Analyza

UML-based Live Programming Environment in Virtual Reality

Link1

kategoria: **programovanie pomocou UML**

programovanie vo VR pomocou UML diagramov
bol uskutocneny vyskym 20 ludi kde sa porovnavalo programovanie vo VR vs v Unity
na pisanie kodu bol pouzity Google cloude speech (neefektivny) a virtualna klavesnica
konverzia medzi UML a zdrojovym kodom prebiehala vdaka AST reprezentacii kodu
Vyskum. otazka: Moze programovanie vo VR znizit cas na vyvoj softveru?
Ucastnici mali za ulohu modifikovat kod hry aby presli 3 rozne levely (napr odstranit class v UML)
Skusenost uzivatelov a praktickost programovania bola vo VR
Cas riesenia uloh bol horsi vo VR
System Usability Scale (SUS) and the User Experience Questionnaire (UEQ)

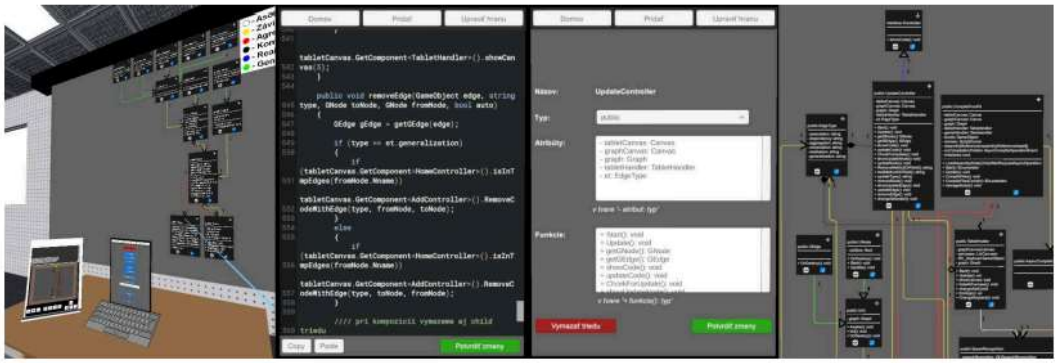


Fig. 1. VR programming environment from left: the whole VR scene, details of VR tablet with source code and class editor, UML class diagram on the wall.

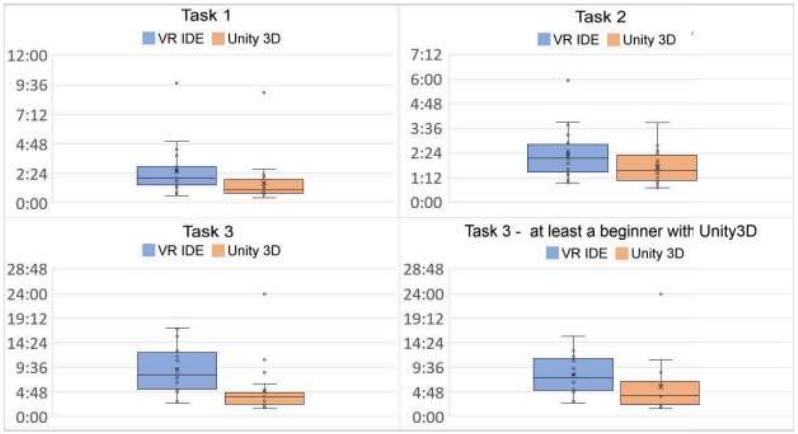


Fig. 5. Comparison of times for each task; times are in minutes.

BabiaXR: Virtual Reality software data visualizations for the Web

Link2

kategoria: programovanie/ vizualicacia pomocou softveru city metaphor

ciel: vybudovat VR softver pre vizualizaciu, analyzu softveru dostupny vsade hlavne prehliadace
Softver postaveny na zakladoch CodeCity3D ktory sluzil na zobrazovanie softveru ako interaktivnej mapy
kazda budova reprezentuje jeden subor a rozmery budovy = parametre suboru
web browser vie zobrazit 8000 budov Oculus browser 1000
plnenie roznych zadani pomocou softveru BabiaXR bolo vo VR rychlejsie ako bez VR
Za mna dostupnejsi softver a moznost zobrazit komplexnejsie systemy

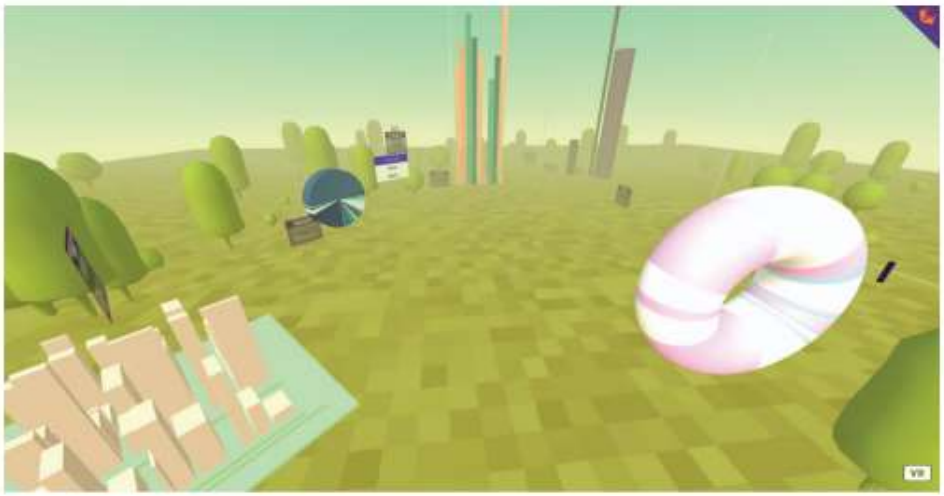


Figure 1: Example of a BabiaXR Scene

Collaborative Modeling and Visualization of Software Systems Using Multidimensional UML

Link3

Snaha o vizualizaciu softveru pomocou UML vo VR
Moznost spolocne naraz vizualizovat softver
Boli vyvtvorene 2D UML diagramy v 3D priestore
Vdaka Aplikacnemu servru a relacnej DB mohli uzivatelia naraz efektivne tvorit UML a komunikovat (cez sockety zrobene)
Kolaborant mozem vidiet aktualne zmeny na projekte aj celkovu historiu zmien
Zmeny su ukladane a tak sa clovek vie vratit k povodnemu zadaniu (undo changes)
Vysledkom je zlepsenie podmienok vyvoja softveru vo VR pomocou UML (aktivna spolupraca, featury dobre)
Je jedna verzia UML nie viacero lokalnych kopii co je vyhoda

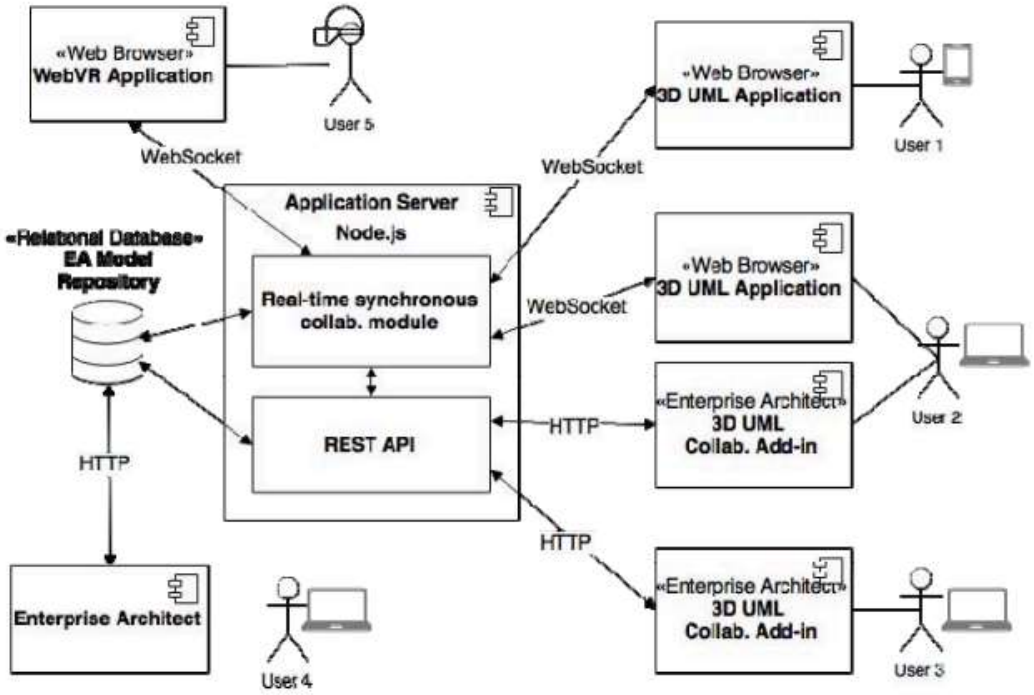


Fig. 2. Collaborative 3D UML Application

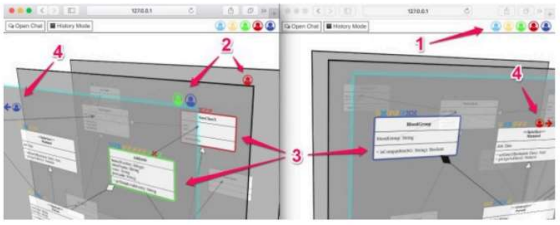


Fig. 6. Presence Awareness Features

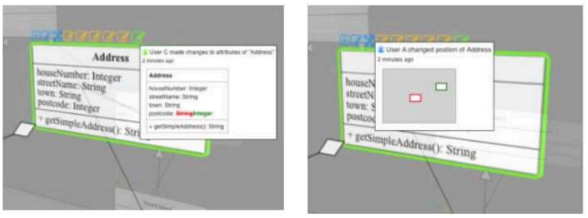


Fig. 8. UML Class and User Action History

Collaborative Software Modeling in Virtual Reality

Link4

kategoria: **kolektivne programovanie / programovanie pomocou UML**

modelovanie softveru vo VR je dolezite kvoli dynamike ktoru mozeme 3D vykonavat
Voice chat vo VR zvuk ktory posleme vie prist v urcitej sile, z nejakeho smeru a dialky
Microsoft hololens mix AR s UML umoznil modelovanie UML ale iba pre jednotlivca
App umoznuje kolektivne UML softver modellin
Prostredie ma nekonecne plus Avatar ktory simuluje moje pohyb vo VR
UML su 3D tvare kociek, text je na stene kocky (moze byt skryty), spojenia su rurky
Vyskum 24 studentov (12 skupin) parove UML klasicke pomocou LucidChart a programovanie vo VR
12/20 ludi nepouzivalo VR aktivne tak cas na modelovanie vo VR bol vacsi ako normale 10/12 skupin to mali takto
rozdiel medzi VR a veb bol 5 minut priemerne
78/100 uzitocnost UML vo VR, ludia mali lepsi pocit z kolektivnej spoluprace vo VR ako normalne
Vacsia chybovost pri modelovani vo VR
Negativny fakt je sposobeny nedostatocnou skusenostou ucastnikov s VR svedcia o tom caste nechcene kliknutia a premiestnovania objektov

staznosti na virtualnu klavesnicu a chybajucu moznost automaticky pripnut (uchytit) graf na nejaku poziciu
Wilcoxon Signed-Rank test

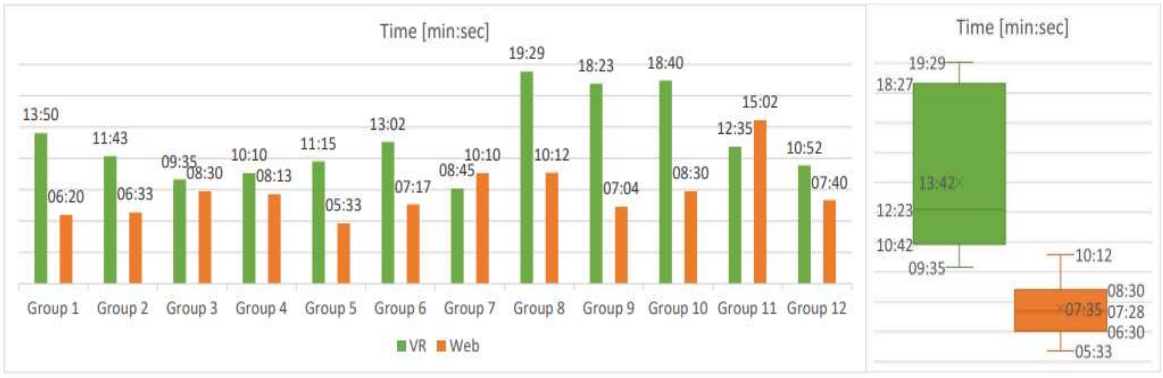
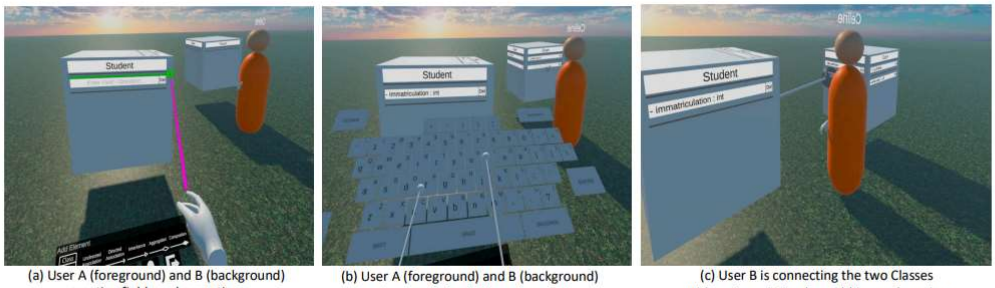


Fig. 5. The error rates of the recorded participant in each group.



