránenie duplicity pomocou Preprocesoru

Pri písaní diagramov, ktoré obsahujú komplikované vzťahy medzi entitami môžeme naraziť na situáciu, kde budeme veľa krát za sebou písanie ten istý názov triedy alebo metódy. S využitím makier preprocessoru môžeme túto duplicitu ľahko odstrániť.

@startuml

'Bez proprocessoru

```
package "class Filter representation" {
    class ObjectStructure
    class Element {
        +{abstract}register(Visitor v)
    }
    class Mapper {
        +register(Filter f)
    }
    class Klient
    class Visitor {
        +{abstract}visitMapper(Mapper m)
    }
    class Filter {
        +visitMapper(Mapper m)
    }
}
```

```
ObjectStructure -down-> Element  
Mapper -up-|> Element
```

```
ObjectStructure <-left- Klient
```

```
Klient -down-> Visitor  
Filter -up-|> Visitor  
}  
@enduml
```

```
@startuml  
'S preprocesorom  
!define o(x) ObjectStructure  
!define e(x) Element  
!define m(x) Mapper  
!define k(x) Klient  
!define v(x) Visitor  
!define f(x) Filter
```

```
package "class Filter representation" {  
    class o()  
    class e() {  
        +{abstract}register(v(x) v)  
    }  
    class m() {  
        +register(Filter f)  
    }  
    class k()  
    class v() {  
        +{abstract}visitMapper(m(x) m)  
    }  
    class f() {  
        +visitMapper(m(x) m)  
    }
```

```
o() -down-> e()  
m() -up-|> e()
```

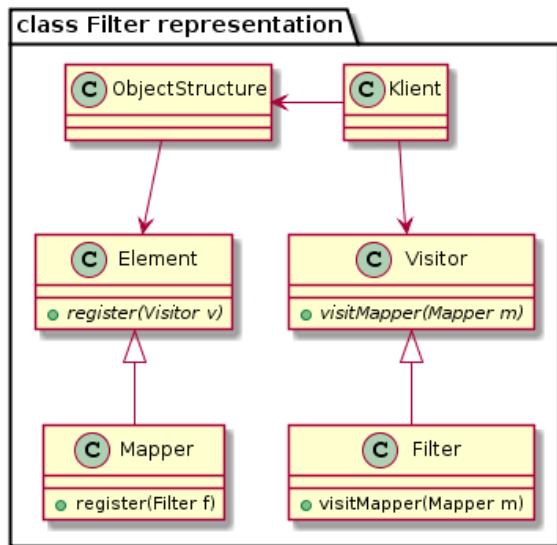
```
o() <-left- k()
```

```

k() -down-> v()
f() -up-|> v()
}
@enduml

```

V oboch prípadoch bude výsledok nasledovný:



V druhom prípade sa rozhodne menej napíšeme a máme možnosť meniť použité názvy tried na jednom mieste namiesto toho aby sme ich museli meniť všade. Stojí za poznámku, že každé definované makro musí mať parameter (v našom prípade x, z ktorého ajtak nečítame). Viac o Preprocesore na [tejto](#) stránke.

Použitie aliasov v sekvenčnom diagrame

V sekvenčných diagramoch odporúčame pri definovaní volaní medzi objektami používať aliasy (skratky). Ich princíp je analogický s predchádzajúcim makrom avšak sú ešte o niečo prehľadnejšie. Aliasy nie sú však podporované v class diagramme.

@startuml

```

participant Client as c
participant Server as s

```

title sd Basic Server call

```

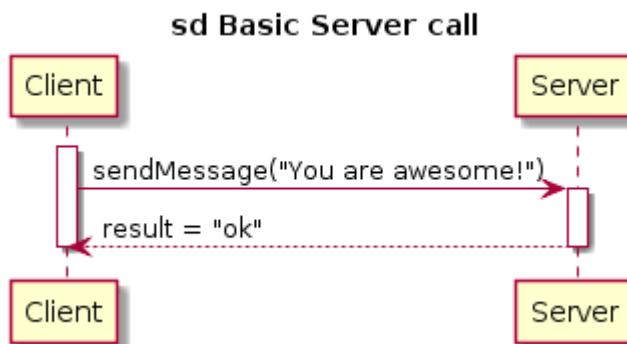
activate c
c -> s: sendMessage("You are awesome!")
activate s
s --> c: result = "ok"

```

```
deactivate s  
deactivate c
```

```
@enduml
```

Výsledok:



Užitočnosť týchto skratiek (a makier) pochopiteľne narastá s narastajúcou komplexitou daného diagramu.

Pravidlá pre súborovú štruktúru

Samotné UML diagramy je potrebné rozdeliť do prehľadnej súborovej štruktúry:

- projekt (názov projektu, napr. 3dsoftviz)
 - doc (inštalačná dokumentácia, vygenerovaná dokumentácia atď.)
 - uml
 - structural
 - class diagrams (korešpondujúce s reálnym kódom)
 - component diagrams
 - behavioral
 - activity diagrams
 - use-case diagrams
 - sequence diagrams
 - state diagrams

Sphinx dokumentacia

Instalacia

Python

- Pre pracu so Sphinxom treba mat nainstalovaný [Python](#).

Python ponuka verzie 2.x a 3.x. Sphinx 1.3 može bezat pod Python 2.6, 2.7, 3.3, 3.4, ale odporucana verzia je 2.7.

- Pre stahnutie a instalovanie externych kniznic pre Python existuje prikaz *pip*. Prikaz uz sa nachadza v oficiálnych verziach Pythonu 3.4.0 alebo 2.7.9.
- Ak prikaz sa nainstaloval automaticky, treba ho stiahnuť zo stránky <https://bootstrap.pypa.io/get-pip.py> a niekam uložiť. V prikazovom riadku treba prejsť do adresára s *get-pip.py* a spustiť nasledovný prikaz:
- `python get-pip.py`

Sphinx

- Prejsť do priečinka s dokumentáciou (tam kde *index.rst* sa nachadza) a pomocou prikazu *pip* nainstalovali Sphinx:
- `pip install sphinx`
 - (sphinx-doc.org)
- Ak treba vytvoriť novú dokumentáciu, pre nastavenie zdrojového adresára a vytvorenie potrebných suborov na pracu so Sphinx treba spustiť prikaz `sphinx-quickstart`
- a odpovedať na otázky. Vyberte si vsetky predvolene odpovede a po vyzve zadajte nazov, autorov a verziu projektu.
- Týmto prikazom budú vygenerované subory *Makefile*, *make.bat* a *conf.py.in*.
 - Vsetky konfigurácie dokumentácie sú v *conf.py.in*.

Attention!

Sphinx-quickstart a vytvaranie týchto suborov generujú novú dokumentáciu! Ak subory *index.rst*, *Makefile*, *make.bat* a *conf.py.in* už existovali, tak sa prepisú!

- Sphinx dokumentácia generuje výstup v rôznych formátoch zo suborov *.rst*.
Podrobnejšie o [RestructuredText](#).

HTML dokumentácia

- Subor *make.bat* povoli vygenerovať dokumentáciu v tom formáte, ktorý potrebujete
- Pre generovanie HTML dokumentácie treba v prikazovom riadku prejsť do priečinka s ReST subormi a *make.bat* suborom a spustiť prikaz `make html`
- Inak generovanie dokumentácie sa da spustiť pomocou CMake v QtCreatore

PDF dokumentacia

Pre generovanie PDF dokumentacie potrebujeme najprv vytvorit Latex dokumentaciu.

Note

Pre pracu s Latex treba mat [TeXlive](#)

Prikazom

make latex

vygeneruje sa Latex dokumentacia, ktorá nasledne sa može konvertovať do PDF pomocou programu [TeXstudio](#).

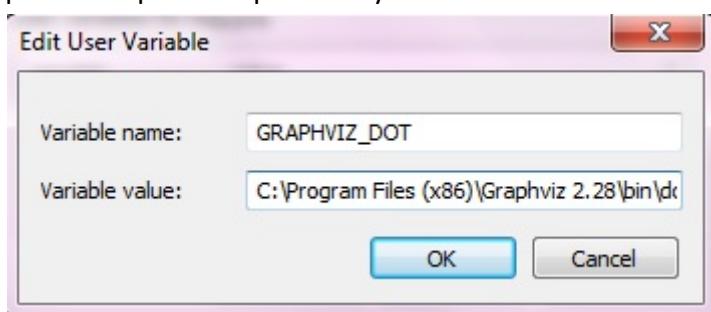
Note

PDF dokumentacia generuje len pomocou prikazoveho riadku a externeho programu, neda sa spustat cez CMake!

PlantUML

Pre pracu s PlantUML nastrojmi v Sphinx treba:

- nainstalovať [Java](#)
- pridať Java do premenných prostredia (environment variable)
- nainstalovať [Graphviz](#)
 - odporúčaná verzia je 2.28
- pridať Graphviz do premenných



-

Note

Hodnota premennej ma byť do *dot.exe*

- pridať Graphviz do extensions v conf.py.in:
 - extensions = ['sphinx.ext.graphviz']
- nainstalovať *sphinxcontrib-plantuml* zo [stránky](#) alebo prikazom
 - pip install sphinxcontrib-plantuml
- pridať plantuml do extensions v conf.py.in:
 - extensions = ['sphinxcontrib.plantuml']
- stiahnuť [plantuml.jar](#)
- pridať do conf.py.in prikaz
 - plantuml = 'java -jar cesta/do/plantuml.jar'
- Attention!
- Dolezite je zmeniť tuto cestu na spravnú, ak máte aktuálnu na Vašom počítači!

- pridavat UML do dokumentacii je mozne pomocou
- .. uml::

```
!include /cesta/do/subor.wsd(txt)
```

alebo

```
@startuml  
PlantUML kod  
@enduml
```

Excel tabulky

- Pre import Excel suborov do dokumentacie treba nainstalovať *excetable* pomocou prikazu

```
pip install sphinxcontrib-excetable
```

- Pridat *excetable* do extensions v *conf.py.in*:

```
extensions = ['sphinxcontrib.excetable']
```

- Importovať tabulky pridaním do .rst suboru:

```
.. excetable:: caption  
:file: path/to/document.xls  
:header: 1  
:selection: A1:B2
```

- Podrobnejšie o [Options](#)

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Just18

Dokumentácia k inžinierskemu dielu

Vedúci tímu: Ing. Peter Kapec, PhD.

Členovia tímu: Bc. Martin Gašpar

Bc. Michal Knapík

Bc. Tomáš Krupa

Bc. Peter Marušin

Bc. Bence Ligárt

Bc. Miloslav Slížik

Bc. Marek Škriečka

Akademický rok: 2017/2018

Obsah

1	Úvod	4
2	Globálne ciele.....	6
2.1	Globálne ciele pre zimný semester	6
2.2	Globálne ciele pre letný semester.....	7
3	Celkový pohľad na systém	7
3.1	Architektúra.....	7
3.2	Dátový model	8
3.2.1	Stručný opis vybraných entít	9
3.3	Diagram tried.....	9
3.4	Moduly	9
4	Zimný semester.....	10
4.1	Aktualizacia zavistlosti projektu a build projektu	10
4.1.1	macOS.....	10
4.1.2	Linux	11
4.1.3	WSL.....	11
4.1.4	Windows	12
4.2	RefaktORIZÁCIA Cmake	13
4.3	OpenPose	14
4.3.1	Úvod	14
4.3.2	Analýza	14
4.4	Vagrant	16
4.5	Provisioning	17
4.6	Continuous integration	17
5	Letný semester.....	18
5.1	Include What You Use (IWYU).....	18
5.2	Migrácia na GitLab a synchronizácia	19
5.3	RefaktORIZÁCIA logovania projektu.....	19
5.4	RefaktORIZÁCIA modulu Data	19
5.5	Analýza knižnice sol2	20
5.6	RefaktORIZÁCIA modulu LeapLib	21
5.7	RefaktORIZÁCIA modulu GitLib a ParserLib	21
5.8	RefaktORIZÁCIA modulov LuaGraph a LualInterface	21

6	Prílohy	21
6.1	Inštalačný manuál.....	21
6.1.1	Linux	21
6.1.2	Windows Subsystem for Linux (WSL)	22
6.1.3	macOS (vytvárané a testované pre verziu 10.13 High Sierra).....	24
6.1.4	Windows.....	26
6.2	Používateľský manuál pre 3DSofviz	31
6.2.1	Ovládacie prvky	31
6.2.2	Záložka GRAPH	32
6.2.3	Záložka CONSTRAINTS	34
6.2.4	Záložka CLUSTERING.....	36
6.2.5	Záložka CONNECTIONS.....	38
6.2.6	Záložka EVOLUTION.....	38
6.2.7	Záložka MORE FEATURES	40
6.2.8	Hlavné okno.....	43
6.2.9	Git repozitár.....	44

1 Úvod

Predkladaný dokument slúži ako technická dokumentácia k predmetu Tímový projekt na Fakulte informatiky a informačných technológií Slovenskej technickej univerzity v Bratislave. Opisuje štruktúru už existujúceho univerzitného projektu 3DSoftviz na vizualizáciu informácií v obohatenej realite a prácu na danom projekte.

Prvá kapitola dokumentu opisuje globálne ciele pre zimný a letný semster, ktoré sa bude snažiť náš tím naplniť. Vzhľadom na rozsah projektu, stav dokumentácie a hlavne požiadavku na podporu viacerých platforiem sme svoju snahu v zimnom semestri venovali hlavne zmenám v infraštruktúre projektu a údržbe. Snažili sme sa vytvoriť vhodné podmienky na plynulé zavedenie continuous integration. Continuous integration pri budúcom vývoji zautomatizuje proces zostavenia a testovania, no taktiež pomôže monitorovať a urýchliť zavádzanie zmien do projektu. Tiež sme sa znažili o vytvorenie replikovateľného vývojového prostredia, ktoré budúcich vývojárov 3DSoftvizu odbremení od počiatočnej konfigurácie či stiahovania závislostí.

Druhá kapitola obsahuje technický popis stavu projektu v čase, keď sme na projekte začali pracovať. Jednotlivé podkapitoly sú venované architektúre, dátovému modelu, triedam a vzťahom medzi nimi a modulom systému.

Ďalšie kapitoly popisujú prácu na projekte z hľadiska analýzy, návrhu, implementácie a testovania nami modifikovaných časti systému.

Záverečná kapitola dokumentu obsahuje inštalačné manuály pre vývojárov na rôznych platformách a tiež používateľský manuál 3DSoftviz-u. Ten sme prebrali po tímových projektoch z minulých rokov a mierne aktualizovali. Nové Inštalačné manuály obsahujú tiež aktualizovaný zoznam potrebných závislostí a prípadné známe chyby vyskytujúce sa počas konfigurácie aj s popisom riešenia.

