FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Informační systém týmových tabulí pro agilní vývoj jako rozšíření NetGenia
Student:	Ing. Peter Liptai
Vedoucí:	Ing. Ondřej Guth, Ph.D.
Studijní program:	Informatika
Obor / specializace:	Webové a softwarové inženýrství, zaměření Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2021/2022

Pokyny pro vypracování

Proveďte analýzu a návrh softwarové tabule pro agilní vývoj v týmu. Aplikace bude pracovat nad daty systému NetGenium, konkrétně s uživateli, úlohami a chybami. Důraz dejte na přehlednost. Navržené řešení implementujte jako prototyp a vhodnými prostředky otestujte.

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 10. ledna 2021 v Praze.

Bakalářská práce

INFORMAČNÍ SYSTÉM TÝMOVÝCH TABULÍ PRO AGILNÍ VÝVOJ JAKO ROZŠÍŘENÍ NETGENIA

Ing. Peter Liptai

Fakulta informačních technologií ČVUT v Praze
Katedra softwarového inženýrství
Vedoucí: Ing. Ondřej Guth, Ph.D.
12. mája 2021

Ing. Peter Liptai. *Informační systém týmových tabulí pro agilní vývoj jako rozšíření NetGenia.* Bakalářská práce. České vysoké učení technické v Praze, Fakulta informačních technologií, 2021.

Obsah

Zoznam obrázkov

Zoznam tabuliek

Zoznam výpisov kódu

Chcel by som poďakovať vedúcemu práce Ing. Ondřej Guth, Ph.D., za cenné rady pri tvorbe bakalárskej práce. Ďalej moja vďaka patrí mojej snúbenici Mgr. Alena Lhotová, ktorá mi vytvorila ideálne prostredie na zvládnutie štúdia a práce zároveň. Ďakujem mojim kolegom z Teamu A a to hlavne Ing. Ondej Švorc, Mgr. Karel Fišer, Ing. Jiři Chlouba, Ph.D., Ing. Jan Miléř a ďalším za testovanie programu. V neposlednej rade ďakujem môjmu oponentovi Mgr. Andor Pathó za to že si našiel čas na oponentúru tejto práce.

Prehlásenie

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací. Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č.121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 citovaného zákona.

V Praze dne 10. května 2020 .....

Abstrakt

V práci je vysvětlen současný systém vývoje softwaru ve firmě Dlubal Software s.r.o. Všechny informace o úkolech a chybách jsou evidovány informačním systémem NET Genium. Ve vývojových týmech probíhá komunikace o úkolech pomocí fyzické přehledové tabule umístěné v kancelářích. Na tabulích jsou informace zkopírovány z NET Genia. Práce vysvětluje nedostatky současného řešení. Proto je vypracované nové řešení – program týmových tabulí, který nahrazuje fyzickou tabuli a odstraňuje zjištěné nedostatky současného řešení. V práci je vypracována analýza a implementace programu týmových tabulí. Závěrečná část práce se zabývá testováním použitelnosti, které odhaluje nedostatky návrhu a implementace. Obsahem práce je také návod k nastavení vývojového prostředí a kompilaci programu na nasazení.

Klíčová slova NET Genium, týmová tabule, SCRUM tabule, SCRUM tým, spolupráce při vývoji softwaru, C ++, Qt

Abstract

The thesis explains the current system of the software development process in the company Dlubal Software s.r.o. All information about tasks and bugs is registered in the information system – NET Genium. Developers teams communicate about the tasks and bugs by using a physical overview board located in the offices. All information on board is copied from NET Genium. The thesis explains the shortcomings of the current solution. Therefore, a new solution is developed – a program Team Table, which replaces the physical board and eliminates the identified shortcomings of the current development process. The analysis and implementation of the program are elaborated in the work. The final part of the work deals with usability testing, which reveals the shortcomings of design and implementation. The appendix includes instructions for setting up the development environment and compiling the program for deployment.