(a) User A (foreground) and B (background) creating fields and operations

(b) User A (foreground) and B (background) entering names for fields and operations

(c) User B is connecting the two Classes with an Association by grabbing and moving

Fig. 2. Screenshots of the collaborative VR modeling environment

VR City: Software Analysis in Virtual Reality Environment

Link5

kategoria: programovanie/ vizualicacia pomocou softveru city metaphor

Vizualizacia softveru pomocou modelu mesta
Steinbrückner and Lewerentz model reprezentoval objekty nasledovne ulica, plot kolo budovy (package), budova (class)
vel roznych pristupov k vizualizacii mesta jeden zaujimavy kazde poschodie budovy reprezentuje inu vlastnost triedy
Model sa formuje z neorientovaneho grafu kde vrchol je class a hrana prepojenie medzi triedami jej vaha zalezi od poctu dependencies
algoritmus layout je odvodený od vizualizačnej techniky vyplnenia priestoru pre multivariačné grafy malého sveta
budova je class, poschodie je funkcia, kocka reprezentuje nejaku hodnotu softveru
viacero vrstiev jedno zobrazuje classes a ich prepojenia, ine iba classes alebo ine ukazuje len konkretnu metodu
su poskytnute tri aspekty historia ako sa class metoda formovala, dalsia verzia sluzi len na analyzu clovek vidi kod, posledna sluzi na vyvoj kod je kompilovatelny
limitacia vo VR max 3x3 metre pohyb => teleportovanie do inych casti mesta

test coverage or for profiling.

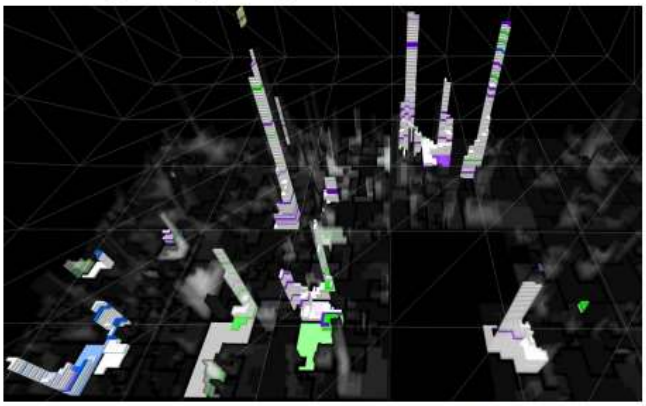


Fig. 10. City in trace mode after playing traces recorded during initialization (blue color assigned) and drawing (green color) in JHotDraw applet. Purple color is assigned to methods involved in both traces.

TABLE I. SYSTEMS UNDER STUDY AND NUMBER OF VISUALIZED FRAGMENTS

System	JHotDraw 7.0.6	JUnit 4
Packages	24	62
Classes	356	1149
Methods	3604	3758
Lines of code	40308	25162
Short SHA	untagged in git	26d6145

Fig. 3. Example of a expanded code label.

Virtual Reality in Software Engineering: Affordances, Applications, and Challenges

Link6

kategoria: pristupy riesenia k prgramovaniu vo VR

rozne pristupy ako zlepsit programovanie pomocou VR
jednou je umiestnovanie kodu na platno hocikde chcem
VR podporuje priestorovu pamat vdaka pohybu ludi popripade pohybu hlavou clovek sa citi byt vtiahnuty do sceny
Bola vyvinuta VR app, kde bol jeden block na pisanie kodu, v zavislosti od kodu sa VR dynamicky prostredie menilo
vdaka tomu clovek vedel hned fixnut chyby
VR pomaha inzinierom vo velkom

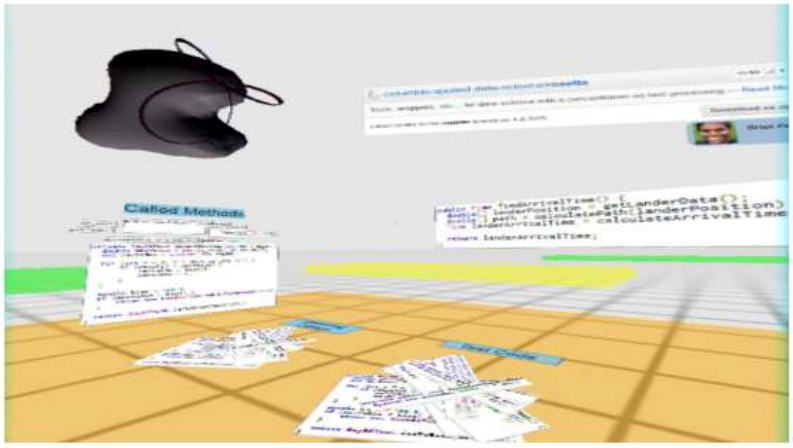


Fig. 3. IMMERSION screenshot. The reviewer is reviewing code to reposition the lander on the comet. The reviewer sees the active method, piles of relevant fragments on the floor, and has expanded one pile into a fragment ring on the left to read the details of those fragments. A model of the comet and the lander's expected flight path is shown in the upper left. The reviewer can walk between code packages on the floor which are color coded according to amount of modification for this review. GitHub details are shown in the upper right.

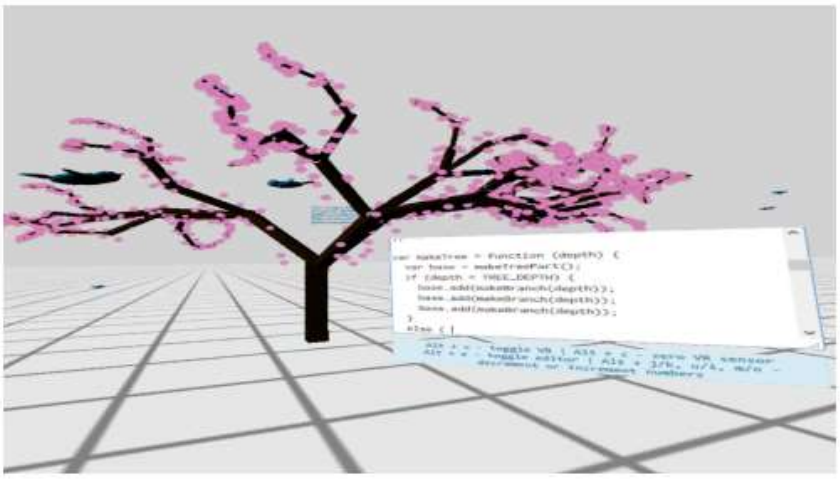


Fig. 2. RIFTSKETCH screenshot. The tree is generated by a recursive algorithm that the user has typed into the floating editor. The flying, animated birds represent tweets pulled in from the Twitter API, also generated by code that the user has entered.

DGT-AR: Visualizing Code Dependencies in AR

Link7

kategoria: AR vizualizacia pomocou grafu

Vizualizacia zavislosti komponentov pomocou inverznej argumented reality
Grafe vrchol reprezentuje subor a hranu zavislost medzi nimi
Vdaka AR vidim na svoj PC plus teda vidim architekturu sotverua alebo coho potrebujem. Teda viem klasicky upravit kod
moznost interakcie s grafom, po kliknuti na vrchol vidime nazov suboru a vsetky jeho hrany su vyraznejsie
graf sa generuje vdaka extnesions zo zdrojaku
vyskum 5 ucastnikov (2 mali Mario vizualiz.) zvysok inu. Mali najst zaujimave suvislosti / anomalie v grafe
problem s kontrastom farieb pri citani textu
problem so zoom vrcholov a ich klikanim kvoli velkemu mnozstvu vrcholov a malej ploche grafu
citanie velkych pismen sposobilo u niektorych unavu
participants were requested to provide feedback

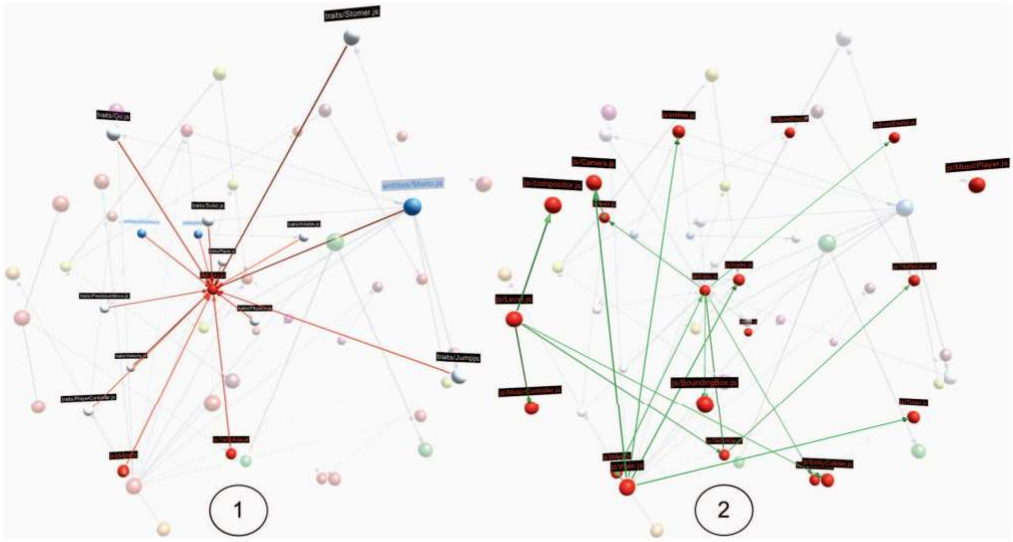


Fig. 3. Dependency highlighting and folder highlighting interactions

CollaVRation: An Immersive Virtual Environment for Collaborative Software Development

Link8

kategoria: **programovanie pomocou UML**

VR pomaha zapracovat ludi na projekty
programovanie vo VR nema negativny vplyv na produktivitu podla vyskumov
vo VR bolo umoznene live programovanie, autokompilacia kodu, zobrazenie programu cez UML
Ciel vyskumu: Je parove programovanie vo VR lepsie ako normalne? (efektivita, praktickost, komunikacia…)
Ulohou bolo dokoncit rozrobenu hru sachu
boli dve casti 1) dvojica programatorov - pohyb figurok 2) tutor ziak - fix bugov, on/of figuriek or events
cas straveny vo VR bol 26m 44s bez 24m55s
cas pristup student tutor bol o 8 minut lepsi
celkovo parove programovanie vo VR praktickejsie a lepsia uzivatelska skusenost
komunikacia bola castejsia vo VR ako normalne
before start due selection : experience questionnaire
at the end: filling prepared questionnaires: 2x SUS, one for the prototype, one for the classic approach, 2x UEQ, same procedure.