Podiel práce na dokumentácii k inžinierskemu dielu

Kapitola	Autor
Globálne ciele pre zimný semester, Aktualizacia závislostí projektu a build projektu - macOS	Peter Marušin
Aktualizacia závislosti projektu a build projektu - Linux	Miloslav Slížik
Aktualizacia závislosti projektu a build projektu - WSL, Formát	Marek Škriečka
Aktualizacia závislosti projektu a build projektu - Windows	Bence Ligárt

Celkový pohľad na systém	Bence Ligárt, Marek Škriečka
Analýza knižnice OpenPose	Martin Gašpar
RefaktORIZÁCIA CMake	Michal Knapík

2 Globálne ciele

2.1 Globálne ciele pre zimný semester

V zimnom semestri sa všetci členovia tímu budú venovať analýze aktuálneho stavu projektu. Na začiatku sa budeme snažiť nakonfigurovať si vývojové prostredia na našich počítačoch, aby sme projekt vedeli spustiť a mohli pracovať na jeho vývoji. Dokumentácia projektu obsahuje inštalačné manuály pre vývojárov vo forme osobitného dokumentu pre každú z troch podporovaných platform: Windows, macOS a Linux (Debian-based distribúcie, napr. Ubuntu). Budeme sa pridržiavať existujúcich príručiek, čím zároveň overíme ich relevantnosť a v prípade potreby ich aktualizujeme.

Projekt so sebou prináša nutnosť inštalácie mnohých závislostí a preto je tiež potrebná synchronizácia inštalovaných verzií na všetkých platformách použitých pri vývoji. Projekt stále závisí na knižnici Qt vo verzii 4, no radi by sme prešli na verziu 5 a taktiež sa po počiatočnej analýze pokúsili o aktualizáciu ostatných závislostí. Ak dokážeme aktualizovať závislosti na novšie verzie, budúcim vývojárom uľahčí počiatočnú konfiguráciu (novšie verzie sú ľahšie dostupné v package manageroch systémov macOS a Linux) a projekt si so sebou nemusí niesť už nevyvájané a nepodporované verzie.

Ďalším cieľom vytvorenie vhodných podmienok na zavedenie continuous integration a continuous development (CI/CD). Pri vývoji multiplatformového softvéru ich dôležitosť narastá - proces zostavovania pre konkrétné platformy či testovania je automatizovaný a tiež je možné monitorovať zostaviteľnosť projektu po vykonaní zmien v repozitári. Projekt je momentálne uložený vo vzdialenom git repozitári na GitHub-e, no v pláne je jeho migrácia na GitLab, ktorý poskytuje natívne mechanizmy pre CI/CD. Podporované je zostavovanie pre všetky 3 platformy. Po odladení do spustiteľnej podoby na všetkých platformách môže byť teda projekt premigrovaný a zavedené CI/CD.

Ďalším cieľom vytvorenie replikovateľného vývojového prostredia s už nainštalovanými závislosťami 3DSoftvizu, nastavenými premennými prostredia a samotným 3DSoftvizom. V tom bude bez ďalšej väčšej konfigurácie možný vývoj projektu. Prostredie chceme vytvoriť pomocou softvéru Vagrant určeného na tvorbu virtuálnych vývojových prostredí prostredníctvom virtualizačných nástrojov ako napr. VirtualBox. Bude predstavovať ďalšiu ľahko použiteľnú alternatívu pre vývojárov, ktorým sa nepodarí projekt rozbeháť na svojom počítači.

Dlhodobým cieľom bude aj údržba už existujúceho kódu, refaktORIZÁCIA, prečistenie dokumentácie a odstraňovanie nájdených bugov. Práci na projekte bolo venovaných mnoho bakalárskych a diplomových prác či tímových projektov.

2.2 Globálne ciele pre letný semester

Kedže väčšinu zimného semestra sme analyzovali existujúci stav projektu a snažili sa o vytvorenie vývojového prostredia, v letnom sa budeme venovať už iba refaktoringu a oprave nájdených chýb.

Členenie kódu v projekte 3DSoftviz nie je v súlade so žiadnym architektonickým štýlom. Časté sú cyklické závislosti, zle členenie modulov či tried. Jednotlivé časti projektu na sebe silno závisia. Našim hlavným cieľom preto je začať s deľbou kódu do modulov a knižníc a zlepšiť logické členenie jednotlivých častí kódu. Jedná sa o veľa práce, no aj ak nestihнемe zrefaktorovať celý projekt, v našej práci môžu pokračovať ďalší študenti.

Chceme tiež zmigrovať projekt na GitLab a zaviesť CI, kedže v zimnom semestri sme to nestihli. Bude potrebné domyslieť aj synchronizáciu vykonaných zmien, kedže na projekte okrem nás pracuje ďalšia skupina študentov v rámci svojich diplomových a bakalárskych prác.

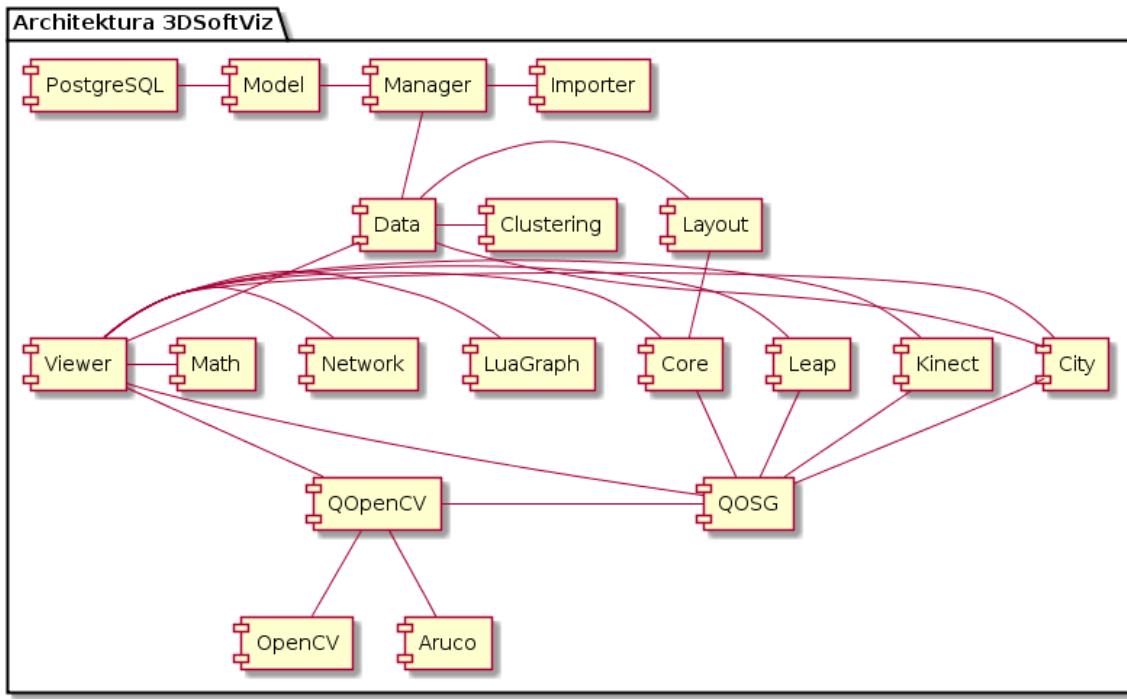
V neposlednom rade budeme pokračovať v refaktORIZácii CMake-u, opravovať nájdené bugy či aktualizovať závislosti projektu (rovako ako počas zimného semestra).

3 Celkový pohľad na systém

Táto kapitola obsahuje celkový pohľad na systém z hľadiska jeho architektúry, dátového modelu, prítomných tried a modulov projektu.

3.1 Architektúra

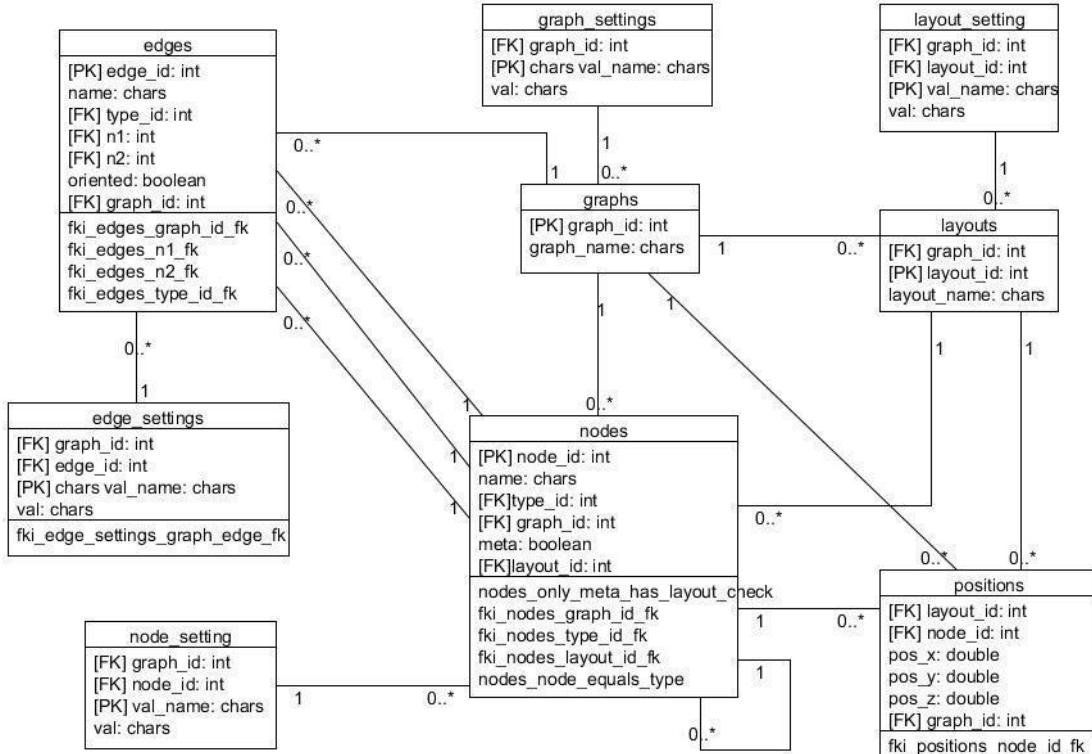
Na obrázku 1 sme znázornili štruktúru projektu 3DSofViz na začiatku semestra. Projekt v predošлом ročníku bol rozšírený o nový modul, pomocou ktorého projekt už dokáže načítať a vizualizovať nový typ grafu - graf modulov. Do architektúry sa tým pridal modul City, ktorý slúži na vytváranie metafory mesta, kde uzly modulov sa znázorňujú ako hierarchické štruktúry poskladané z regiónov, budov a gúľ. Projekt bol rozšírený aj o ďalšie triedy a zmeny, ktoré ale celkovú architektúru nezmenili.



Obrázok 1: Architektúra projektu

3.2 Dátový model

Dátový model databázy sme prebrali od predchádzajúcich prác, zobrazený je na Obrázok 2.



Obrázok 2: Dátový model

3.2.1 Stručný opis vybraných entít

Graphs – predstavuje jednotlivé záznamy grafov,

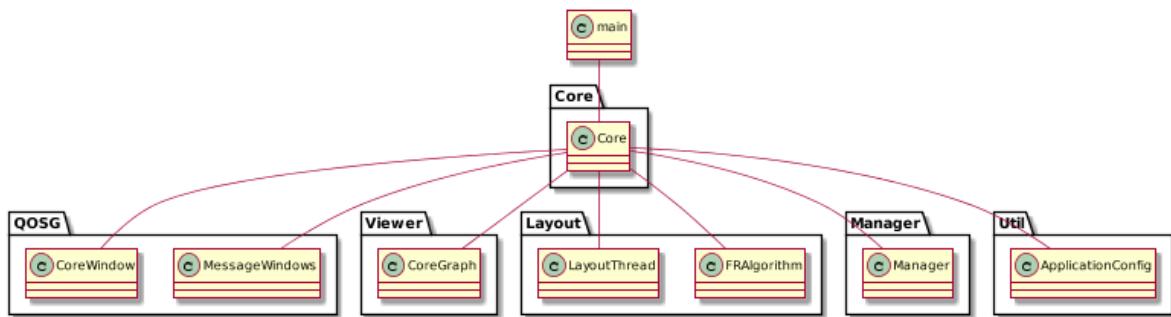
Layouts – obsahuje layout pre daný graf. Predstavuje konkrétné rozloženie množiny uzlov a hrán grafu v priestore,

Nodes – obsahuje uzly a typy v grafe (graf je vytvorený z množiny uzlov popredájaných hranami, pričom každý uzol má svoj typ),

Positions – obsahuje súradnice uzlov v priestore. Pozostáva z troch súradníck, ktoré sú naviazané na konkrétny uzol a layout,

Edges – má uložené hrany spájajúce uzly v grafe. Hrana môže byť orientovaná a neorientovaná a má začiatočný a koncový uzol a typ.

3.3 Diagram tried



Obrázok 3: Diagram tried

Na diagrame tried sme znázornili najdôležitejšie triedy, ktoré používa trieda main a Core.

3.4 Moduly

Core – obsahuje jadro systému, inicializuje základné časti systému.

Data - dátový modul pre opis štruktúry grafu, obsahujúci triedy reprezentujúce jednotlivé prvky grafu (graph, node, edge, type, layout, ...).

Importer - modul pre parsovanie vstupných súborov vo formátoch GraphML, RSF a GXL.

Layout – modul, ktorý má na starosti rozmiestňovanie uzlov v 3D priestore, taktiež obsahuje implementácie layout algoritmu a triedy pre pridávanie ohraničení rozmiestnenia.

Manager - modul pre prácu s grafom.

Math - model pre rozšírenie práce s kamerou.

Model – modul pre komunikáciu systému s databázou. Funkcionalitou je mapovanie objektov do databázy, vytvorenie spojenia s databázou a základne funkcie výberu a uloženia grafu. Taktiež umožňuje uloženie uzlov aj s ich atribútmi a viacero rozmiestnení pre 1 graf.

Network - modul pre podporu kolaboratívnej práce nad grafo. Poskytuje klient/server funkcionalitu.

Noise - modul pre vytvorenie generovaného 3D priestoru pre pozadie.

OsgBrowser - modul zahŕňa viazanie udalostí pre jednotlivé klávesy a akcie myši medzi

rozhraniami Qt a OpenSceneGraph a vizualizáciu načítaných grafov.