Keywords NET Genium, Team Table, SCRUM board, SRCUM team, software development cooperation, C++, Qt

Zhrnutie

Motivácia

Vytvoriť nástoj, ktorý uľahčí každodenné úlohy pri vývoji softvéru. Zníži chybovosť ľudského faktoru a ušetrí čas.

Cieľ práce

Cieľom práce je vytvoriť program tímových tabúľ. Program má zefektívniť prístup informáciám, ktoré sú potrebné na komunikáciu ohľadne úloh vývoja softvéru. Tým program zníži kognitívnu záťaž na členov tímu pri prístupe k daným informáciám. Program má odbúrať nedostatky fyzickej tabule a priniesť nové funkcionality.

Postup

Ako prvé bolo potrebné vytvoriť analýzu problematiky. Zber informácií prebiehal z viacerých vývojárskych tímov, aby výsledné riešenie každému tímu vyhovovalo. V práci bol spracovaný prehlad existujúcich programov, ale žiadny nesplňoval naše požiadavky. Práve pre to bol vytvorený návrh programu, ktorý bol implementovaný. Program bol nasadený a bola prevedený test používateľnosti.

Výsledky práce

Výsledkom je program, ktorý je nasadený vo vývoji. Program je používaný denne viac ako 50 ľudmi, ktorý sú s ním spokojný a pomáha im v komunikácií ohľadne úloh.

Záver

Podarilo sa urobiť analýzu, ktorá zachytáva potreby všetkých tímov. Z tejto analýzy vytvoriť návrh riešenia a ten implementovať. Program sa podarilo nasadiť do vývoja a otestovať. Výsledky testu poukázali na nedostatky a potrebné rozšírenia, ktoré bude v budúcnosti potrebné zapracovať. Avšak žiadny z nedostatkov nie je v rozpore z navrhnutým riešením a bude jednoducho implementovaný.

Zoznam skratiek

CRUD	Create, Read, Update and Delete
RGB	Red Green Blue

Slovník termínov

code review	posudok kódu
merge	operácia GIT Merge
merge message	súhrná správa obsahujúca všetky lístky, ktoré sú mergované do vývojovej vetvi grandmaster
SCRUM	metodika agilného vývoja
SCRUM Master	vedúca rola člena tímu určená metodikou SCRUM
RGB model	RGB je aditívny farebný model, pri ktorom svetlo želanej farby vzniká zmiešaním červeného, zeleného a modrého svetla vhodnej intenzity.
statusMapping	stav lístku

Prekladový slovník

ticket	lístok
fixing	opravuje
has task	má task
factory pattern	návrhový vzor továreň

Kapitola 1
Úvod

Vývoj softvéru veľkých projektov vyžaduje kooperáciu viacerých ľudí. Jeden z kľúčových faktorov efektivity procesu vývoja je komunikácia. V súčastnosti je na komunikáciu vo firme Dlubal Software s.r.o. využívaná fyzická priehľadová tabula umiestená v kancelárií. Toto riešenie má mnoho nedostatkov, ktoré sú popísane v kapitole 4.1. Najzásadnejšia nevýhoda pramení zo súčasnej globálnej pandémie, kedy bola zamestnancom v profesiách, ktoré to umožňujú, nariadená práca z domova. Vývoj softvéru z domova možný je, avšak presun zamestnancov z kancelárií mal za následok absenciu a nedostupnosť spomínanej tabule. Informácie nutné na komunikáciu, ktoré boli prehľadnou formou dostupné na tabuli, sú v súčastnosti dostupné iba v menej prehľadnej forme v informačnom systéme NET Genium. Táto situácia spomaľuje vývoj a pridáva kognitívnu záťaž jednoduchým a často opakovaným akciám spojených s komunikáciou pri vývoji softvéru.

Nedostupnosť tabule zkomplikovala komunikáciu vo vývoji tímu autora tejto práce, preto sa tomuto problému rozhodol venovať vo svojej bakalárskej práci. Navrhuje a vytvára preto program tímových tabúľ, ktorý ponúkne informácie dostupné na fyzickej tabuli zamestnancom pri práci z domova.