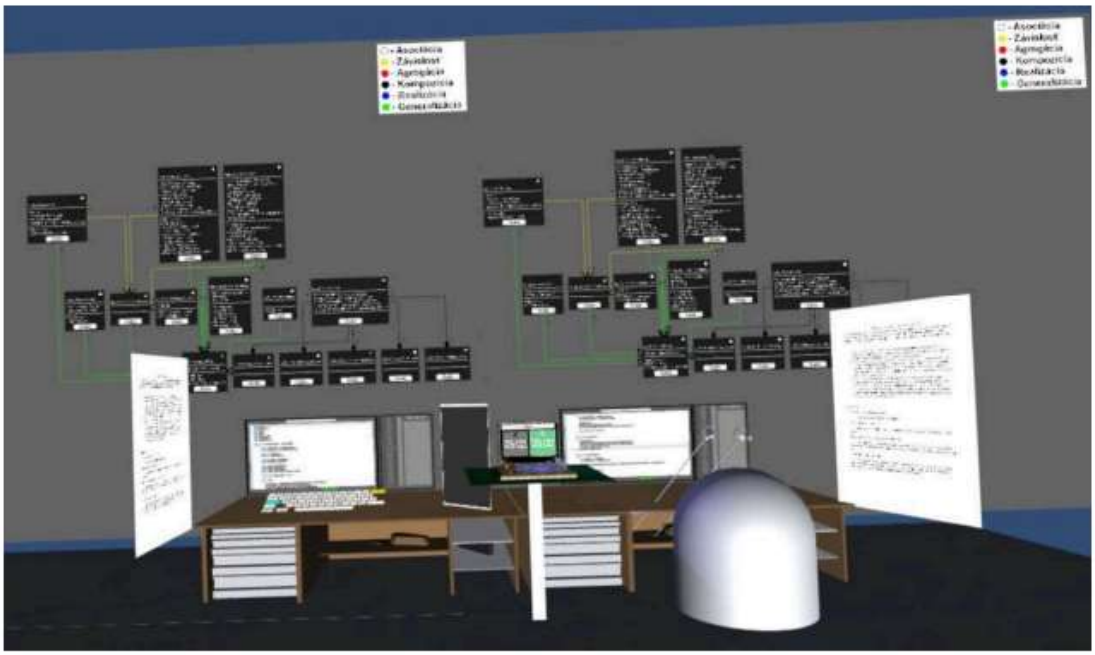


Fig. 1. VR scene for collaborative programming containing 2 virtual desks and background wall with UML diagrams. The capsule on the right is the user's avatar.

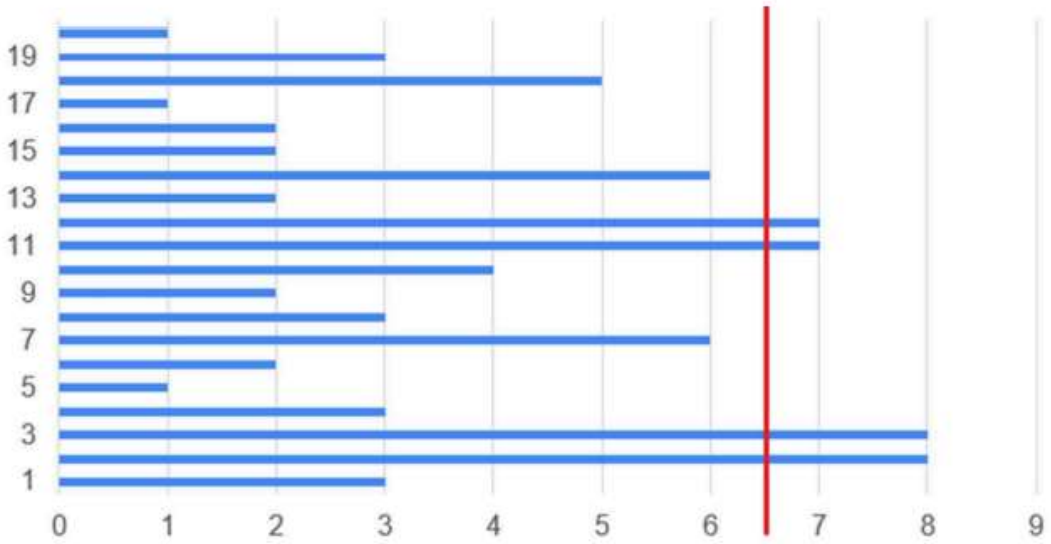


Fig. 5. Participant's previous experience with programming in C# and Unity.

Program Comprehension in Virtual Reality

Link9

kategoria: AR programovanie

vyskum sa venuje programovaniu vo VR
bolo 26 ucastnikov podelenych na polovicu cast robila ulohy vo VR a klasicky
otazky: Ako velmi sa lisi koncentracia a produktivita v programovani vo VR
Ulohou bolo napisat vystupy kodu a kratku textovu dokumentaciu o com kod bol. Bolo 8 mini java projektov kazdy mal iny rozsah max kolo 80 riadkov
Koncentracia vo VR bola horsia, je to aj neskusenostou s VR
Produktivita nebola ovplyvnena vo VR oproti normalu
Klavesnica bola v jednom trackeri a mys druhom. Pisanie sa prejavovalo na virtualnej klavesnici
NASA TLX survey

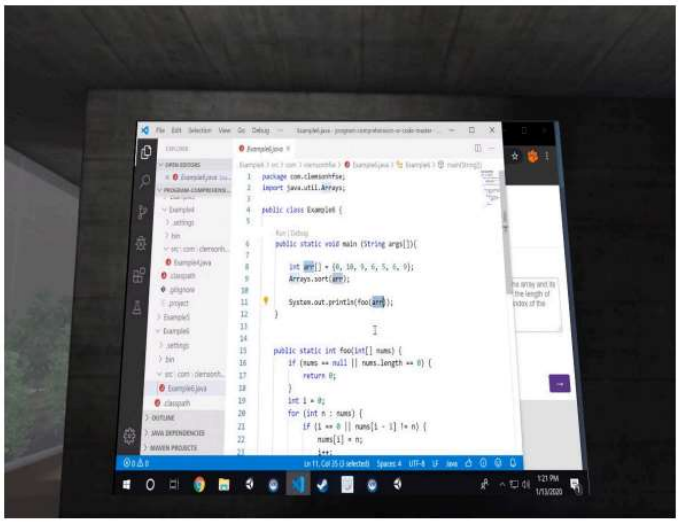
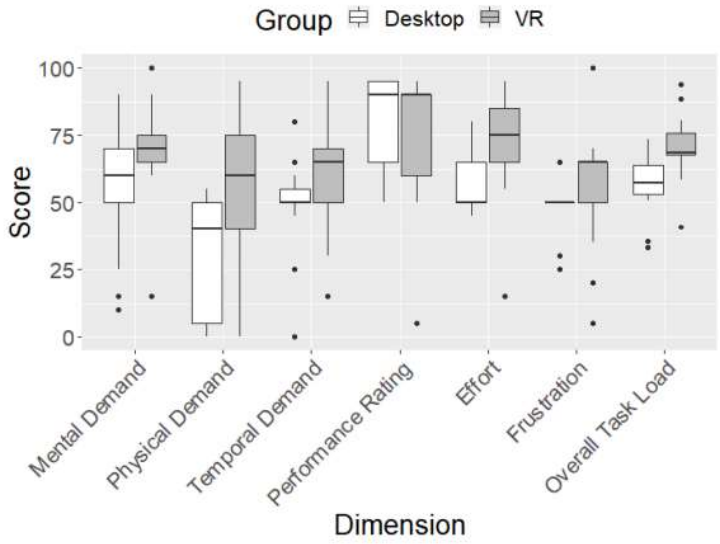


Figure 1: The virtual reality environment used by the programmer to complete source code comprehension tasks. This figure shows the desktop mirror as the programmer comprehended source code.

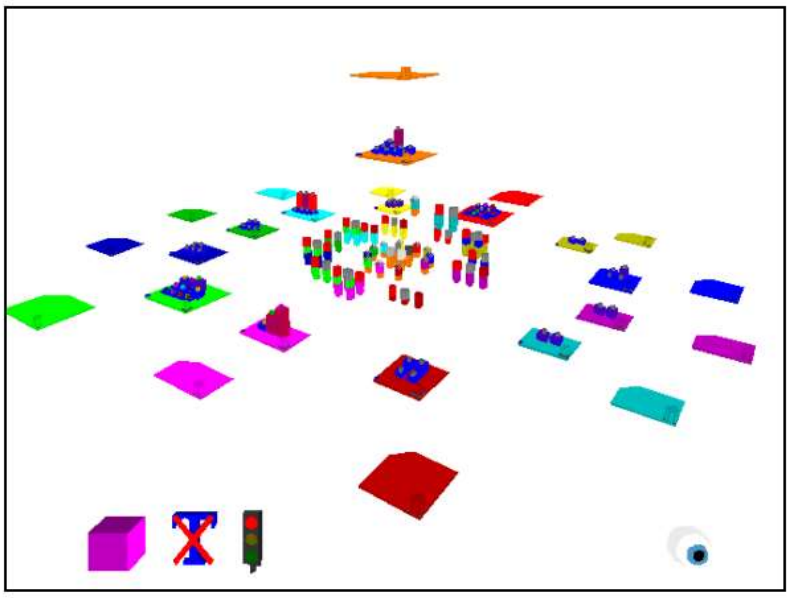


Visualising software in virtual reality

Link10

kategoria: vizualizacia softveru vo VR

riesi vizualizaciu softveru a aspekty na ktore netreba pri nej zabudat
Treba zvolit vhodnu objektovu reprezentaciu, ktora bude aj relativne jednoducha, parametre objektu mozu repre vlastnosti casti softvera
treba dbat aj na kvalitu a mnozstvo info ktore budu zverejnene od danom objekte
vhodnu navigaciu v prostredi aby sa uzivatel nestratil
viac levelov poskytnutia info prvy krat vidi nieco pre detaily klikne na konktretnu oblast co ho zaujima
zrobeny prototyp FileVis na prezentaciu projektov v C++, priecinok bol box subor ikonka farby podla typu .c alebo .h
Blok funkcie vyska = velkost farba = komplexnost
boli zrobene 3 vizualizacie roznych systemov vysledky v tabulke



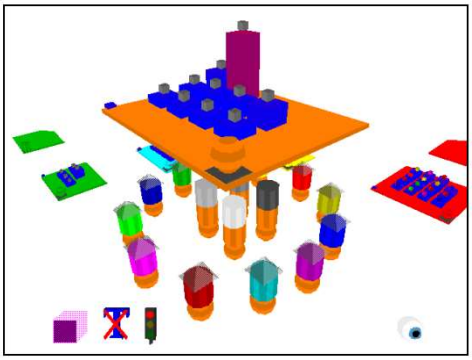


Figure 2. Illustrating a file containing some low-detail function representations.

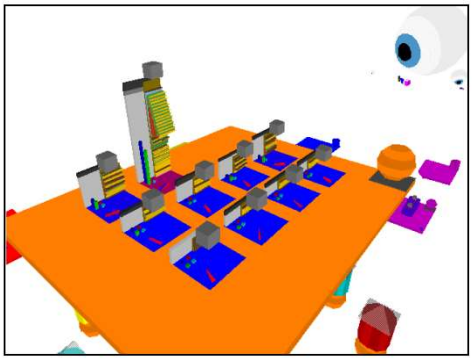


Figure 3. The high-detail function representations appear as the viewer moves closer.

IDEvelopAR: A Programming Interface to enhance Code Understanding in Augmented Reality

Link11

kategoria: **programovanie pomocou UML**

Vyvinuty **IDEVELOPAR** ktory zoberie casti kodu z codebubbles a premietne ich z konecneho do nekonecneho priestoru
Code bubles navigacia kodu je znazornena vizualnymi ciarami lepsie ako preklikavanie medzi oknami v klasickom IDE
Appke vieme menit kod, zobrazovat strukturu kodu, kompilovat kod
Historia navigacie je v podobe stromu, so zobrazenim sa vieme celkovo hrat (zoom in/out, zobrazenie konstruktora triedy alebo cely kod classy)
zmena kodu je plynule sycnhronizovana do id a pouzivame na to bluetooth klavesnicu
Fixovanie kodu je lahsie vdaka navigacii ide postupne cez classy metody az kym nedostane k chybe
questionnaire containing a System Usability Scale (SUS) part [12], [13], a User Experience Questionnaire (UEQ)
boi zrobene timy studentov ktori mali pomocou tohto softveru najst 3 chyby a fixnut ich jeden mal headset menil kod druhy navigator mal PC s live prenosom VR a svojim intelij
SUS score 72.19 (with 95% confidence interval [52.72 - 85.00]) as well as improved SUS score for small samples. 69.45 indicate that the participants perceived the overall usability of the prototype as acceptable
predpoklad rastu uzitocnosti skusenostou uzivatela s touto technologiou
scrollovanie kodu pomocou oci sa niektorym nepacilo, chybala im info ci premenna sa pouziva niekde inde v kode

Fig. 1. Hand Menu. Menu for controlling basic functions (switch to programming mode, run code on IDE and show project view). The menu appears as soon as the user raises a hand and turns its palm towards their face.