QOSG – modul pre prácu s grafickými prvkami softvéru. Má na starosti vytvorenie hlavného okna a prácu s pomocnými oknami a widgetami.

Util - zabezpečuje konfiguráciu nastavení aplikácie a funkcie pre vyčistenie pamäte.

Viewer - modul zabezpečuje pohyb v 3D priestore a prácu s kamerou. V module sa tiež pripravuje graf a jeho pre zobrazenie a vytvorenie 3D kocky pre pozadie.

Kinect – modul pre komunikáciu a ovládanie zariadenia Kinect. Medzi jeho funkcionalitu patrí získavanie informácií a ich nasledovné spracovanie. Obsahuje detegovanie gest, ktoré nahradzajú ovládanie myšou, otáčanie a pohyb grafu a gestá pre ďalšie ovládanie.

Speech - implementuje funkcionalitu rozpoznávania hlasu.

OpenCV - zabezpečuje rozpoznávanie tváre na obraze z kamery a poskytuje funkcionalitu pre správu kamier.

QOpenCV – obsahuje okno pre ovládanie rozpoznávania tváre, značky a ovládanie video pozadia.

Aruco – obsahuje funkcionalitu, ktorá vie rozpoznávať značky z kamery použitím knižnice Aruco.

5DTGloves – zabezpečuje detegovanie gesta ruky a vykonávanie korešpondujúcich akcií.

Leap senzor – deteguje dve ruky používateľa v 3D priestore a sleduje pohyby rúk až na úroveň článkov prstov.

3D myš – poskytuje ovládanie kamery pomocou tohto zariadenia.

4 Zimný semester

4.1 Aktualizacia závislosti projektu a build projektu

4.1.1 macOS

Na platforme macOS projekt beží s verziami závislostí:

- OpenSceneGraph 3.5.6
- Qt 5.9.2
- OpenCV 2.4.13
- CMake 3.9.5

Kompilujeme všetci Clangom a projekt sa nám podarilo dostať do spustiteľnej podoby na verzích macOS 10.13 (High Sierra), . Aktualizovali sme Qt zo 4 na 5 a tiež sme projekt rozbehali s takmer najnovším OSG. Pri Qt bolo ale potrebné samostatne si stiahnuť a zostaviť modul osgQt, kt. už vo verzii 5 nie je distibuovaný spolu s OSG. Modul bol neskôr pridaný ako závislosť do repozitára projektu.

Pri počiatočnom testovaní sme sa aj na systéme macOS často stretávali s problémami ohľadom dostupnosti špecifických verzií závislostí.

Zaujímavé v tomto kontexte je hlavne porovnanie systémov macOS s Debian-based distibúciami Linuxu. Oba podporujú správu softvérových balíkov pomocou package management systémov. Repozitáre apt-u sú však centrálnie regulované a často ich správcovia pri vydaní novej verzie balíka chvíľu počkajú, kým ju začlenia do ekosystému package managera (či už z obozretnosti pred nespoľahlivosťou alebo nedostatočnej podpore).

V ekosystéme Homebrew na macOS balíky nie sú až tak striktne regulované, tvorcovia zvolili prístup "bleeding edge" - nové verzie sú často začlenené do package managera hned po ich vydaní aj napriek riziku nespoľahlivosti, nedostatočnej podpore či nepripravenosti vývojárov na aktualizáciu. Problém hlavne nastáva, nová verzia nahradí starú a tá už nie je priamo dostupná v package management systéme.

To presne často zaskočilo aj integrátorov na systémoch macOS. Homebrew napríklad počas semestra aktualizoval verziu OpenSceneGraph-u na 3.5.7. 3DSoftviz ale zatiaľ funguje s verziou 3.5.6, ktorá už nie je v Homebrew dostupná. Preto boli niektorí z nás nútení závislosť následne sami získať z oficiálneho git repozitára v správnej verzii a zbuildovať.

Z tohto hľadiska je macOS pri našom projekte ľažšie udržiavateľný ako linuxové distribúcie s apt-om. Komunita apt-u navyše disponuje veľkým množstvom Personal package archives (PPA) - súkromných repozitárov, kde sú často balíky aj vo verziach, ktoré už nie je možné získať z oficiálnych zdrojov.

4.1.2 Linux

Projekt už bol na linuxe funkčný, našou úlohou bolo regresne otestovať projekt s novými verziami knižníc. Konkrétnie OpenSceneGraph 3.5.6 a Qt 5.9.1. Prechod z Qt4 na Qt5 si vyžiadal úpravy v programe pretože knižnica nie je späť kompatibilná. OpenSceneGraph vo verzii 3.5.6 oddelil osgQt do samostatného repozitára, čo si vyžadovalo úpravu v CmakeList.txt.

Po týchto úpravách je projekt bez problémov spustiteľný.

4.1.3 WSL

Kedže pri buildovaní projektu na platorme Windows vznikali viaceré problémy rozhodli sme sa otestovať novú funkcionality systému - Windows Subsystem for Linux. Tento modul umožnuje za behu systému Windows spustiť systém Linux, konkrétnie Ubuntu vo verzii 16.04. Tento subsystém je priamo súčasťou OS Windows a nefunguje ako virtuálny stroj. Po nainštalovaní je k dispozícii terminál, cez ktorý je celý systém ovládaný. Tento systém

podporuje ovládanie len z terminálu, avšak výstup sa dá exportovať na lokálny server, takže s grafickými aplikáciami nie je problém.

Pri inštalovaní a buildovaní projektu sa prišli na nedostatky v podobe podpory oficiálnych závislostí, ktoré projekt 3D softvizi potrebuje. Ako tím sme sa dohodli, že projekt aktualizujeme na čo najnovšie knižnice, no WSL malo v repozitári len staršie verzie. Jednou z dôležitých knižník bola knižnica Qt vo verzii 5, ktorá bola kvôli tomu inštalovalaná z externých úložísk.

Projekt sa napokon cez tento systém podaril rozbehnuť a aj nainštalovať. Overením tejto funkcionality sme pridali ďalšiu možnosť inštalácie projektu na operačnom systéme Windows, čo výrazne ušetrí čas, pretože inštalovanie 3DSoftviz na tejto platforme je náchylnejšie na chyby.

4.1.4 Windows

Projekt na windows sme skúšali rozbehať s rôznymi verziami závislostí a knižníc. Najprv sme postupovali podľa existujúceho inštalačného manuálu, s MSVC 12 (Microsoft Visual Studio 2013) a s aktualizovanými verziami knižníc, pričom všetky komponenty sme mali 32 bitové. Projekt sa nám podarilo úspešne buildnúť a nainštalovať, ale pri spustení projektu vyskytla chyba (chybová hláška: Error: OpenGL version test failed, requires valid graphics context., FATAL [default] CRASH HANDLED; Application has crashed due to [SIGSEGV] signal). Na fórách sme našli potenciálne riešenia na problém, medzi ktorými bola aj zlá verzia Qt, sme ich vyskúšali, ale stále sme sa dostali ku tej istej chybe.

Následne sme prešli na MSCV 14 (Microsoft Visual Studio 2015), pričom knižnice a závislosti sme nechali 32 bitové. Projekt sme úspešne buildli a nainštalovali ale taktiež sa vyskytla chyba pri štarte (The procedure entry point ?defaultConnection@QSqlDatabase@@2PBDB could not be located in the dynamic link library C:\Timak\3dsoftviz_install\bin\3DSoftviz.exe).

Nakoniec s MSVC 14.11 (Microsoft Visual Studio 2017) sa nám podarilo rozbehať projekt. K úspešnému rozbehaniu projektu sme potrebovali aktualizovať závislosti na nasledujúce verzie:

- Microsoft Visual C++ (MSVC) 14.11 - Microsoft Visual Studio 2017
- CMake 3.9.5
- OpenSceneGraph 3.4.1
- Qt 5.9.2
- OpenCV 2.4.13.4
- Boost 1.65.1

Na Windows na prácu s projektom v predošlých rokoch používali 32 bitové verzie závislostí. Kedže pri buildovaní projektu podľa starého inštalačného manuálu pre platformu Windows sme mali viac problémov, ktoré neboli spomenuté v inštalačnom manuáli a Microsoft Visual Studio 2013 už nie je voľne dostupný pre študentov fakulty, sme sa rozhodli aktualizovať všetky závislosti na 64 bitové verzie a Microsoft Visual Studio na najnovšiu verziu (2017).

OpenSceneGraph sme našli prebuildovaný s MSVC 14.11 na internete, ale k OpenCV sme našli len zdrojové súbory. Existujúce prebuildované verzie neboli kompatibilné s ostatnými závislosťami. OpenCV sme teda museli buildnúť. Na buildnutú verziu sme potom dali odkaz na stiahnutie do inštalačného manuálu, aby to nemusel každý buildovať.

Qt sme tiež aktualizovali z verzie 4 na 5. QtCreator už sme nemuseli zvlášť inštalovať, pretože na rozdiel od predošlých tímov Qt sme inštalovali cez online inštalátor, ktorý automaticky nainštaluje aj najnovšiu verziu QtCreatora. QtCreator sme teda tiež aktualizovali na aktuálnu najnovšiu verziu.

Problémy boli aj so submodulom libnoise, v ktorom sme tiež museli urobiť niekoľko zmien kvôli novej verzii MSVC. Tak isto sme museli aktualizovať aj moduly Leap a Leap-Orion na najnovšie 64 bitové verzie zo staršej 32 bitovej. K úspešnému buildu a spusteniu projektu sme museli urobiť ešte zmenu v súbore \src\Clustering\Figures\Sphere.cpp, tiež kvôli novej verzii MSVC. Poslednú zmenu sme museli urobiť v súbore \resources\scripts\app\main.lua, kde sme museli zakomentovať prvý riadok.

Projekt na platforme Windows sa nám nakoniec podarilo rozbehať, ale bolo to časovo náročné. V našom tíme primárne sa bude pracovať na platformách Linux a MacOS a Windows bude slúžiť len ako testovacia platforma, pretože aj pri doterajších zmenách chyby sa vyskytli väčšinou na Windows.

4.2 RefaktORIZÁCIA Cmake

Hlavný CMakeLitst.txt súbor je veľký a pomerne neprehľadný, primárne sme sa snažili dostať buildovanie externých dependencii do samostatného CMakeLists súboru. Tento cieľ sa podarilo splniť, no počas vykonávania úlohy sa ukázalo že bude potrebné zjednodušiť aj iné časti hlavného cmake súboru a teda po splnení jednej úlohy nám pribudli ďalšie, a však tie by po dokončení mali ešte viac zjednodušiť prácu s cmake súbormi.

Prenesenie buildovania externých dependencii do samotného cmake súboru sme vykonali pomocou cmake príkazu: `add_subdirectory(dependencies)`, kde parameter predstavuje adresár vrámcí hlavnej úrovne projektu. V adresáre sú všetky externe dependencie spolu so svojimi cmake súbormi. Do tohto adresára sa pridal nový CMakeLists.txt súbor kde sú cmake príkazy potrebné pre buildovanie všetkých externých závislostí. Výsledkom je jednoduchšia možnosť práce s buildovaním externých projektov.

Ďalej sme pracovali na presunutí priradovania zdrojových a hlavičkových súborov, tie sa do hlavného CMakeLists.txt súboru zapracovali pomocou príkazov add_subdirectory(src) a include(include/Headers.cmake). Vytvorili sme teda nový CMakeLists.txt súbor v podadresári src/ a nový Headers.cmake súbor v podadresári include/ kde sa presunuli potrebné cmake skripty z hlavného súboru. Taktiež sa vytvoril CMakeLists.txt súbor v podadresári tests/ ktorý sa zapracuje do hlavného súboru už spomínaným príkazom add_subdirectory(tests).
V rámci refaktORIZÁCIE sme sa rozhodli zapracovať do procesu buildovania aj niekoľko nových prvkov, ide o pridanie git informácií ako aktuálna vetva, posledný commit a pod. ktoré sa majú zobrazovať počas procesu buildovania. Ďalej sme pridali možnosť používania nástroja ccache ktorý má za úlohu zrýchliť proces buildovania.

4.3 OpenPose

4.3.1 Úvod

Detekcia prstov na ruke je jedna z najdôležitejších častí projektu a jej správnosť a presnosť predstavuje kľúč k úspechu celého projektu. Rôzne tvary a veľkosti prstov, množina uhlov, v ktorých sa prsty môžu nachádzať a ďalšie faktory znamenajú potrebu implementácie komplexného riešenia, vlastného alebo existujúceho.

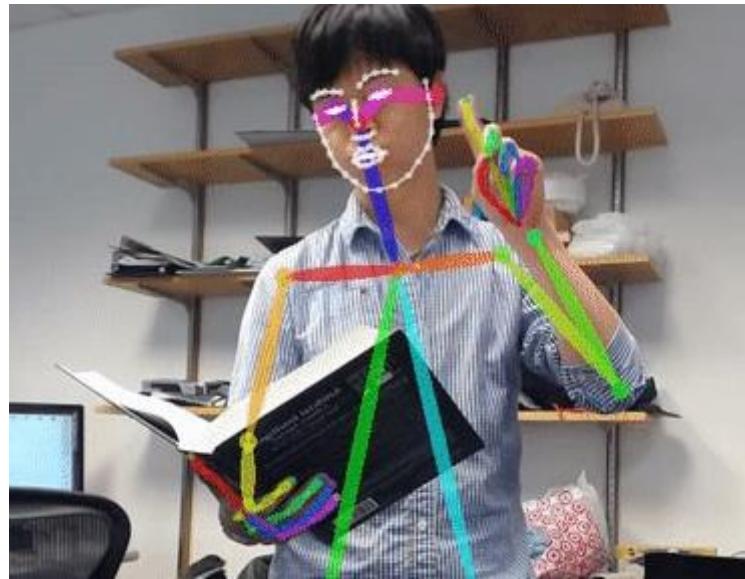
Existencia knižníc s požadovanou funkcionality nás môže viesť k predpokladu, že integrácia takejto knižnice/produkту, do nami vyvájaného systému môže byť efektívnejšia, ako do času tak aj do výsledku, ako vytváranie vlastného riešenia. Rôznorodosť riešení ale so sebou nesie aj rozdiel v použitých technológiách a obmedzeniach s tým súvisiacimi, náročnosti integrácie a požiadavkách na výpočtový výkon, či rozdiel medzi deklarovanými a skutočnými vlastnosťami.