Práca vysvetľuje súčasné riešenie komunikácie v kapitole 2. Následne sú predstavené existujúce riešenia problému tímových tabúľ v kapitole 3. Je vysvetlené prečo žiadne existujúce riešenie nevyhovuje. Preto je navrhnuté vlastné riešenie tímovej tabule, ktoré je analyzované a navrhnuté v kapitole 4, na ktorú nadväzuje popis implementácie programu tabule v kapitole 5. Implementácia je nasadená a otestovaná v kapitole 6.

Kapitola 2
Súčasné riešenie

Súčasné riešenie opisuje podporu organizácie vývoja na projekte RFEM6. V súčastnosti na tomto projekte pracuje 150 zamestnancov, ktorý sú rozdelený do ôsmych tímov.

V rámci tímu prebieha intenzívna komunikácia ohľadne aktuálnej práce. Na každej pracovnej úlohe spolupracujú minimálne traja zamestnanci. Medzi tímová komunikácia je menej častá. Typicky nastáva pri prácach, ktoré ovplyvňujú časti projektu, za ktoré sú zodpovedné iné tímy.

Primárny nástroj na tímovú komunikáciu je fyzická tabula, ktorú môžme vidieť na obr. 2.1, umiestnená v každom tíme. Na tejto tabuli sú informácie o stave rozpracovaných úloh a osobách, ktoré na nich pracujú. Pokrok na úlohách spúšťa procesy nastavené v tíme, preto je dôležité, aby informácie o stave boli dostupné rýchlo a prehľadne.

Ďalší použitý nástroj je informačný a komunikačný portál NET Genium. Tento nástroj z hľadiska vývoja slúži ako register všetkých úloh a ich histórie. Všetky úlohy na tabuli sú iba kópia dát v NET Geniu, ktoré je zodpovedné za ich archiváciu.

2.1 Popis aplikácie NETGenium

„NET Genium je informační portál, který plní úlohu inteligentního úložiště dat veškerých firemních agend a slouží jako nástroj pro firemní komunikaci.“[16]

2.1.1 Základné stavebné prvky

Z hľadiska tejto práce sa budeme zaoberať dvoma stavebnými prvkami NET Genia, a to editačnými formulármi a náhľadovými tabuľkami. NET Genium obsahuje ďalšie prvky, avšak z hľadiska tejto práce niesu relevantné.

Editačný formulár slúži na pridanie, úpravu alebo odstránenie práve jedného záznamu v databázovej tabuľke. Typicky nie sú všetky atribúty editovatelné pomocou formuláru. Napríklad dátum, čas a autor úpravy sú typicky vyplnené automaticky pri každej zmene. Tabuľky môžu mať zapnutú funkciu histórie. Každá zmena sa ukladá ešte v tabuľke s rovnakým názvom, ktorý je rozšírený príponou „_history“.

Náhľadová tabuľka slúži na zobrazenie jednej tabuľky z databázy. Tento ovládací prvok poskytuje možnosti filtrácie a vyhľadávania.

Pomocou týchto dvoch prvkov vieme vykonávať všetky CRUD operácie nad dátami jednej tabuľky.

Obr. 2.1 Tabula v tíme A

2.1.2 Použitie na podporu vývoja

Práca na projekte je rozdelená na dva hlavné typy úloh – Todos a Bugs. Tieto typy sú reprezentované tabuľkami v databázy. Na prácu s nimi používame editačný formulár a náhľadovú tabuľku popísanú v kapitole 2.1.1.

Todo je zamerané na nový vývoj. Každé Todo musí mať určený práve jeden z typov:

Task – vývoj novej atomickej funkcionality,

Story – vývoj novej funkcionality, môže byť rozdelený na menšie atomické Task,

Refactoring – refactoring už hotovej funkcionality,

Others – scripty automatického testovania, jednotkové testy, atď.