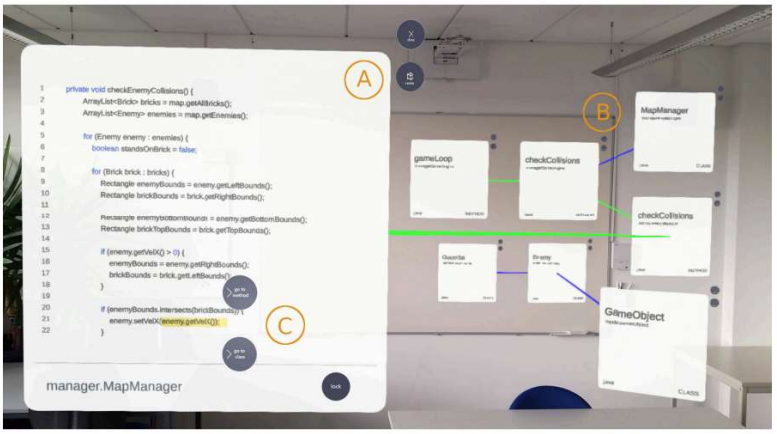
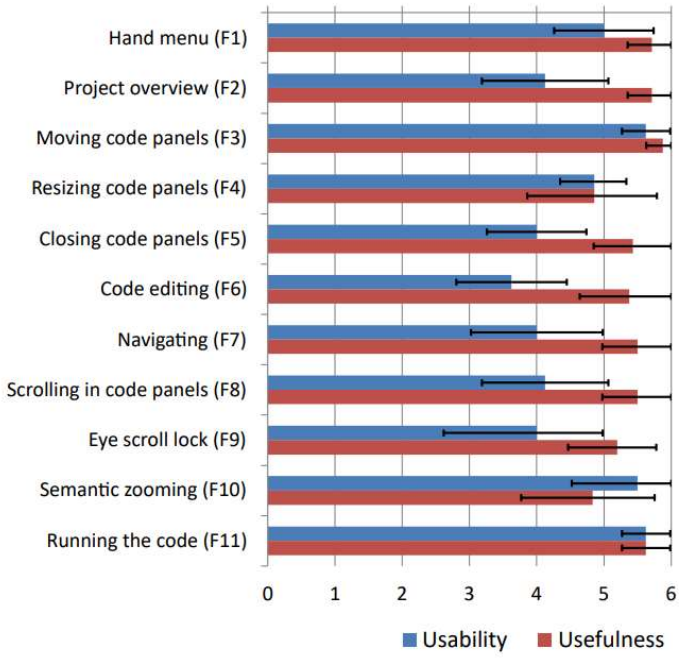


Fig. 3. IDEVELOPAR in practice. The exemplary procedure from section III is shown. **(A)** Opened code panel **(B)** Two independent, visual linked emerging code panel trees shown at a different level of detail (semantic zooming) **(C)** Code navigation, when clicking on a followable code element



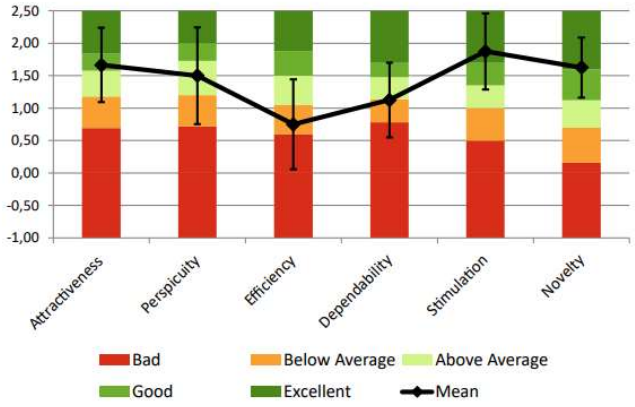


Fig. 5. Results of the User Experience Questionnaire. The error bars show the 95% confidence interval.

XRaSE: Towards Virtually Tangible Software using Augmented Reality

Link12

kategoria: programovanie/ vizualicacia pomocou softveru city metaphor

Ide o zobrazenie architektury softveru do Argumented reality
Ako inspiraciu zobrali codeCity VR kde mohli odpozorovat ako vyuziva vnemy pouzivatela na pochopenie danym veciam
Celkovo sa pomocou GrafuUM vykresli architektua softveru, program si vie zrekonstruovat aj nejake metody premenne ktore su potrebne
V zavislosti od architektury su tri stavy vyvoja jeden navrhne vhodnu topologiu na zobrazenie architektury
druhy riesi osvetlenie objektov napriklad aby to kolabovalo s AR a nebol nejaky objekt priesvitny az neviditelny.
Takisto sa riesi zobrazenie mapy prostredia idealne na jednej velkej rovine kde uzivatel ma na dohlad kazdu cast
uzivatel si vie objekt vytvorit popripade odstranit cele
Softver bol vyvinuty pomocou Unity,MS mixed reality a zobrazoval Java projekty
Otazky: Ako je kognitívna záťaž, angažovanosť, použiteľnosť a zapamätanie. Nakoniec tiež skúmame využitie biometrických skenerov na kvantitatívnu analýzu vyššie uvedených kvalitatívnych parametrov.

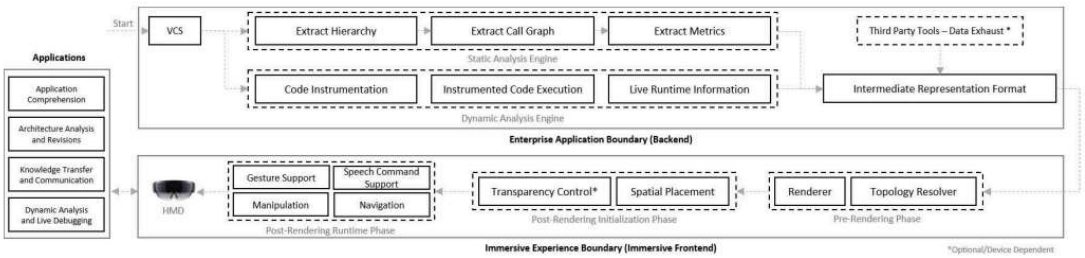


Fig. 1. XRaSE System Overview: An AR based immersive application visualization and comprehension framework.

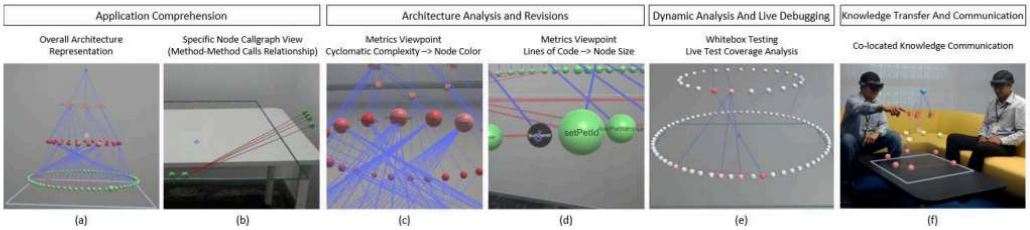


Fig. 2. Immersive software representation using XRaSE. (a-d) Depicting static 3D structure, overlaid metrics, navigation and manipulation viewpoints of open-source Petclinic application in a Microsoft HoloLens HMD. (e) Dynamic test coverage analysis by executing an existing test suite. (f) Collaborative knowledge communication using multiple HMDs.

Simplifying Robot Programming using Augmented Reality and End-User Development

Link13

kategoria: programovanie vo VR

Dolezite pri robotoch generally aby bolo programovanie nezavisle od vyrobcu, mozne debug chyby, jasne vidie parametre robota
vyvinute prostredie SPEARED s pouzitim AR na programovanie robotov vsetko vid vyssie moze clovek vidiet v AR cez headset
Pridany simulator kde mozeme vidiet premietnuty nas stav kodu. Lepsi up to date ako keby sme mali cakat kym sa program nahra do robota a ten zacne vykonavat pohyby
Bolo už historicky riešené programovanie robota pomocou AR v offine režime
fyzická simulácia zabezpečuje pohyb robota a detekciu kolízie pri pohybe
User interface zabezpečuje tok inštrukcií do end point častí robota
V headsete má človek robota jeho kód a interface so zoznamom príkazov na interakciu s robotom
Non Ar device PC obsahuje kód robota a interface na komunikáciu Kód musí byť vždy preložený pomocou inverznej kinematiky
bol spraveny rozhovor s odbornikmi a dole tabulke su vysledky miery suhlasu pre jednotlive tvrdenia

odbornici pozitivne ocenili simulaciu robota jeho kodu do AR
takisto DLL kodenie (prikazy v podobe bloku s troma parametrami) umoznilo lepsi pristup k robotom pre laikov a znizilo nutnost ucit sa novy jazyk zakazdym
Celkovo uzivatel porozumel lepsie kodu
Detekcia kolizie ma potencial zlepsovat sa takisto support pre no AR zariadenia

Fig. 6: Robot Code, Robot Model in an AR application and block-based DSL Code (f.l.t.r.)

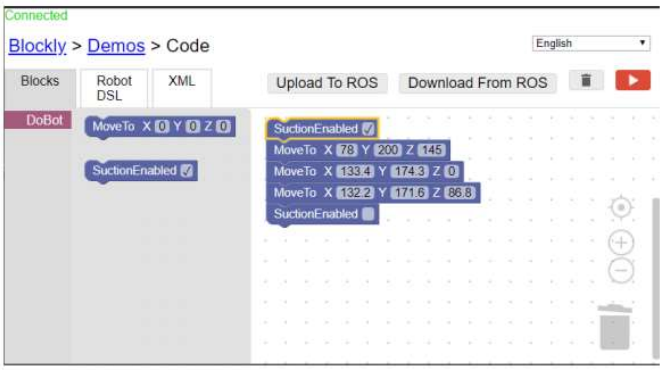


Fig. 8: Adapted Blockly environment with custom blocks (above) and for robot code (below)

	Experts Feedback (n=9)				
Statement	**Median**	**Average**	**Min**	**Max**	**Std. Dev.**
1.1 Defect finding through visualization	6	5,44	1	7	1,71
1.2 Coordinate handling target sphere	6	5,67	2	7	1,41
1.3 Environmental constraints through robot projection	7	6,44	5	7	0,68
1.4 Easy adaptation of existing code	5	5,22	3	7	1,47
2.1 Validation Through Visualization	7	6,11	2	7	1,52
2.2 Step-by-step defect finding	6	6,22	5	7	0,63
3.1 Smart collision detection error finding	7	6,56	5	7	0,68
3.2 Time ghosts object localization	6	5,89	3	7	1,29
3.3 Time ghosts Environmental information	4	4,00	1	7	1,63
3.4 Environmental information through object detection	7	6,33	5	7	0,82
4.1 Understandability through uniform representation	6	6,33	5	7	0,67
4.2 Defect finding through uniform representation	6	6,22	5	7	0,79
4.3 Multi-device code modification	7	6,67	6	7	0,47
4.4 Learnability through code abstraction	7	6,33	5	7	0,82
4.5 Entry-barrier reduction through uniform code	7	6,78	6	7	0,42

Fig. 10: Feedback to Statements Overview

Toward a VR-Native Live Programming Environment

Link14

kategoria: programovanie pomocou AST blokov

programovanie pomocou drag and dropu programovych prvkov vo VR cim sa minimalizuje pisanie kodu
program je ulozeny v syntatickom strome rozdeleny na blok. Blok = 3D kocka vo VR - pouziva ho Squeak co je system pre live programovanie
clovek moze menit nazov stitku na bloku alebo asik aj text vo vnutri bloku. Vseobecne moze mazat bloky pridavat prepajat to vsetko drag and dropom
podpora grafickeho pisania system rozozna co napisem virtualnou rukou. Nieco ako scratch. System ponuka automaticke doplnanie nazvov funkcii popripade aj metod ci tried s parametrami
Experiment sa zameral na pracu s app. Prvej casti riesili intuitivnost. Druhej poskytli navod a riesili ulohy k dispozicii bol aj instruktor
Dali im nakodenu slnecnu sustavu vo VR anechali ich nakodit rotovanie kolo slnka alebo setnut rychlost objektov
V prvej casti bez vysvetlenia boli problemy s oznacenim textu nakolko mal maly font a aj chvenie ruky mohlo kurzor posunut uplne inde
ludia sa citili byt menej produktivni ako na desktope ale ocenili rychlu spatnu vazbu pri programovani alias vedeli ci to ide bo nejde
ucastnikom chybalo prezeranie definicii tried, metod dostupne bolo iba prezeranie instancii tried
Zadavanie textu bolo viac menej bezproblemove