To je dôvod, prečo je každú knižnicu potrebné najskôr analyzovať, otestovať a až potom pristúpiť k jej integrácii do aplikácie resp. zvolenie si cesty vlastného riešenia. Čas, ktorý sa spotrebuje takouto analýzou je rozhodne menší ako ten, ktorý sa minie pri snahe použiť niečo, čo nebude napĺňať naše potreby

4.3.2 Analýza

OpenPose je nová knižnica pre detekciu kľúčových bodov tela, ruky a tváre v reálnom čase, napísaná v jazyku C++, využívajúca technológie CUDA, OpenCV a Caffe. Jej vznik sa datuje k aprílu tohto roka (2017), čo neprispieva k množstvu informácií, ktoré o nej je možné získať. Jej väčšina je sústredená priamo na GitHube¹, kde sa aj riešia problémy pri inštalácii, nápady na rozšírenie použitia resp. otázky k vývoju v budúcnosti.

¹ <https://github.com/CMU-Perceptual-Computing-Lab/openpose>



Obrázok 4: Detekcia kľúčových bodov tela pri jednej osobe²



Obrázok 5: Detekcia kľúčových bodov tela pri dvoch osobách³

Verzia 1.0.0 vyšla len 8. júla 2017. Súčasná verzia je 1.2.0 a hoci je písaní chystaní update na verziu 1.2.1 žiadne bližšie informácie o dátume alebo obsahu tohto updatu nie sú k dispozícii⁴. Je dostupná zadarmo(pri nekomerčnom použití). Podľa dostupných informácií môže byť na ako zdroj na vstup obrázok, video, webkamera, či IP kamera alebo Kinect⁵, pri

² <http://www.cs.cmu.edu/~yaser/>

³ <http://www.researchcareer.com.au/archived-news/code-released-for-better-body-tracking>

⁴ https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/release_notes.md

⁵ https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/demo_overview.md

ktorom niektorí používatelia riešili aj použitie ako obyčajnej webkamery⁶. Výstupom je grafická obrazovka, kľúčové body identifikované na vstupnom obrázku/videu vo formáte JSON, XML a alebo renderované video/obrázok už s kľúčovými bodmi vo formáte JPG, PNG, AVI.⁷

Knižnica OpenPose využíva technológiu CUDA, kvôli uplatňovaniu hlbokého učenia. Táto technológia je vlastnená spoločnosťou Nvidia, a je dodávaná výhradne ku grafickým akcelerátorom tejto spoločnosti. Nie je v pláne pridanie podpory pre konkurenčné AMD (FireStream)⁸, čo predstavuje určité obmedzenie v oblasti hardvéru potrebného k práci. Tiež podpora medzi operačnými systémami je limitovaná na Windows 8 a 10, Ubuntu 14 a 16 a Nvidia Jetson TX2. Hoci tvorcovia uvádzajú, že knižnica bola použitá aj na iných OS ako napr. macOS, CentOS, oficiálne podporované nie sú⁹. Výhodou ale môže byť, že pre spomínané operačné systémy existujú dobré návody obsahujúce všetky kroky, ktoré treba vykonať pre plnohodnotné využívanie funkcií knižnice. Pre platformu Windows tiež existuje aj demo. Je vhodné spomenúť, že dokumentácia je automaticky generovaná cez doxygen, čo je vhodné vzhľadom na rovnaký spôsob generovania aj nášho projektu

Na stránke projektu je dostupných pomerne veľa informácií¹⁰, ale vzhľadom na dĺžku existencie projektu, informácií z iných zdrojov je veľmi málo, čo prispieva k argumentu, že bez vyskúšania funkcionality nie je možné urobiť kvalifikované rozhodnutie.

Autori deklarujú, že v prípade nájdenia videa, kde ich algoritmus nefunguje dobre sú ochotní ho upraviť/vylepšiť a zároveň prosia o nahlásenia chýb alebo pridanie vlastných vylepšení, ktoré by mali význam pre ostatných používateľov. Veľkým problémom, ktorí sa snažia niektorí používatelia zmierniť vlastnými úpravami je rýchlosť resp. výpočtová náročnosť, no napriek pokusom o jej zlepšenie, ak je strata presnosti príliš veľká, napr. rovnaká ako nárast rýchlosťi, autori takéto riešenie neimplementujú, hoci niektorým by aj nižšia presnosť pri zvýšení rýchlosťi mohla vyhovovať.

Autori deklarujú, že výpočtový výkon potrebný pre detekciu kľúčových bodov je rovnaký bez ohľadu na počet detekovaných ľudí z obrazu.

4.4 Vagrant

Z dôvodu zložitosti nainštalovania všetkých závislostí pre build 3dsoftvizu sme sa rozhodli použiť Vagrant na vytvorenie vývojového prostredia.

⁶ <https://github.com/CMU-Perceptual-Computing-Lab/openpose/issues/189>

⁷ <https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/output.md>

⁸ <https://github.com/CMU-Perceptual-Computing-Lab/openpose/issues/45>

⁹ <https://github.com/CMU-Perceptual-Computing-Lab/openpose/blob/master/doc/installation.md#operating-systems>

¹⁰ <https://github.com/CMU-Perceptual-Computing-Lab/openpose/tree/master/doc>

Vagrant je softvér, ktorý umožňuje vytvorenie a údržbu prenosnej VM. Vstup pre vagrant je Vagrantfile, ktorý obsahuje informácie:

- Aký VM nástroj použiť(VMware,Virtualbox atď.)
- Koľko HW pridelí virtuálnemu stroju
- Aký provisioning má pri prvom spustení vykonať
- Aký OS má použiť

VM sa spustí príkazom: \$ vagrant up

Prebehne provisioning a používateľ má k dispozícii funkčné vývojové prostredie.

4.5 Provisioning

Provisioning je príprava prostredia, zahŕňa inštaláciu správnych verzií knižníc potrebných pre build 3dsoftvizu. Na provisioning využívame Ansible – program, ktorý vykoná provisioning.

Príklad ansible syntaxe pre nainštalovanie gitu pre systém, ktorý využíva APT package manager:

```
- name: install git  
  apt: name=git state=present
```

Name - názov úlohy

Apt - zadefinuje, že sa má na inštaláciu použiť apt package manager

Name = git - názov balíku, s ktorým sa bude manipulovať

State = present – balík má byť nainštalovaný po dokončení príkazu

Znalosti získané pri vytváraní provisioningu cez ansible plánujeme využiť pri CI.

4.6 Continuous integration

S plánovanou migráciou z githubu na gitlab sme plánovali zaviesť aj CI. Idea bola taká, že sa využije .yml súbor využitý pri provisioningu, ktorý sa rozšíri o príkazy na build. Týmto súborom vytvoríme docker image. Pokiaľ celý proces prebehne bez chyby, tak je program skompilovateľný a môže prebehnúť merge branche na gite.

Pri riešení CI sme narazili na problém, keď gitlab CI runner stále nepodporuje spustenie testu iba pri merge branche, ale iba pri každom commite. Čo nie je pre náš projekt vhodné, pretože provisioning + build trvá cca 30-40 minút.

Čiže CI je pripravené a otestované lokálne. Čaká sa migráciu projektu na gitlab a taktiež nato kým gitlab implementuje, potrebnú funkcionality do ich CI runneru. Čo by malo byť už čoskoro. Pokiaľ sa tak nestane tak budeme uvažovať nad workaroundsom.

5 Letný semester

5.1 Include What You Use (IWYU)

V začiatocnej fáze rozčleňovania projektu je potrebné upraviť includnuté knižnice v triedach. Snažili sme sa o voľnejšiu previazanosť – zníži sa pravdepodobnosť, že trieda je závislá na knižničiach, ktoré nie sú includované priamo ťaou. Môžeme odhaliť cyklické závislosti a bude ľahšie existujúce triedy rozdeľovať či presúvať.

Projekt pri zostavovaní používa nástroj Cotire¹¹ (optional). Ten urýchľuje proces zostavenia pomocou techník *Single Compilation Unit* a *Precompiled Header* – počas zostavenia spája všetky triedy do jedného súboru, ktorý je následne skompilovaný rýchlejšie. To ale tiež znamená, že v triedach môžu chýbať potrebné includnuté knižnice a binárku bolo možné zostaviť len vďaka „zlepreniu“ všetkých tried dokopy Cotire-om (vždy sa našla aspoň jedna ďalšia trieda, ktorá obsahovala chýbajúci include).

Na analýzu a následnú úpravu include-ov sme použili nástroj Include What You Use (IWYU)¹². Ten analyzuje jednotlivé symboly (dátové typy, funkcie či makrá) v .cpp/.h súbore a kontroluje, či sú includnuté headre, ktoré symboly poskytujú. Vie odhaliť chýbajúce includy, no tiež zbytočne includnuté headre.

Nástroj je vo verzii alpha – úpravu include-ov nebolo možné automatizovať a prechádzali sme väčšinu tried projektu manuálne. Nástroj poskytuje aj možnosť zadefinovania pragiem¹³, ktorými je možné upraviť výsledky jeho analýzy – možno označiť includy, ktoré označil ako nepotrebné za nutné a naopak.

```
/Users/peto.marusin/VizReal/sources/LeapLib/include/LeapLib/LeapActions.h should add these lines:  
namespace Leap { struct Vector; }  
namespace LeapLib { class LeapManager; }  
  
/Users/peto.marusin/VizReal/sources/LeapLib/include/LeapLib/LeapActions.h should remove these lines:  
- #include "LeapLib/LeapManager.h" // lines 5-5  
  
The full include-list for /Users/peto.marusin/VizReal/sources/LeapLib/include/LeapLib/LeapActions.h:  
#include <Leap.h> // for Gesture, Hand  
#include "LeapLib/DirectionDetector.h" // for DirectionDetector, DirectionD...  
#include "LeapLib/LeapExport.h" // for LEAPLIB_EXPORT  
namespace Leap { struct Vector; }  
namespace LeapLib { class LeapManager; }  
---
```

Obrázok 6 Ukážka výstupu IWYU pre LeapActions.h

IWYU vo verzii 0.8 bol testovaný na macOS a Linuxu. Boli problémy na macOS 10.13 (High Sierra) s Clangom (asi zapríčinené Apple verziou Clangu). Stále však bolo možné IWYU používať.

Inšpekcii includov sme okrem uvoľnenia previazanosti začali s úmyslom zmenšenia ich celkového počtu (urýchlenie zostavenia). To sa nám však asi nepodarilo. Zmeny v develop

¹¹ <https://github.com/sakra/cotire>

¹² <https://github.com/include-what-you-use/include-what-you-use>

¹³ <https://github.com/include-what-you-use/include-what-you-use/blob/master/docs/IWYUPragmas.md>

vetve činili 792 nových a 550 odstránených riadkov kódu. Aj s includmi navyše sú však triedy menej previazané. Najlepšie by bolo robiť inšpekciu include-ov 3DSoftizu aj naďalej počas celého vývoja. Nástroj je možné aktivovať prepínačom v hlavnom CMake-u projektu. Pri jeho použití musí byť vypnutý Cotire.

5.2 Migrácia na GitLab a synchronizácia

V prvej tretine semestra prebehla migrácia hlavného repozitára z GitHub-u na GitLab (príprava na nasadenie CI). Výhodou GitLab-u sú tiež privátne repozitáre aj pri free účte – je možné rozšíriť kód o časti, pri ktorých si vlastník nepraje ich zverejnenie.

Tímový projekt, diplomanti aj bakalári majú separátne repozitáre. Synchronizáciu zmien sme riešili vytvorením synchronizačnej vetvy v každom repozitári. Synchronizácia prebiehala raz týždenne (piatok). Dovtedy sme sa snažili vyriešiť všetky čakajúce pull-requesty. Následne sme zosynchronizovali synchronizačnú vetvu s aktuálnou verzou develop vetvy a vytvorili pull request *sync vetva TP -> sync vetva DP/BP*. Synchronizáciu je možné robiť obojsmerne. Sync vetva by mala po synchronizácii obsahovať rovnaké commity vo všetkých repozitároch, ktoré pracujú na 3DSoftvize, čo aj platilo. Tento spôsob synchronizácie sa v praxi celkom osvedčil.

5.3 RefaktORIZÁCIA logovania projektu

Pri analýze refaktORIZÁCIE sme zistili, že vývojári na projekte 3DSoftviz nedodržovali konvencie a metodiku správneho logovania. Navyše na logovanie využívali rôzne metódy z rôznych knižníc a niekedy logy nedávali zmysel a boli tu zanechané len ako pomocné výpisy.

Preto sme pristúpili na prepísanie všetkých logovacích správ (ich počet bol na jednotný formát podľa metodiky a využili sme pri tom jednotne knižnicu *easylogging*, ktorá je dostupná pre C++. Tie, ktoré sa ukázali ako nepotrebné pre projekt a potreby odhaľovania chýb sme zmazali.

Pri logovaní teraz využívame všeobecnú štruktúru:

```
LOG( WARNING/ERROR/FATAL ) << "MENO_BALIKU/MENO_TRIEDY/MENO_FUNKCIE(PARAMETRE)"
```

5.4 RefaktORIZÁCIA modulu Data

Hlavným cieľom zrefaktorovania modulu Data bude jeho oddelenie do samostatnej knižnice ktorá nemá žiadne závislosti na zvyšok 3Dsoftviz, ani na externé knižnice. Výsledkom bude tiež zlepšená hierarchia tried a odstránenie nepotrebných súčastí modulu Data.

RefaktORIZÁCIA bude prebiehať vo viacerých fázach. V prvej fáze sa vytvorí hierarchia tried pre entitu Graph. Táto hierarchia zníži komplexnosť zdrojového kódu a umožní väčšiu modularitu celkového projektu.

V ďalšej fáze sa budeme zaoberať odstraňovaním interných závislostí na iné triedy v rámci projektu 3Dsoftviz, ako aj externých knižníc. Pri analýze sme zistili, že niektoré metódy používali smart pointer z knižnice OSG, pričom nevykonávali žiadnu logiku spojenú s touto knižnicou. Tento pointer sa dá nahradí jedným zo smart pointrov, ktoré sa nachádzajú v balíku std.

V ďalšej fáze chceme odstrániť závislosť modulu Data na DAO triedach. Tieto činnosti by sa mali vykonávať v inom moduli, keďže slúžia na načítanie a ukladanie dát do databázy. Našim cieľom bude ich presunutie do namespacu GraphManager.

V poslednej fáze sa budeme zaoberať s odstránením zbytočnej previazanosti entity Type na triedu ApplicationConfig, cez ktorú číta nastavenia. Tento nedostatok napravíme tak, že všetky nastavenie budú posielat v konštruktore pri vytváraní nových inštancií tejto entity.

5.5 Analýza knižnice sol2

Knižnica Sol2 slúži rovnako ako Diluculum na získavanie dát z Lua scriptov, tzv. binding pre C++. Ide go-to framework, s minimálne podľa dokumentácie, ľahko použiteľným API.