Práca typu **Bug** je zameraná na opravu už hotových funkcionalít. Bug musí mať určený práve jeden z typov:

Branch – oprava chyby, ktorá sa prejavuje vo vývojovej vetve, ktorá ešte nebola namergovaná do vývojovej vetvy „grandmaster“. Branch bug môže byť pridelený konkrétnemu Todo, v prípade ak sa chyba vyskytuje práve v ňom.

Global – oprava chyby, ktorá sa prejavuje vo vývojovej vetve „grandmaster“.

Obe tieto tabuľky majú zapnutú históriu, ktorá je mimoriadne dôležitá pre tímové postupy práce.

Todos a Bugs tabuľky majú analogické atribúty:

ID – unikátne identifikačné číslo

Type – typ Todo Task, Story, Refactoryng, Other alebo Bugu Branch, Global

Status – stav, v ktorom sa úloha nachádza

Subject – krátky textový popis úlohy

Team – tím, ktorému je úloha pridelená

Programmer – osoba, ktorá úlohu implementuje a Merguje

Tester – osoba, ktorá vykonáva testovanie

Reviewer – osoba, ktorá vykonáva code review implementácie

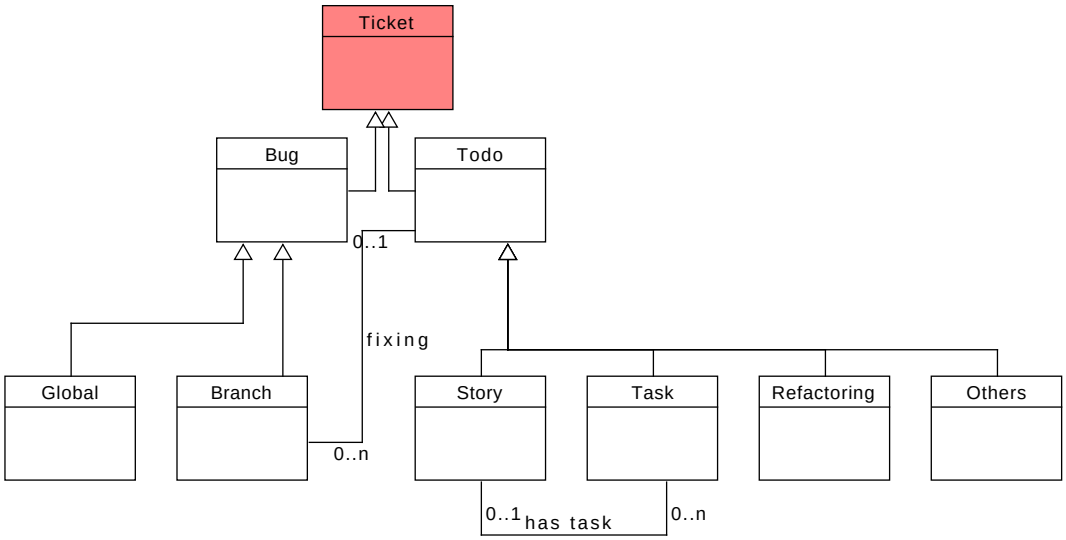
Informácie v daných atribútoch sú zachytené na lístkoch na fyzickej tabuli. Podla týchto informácií je jednoduché Todo alebo Bug v NET Geniu vyhladať. Zároveň je na lístkoch zachytená aj história zmien stavu.

2.2 Popis fyzickej tímovej tabule

Fyzická tabula umiestnená v tíme sa skladá zo šiestich zón, do ktorých sú umiestňované jednotlivé lístky.

2.3 Hierarchia lístkov

Hierarchia lístkov – Story môže byť rozdelené na viac lístkov typu Task. Todo lístky môžu obsahovať Bugy typu Branch. Viz doménový model hierarchie lístkov obr. 2.2.