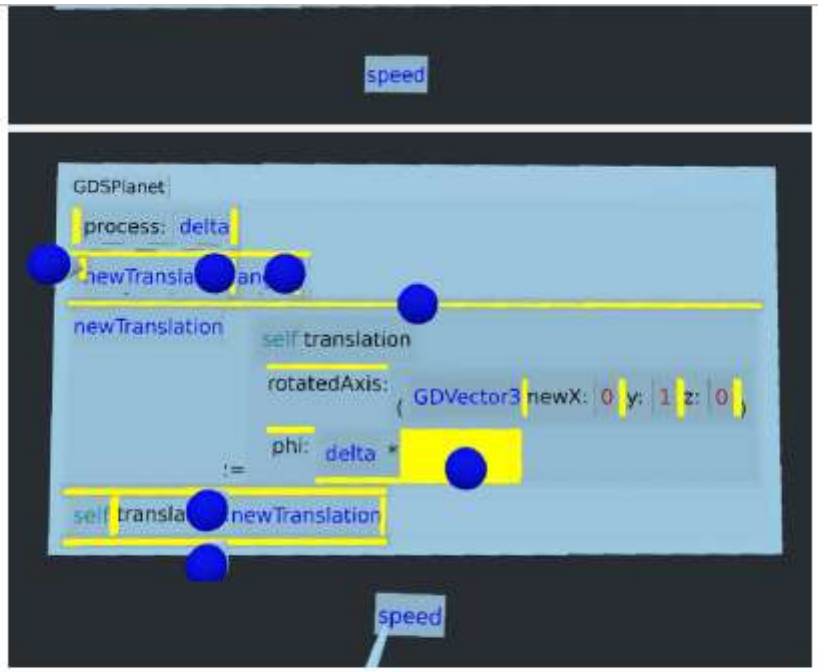


Figure 3. At the top, a collection of blocks forming a method and a single block containing the identifier "speed" are depicted. After picking up the identifier block (below), blue spheres mark possible insert positions in the method.

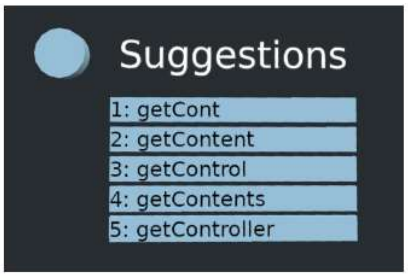


Figure 5. The suggestions menu after the user enters the string "getCont" into a block.

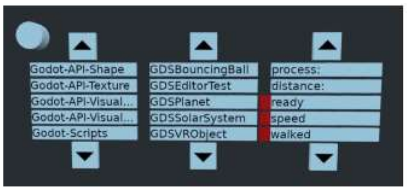


Figure 6. The code browser. The layout is similar to the Squeak/Smalltalk system browser, with code categories on the left, classes in the center, and instance methods on the right. A red button is shown next to subclasses of the VR morph class, allowing users to instantiate the class in the 3D world.

Overcoming Issues of 3D Software Visualization through Immersive Augmented Reality

Link15

kategoria: programovanie/ vizualicacia pomocou softveru city metaphor

1) Ako môže pohlcujúca rozšírená realita pomôcť minimalizovať problémy s použiteľnosťou 3D softvérových vizualizácií?
Ako dobre je podporovaný výber v rôznych 3D softvérových vizualizačných technikách zobrazených v imerznej rozšírenej realite?
Ako čitateľnosť textu ovplyvňuje vývojárov, ktorí používajú rôzne 3D softvérové vizualizačné techniky zobrazené v imerznej rozšírenej realite?
Lepsia navigacia systeme ale nadalej pretrvavany problem s citetelnostou kodu
na vyskum bolo vybrate zobrazenie softveru pomocou city metaphory udajne lepsie na porozumenie softveru
Boli urobene viacere vyskumy s roznymi vizualizacnymi pristupmi
okluzia alias problem s detekovanim objektov bol vyrieseny stolovym view kde uzivatel moze vidiet vsetky budovy viac problem normal ako VR
citatelnost bola zla nakolko zla farba pisma, fakt ze zobrazuje 3D tvare a aj to ze zobrazuje len ked na neho namierime kurzorom (nazvy tried)
clovek bol schopny identifiovat triedy rovnakeho typu na zkalade rovnakych vlastnosti objektu napriklad farba vyska => z coho plynie challenge zobrazit vhodne velke a strukturovane city

Figure 3: The space-time cube visualization technique displayed in immersive augmented reality.

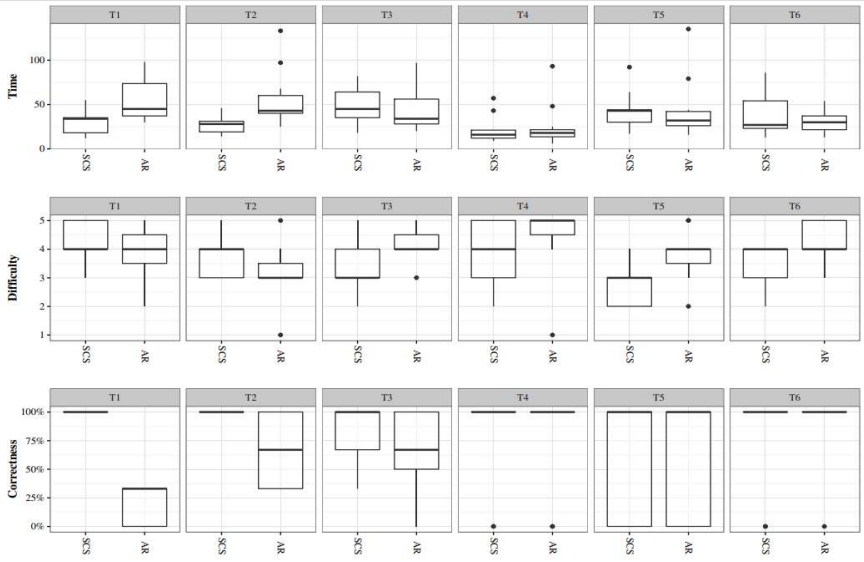
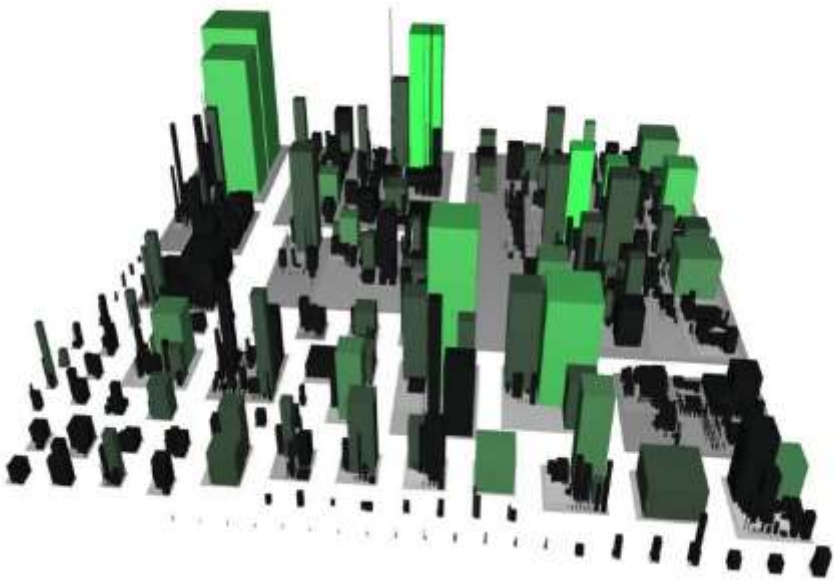
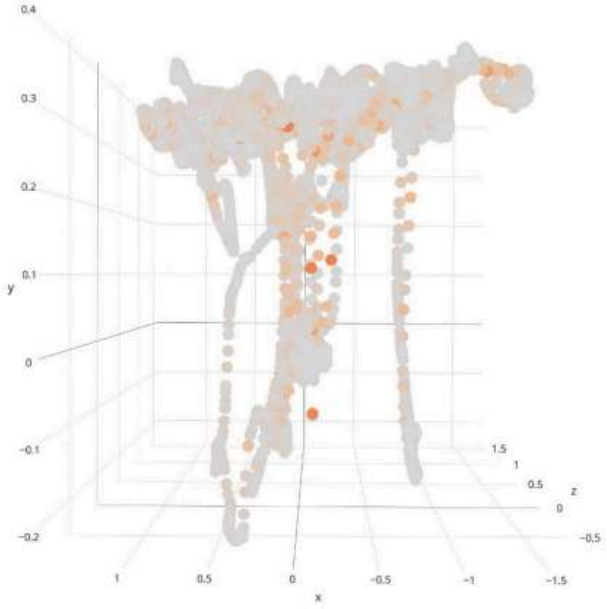


Figure 4: On the top, *time* (in seconds) that participants required to complete the tasks. In the middle, the *difficulty* perceived by participants when completing the tasks. At the bottom, the percentage of *correctness* achieved by participants.

Vysledky narocnosti casu a i. riesenych uloh



(a) The city visualization of Azureus.



(a) Navigation of 3D city visualizations: Standing.

VR-UML: The Unified Modeling Language in Virtual Reality

Link16

kategoria: **programovanie pomocou UML**

Imerzívny zážitok: VR-UML poskytuje pohlcujúci zážitok z modelovania softvéru v 3D priestore.
Vizualizácia a navigácia: Umožňuje intuitívne zobrazenie a navigáciu v komplexných softvérových modeloch a ich vzájomných vzťahoch. Interakcia s modelmi:
interagovanie navigacia vo VR tak aby sa znizil pocit nevolnosti co je casty problem
Podpora hypermodelovania: VR-UML podporuje súčasné zobrazenie viacerých heterogénnych modelov, čo umožňuje hlbšiu analýzu naprieč rôznymi typmi diagramov a záujmami zainteresovaných strán.
Ucastnici ocenili intuitivnost a jasnost struktury
Taktiez sa zobrazovanie vo VR vyhlo opakovaniu tych istych objektov
nevyhodou bola opat citatelnost kodu
takisto nejednotnost v interakcii s VR elementmi
86% ludi suhlasilo ze bola zobrazena kvalistna struktura softveru
71 % ludi uviedlo ze VR modeling je lepsi
Tasks: 1. Multi-diagram elements: which elements with the same name recur in multiple diagrams and how often? 2. Change the attribute “email” in the Customer class from public to private. 3. Change the relation multiplicity between Customer to Shopping Cart to 1-1. 4. Create the Model-View-Controller (MVC) pattern in the class diagram in non-VR (see Fig. 14) and VR-UML

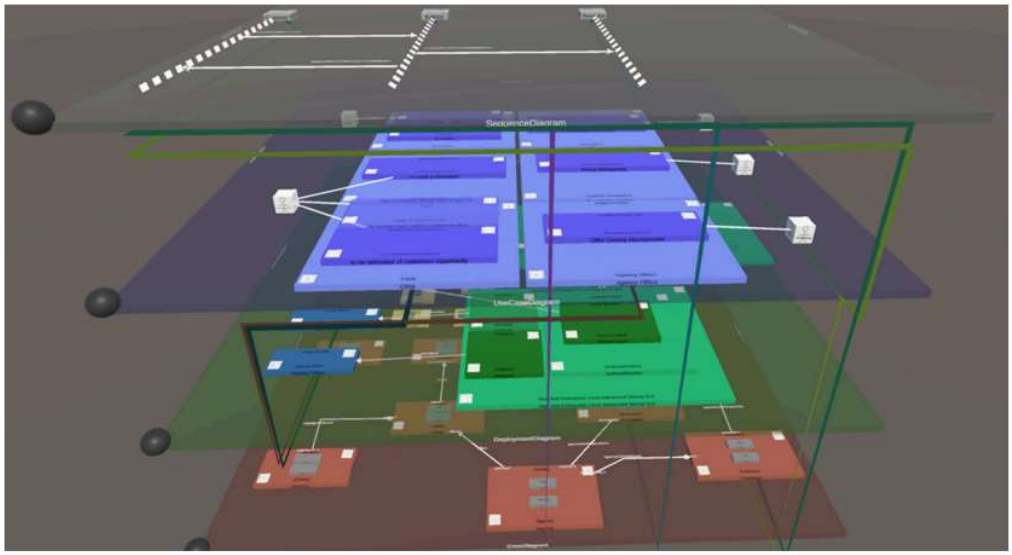


Fig. 7. VR-UML stacked hyperplane visualization of travel agency model.