V podstate sa jedná o header-only knižnicu. Je zabezpečená podpora Lua 5.1, 5.2 a 5.3 a LuajIT (Just-In-Time-Compiler for Lua) a tiež C++14. Z Githubu projektu je zrejmé, že na projekte sa stále pracuje, ale informácie o budúcich verziách tam nie sú. Poskytuje tutoriál, ktorý obsahuje predovšetkým príklady z krátkych kódov s minimom textu. Viac je možné zistiť až z dokumentácie, ktorá by mohla byť napísaná aj prehľadnejšie.

Sol2 podporuje základné konštrukcie ako je podpora stavov, tabuliek, nastavovanie resp. získavanie hodnôt rôznych typov. Tabuľka predstavuje hlavný bod interakcie medzi jazykom Lua a C++. K dispozícii sú dva druhy tabuliek, globálne a neglobálne, no obe poskytujú rovnaký interface a všetky globálne tabuľky sú konvertovateľné na neglobálne. Je k dispozícii aj operator[] proxy.

Sol2 dáva k dispozícii aj CMake script, avšak dokumentácia uvádza, že je primárne určený na testovacie účely. Pri problémoch so CMake súborom nie je poskytovaná plná podpora, nakoľko ide o výtvor dobrovoľníka.

Vytváranie objektov funguje cez deklaráciu sol2:: . Volanie funkcií cez protected_function slúži nielen na odchytávanie errorov a ale aj ich spracovanie. Podľa dokumentácie akýkoľvek C++ alebo Lua error by mal byť odchytaný a spustený cez voliteľný error_handler.

Stránka knižnice obsahuje aj informácie o možných eroroch, ktoré sa pri jej používaní môžu vyskytnúť, uvádza problém s LuajIT na Mac OS X, problém s Visual Studio množstvo ďalších.

5.6 RefaktORIZÁCIA modulu LeapLib

Hlavným cieľom pri refaktORIZácii tohto modulu bolo odstrániť z neho všetky nepotrebné závislosti a tým umožniť jeho osamostatnenie od 3dSoftvizu. Momentálne je tento modul používaný už ako samostatná knižnica s vlastným namespace bez závislosti na 3dSoftvize. Pri oddelení tejto novo vzniknutej knižnice bolo potrebné vytvoriť v Softviz::Leap balíku novú tiredu Listener, ktorá dedí z triedy LeapLib::LeapListener, pretože nebolo možné odstrániť všetky závislosti z pôvodnej LeapLib a tak bolo najvhodnejším riešením niektoré časti rozdeliť do viacerých vrstiev.

5.7 RefaktORIZÁCIA modulu GitLib a ParserLib

Mobil GitLib bol oddelený od časti Repository/Git do samostatnej knižnice bez akýchkoľvek závislosti na 3dSoftvize, spolu s ním boli presunuté aj jeho testy.

Modul Importer::Parsing bol oddelený do samostatnej knižnice, čím sa opäť podarilo o čosi zmeniť veľkosť samotného 3dSoftvizu. Táto novo vzniknutá knižnica má na starosť parsovanie zdrojových súborov jazyka Java. Do tejto knižnice linkujeme externú knižnicu *parserlib*, výsledná knižnica je používaná samotným 3dSoftvizom.

5.8 RefaktORIZÁCIA modulov LuaGraph a LualInterface

Cieľom bolo oddeliť zdrojové kódy týkajúce sa tried LuaGraph a LualInterface do samostatných znovupoužiteľných celkov, bez spätných závislostí na hlavnú aplikáciu či iné nesúvisiace moduly. Z oboch modulov vznikli samostatné verziované repozitáre, ktoré sú zahrnuté do projektu ako git submoduly. Sú plne samostatne zostaviteľné a inštalovateľné, no hlavne schopné zahrnutia do akéhokoľvek iného projektu. Takisto sú pripravené na pokrytie testami. Všetky závislosti projektu sú vyriešené možnosťou vyhľadania v súborovom systéme alebo samostatným zostavením.

6 Prílohy

6.1 Inštalačný manuál

6.1.1 Linux

Názvy knižníc sú ukázané pre Ubuntu 17.10

1. Cez Apt nainštalovať nasledujúce programy
 - Git - git
 - OpenSceneGraph - libopenscenegraph-dev //update na verziu 3.4.0
 - Qt5 - qt5* // update na qt 5.9.1

- Qt5WebEngine - qtwebengine5
- QtCreator - qtcreator
- Boost - libboost-all-dev
- Lua - liblua5.1-0-dev
- OpenCV - libopencv-dev
- FreeGlut3 - freeglut3-dev
- CMake - cmake // verzia 3.9

Program je otestovaný aj s OpenSceneGraph verziou 3.5.6 , ale tá nie je momentálne v Ubuntu repozitári a ani v PPA, čiže treba build zo source kódu

Príkaz na inštaláciu závislostí : *\$ sudo apt install git,libopenscenegraph-dev,qt5*,qtwebengine5,qtcreator,libboost-all-dev,liblua5.1-0-dev,libopencv-dev,freeglut3-dev,cmake*

1. Klonovať cez Git projekt 3dsoftviz
 - git clone ** url repozitára **
 - git submodule update --init --recursive
2. Zbuildovanie projektu
 - Otvoriť QtCreator
 - File >> Open file or project >> Otvoriť CmakeList.txt v zložke 3dsoftviz
 - Vytvoriť projekt
 - Ak úspešne (dá sa kliknúť na FINISH) >> Kliknúť FINISH
 - Ak neúspešne – Nájsť chybu vo výpise
3. Nastaviť build v QtCreator-i
4. Projects >> Build and Run >> Build Build Settings >> Add >> Clone Selected >> pomenovať “unity” - automaticky prepne na unity build mode Build Steps >> Details >> zaškrtnúť install_unity
5. Klonovať cez Git projekt 3dsoftviz (AlphaReach klonuje z repozitára Cimox)
 - git clone ** url repozitára **
 - git submodule update --init --recursive
6. Nastavenie počtu jadier na buildovanie projektu... Details >> Additional arguments “-jN”, kde N reprezentuje počet VIRTUÁLNYCH jadier

6.1.2 Windows Subsystem for Linux (WSL)

Inštalácia WSL

Note:

Inštalácia WSL je možná len pre 64-bitovú verziu Windows 10 a build 1607+

1. Spustiť **PowerShell** ako **administrátor** a spustiť nasledujúci príkaz:
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux

2. Spustiť **CMD** a spustiť:

bash

3. Potvrdiť inštaláciu a nastaviť meno a heslo

4. WSL nepodporuje defaultne GUI aplikácie, no ich obraz sa dá exportovať na XServer:

- nainštalovať Xlaunch (alebo iný)
- v nastaveniach nastaviť port výstupného displeja
- v terminály zadať export DISPLAY=:port (ten ktorý sme nastavili v Xlaunch)

Inštalácia projektu

1. Nainštalovať prostredníctvom Synaptic nasledujúce programy

- **Git** - git
- **OpenSceneGraph** - libopenscenegraph-dev
- **Boost** - libboost-all-dev
- **Lua** - liblua5.1-0-dev
- **OpenCV** - libopencv-dev
- **FreeGlut3**
 - freeglut3
 - freeglut3-dev
- **CMake** [3.5.0](#)
 - inštalovať zo source files, nie Synaptic

2. Nainštalovať Qt5 cez ppa (cez Synaptic je dostupná len verzia 4):

`sudo add-apt-repository ppa:beineri/opt-qt591-xenial`

`sudo apt-get update`

`sudo apt-get install qt59-meta-full`

3. Pre správne fungovanie je potrebné nastaviť flag pre spustiteľnosť knižníc Qt príkazom:

- `sudo execstack -c /opt/qt59/lib/*`
- v súčasnej verzii execstack nie je dostupný, treba ho najskôr nainštalovať

4. Klonovať cez Git projekt 3dsoftvizQt

- `git clone ** url repozitára **`
- `git submodule update --init --recursive`

5. Prepnúť sa do aktuálnej vetvy:

- `git checkout vetva`
- `git submodule update --init --recursive`

6. Zbuildovanie projektu

- Otvoriť QtCreator príkazom:

- ```
sudo ./opt/qt59/bin/qtcreator
```
- V QtCreator File >> Open file or project >> Otvoriť CmakeList.txt v zložke 3dsoftviz
  - Vytvoriť projekt
    - Ak úspešne (dá sa kliknúť na FINISH) >> Kliknúť FINISH
    - Ak neúspešne – Nájsť chybu vo výpise

## 7. Nastaviť build v QtCreator-i

Projects >> Build and Run >> Build Build Settings >> Add >> Clone Selected >> pomenovať “unity” - automaticky prepne na unity build mode Build Steps >> Details >> zaškrtnúť install\_unity

Nastavenie počtu jadier na buildovanie projektu... Details >> Additional arguments “-jN”, kde N reprezentuje počet VIRTUÁLNYCH jadier

8. V Tools >> Options >> Build & Run >> QtVersions je potrebné cez Add pridať /opt/qt59/bin/qmake
9. V Tools >> Options >> Build & Run >> Kits je potrebné nastaviť verziu Qt5.9..
10. Spustiť buildovanie
11. Po buildovaní v Projects >> Run >> Add >> Custom Executable nastaviť Executable: {project\_dir}/\_install/bin/3DSoftviz  
Working directory: {project\_dir}/\_install/bin
12. Spustiť projekt

### 6.1.3 macOS (vytvárané a testované pre verziu 10.13 High Sierra)

1. Inštalácia Homebrew príkazom:

```
/usr/bin/ruby -e "$(curl -fsSL
\https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

2. Nainštalovať git:

```
brew install git
```

3. Nainštalovať OpenCV - MUSÍ byť verzia 2.x, nie 3 a vyššia! (po inštalácii skontrolovať príkazom brew info opencv@2, vyhovujúca verzia je napr. 2.4.13.4):

```
brew install opencv@2
```

4. Vytvoriť symbolické linky pre opencv@2:

```
brew link opencv@2 --force
```

5. Nainštalovať OpenNI2:

- možné pomocou Homebrew nasledovne:

- brew tap homebrew/sciencebrew tap totakke/openni2brew install openni2
- dostupné [na stiahnutie v Google Drive](#)

6. Nainštalovať Boost:

```
brew install boost
```

7. Nainštalovať CMake:

```
brew install cmake
```

8. Nainštalovať Qt (preferovaná verzia 5.9.2):

```
brew install qt
```

9. Vytvoriť symbolické linky pre Qt:

```
brew link qt --force
```

8. skontrolovať verziu a symbolické linky pre Qt príkazom: qmake --version. Ak je výstup command not found, skontrolovať v Homebrew: brew info qt. Ak je Qt nainštalované, no nevieme nájsť qmake, Homebrew nevytvoril správne symbolické linky. Skúsiť brew unlink qt a znova brew link qt --force, prípadne reinstall

9. Nainštalovať sphinx:

```
pip install sphinx
```

alebo

```
brew install sphinx
```

10. Nainštalovať astyle:

```
brew install astyle
```

11. Nainštalovať pcl:

```
brew install pcl
```

12. Nainštalovať doxygen:

```
brew install doxygen
```

13. Nainštalovať cppcheck:

```
brew install cppcheck
```

14. Nainštalovať ovládače pre 3D myš dostupné na odkaze alebo v Google Drive projektu.

15. Nainštalovať Qt Creator:

```
brew cask install qt-creator
```

16. Naklonovať projekt:

```
git clone https://github.com/BergiSK/3dsoftviz --recursive
```

17. Otvoriť Terminál, cd do priečinka ~/3dsoftviz a aktualizovať submoduly:

```
git submodule update --recursive
```

18. Otvorit' QtCreator a otvorit' CmakeLists.txt z priečinka, kde bol naklonovaný projekt.

## Nastavenie Qt Creatora:

- cmd + , pre otvorenie nastavení
- skontrolovať záložku Qt Versions, ak je dostupné Qt 5.x, nechať tak. Ak nie, treba nájsť binárku qmake (defaultne po inštalácii Qt cez Homebrew v /usr/local/bin/).
- skontrolovať záložku CMake. Ak nie je detekovaný žiadny, treba nájsť binárku cmake (defaultne po inštalácii CMake cez Homebrew v /usr/local/bin/).
- ísť do zákožky Kits.
- Zvoliť konfiguráciu a vpravo kliknúť na Clone

- Klinúť na novú konfiguráciu (dole sa otvorí editačné okno)
- Name zvoliť napr. unity
- C aj C++ Compiler zvoliť Clang
- Qt version zvoliť Qt 5.x
- CMake Tool zvoliť na CMake zo záložky CMake
- Vpravo hore klikúť na Make Default
- Potvrdiť tlačidlom OK

### **Build & Run konfigurácia:**

- V ľavom menu Qt Creatora kliknúť na Projects
- Kliknúť na Build pri unity
- Edit build configuration zvoliť na Debug
- Build directory zvoliť na ~/3dsoftviz/build
- Pri položke Build steps kliknúť na Details a zaškrtnúť install\_unity
- Kliknutím na Build project (kladivo) projekt zbuildovať
- Naľavo kliknúť na Run pri unity
- Pri Run configuration rozbalíť Add menu a zvoliť Custom Executable
- Zvoliť binárku 3dsoftviz/\_install/bin/3DSoftviz
- Working directory nastaviť na 3dsoftviz/\_install/bin

### **Známe problémy, riešenia a rady**

#### **Q: Build error "/usr/local//mkspecs/macx-clang"**

CMake Error at /usr/local/lib/cmake/Qt5Core/Qt5CoreConfig.cmake:15 (message): The imported target "Qt5::Core" references the file "/usr/local//mkspecs/macx-clang"

**A:** Chyba pravdepodobne je na strane Homebrew, pri inštalácii Qt5 nevytvorí dve symlinky (ani po zavolení brew link). Musíte ich vytvoriť ručne (ak treba, upravte verziu v ceste, návod bol robený pri nainštalovanej 5.9.2):