Obr. 2.2 Doménový model hierarchie lístkov

2.4 Lístok na fyzickej tabuli

Farba je najdôležitejšia vlastnosť každého lístku. Reprezentuje väzby na ostatné lístky na tabuli. Závislé lístky majú spravidla rovnakú farbu, čo pomáha veľmi rýchlej orientácii v stave rozpracovanosti celku. Všetky Bugy typu Global majú rovnakú červenú farbu. Bugy typu Branch, ktoré nemajú závislosť na žiadnom Todo, majú farbu bledo červenú.

Prednú stranu lístka tvorí skratka typu Todo alebo Bugu, identifikačné číslo a krátky textový popis.

Príklad prednej strany lístka typu User Story s identifikačným čislom 1689 a popisom „Results in main view“:

```
US-1689: Results in main view
```

Na zadnú stranu lístku sa zapisuje história pohybu po tabuli. Pri každom pohybe lístku je na zadnú stranu napísaný zápis presunu. Vo forme iniciál osoby, ktorá lístkom posúva a stav, do ktorého lístku dáva.

Príklad zadnej strany:

```
P.L. - Fixed
K.F. - Review
P.L. - Fixed
M.M. - ToMerge
P.L. - MergedInBranch
K.F. - Tested
```

Tested je posledný stav, v ktorom sa lístky nachádzajú. V tomto stave je úloha už namergovaná vo vývojovej vetve grandmaster.

2.5 Zóny fyzickej tabule

2.5.1 Tabuľka

Tabuľka je dominantný prvok na tabuli. Riadky určujú člena tímu, ktorý sa zaoberá lístkom. Stĺpce určujú stav lístka.

2.5.2 Zásobník naplánovaných lístkov

Do tohto zásobníku pridáva SCRUM Master alebo iná poverená osoba lístky, na ktorých sa bude pracovať v najbližšej dobe.

2.5.3 Zásobník rozpracovaných lístkov

Do tohto zásobníku sa presúvajú lístky z panelu naplánovaných lístkov 2.5.2 v momente, keď sa na nich začne pracovať. Pri tomto posune sa musí vyrobiť kópia lístku, ktorá je umiestnená do tabuľky 2.5.1. Do tejto kópie sa zapisuje na zadnú stranu história posunov. V tomto zásobníku sa nachádzajú iba nadradené lístky.

Príklad: Ak sa v tomto zásobníku nachádza lístok typu User Story, ktorý je nadradený iným lístkom typu Branch Bug alebo Task, je v tomto zásobníku iba nadradený lístok User Story.

2.5.4 Zásobník lístkov aktuálne mimo tím

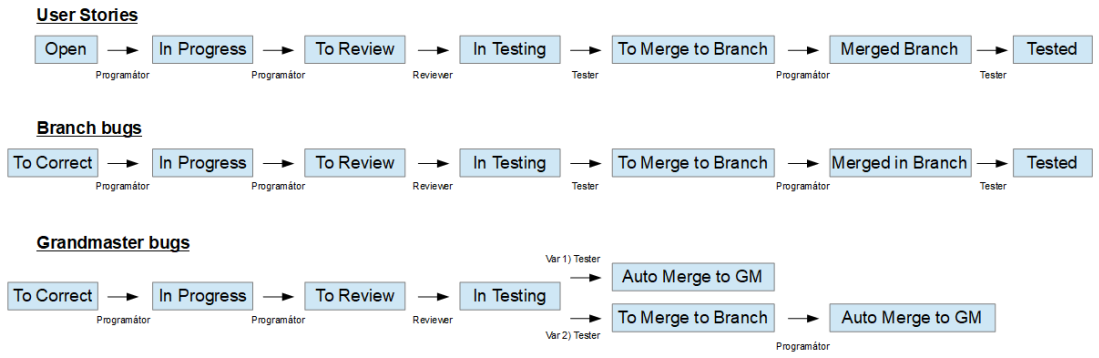
Zásobník sa nachádza v tabuľke 2.5.1. V tomto priestore sú lístky, na ktorých pracuje aktuálne osoba, ktorá nie je člen tímu. Typicky môže isť o testovanie alebo Code Review.

2.5.5 Zásobník lístkov namergovaných v team master vetvi

Obsahuje lístky, ktoré sú namergované v team master vetvi.