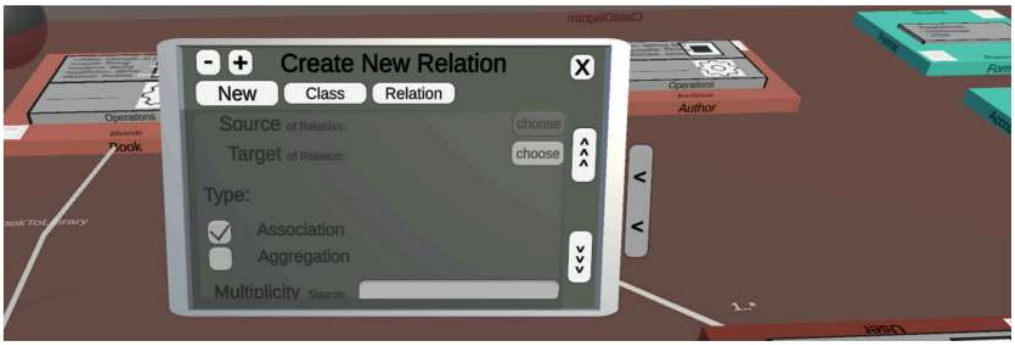


Fig. 9. VR-UML create relation modeling support with virtual tablet.

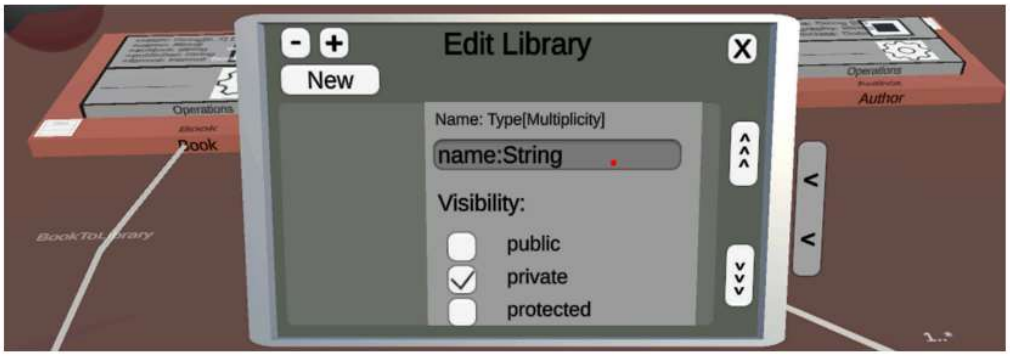
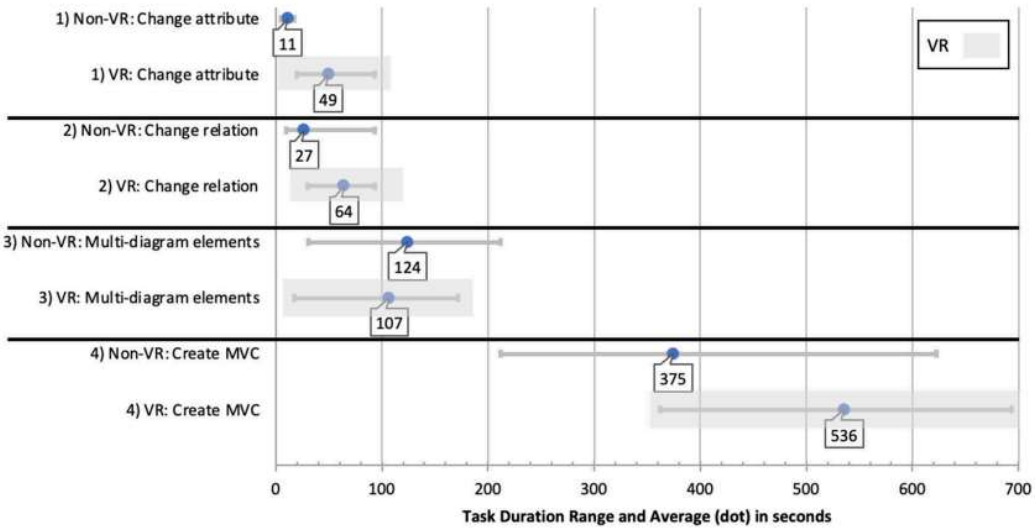


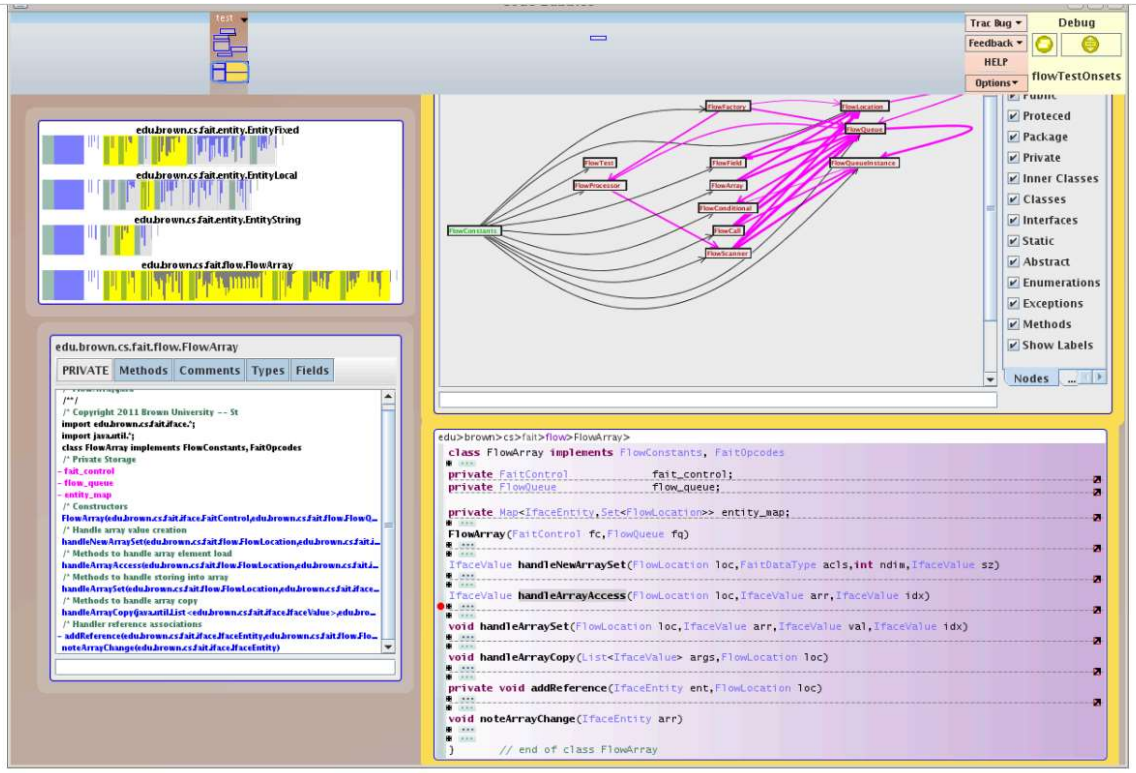
Fig. 10. VR-UML attribute modeling support via virtual tablet.

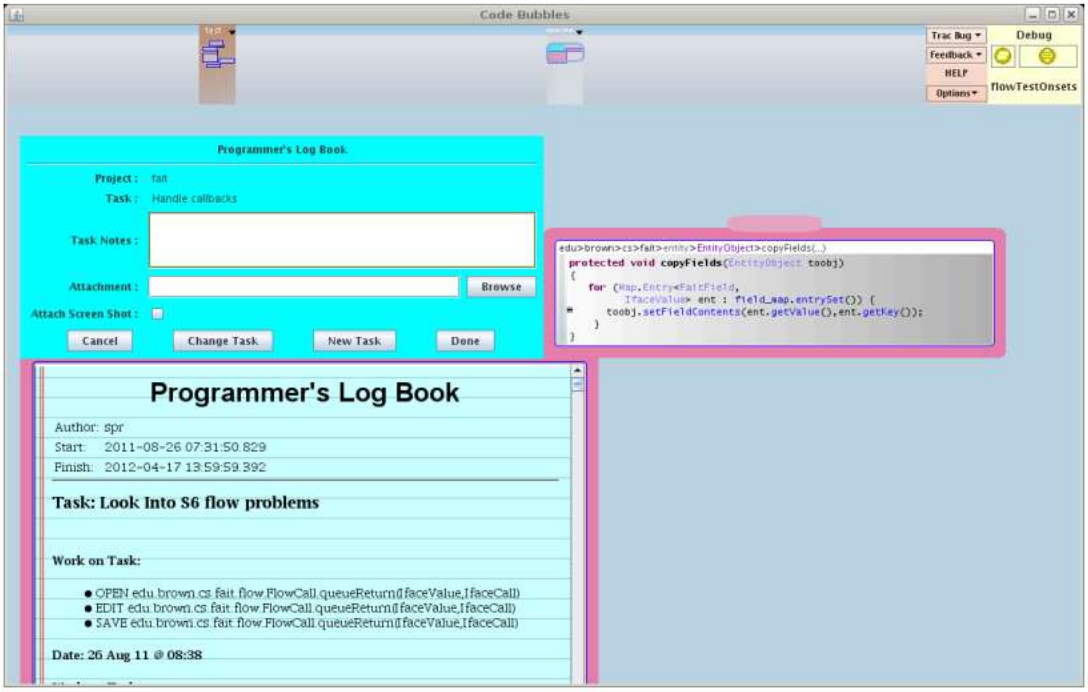


Code Bubbles: A practical working-set programming environment

[Link17]

Code Bubles zobrazuje na platne viacere bublinky roznych tipov (dokumentacia, zdrojovy kod, poznamky)
bubliny sa mozu zoskupovat do roznych skupin
Bubliny su usporiadane
pri debugovani je kazda metoda, trieda, premenna schopna zobrazit sa do jednej bubliny, plus zvlast pole na chyby ...
Pri debugu vnaranim do novych objektov sa vytvaraju nove bubliny
Je mozne kolaborativne programovanie
Program je ulozeny v cloude
Je mozne robit aj Unit testy




Code Bubbles
Trac Bug
Feedback
HELP
Options
Debug
flowTestOnsets
Programmer's Log Book
Project : fait
Task : Handle callbacks
Task Notes :
Attachment :
Browse
Attach Screen Shot :
Cancel
Change Task
New Task
Done
edu>brown>cs>fait>entity>EntityObject>copyFields(...)
protected void copyFields(EntityObject toobj)
{
for (Map.Entry<FaitField,
IfaceValue> ent : field_map.entrySet()) {
toobj.setFieldContents(ent.getValue(),ent.getKey());
}
}
Programmer's Log Book
Author: spr
Start: 2011-08-26 07:31:50.829
Finish: 2012-04-17 13:59:59.392
Task: Look Into S6 flow problems
Work on Task:
OPEN edu.brown.cs.fait.flow.FlowCall.queueReturn(IfaceValue,IfaceCall)
EDIT edu.brown.cs.fait.flow.FlowCall.queueReturn(IfaceValue,IfaceCall)
SAVE edu.brown.cs.fait.flow.FlowCall.queueReturn(IfaceValue,IfaceCall)
Date: 26 Aug 11 @ 08:38

OctoBubbles: A Multi-view interactive environment for concurrent visualization and synchronization of UML models and code

Link18

Octobubles sluzi na vizualizaciu softveru na nizkej urovni az po vysoku
sluzi aj na synchronne programovanie, zmena UML premietne do kodu a naopak
Vdaka Java parseru sme schopni rozbit zdrojovy kod na AST strom
Otazky: Q.1 Aké sú názory používateľov na nápad, použiteľnosť a efektivitu OctoBubbles? Q.2 Aké sú chýbajúce a požadované funkcie?
Prieskume uzivatelia hodnotili pozitivne moznost porozumiet kodu a softveru na nizkej urovni vdaka lahkemu preskakovaniu medzi UML blokmi
Umple umoznuje vytvarat z UML grafu kod v hocijakom jazyku je lepsi v zobrazovani subeznych vizualizacii bo je na dvoch platnach, co Octobubles ani nepodporuje
nedostatok je ze nepodporuje súbežnú vizualizáciu modelov správania softvéru (napr. sekvenčné diagramy) a ich pridruženého zdrojového kódu.