```
ln -s /usr/local/Cellar/qt/5.9.2/mkspecs /usr/local/mkspecsln -s
/usr/local/Cellar/qt/5.9.2/plugins /usr/local/plugins
```

Ak príkaz zlyhá, pretože symlinky už existujú, zmažte ich a skúste znova.

### **6.1.4 Windows**

Tento návod bol úspešne otestovaný na operačnom systéme Windows 10 Pro.

Na inštaláciu potrebujeme klonovať projekt 3DSoftViz (Tím č.18 klonuje z /pmarusin/3dsoftviz) z Githubu a stiahnuť nasledovné: Nasledujúce programy neinštalovať do Program Files, PATH nesmie obsahovať medzeru!:

- Microsoft Visual Studio 2017 (môže byť nainštalovaný štandardne do Program Files)
- CMake (v3.9.5)
- OpenSceneGraph (v3.4.1) (jeden z nasledujúcich)
  - Full pack
  - Release + Debug + Knižnice 3rd party dependencies VC14.1
- Qt (v5.9.2) a Qt Creator (v4.4.1) - cez Qt online installer
- OpenCV (v2.4.13.4) (jeden z nasledujúcich)
  - Built - buildnuté

- Source - iba zdrojové súbory, treba buildnúť
- Boost (v1.65.1)
- RapidEE - program na prácu s premennými prostredia
- Kinect for Windows SDK 1.8
- OpenNI2 (v2.2)
- NiTE (v2.21)
- Debugging Tools for Windows 10 - WinDbg - Win 10

### **Postup inštalácie:**

**Uvedené programy neinštalovať do Program Files, PATH nesmie obsahovať medzeru!**

**Idealne všetky programy, knižnice, atď. dať do jedného adresára. Ideálne je mať všetko na spoločnom mieste kvôli prehľadnosti, napr. C:/Timak/**

1. Nainštalovať Microsoft Visual Studio 2017 (*môže byť nainštalovaný štandardne do Program Files*) spolu aj s rozšíreniami:
  1. VC++ 2017 v141 toolset
  2. Windows 10 SDK for Desktop C++
  3. Visual C++ tools for CMake
  4. Visual C++ ATL support
  5. MFC and ATL support
  6. C++/CLI support
2. Nainštalovať CMake. (Cesta je v dokumente označená ako %CMAKE\_DIR%). Pri inštalácii zvoliť "Add CMake to the system for all users".
3. Nainštalovať Qt a Qt Creator (*mal by sa nainštalovať automaticky s Qt*):
  1. Nainštalovať Qt online installer (%QT\_INSTALLER\_DIR%)
  2. Spustiť MaintenanceTool.exe > Skip > Add or remove components
  3. Zvoliť z Qt 5.9.2 nasledujúce veci:
    1. mserv2017 64-bit
    2. všetko od Qt Charts po Qt Script
  4. Zvoliť z Tools Qt Creator 4.4.1 CDB Debugger Support
  5. Nainštalovať zvolené položky (%QT\_INSTALLER\_DIR%/5.9.2/mserv2017\_64)
4. Vytvoriť zložku OpenSceneGraph (%OSG\_DIR%)
5. V prípade stiahnutia Full pack:
  1. Rozbalíť zložky do %OSG\_DIR%
  2. Vynechať nasledujúci krok.
6. V prípade stiahnutia Release + Debug + Knižnice 3rd party dependencies VC14.1:
  1. Vytvoriť zložku Release v zložke %OSG\_DIR%
  2. Rozbalíť do nej zložky zo stiahnutého Release
  3. Vytvoriť zložku Debug v zložke %OSG\_DIR%
  4. Rozbalíť do nej zložky zo stiahnutého Debug
  5. Vytvoriť zložku 3rdParty v zložke %OSG\_DIR%

6. Rozbalíť do nej zložky zo stiahnutého 3rd party dependencies VC14.1
7. V prípade stiahnutia zbuildovaných súborov OpenCV
  1. Rozbalíť zložky do %OPENCV\_DIR%
  2. Vynechať nasledujúci krok.
8. V prípade stiahnutia iba zdrojových súborov OpenCV
  1. Rozbalíť OpenCV do %OPENCV\_DIR%/source
  2. Vytvoriť zložku install a build v %OPENCV\_DIR%
  3. Spustiť CMake (cmake-gui.exe)
    1. source code > %OPENCV\_DIR%/source
    2. binaries > %OPENCV\_DIR%/build
    3. nastaviť 3rdparty:
      - stlačiť Add Entity
      - zadať Name: 3rdparty
      - zvoliť Type: PATH
      - zadať Value: %OPENCV\_DIR%/source/3rdparty
      - stlačiť OK
    4. stlačiť Configure
    5. nastaviť generator na Visual Studio 15 2017 Win64
    6. stlačiť Finish
    7. čakať na výpis
    8. nastaviť CMAKE\_INSTALL\_PREFIX na %OPENCV\_DIR%/install
    9. stlačiť Generate
      - ak došlo k erroru: File > Delete Cache a skúsiť znova
  4. Nájsť súbor OpenCV.sln v %OPENCV\_DIR%/build
  5. Otvoriť súbor vo VS2017 (**ako správca!**)
  6. Nastaviť Solution Configuration na Debug, Solution Platform na x64
  7. Nájsť projekt ALL\_BUILD > pravý klik > build
  8. Po skončení nájsť projekt INSTALL > pravý klik > build
  9. Nastaviť Solution Configuration na Release
  10. Nájsť projekt ALL\_BUILD > pravý klik > build
  11. Po skončení nájsť projekt INSTALL > pravý klik > build
  12. Presunúť nainštalované súbory zo zložky install do %OPENCV\_DIR%
  13. Zložky install, build a sources môžu byť vymazané z %OPENCV\_DIR%- 9. Nainštalovať Boost (%BOOST\_DIR%)
- 10. Otvoriť súbor environment.txt a upraviť v ňom cesty k programom a knižniciam, ktoré sa nachádzajú na začiatku súboru.
  1. Spustiť powershell ako správca
  2. Skopírovať do powershellu obsah celého súboru
  3. Vynechať pridávanie systemových premenných cez RapidEE (nasledujúci krok) a vykonať len kontrolu, či sa cesty spravne nastavili.
- 11. Nainštalovať a otvoriť RapidEE, v ktorom sa vykonajú tieto zmeny (**ako správca!**):

1. do PATH pridať premenné:
    1. %CMAKE\_DIR%/bin
    2. %QT\_INSTALLER\_DIR%/5.9.2/msvc2017\_64/bin
    3. %QT\_INSTALLER\_DIR%/Tools/QtCreator/bin
    4. %OSG\_DIR%/Debug/bin
    5. %OSG\_DIR%/Release/bin
    6. %OSG\_DIR%/3rdParty/x64/bin
    7. %OPENCV\_DIR%/x64/vc15/bin
  2. Vytvoriť premennú CMAKE\_INCLUDE\_PATH a pridať:
    1. %OSG\_DIR%/Debug/include
    2. %OSG\_DIR%/Release/include
    3. %OSG\_DIR%/3rdParty/x64/include
    4. %OPENCV\_DIR%/include
  3. Vytvoriť premennú CMAKE\_LIBRARY\_PATH a pridať:
    1. %OSG\_DIR%/Debug/lib
    2. %OSG\_DIR%/Release/lib
    3. %OSG\_DIR%/3rdParty/x64/lib
    4. %OPENCV\_DIR%/x64/vc15/lib
  4. Vytvoriť premennú BOOST\_INCLUDEDIR a pridať: %BOOST\_DIR%/boost
  5. Vytvoriť premennú BOOST\_LIBRARYDIR a pridať: %BOOST\_DIR%/libs
  6. Vytvoriť premennú BOOST\_ROOT a pridať: %BOOST\_DIR%
  7. Vytvoriť premennú OPENCV\_DIR a pridať: %OPENCV\_DIR%
12. Nainštalovať WinDbg.

#### Inštalácia projektu:

1. Naklonovať projekt 3DSoftViz cez git shell (%3DSoftViz%)
2. Cez command line prejsť do naklonovaného projektu a zavolať *git submodule update --init --recursive*
3. Vytvoriť v priečinku %3DSoftViz% priečinky \_build a \_install
4. Spustiť QtCreator. Tools > Options... > Build and Run:
  1. záložka CMake – ak je nainštalovaný CMake, tak auto-detected, inak pridať manuálne %CMAKE\_DIR%/bin/cmake.exe
  2. záložka Compilers – ak existuje VS2017, tak sú auto-detected
  3. záložka Qt Versions – ak je nainštalovaný Qt, tak auto-detected, inak zadaj cestu %QT\_INSTALLER\_DIR%/5.9.2/msvc2017\_64/bin/qmake.exe
  4. záložka Debuggers – ak je nainštalovaný WinDbg, tak sú auto-detected, inak pridať manuálne C:/Program Files (x86)/Windows Kits/10/Debuggers/x64/cdb.exe
  5. záložka Kits – vytvoriť nový a nastaviť hodnoty nasledovne:
    1. Name: Local PC
    2. Device type: Desktop

3. Compiler: C: Microsoft Visual C++ Compiler 15.0 (amd64)
4. Compiler: C++: Microsoft Visual C++ Compiler 15.0 (amd64)
5. Debugger: Auto-detected CDB at C:/Program Files (x86)/Windows Kits/10/Debuggers/x64/cdb.exe
6. Qt version: Qt 5.9.2 MSVC2017 64bit
7. CMake Tool: System CMake at %CMAKE\_DIR%/bin/cmake.exe
6. záložka General – nastaviť Default build directory: %3DSoftViz%/\_build
7. Potvrdiť – OK
5. File > Open File or Project... > vybrať CMakeLists.txt z %3DSoftViz%
6. Skontrolovať výpis v časti 6 General Messages, na konci výpisu musí byť
7. Generating done
8. CMake Project was parsed successfully.
9. Vybrať Projects > Build & Run > Local PC > Build, v časti Edit build configuration kliknúť na Add > Clone selected, nazvať „unity“
10. Prejsť na vytvorený build config. „unity“, v časti Build Steps otvoriť Details a vypnúť pri build step Build: cmake.exe --build . --target možnosť *all* a označiť *install\_unity*
11. Stlačiť Build (kladivo vľavo dole - potrebné spraviť znova po každej následnej úprave systémových premenných)
12. Po úspešnom zbuildovaní vybrať Projects > Build & Run > Local PC > Run, v časti Run pridať Add > Custom Executable a nastaviť:
  1. Executable: %3DSoftViz%/\_install/bin/3DSoftviz.exe
  2. working directory: %3DSoftViz%/\_install/bin/
13. Spustiť program pomocou zeleného tlačidla Run (vľavo dole)

V prípade, že aplikácia ihneď po spustení crashne, napriek úspešnému buildu, jedná sa pravdepodobne o problém s grafickou kartou. Na notebookoch, ktoré majú externú grafickú kartu NVidia, je v tomto prípade treba cez Nvidia Control Panel nastaviť jej použitie pre 3DSoftViz.exe

### Rozšírenie 3DSoftviz o Kinect

1. Nainštalovať Kinect for Windows
2. Skontrolovať v RapidEE či sa vytvorila premenná %KINECTSDK10\_DIR%, keď nie, vytvoriť a pridať: C:/Program Files/Microsoft SDKs/Kinect/v1.8
3. Nainštalovať OpenNI2 (%OPENNI2\_DIR%)
4. Skontrolovať v RapidEE či sa vytvorili premenné (ak na konci majú 64, vymazať 64):
  1. %OPENNI2\_INCLUDE%, keď nie, vytvoriť a pridať: %OPENNI2\_DIR%/Include/
  2. %OPENNI2\_LIB%, keď nie, vytvoriť a pridať: %OPENNI2\_DIR%/Lib/
  3. %OPENNI2\_REDIST%, keď nie, vytvoriť a pridať: %OPENNI2\_DIR%/Redist/
  4. %OPENNI2\_ROOT%, keď nie, vytvoriť a pridať: %OPENNI2\_DIR%
5. Nainštalovať NiTE2 (%NITE2\_DIR%)
6. Skontrolovať v RapidEE či sa vytvorili premenné (ak na konci majú 64, vymazať 64):

1. %NITE2\_INCLUDE%, keď nie, vytvoriť a pridať: %NITE2\_DIR%/Include/
  2. %NITE2\_LIB%, keď nie, vytvoriť a pridať: %NITE2\_DIR%/Lib/
  3. %NITE2\_REDIST%, keď nie, vytvoriť a pridať: %NITE2\_DIR%/Redist/
  4. %NITE2\_ROOT%, keď nie, vytvoriť a pridať: %NITE2\_DIR%
7. Pridať do premennej CMAKE\_INCLUDE\_PATH:
1. %OPENNI2\_INCLUDE%
  2. %NITE2\_INCLUDE%
8. Pridať do premennej CMAKE\_LIBRARY\_PATH:
1. %OPENNI2\_ROOT%/Driver
  2. %OPENNI2\_REDIST%
  3. %OPENNI2\_REDIST%/OpenNI2/Drivers
  4. %OPENNI2\_LIB%
  5. %NITE2\_ROOT%/Samples/Bin/OpenNI2/Drivers
  6. %NITE2\_LIB%
9. Pridať do premennej PATH:
1. %OPENNI2\_REDIST%/OpenNI2/Drivers
  2. %OPENNI2\_REDIST%
  3. %NITE2\_REDIST%
  4. %NITE2\_ROOT%/Samples/Bin
10. Spustiť CMake a skontrolovať vo výpise:
1. OpenNI2 FOUND
  2. NITE2 FOUND
  3. KINECTSDK FOUND

## 6.2 Používateľský manuál pre 3DSoftviz

**Okno s aplikáciou je rozdelené na tri základné časti:**

- menu
  - File – načítanie grafu zo súboru, z databázy, uloženie grafu, uloženie layoutu, ukončenie aplikácie
  - Settings – nastavenia aplikácie; konfiguračný súbor používa bodkovú notáciu, ktorá umožňuje identifikovať význam konfiguračnej premennej
  - Help
  - Test - pušta základné grafy pre rýchle testovanie (100-uzlový, 500-uzlový, Veolia, Lua Graph, Module Graph)
- hlavné okno - zobrazuje graf a umožňuje s ním používateľovi interagovať
- ovládací panel – nástroje pre prácu s grafom

### 6.2.1 Ovládacie prvky

**Ovládanie kamery:**

- Vyber prvkov grafu - ľavé tlačidlo myši + pohyb myšou

- Otáčanie kamery okolo grafu - pravé tlačidlo myši + pohyb myšou
- Ovládanie priblíženia obrazovky - koliesko na myši
- Ovládanie pomocou klávesnici:
  - Hore - PgUp
  - Dole - PgDn
  - Vľavo - šípka vľavo
  - Vpravo - šípka vpravo
  - Dopredu - šípka hore