2.5.6 Pravidlá postupnosti posúvania lístkov

Na obrázku 2.3 je odporúčaná postupnosť stavov lístka a role, ktoré stav menia.



Obr. 2.3 Pravidlá postupnosti stavov lístkov

2.5.7 Pravidlá zápisu informácií na lístky

V tejto zóne sú príklady, ako má vyzerať predná strana lístkov jednotlivých typov Todo a Bug. Ďalej sú tu vypísané skratky stavov a príklad korektného zápisu histórie presunov po tabuli, ktorá sa má nachádzať na zadnej strane lístku.

Kapitola 3
Existujúce riešenia

Na trhu existuje mnoho riešení na podporu komunikácie pri agilnom vývoji softvéru. V tejto kapitole budú jednotlivé riešenia vymenované a v priehľadovej tabuľke bude prehľad splnenia potrebných funkcií a vlastností ktoré požadujeme.

3.1 Kritéria vyhľadávania existujúcich riešení

Do výberu boli zaradené riešenia ktoré obsahujú užívateľské rozhranie podobné fyzickej tabuli. Sú to hlavne programy ktoré poskytujú riešenie pre agilný vývoj a majú užívateľské rozhranie „Scrum Board“ alebo „Kanban board“.

3.2 Potrebné funkcie a vlastnosti

Užívateľské rozhranie použitého nástroja musí byť graficky podobné fyzickej tabuli.

Ovládanie prvkov užívateľského rozhrania musí byť intuitívne ekvivalentné užívateľským skúsenostiam s prácou s fyzickou tabulou.

Vlastné rozhranie umožňujúce bezpečnú obojstrannú synchronizáciu dát v NET Geniu. Všetky dáta vytvorené v nástroji k úlohám musia byť v NET Geniu.

3.3 Prehľad existujúcich riešení

Prehľad existujúcich riešení je v tabuľke 3.1. Stĺpce označujú potrebné funkcie a vlastnosti, jednotlivé riadky sú existujúce riešenia.

3.4 Záver

Z prehľadu v tabuľke 3.1 je zrejmé, že žiadne existujúce riešenie plne nespĺňa všetky požiadavky na tímovú tabuľu. Najzásadnejší problém všetkých spomenutých riešení je veľké množstvo ďalších funkcionalít. Tieto funkcionality by bolo potrebné zakázať, čo nie je možné, pretože používanie funkcionalít generuje dáta. Všetky tieto dáta nie je možné ukladať do NET Genia. Tak tiež nebolo nájdené riešenie, ktorého užívateľské rozhranie odpovedá zaužívaným procesom na tímovej fyzickej tabuli.

Pre nevhodnosť existujúcich riešení bolo rozhodnuté, že bude vypracované vlastné riešenie programu tímových tabúľ.

Tabuľka 3.1 Prehľad existujúcich riešení

Produkt	Užívateľské rozhranie				Ovládanie	Vlastné rozhranie
	Farby lístkov	Riadky členov tímu	Stĺpce stavov	Postupnosť stavov		
monday.com	✗	✗	✓	✗	✗	✓
www.smartsheet.com	✓	✗	✓	✗	✗	✓
www.meistertask.com	✗	✗	✓	✓	✗	✓
kanbanize.com	✓	✗	✓	✗	✗	✓
hubstaff.com	✗	✗	✓	✓	✗	✗
clickup.com	✗	✗	✓	✓	✗	✓
www.wrike.com	✗	✗	✓	✗	✗	✓
trello.com	✗	✗	✓	✓	✗	✓
asana.com	✗	✗	✓	✓	✗	✓
kanbantool.com	✗	✗	✓	✓	✗	✓
www.planview.com	✓	✗	✓	✓	✗	✓
www.zoho.com	✗	✗	✓	✗	✗	✗
kanbanflow.com	✓	✓	✓	✓	✗	✓

Kapitola 4
Analýza a návrh

Predmetom tejto kapitoly je analýza a návrh aplikácie digitálnej verzie tímovej tabule.