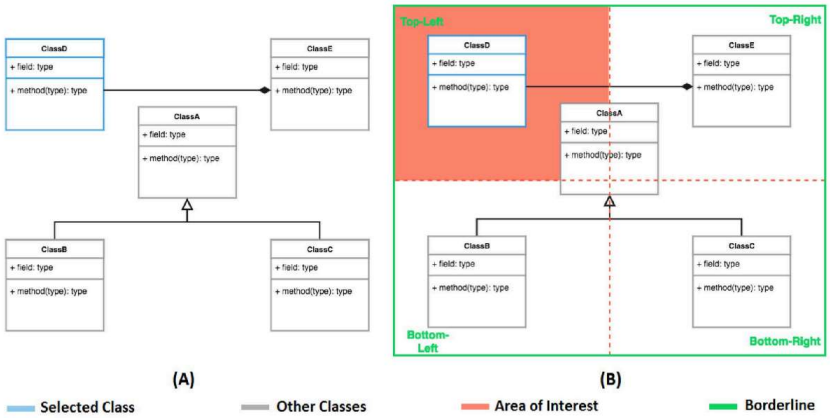


Fig. 2. The scaling approach: steps A and B

TABLE I
PERCEPTIONS REGARDING THE USABILITY OF OCTOBUBBLES

Usability Measurement	Med	Q1	Q3	IQR
Willing to use the system frequently	4	3	4	1
Complexity of the system	1	1	1	0
Ease of use	5	4	5	1
Need of support to use the system	1	1	1	0
Integrity of various functions	5	4	5	1
Inconsistency in the system	1	1	1	0
Intuitiveness	4	4	5	1
Cumbersomeness to use	1	1	2	1
Feeling confident when using the system	4	4	5	1
Required learning-effort	1	1	2	1

Fig. 3. The scaling a

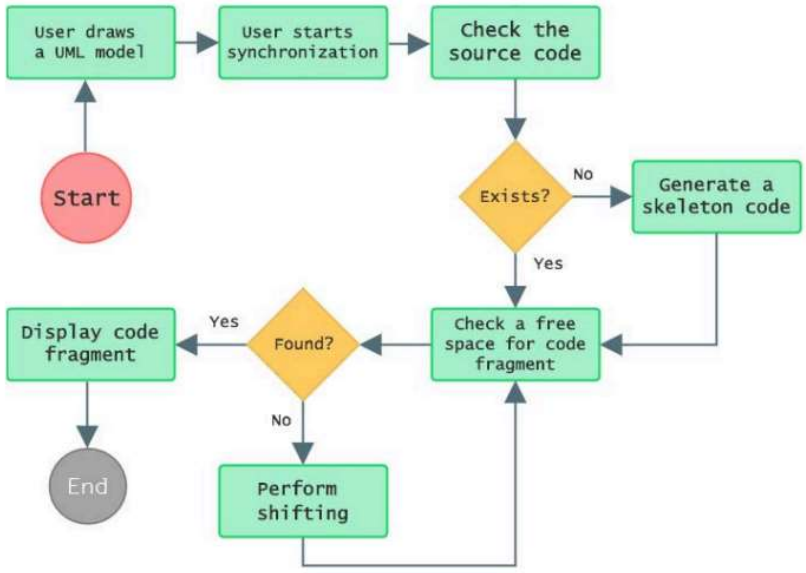


Fig. 4. The synchronization and visualization process of *OctoBubbles*

Generating Java Code from UML Class and Sequence Diagrams

Link19

kategoria: **programovanie pomocou UML** tento

na develop app sa pouziva UML class diagram a par sekvencnych
class diagram nam definuje normal, abstract, inherited classes a takisto aj interfacy
kazda trieda obsahuje informacie o metodach a premennych ktore pouziva
pre kazdu triedu existuje sequence diagram ktory charakterizuje spravanie jej jednotlivych metod a premennych
niekde sa pouzivaju aj len class diagramy na generovanie kostry kodu a niekde aj sequence na generovanie behavioral kodu

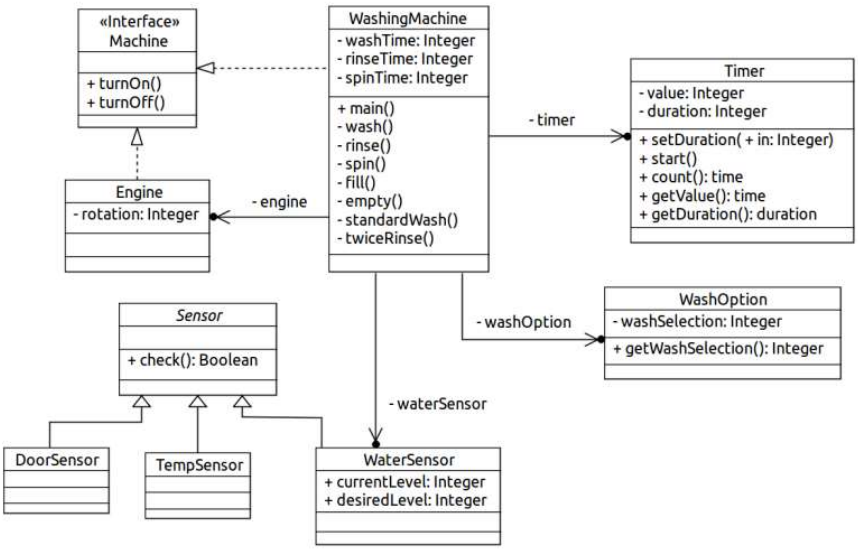
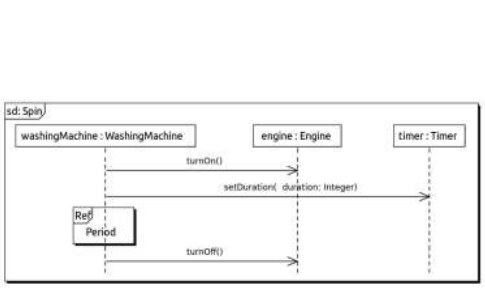
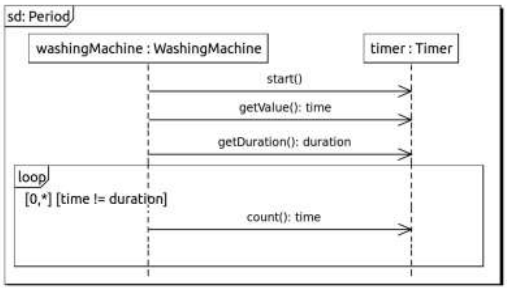


Figure 1. Class diagram - Structural view of Washing Machine.



(a)



(b)

UML executable: A comparative study of UML compilers and interpreters

Link20

kategoria: **programovanie pomocou UML**

snaha docielit vyvoj softveru len pomocou UML => potreba vhodne specifikovat interpretaciu UML pri kompilacii na kod
na vyvoj kodu sa pouziva z UML class, sequence a activy diagram
transformacia UML je generovanim kodu alebo modelovanim
pre Generovanie kodu su jasne definovane pravidla modelovania UML, zmeny v kode su neziaduce jedine v UML => vznika nam pak kod v C
pri modelovani sa UML viac nasobne transformuje az sa z neho postupne vymodeluje nejaky kod,
je dolezite spravne zadefinovat si semantiku a pravidla na tvorenie UML diagramov
posledny obrazok zachytava styri rozne pristupy pri generovani kodu z UML

leting a UML virtual machine.

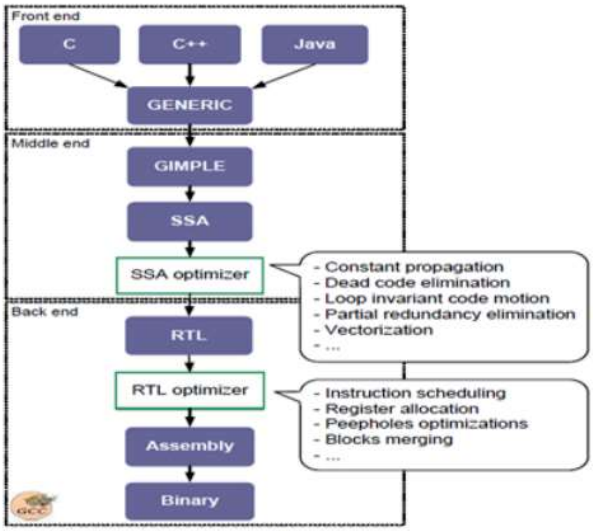


Fig.2. GCC architecture [2].

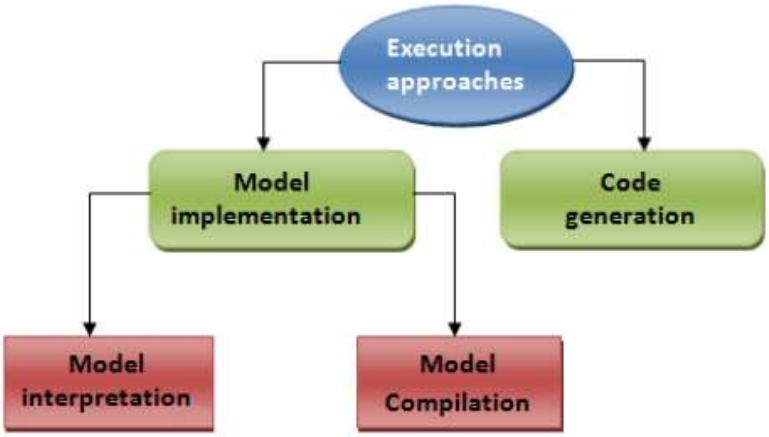


TABLE II. COMPARATIVE ANALYSIS ON EXECUTING UML APPROACHES.

	ACTi interpreter [10].	**UML-compiler [7].**	**A UML Virtual [9].**	**UML compiler (GUML) [2].**
Type	UML virtual machine	UML compiler	UML virtual machine	Front end UML in GCC
Optimization supported	No	No	-eliminates steps for intermediate and final code generation - increases the portability	Reduce assembly code size
Grammar Supported	No	Context free grammar	No	No
fUML supported	No	No	No	Yes
ALF supported	No	No	no	No
OCL supported	No	No	no	No
Extendibility	Yes	Yes	yes	Yes
Analysis Phase	Yes	Yes	No	Performed by UML modelers
Execution phase	No	No	Yes	Yes
Is Compiler	No	Yes	No	Yes
Is Interpreter	Yes	No	Yes	No

[7] I. Chanda, A. Kapiilal, and S. Sengupta, "UML compil

UML to code, and code to UML

Link21

UML je vo forme XML
UML ma stromovu strukturu, listy obsahuju skripty ktore zodpovedaju za vykonanie prekladu
najpouzivanejsie frameworky su MD2 a Axion
Program zlozeny z 3 komponentov
Jeden poskytuje xml mapu (UML) ktorej robi clovek zmeny, potom je tam prevodnik ktory prevedie na zdrojovy kod a nasledne je tam este komponent na sledovanie zmien vo vyslednom kode (kvoli tomu aby app bola reaktivna live)
clovek si moze vytvorit UML z kodu ktoreho chce
code generation funguje vdaka XML iteratiru ktory zoberie XML mapu celu ju prejde a vyberie z nej klucove slove s ktorymi si dalej potom robi
refractoringEngine je zodpovedný za konverziu kódu,
assembler je zodpovedný za písanie kódu
Po zapísaní kódu do príslušných súborov FileListener počúva zmeny v súbore. Ak sa zistí zmenu, odošle udalosť do CodeTracer ktory rozhodne, či zmena ovplyvní kod
Podpora zatial len pre Javu

```
<component>
     <package>package_name/root</package>
     <type>class/interface/abstract class</type>
     <keyword>final</keyword>
     <class_name>Class_name</class_name>
     <access_modifer>public/protected/default</access_modifier>
     <></>
     <field>
          <access_modifer>public/private/protected/default</access_modifier>
          <key_word>volatile/final/static</key_word>
          <Type>Types</Type>
          <array_size>0…*</array_size>
          <identifier>Name</identifier>
          <default_value>default_value</default_value>
     </field>
     …
     <method>
          <access_modifier>public/private/protected/default</access_modifier>
          <key_word>static/synchronized/final</key_word>
          <return_type>type/void</return_type>
          <method_name>Name</method_name>
          <parameters>
               <parameter>
                    <array>
                         <Type>Type</Type>
                         <parameter_name></parameter_name>
                    </array>
               </parameter>
               …
          </parameters>
     </method>
     <extends>
          <parent>
               <parent_type>class/interface</parent_type>
               <parent_name>Component_Name</parent_name>
          </parent>
          …
     </extends>
</component>
```