  - Dozadu - šípka dole

Inicializácia automatického pohybu začne po stlačení kláves Alt + Shift a kliknutím myši na zvolenú hranu, či uzol. V závislosti od nastavenia aplikácie sú pred inicializovaním pohybu ešte automaticky vybrané body záujmu. Pokiaľ je automatický výber uzlov vypnutý, body záujmu je možné zvolať manuálne myšou alebo stlačením klávesy Q (pre náhodný výber uzlov alebo pre výber uzlov pomocou metrík). Automatické použitie metrík je možné vypnúť v nastavení aplikácie pomocou parametra „Viewer.PickHandler.SelectInterestPoints“ nastaveného na hodnotu 1.

#### Iné ovládacie prvky:

- Kláves "T" – skrytie všetkých ovládacích prvkov
- Kláves "S" - štatistiky vykreslovania
- Kláves "Shift" - pridávanie ďalších objektov do výberu
- Kláves "Ctrl" - odstránenie objektov z výberu

#### 6.2.2 Záložka GRAPH



- manipulácia s prvkami grafu (no-select mód), pohyb vybraných uzlov v priestore



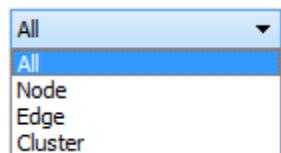
- výber jedného prvku grafu (single-select mód)

Umožňuje sústredenie sa na práve jeden objekt – môže to byť hrana aj uzol.



- výber viacerých prvkov grafu (multi-select mód)

Umožňuje vybrať v trojrozmernom zobrazení viacero objektov naraz.



- typ výberu: všetko, iba uzly, iba hrany, klastre



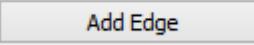
- centrovanie pohľadu vzhľadom na vybraný prvak grafu

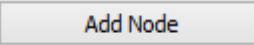
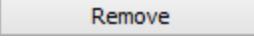
V prípade, že nie je označený žiadny element, kamera bude vycentrovaná na ťažisko grafu

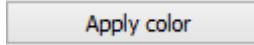
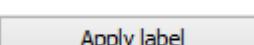


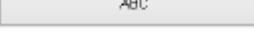
- pridanie meta uzla do grafu

-  - odstránenie vybraných meta uzlov z grafu
-  - ukotvenie vybraných uzlov na aktuálnej pozícii, t. j. nebudú sa pohybovať v závislosti od pôsobenia síl ostatných uzlov
-  - uvoľnenie ukotvených uzlov

 - pridanie hrany medzi dvomi vybranými uzlami  
Umožňuje pridať hranu medzi dvoma vybranými uzlami, kde ešte nie je hrana, inak končí akcia chybovou hláškou. Ideálne je čierou šípkou vybrať jeden uzol a bielou šípkou ho nastaviť na také miesto, kde sa ho dá spojiť s druhým uzlom - je potrebné mať nastavenie Node v takomto prípade spolu s Multi-select mode.

 - pridanie uzla do stredu pohľadu  
 - odstránenie vybraných elementov (uzly alebo hrany)  
Ak sa rozhodneme pre zmazanie hrany, uzly prepojené s touto hranou v grafe zostávajú.

,  - zafarbenie zvolených uzlov a hrán farbou vybranou z palety nad tlačidlom  
 - aplikovanie textového označenia na vybrané uzly podľa textu z pola nad tlačidlom

 - zapnutie/vypnutie zobrazovania popisov uzlov a hrán  
,  - spustenie/zastavenie rozmiestňovania (animovania) uzlov grafu

 - zmena odpudivých síl pôsobiacich medzi uzlami  
,  - výber vizuálnej reprezentácie uzla (square, sphere) a výber vizuálnej reprezentácie hrany (quad, cylinder, line)

- pridanie nového manipulátora (geometria kocky) do scény, ktorý umožňuje pohyb kamery ľavým tlačidlom myši

## Dragger\_rotation

- pridanie nového manipulátora (geometria gule) do scény, ktorý umožňuje rotáciu grafu okolo jeho stredu ľavým tlačidlom myši

## City layout

- reprezentácia grafu ako vizualizačnej metafory mesta (namiesto klasického grafu).

### 6.2.3 Záložka CONSTRAINTS

 - aplikovanie priestorového ohraničenia: povrch gule

 - aplikovanie priestorového ohraničenia: obsah gule

 - aplikovanie priestorového ohraničenia: rovina

 - aplikovanie priestorového ohraničenia: zjednotenie gule a roviny

Zjednotenie gule a roviny je vhodné pre zobrazenie grafov s hustým stredom, alebo na veľké grafy.

 - aplikovanie priestorového ohraničenia: kružnica

Aplikovanie obmedzenia na kružnicu na uzly v celom grafe je vhodné pre veľmi riedke grafy alebo na grafy s pravidelnou štruktúrou. Pri hustých grafoch sa hrany medzi uzlami prekrývajú

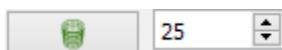
 - aplikovanie priestorového ohraničenia: kužeľ

Obmedzenie na kužeľ je vhodným riešením v prípadoch, kedy má jeden uzol výrazne vyšší počet hrán ako ostatné uzly.

 - aplikovanie priestorového ohraničenia: kužeľový strom

Po aplikácii sa uzly rozdelia do skupín podľa spoločného rodiča. Na tieto skupiny sa aplikujú obmedzenia na kužeľ, ktoré sú následne obmedzené na roviny v závislosti od hĺbky uzlov v strome. Kužeľový strom sa aplikuje automaticky na celý graf na základe používateľom vybraného koreňového uzla. Jedine v prípade, že graf nie je spojitý, tak sa aplikuje iba na komponent, ktorý obsahuje koreňový uzol.

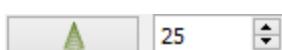
 - odstránenie vybraných priestorových ohraničení



25

- aplikovanie priestorového ohraničenia: povrch valca

Vloží do scény tzv. bod záujmu, od ktorého sú uzly zobrazovaného grafu odťaľané do tvaru valca. Polomer valca sa dá nastaviť pomocou číselníka.



25

- aplikovanie priestorového ohraničenia: povrch kužeľa

Vloží do scény tzv. bod záujmu, od ktorého sú uzly zobrazovaného grafu odťaľané do tvaru kužeľa. Polomer kužeľa sa dá nastaviť pomocou číselníka. Veľkosť kužeľa sa nastavuje automaticky podľa toho, kam sa používateľ prostredníctvom kamery pozera.



- aplikovanie radiálneho rozmiestnenia na označené uzly

Odporúča sa používať pri stromovom type grafu. Použitie rozmiestnenia na označené uzly dáva používateľovi nové možnosti ako zväčšenie priestoru označením uzlom, alebo manuálne zhlukovanie uzlov.



- výber módu vykreslenia radiálneho rozmiestnenia (drôtený, plný)

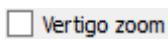


- nastavenie módu 2D/3D radiálneho rozmiestnenia



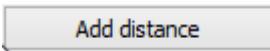
1. nastavenie veľkosti rozmiestnenia
2. nastavenie priečinnosti rozmiestnenia
3. nastavenie počtu zobrazených gúľ
4. nastavenie faktora zosilnenia odpudivých síl v radiálnom rozmiestnení pre uzly, ktoré nie sú na rovnakej vrstve
5. nastavenie faktora zosilnenia odpudivých síl v radiálnom rozmiestnení pre uzly, ktoré sú na rovnakej vrstve

Veľkosť radiálneho zobrazenia sa dá nastaviť v rozmedzí 0 – 300, parameter priesvitnosti 0 - 100 %, veľkosť faktora zosilnenia odpudivých síl sa nastavuje medzi hodnotami 1 - 5000.

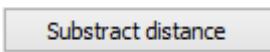


- prepínač medzi normálnou a vertigo kamerou

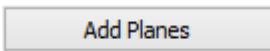
Tento mód kamery je vhodné použiť vtedy, keď chce používateľ meniť dva rôzne pohľady na graf: lokálny pohľad, pri ktorom môže používateľ s väčšou presnosťou skúmať jednotlivé uzly a vzťahy medzi nimi a globálny pohľad, pri ktorom môže používateľ skúmať vzťahy medzi uzlami a rozloženie uzlov v daných hĺbkach kostry grafu v globálnom kontexte.

 Add distance

- zvýšenie vzájomnej vzdialenosťi medzi rovinami

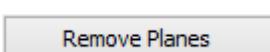
 Subtract distance

- zníženie vzájomnej vzdialenosťi medzi rovinami

 Add Planes

- pridanie dvoch paralelných rovín

Obmedzenie na roviny sa aplikuje pri grafoch s minimálnou maximálnou hĺbkou kostry grafu hodnoty 2. Koreňový uzol v kostre grafu určí program - vyberie uzol s najväčším počtom hrán. Pri zrušení obmedzenia sa uzly „odpoja“ od roviny.

 Remove Planes

- odobranie dvoch paralelných rovín

 1



- zmena násobiča odpudivých síl medzi uzlami

Násobič odpudivých síl medzi uzlami je na začiatku nastavený na 1 kvôli prvému pridaniu dvoch rovín do priestoru - nechceme, aby sa hneď zväčsili odpudivé sily.

 X

- vypnutie všetkých predchádzajúcich obmedzení

#### 6.2.4 Záložka CLUSTERING

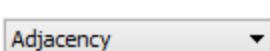


- zlúčenie vybraných uzlov

Umožňuje zlúčiť vybrané uzly do jedného spoločného uzla. Takýto uzol sa bude v pokračovaní zobrazovať modrou farbou.



- zrušenie zlúčenia vybraných uzlov

 Adjacency

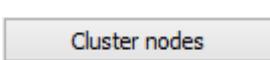


- definovanie algoritmu, ktorým sa bude zhlukovať graf (adjacency, leafs, neighbours)

Depth:

 1

- nastavenie počtu rekurzí pre vybraný algoritmus

 Cluster nodes

- spustenie zhlukovania nad aktívnym grafom

Ak zhlukovanie trvá viac ako 1 sekundu, objaví sa indikátor postupu.

 Edge Bundling

|                                                                                       |                                                                                         |
|---------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------|
|  1 |  2   |
|  3 |                                                                                         |
| alpha:                                                                                |  100 |
|                                                                                       |  4   |

1. spustenie algoritmu na zväzovanie hrán

2. pozastavenie algoritmu na zväzovanie hrán
3. úplne zastavenie algoritmu na zväzovanie hrán a zobrazenie pôvodného grafu
4. vstupné pole na zadanie konštanty, určujúcej silu akou sú hrany k sebe počas zväzovacieho algoritmu pritáhované

**Po použití funkcie zhlukovania, sa odkryjú nasledujúce možnosti:**

Opacity:

auto

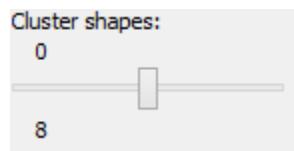
selected

auto - automatická priehľadnosť - mení sa na základe vzdialosti zhlukov od kamery

selected - priehľadnosť označeného zhluku – pomocou posuvníka(nižšie) sa mení priehľadnosť len označených zhlukov



- posúvaním upravíme priehľadnosť označených zhlukov



- posúvaním sa mení prahová hodnota, pri ktorej sa menia tvary zhlukov

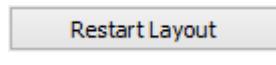
Spodné číslo udáva, koľko uzlov obsahuje daný zhluk (v tomto prípade 8).

**Pri označení konkrétneho zhluku sa odkryjú nasledujúce možnosti:**

- kliknutím zmeníme označený zhluk na obmedzovač

Obmedzuje pozície uzlov tak, aby z neho nevyšli von. Keď obmedzovač posunieme dostatočne ďaleko, t.j. mimo pôvodnej pozície uzlov, uzly sa začnú lepiť na jeho stenu a posúvať spolu s ním. Ignoruje príťažlivé a odpudivé sily medzi ním a ostatnými uzlami grafu (posunutie zhluku bez obmedzovača spôsobí posun celého grafu za týmto zhlukom).

Obmedzovač začína svoje pôsobenie ako kocka, je možné zmeniť jeho tvar naťahovaním a stlačením.



- znovurozmiestnenie uzlov v priestore po tom, ako sa nalepia na hranu obmedzovača

Repulsive force

- upravenie odpudivej sily medzi uzlami v označenom zhluku

Čím je hodnota väčšia, tým budú uzly ďalej od seba.

**Ďalšie funkcie obmedzovača:**

Ak na zhluk zaregistrujeme obmedzovač, môžeme s ním jednoducho pohybovať a meniť jeho tvar pomocou klávesových skratiek a myši:

- Pohyb – metóda ťahaj a pust ( drag & drop )

- Zmena veľkosti – držíme **Ctrl** a točíme kolieskom myši
- Zmena tvaru
  - na osy x – držíme **X** a **Ctrl** a točíme kolieskom myši
  - na osy y – držíme **Y** a **Ctrl** a točíme kolieskom myši
  - na osy z – držíme **Z** a **Ctrl** a točíme kolieskom myši

## 6.2.5 Záložka CONNECTIONS

**Nick:**  - napísanie mena, pod ktorým bude používateľ vystupovať v kolaborácii

**Host session**  - spustenie/zastavenie servera

**Host:**  - napísanie IP adresy servera

**Connect to session**  - pripojenie(odpojenie) ku(od) kolaborácií

**Collaborators:**

- zoznam používateľov (zoradený abecedne), v ktorom je možné jedného vybrať a použiť nasledujúce funkcie:

**Spy**  
 **Center**  
 **Shout** - po výbere si môžeme zvolať jednu funkciu z dvojice: *Spy* (špehovať) a *Center* (centrovať).

Po aktivovaní funkcie Spy získa používateľ pohľad iného používateľa, ktorý je priebežne aktualizovaný – znamená to, že po hybom sledovaného používateľa sa aktualizuje aj pohľad sledujúceho. Po aktivácii Center nasmeruje pohľad používateľa tak, aby v jeho strede bol iný používateľ. Pri centrovaní platí to isté, čo pri špehovaní – teda pri aktualizácii polohy centrovaného používateľa sa natáča aj pohľad centrujúceho používateľa. Po označení polička Shout sa ostatným používateľom v scéne zobrazí pri vašom mene ikona znázorňujúca, že sa pokúšate upútať pozornosť.

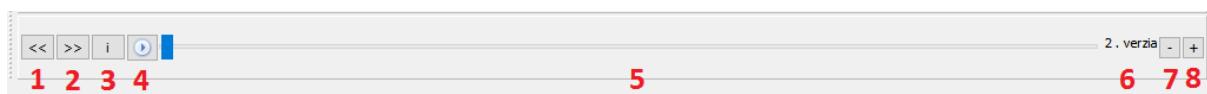
**Avatar scale**  - nastavenie veľkosti avatarov v scéne

Avatar je kužeľ, ktorého kruhová podstava znázorňuje smer, ktorým sa používateľ pozera.

## 6.2.6 Záložka EVOLUTION

- Po rozkliknutí tabu Evolution (1) sa zobrazia možnosti evolúcie

1. Lifespan - možnosť ponechania vymazaných uzlov vo vizualizácii. Prednastavená hodnota 0 znamená, že vymazané uzly sa automaticky vymažú z grafu. V prípade hodnoty väčšej ako 0 vymazané uzly v grafe zotrívajú o verzie dlhšie podľa nastavenej hodnoty
2. Change commits - prepínač spracovania Git repozitáru. Ak je zaškrtnutý, inicializuje sa spracovanie na úroveň grafu volaní. V opačnom prípade - na úroveň histórie Git repozitáru
3. Kombo box s výberom vizualizácie - prepínanie sa medzi jednotlivými možnosťami vizualizácie grafu volaní
  - *LuaStat* - vizualizácia softvérových metrík pomocou analýzy Lua zdrojového kódu
  - *Difference* - pohľad na zmeny, ktorými softvér prešiel pri prechode na novú verziu
  - *Changes* - aktivovanie filtrovania nad práve aktívnu vizualizáciu
4. Kombo box s výberom filtra - výber vhodnej skupiny filtra
  - Prednastavená možnosť *All* - všetky prvky grafu sú zobrazené
  - *Authors* - filtrovanie podľa autorov zmien v softvéri
  - *Structure* - filtrovanie podľa štruktúry
5. Kombo box zo zoznamu možností - možnosti zavisia od vybraneho filtru
  - zoznam autorov s možnosťou zobrazenia zmien všetkých autorov - *All*
  - štruktúra - *Files* (zobrazí v grafe volaní len uzly reprezentujúce adresáre a súbory), *Local Functions* (zobrazí rozšírenú možnosť *Files* spolu s uzlami lokálnych funkcií), *Global Functions* (zobrazia sa uzly možnosti Local Functions spolu s uzlami globálnych funkcií) a *Modules* (zobrazí všetky štruktúry, ktoré sa v grafe nachádzajú)

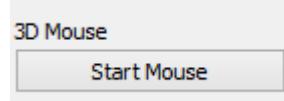


- panel ovládania evolúcie

1. Prechod na predchádzajúcu verziu - možnosť, kedy sa stav grafu vráti o jednu verziu dozadu
2. Prechod na nasledujúcu verziu - možnosť, kedy sa stav grafu posunie o jednu verziu dopredu
3. Tlačidlo informácií o verzii - zobrazí informácie o aktuálne zobrazenej verzii. Medzi zobrazené informácie patrí identifikátor, autor a dátum commitu spolu so zoznamom súborov, ktoré boli zmenené
4. Sputenie/zastavenie animácie - aktivovanie/zastavenie automatického prechodu na novú verziu
5. Posuvník - presun na konkrétnu verziu pomocou skokového prechodu medzi verziami
6. Indikátor verzie - poskytuje informáciu o aktuálne zobrazenej verzii
7. Spomalenie animácie - regulovanie rýchlosťi animácie

## 8. Zrýchlenie animácie - regulovanie rýchlosťi animácie

### 6.2.7 Záložka MORE FEATURES



- zapnutie 3D myšky (musí byť aktivovaný driver)

**Camera rotation** - ak je zaškrtnuté, kamera nasmerovaná na graf sa pohybuje na základe pohybu tváre, značky alebo rúk, inak sa na základe týchto akcií rotuje samotný graf

**Camera enabled** - povoľuje použitie kamery

- otvorenie okna pre prácu s kamerou

- otvorenie okna pre prácu s hlasovým ovládaním

#### Note

Speech je momentálne vylúčený z projektu

- zapnutie ovládania pomocou Leap Senzor-u

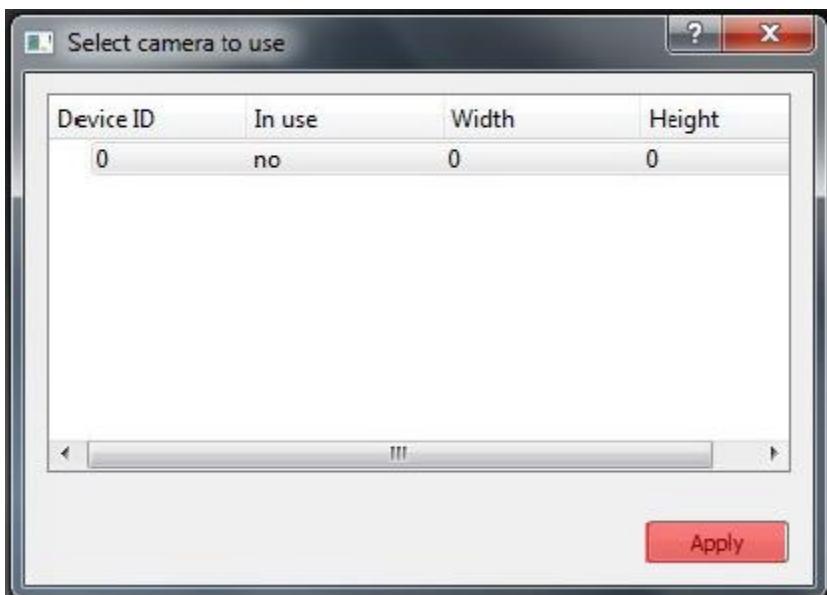
#### Okno pre prácu s kamerou



- prispôsobenie ľavej strany okna pre ovládanie funkcionality rozpoznávania tváre (pri zapínaní treba zaškrtnúť Camera rotation a Camera enabled).

- zvolenie kamerového zariadenia a následným potvrdením objavenie záberu z kamery

Ukončiť túto akciu je možné tlačidlom „StopFaceRec“ (ak používateľ zatvoril okno, môže ho vrátiť na grafický interface opäťovným kliknutím na „StartCamera“ a potom pozastaviť detekciu). V prípade detegovanej tváre (detekcia je reprezentovaná zeleným obdĺžnikom) sa kamera alebo graf pohybuje vďaka pohybu tváre.



- okno pre výber snímacieho zariadenia

Face Recognition

Marker

- prispôsobenie ľavej strany okna pre ovládanie funkcionality rozpoznávania značky

Start Marker Detection

- zvolenie kamerového zariadenia a následným potvrdením objavenie záberu z kamery určenej pre rozpoznávanie značky a graf sa začne otáčať a pohybovať so značkou

Background - nastavenie aktuálne snímanie ako pozadie pre graf

Je potrebné zmeniť parameter „Viewer.SkyBox.Noise“ v konfiguračnom súbore na hodnotu 2 alebo 3 (odporúčané je 3).

Marker is behind - prepínanie medzi pohybom podľa značky ako keby sa kamera pozerala na používateľa a naopak

Correction - zapnutie korekcie

Update cor. param. - nastavenie korekčných parametrov

Podľa predvolených nastavení sa značka pohybuje tak, ako keby sa kamera pozerala vo vodorovnom smere. Ak by sa pozerala napr. na stôl pod miernym sklonom dole, graf by sa pri posúvaní značky po stole neposúval korektne. Preto je možné nastaviť korekčné parametre. Najskôr je potrebné nastaviť značku do polohy, kedy je detegovaná na spodnom okraji a následne stačiť toto tlačidlo. Po nastavení sa aktivuje opcia „Correction“ (uvedené vyššie), ktorou je možné zapnúť korekciu.

#### **Change Markers**

- zmena spôsobu použitia značky v prípade, že používateľ má k dispozícii len jednu značku

**NoVideo** - vypnutie/zapnutie zobrazenia videa

Toto prepínanie a vypnutie zobrazenia video má vplyv len na zobrazenie v rámci tohto ovládacieho okna „Face Recognition“ and „Marker Detection“ a neovplyvňuje to ani voľbu kamery pre video pozadie.



- Interakcia s vizualizáciou v obohatenej realite

- Možnosť *Custom light*, vyznačená modrou, ktorá slúži na prepínanie vlastného a základného zdroja svetla
- Možnosť *Shadow*, vyznačená žltou, ktorá slúži na zapínanie a vypínanie generovania tieňov
- Možnosť *Base*, vyznačená červenou, ktorá slúži na zobrazenie a skrytie základne
- Možnosť *Axes*, vyznačená ružovou, ktorá slúži na zobrazenie a skrytie pomocných osí
- Tlačidlo *Center graph*, vyznačené svetlo modrou, ktoré slúži na umiestnenie grafu nad stred základne

#### **Okno pre prácu s kinectom a arucom**

**Start kinect**

- zapnutie detekcie

**Kinect Snapshot**

- zachytenie kádra s následnou možnosťou dať ho na pozadie

**Turn on Marker Detection**

- zapnutie rozpoznávania značiek

**Turn off cursor**

- prepínanie medzi detekovaním ruky pre manipuláciu grafu alebo kamery v podobe rotovania a medzi detekovaním ruky pre funkciu “klik” (pohyb ruky do hĺbky, nie vertikálne alebo horizontálne)

**Turn off zoom**

- vypne možnosť približovania



- nastavenie práce s arucom

**Start projective AR view** - zobrazenie okna projekčného zobrazenia

| Projector |          |
|-----------|----------|
| FOV:      | 30,00 Ā° |
| Pos X:    | -0,665 m |
| Pos Y:    | -1,345 m |
| Pos Z:    | 0,825 m  |
| Dir X:    | -0,085 m |
| Dir Y:    | 1,345 m  |
| Dir Z:    | -0,587 m |

- v projekčnom zobrazení - Projector - spinboxy na zmenu parametrov projektora (odhora) - zorné pole, pozícia (súradnice x, y, z), smer projekcie(súradnice x, y, z)

| Viewer |          |
|--------|----------|
| FOV:   | 90,00 Ā° |
| Pos X: | -1,880 m |
| Pos Y: | -0,950 m |
| Pos Z: | 1,720 m  |
| Dir X: | 1,130 m  |

- v projekčnom zobrazení - Viewer - spinboxy na zmenu parametrov pozorovateľa (odhora) - zorné pole, pozícia (súradnice x, y, z), smer projekcie (súradnice x, y, z)

| Graph        |                                     |
|--------------|-------------------------------------|
| Pos X:       | -0,750 m                            |
| Pos Y:       | -0,250 m                            |
| Pos Z:       | 0,250 m                             |
| Radius:      | 0,500 m                             |
| Place graph: | <input checked="" type="checkbox"/> |

- v projekčnom zobrazení - Graph - spinboxy na zmenu parametrov grafu (odhora) - pozícia (súradnice x, y, z), polomer, checkbox Place graph na potvrdenie použitia parametrov grafu (štandardne označený)

**Apply scene** - potvrdenie zadaných parametrov scény

#### Hlasové príkazy pre Speech

- select all nodes - vybranie všetkých uzlov
- select left side - vybranie uzlov na ľavej strane
- select right side - vybranie uzlov na pravej strane
- clear screen - zrušenie vybratia uzlov
- sphere - sformovanie gule pre vybrané uzly
- unset restrictions - návrat k pôvodnému stavu - zrušenie akcie "sphere"

#### 6.2.8 Hlavné okno

**edges filter** - filtrovanie hrán

nodes filter - filtrovanie uzlov

Príklady príkazov:

- "params.type like 'file' or params.type like 'directory'"
- "params.name like 'init%.lua' and params.type like 'function'"
- "params.type like 'function'"

Filter je navrhnutý pre grafovú vizualizáciu softvéru s využitím softvérových metrík jazyka Lua a je vyhodnotený po stlačení klávesu „Enter“.

Load function calls

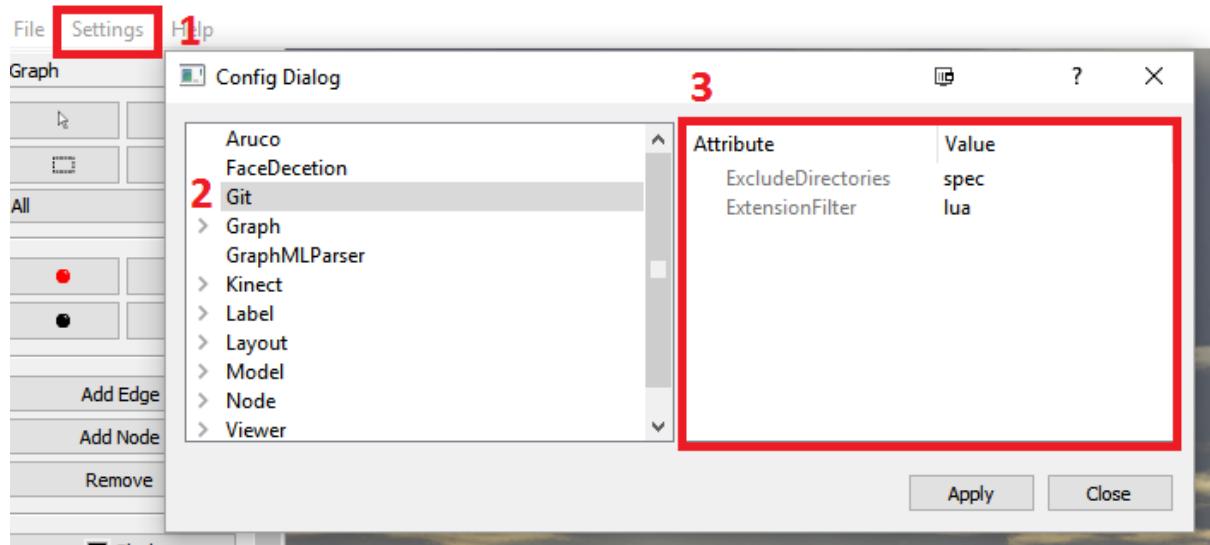
- zobrazí dialóg pre výber súborov a po vybratí vykreslí do poľa pod tlačidlom graf volaní funkcií týchto súborov

Pri označení práve jedného vrcholu v poli sa zobrazí stromová štruktúra informácií o tomto vrchole.



- prepínanie medzi zobrazovaním jedného prehliadača pre každý uzol a zobrazovaním jedného prehliadača pre všetky vyznačené uzly

### 6.2.9 Git repozitár



1. Settings / Options - zobrazenie dialógového okna s konfiguráciou
2. Možnosť Git - zobrazia sa možnosti konfigurácie spracovania Git repozitáru
3. Možnosti konfigurácie spracovania Git repozitáru
  - vyčlenenie adresárov (ExcludeDirectories) - ľubovoľný počet názvov adresárov oddelených znakom ",". Pre zadanú hodnotu sa pri spracovaní Git repozitáru odignorujú všetky súbory, ktoré vo svojej relatívnej ceste obsahujú adresár spec.
  - ExtensionFilter - funguje obrátene, pričom ponecháva len tie súbory, ktorých koncovka súboru sa zhoduje s jednou zo zadaných hodnôt. Hodnota taktiež

môže obsahovať viacero koncoviek súborov, pričom musia byť oddelené znakom ","