Fig.1: XML Template

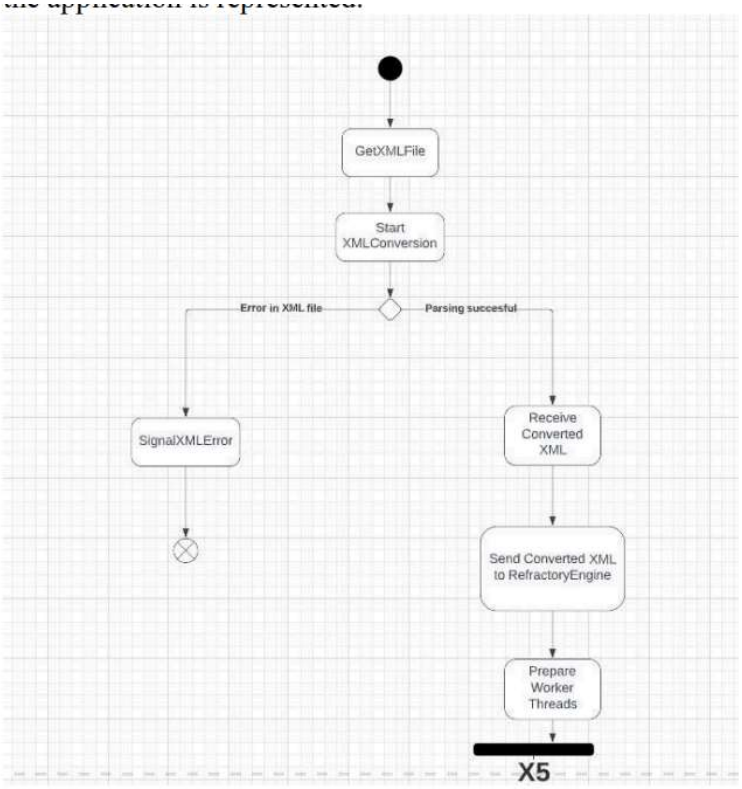
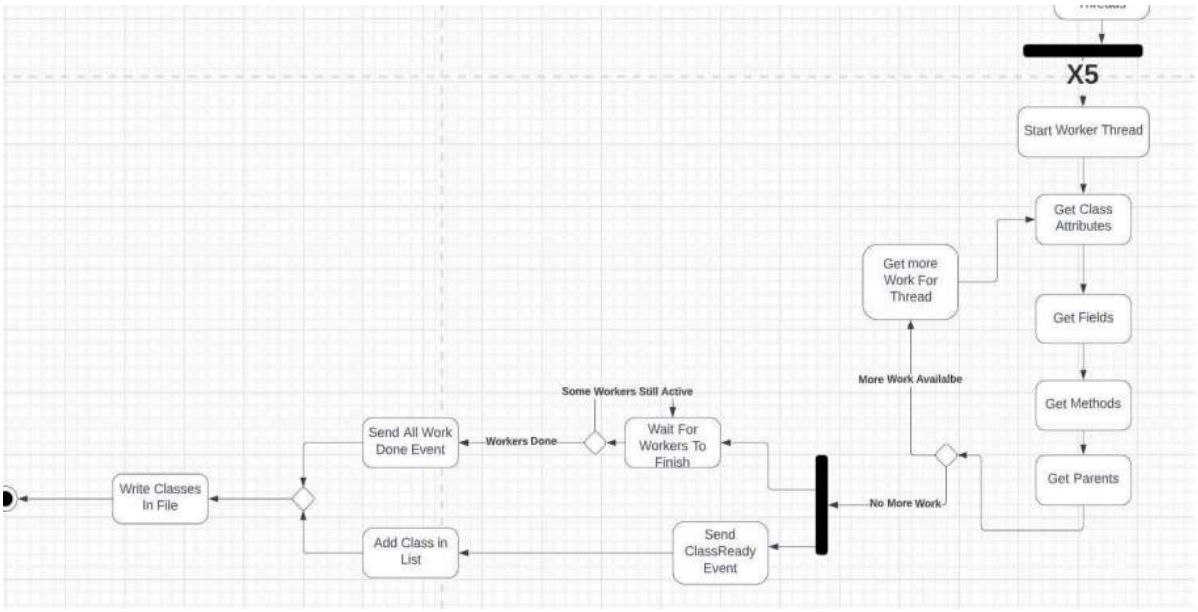


Fig. 4. Activity diagram first page

Postup konverzie UML na kod part 1



Postup konverzie UML na kod part 2

Table II. Experimental results using the resource monitor.

CPU type	Number of classes	CPU consumption average based on time	Time (MS)
I7 11800H	10	0.03	0.028
	100	0.64	0.45
i5 8350U	10	0.64	0.05
	100	2.56	0.75
I7 8550U	10	0.48	0.036
	100	0.7	0.56
AMD Ryzen 7 3700 U	10	0.5	0.03
	100	0.75	0.7

APP konzumacia CPU casu konverzie UML na kod pri roznych procesoch

Unreal vs Unity vs Godot

link22

herne enginy sa pouzivaju na vyvoj hier v 2D alebo 3D
sa naprogramovane pomocou C alebo C++
vyvoj hier ale takisto aj moznost vizualizacie veci do VR alebo AR
Unity ma architekturu zalozenu na komponentoch co umoznuje vysoku skalovatelnost a modularitu
Unity poskytuje siroku moznost uz hotovych komponentov -> vhodne pre zaciatocnikov
Unreal poskytuje siroku paletu filmovych nastrojov vdaka ktorym maju hry krasnu realisticku grafiku
Unreal tvorba hry prebieha formou blueprint kde clovek nemusi mat ziadne znalosti programovania
Godot je open source vdaka comu sa moze komunita ludi podielat na jeho vyvoji a fixovani, ziadna sluzba neni platena
Godot ma stromovu strukturu scen vdaka comu poskytuje intuitivny sposob spravy hernych objektov

XR Interaction Toolkit vs Microsoft Mixed Reality Toolkit

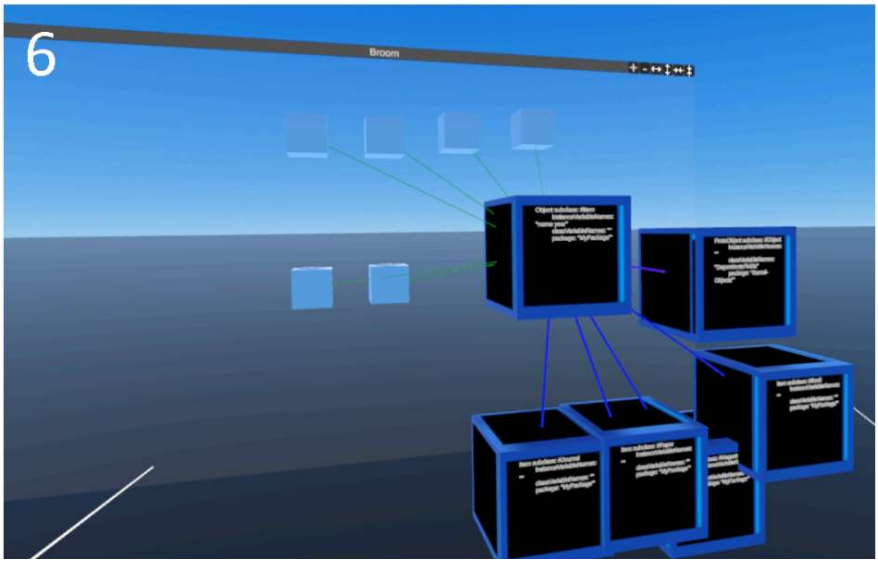
Link23, Link24

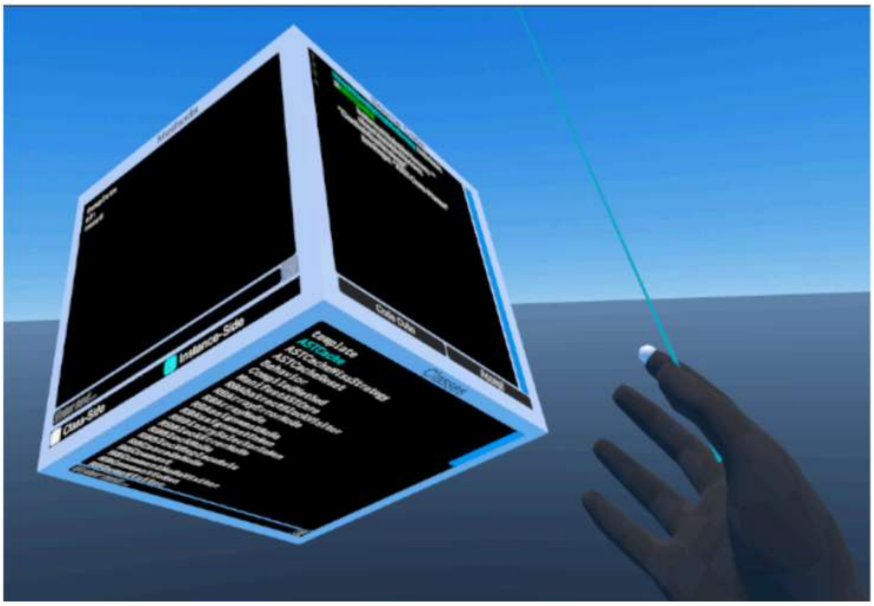
XR poskytuje sadu komponentov vdaka comu vieme vyrobit rozny herny vizual
MRTK poskytuje sadu uz hotovych scen vdaka comu je to idealnejsie pre zaciatocnikov
XR je najviac kompatibilny s UNITY
MRTK je v tomto smere flexibilnejsi podporuje viacere verzie Unity a fixuje viacero svojich verzii
MRTK podporuje zazitky VR a AR pomocou roznych pluginov co mu rozsiruje moznosti zazitkov
XR ma zabudovanu v sebe moznost AR aj VR
MRTK ma viacero verzii ktorym robi technicky support co nie je vyhodne z hladiska stability
XR poskytuje jednu stabilnu verizu ktoru pravidelne aktualizuje
XR narocnejsi na pracu MRTK vhodnejsi pre zaciatocnikov
Meta poskytuje C# rozhranie pre Meta OpenXR runtime
je open source
kompatibilita verzii s projektom (ak je k dispozicii novsia verzia tak ju clovek vie stiahnut bez potreby vytvorit novy projekt a naslednej editacie)
ponuka siroku skalu nastrojov pre pokrocilejsi vyvoj aplikacii
automaticke dotiahnutie vsetkych dependencies
poskytuje cloud ulozisko cize vieme pracovat na vyvoji viaceri
pristup k starsim verziam

VR-Based User Interactions to Exploit Infinite Space in Programming Activities

link25

Objektovo orientovane programovanie pomocou kociek
Kazda kocka predstavuje jednu class
Jazyk zatial len Python a Pharo
Kocka obsahuje zoznam metod triedy , premennych, info o jej zavisloastiach ...
Jedna stena sluzi na editaciu kodu metody, ktoru si vyberieme
IMPLEMENTÁCIA VRIDE je implementovaný pomocou Unity
Komunikácia so serverom sa jednoducho vykonáva pomocou požiadavky POST,
Vidíme, že využitie priestoru je takmer 24 m3 -> vyuzitie nekonecnosti VR
pomocou metly vie objekty naraz odstranovat ...




Instance-Side
Class-Side
template
ASTCache
Behavior
Classes