4.1 Nevýhody súčasného riešenia

- Strata času:
  - Manuálna výroba lístkov.
  - Duplicitná práca pri posunoch na tabuli. Je potrebné presun zapísať na lístok a zároveň zmeniť v NET Geniu.
- Chybovosť – bežné chyby pri práci s tabuľou:
  - Chybne opísané čísla lístkov – komplikuje hľadanie Todo alebo Bugu v NET Geniu.
  - Zlý formát popisu.
  - Nevyplnená / chybne vyplnená história.
  - Posun lístkov na nevhodné destinácie. Napriek vyznačeným tímovým pravidlám postupnosti posunov.
- Potrebné sledovať skladové zásoby kancelárskych potrieb. A to:
  - farebné papiere,
  - farebné špendlíky,
  - písacie potreby.
- Nedostupnosť tabule pri práci z domova.

4.2 Hlavné procesy aplikácie

Aplikácia má slúžiť ako digitálna náhrada fyzickej tabuli umiestnenej v jednotlivých tímoch. Musí nahradiť kompletnú funkcionalitu fyzickej tabule a zároveň odstrániť nevýhody používania fyzickej tabuli.

4.2.1 Čítanie informácií z tabule

Program zobrazí tabule všetkých tímov, ktorých je užívateľ členom.

Zobrazené lístky musia obsahovať všetky informácie, ktoré boli dostupné na fyzickej tabuli. Musia rešpektovať farebnosť podla hierarchie lístkov zachytené na obr. 2.2.

4.2.2 Modifikácia Todo a Bug

Presun lístka upravuje dáta v NET Geniu, odbúrava potrebu dvojitého nastavovania. Každý posun vytvára záznam do histórie.

4.2.3 Merge do vetvi grandmaster

Presun všetkých lístkov z tímovej master vetvi do vetvi grandmaster.

Pri operácií merge do vetvi grandmaster, viz kapitola 4.2.3, je potrebné informovať ostatné tímy o obsahu nových funkcionalít a opravách chýb, ktoré sa do vetvi grandmaster dostanú. Tento proces vygeneruje správu o všetkých lístkoch v tímovej vetve master.

4.3 Špecifikácia požiadaviek

4.3.1 Funkčné požiadavky

4.3.1.1 F01 – Detekcia prihláseného užívateľa

Program bude spúšťaný z firemných počítačov, do ktorých sa prihlasujú zamestnanci pomocou doménových účtov. Užívateľ programu bude vždy identický s prihláseným užívateľom operačného systému.

4.3.1.2 F02 – Zobrazenie tabúľ

Zobrazenie tabúľ tímov, ktorých je užívateľ členom. Podla počtu nájdených tímov môžu nastať tri scenáre:

- **Užívateľ nie je členom žiadneho tímu.** Program vypíše chybovú hlásku a vypne sa.
- Užívateľ je členom **viac ako jedného** tímu. Program zobrazí prvú tabulu v abecednom poradí. V nastavení programu je možné prepnúť tím, po jeho prepnutí sa zobrazí vybraná tabula.
- Užívateľ je členom **práve jedného** tímu. Program zobrazím tabuľu daného tímu.

4.3.1.3 F03 – Zobrazenie lístkov prehľadnou formou

Pravidlá farebnosti lístkov sú rovnaké ako na fyzickej tabuli opísané v kapitole 2.4. Je potrebné zvoliť farby tak, aby boli od seba dostatočne odlišné.

Podla uskutočneného testovania sa ukazuje, že maximálny počet dostatočne vizuálne odlišných farieb, je 59. Tento počet nezahŕňa červenú farbu pre Global Bug a bledo červenú farbu pre nepriradený Branch Bug.

Pri používaní programu sa možno zistí, že daný počet farieb je pre niektoré tímy nedostatočný. To znamená, že na ich tímovej tabuli je potrebný počet rôznych farieb väčší ako 59. Preto je potrebné v implementácií myslieť na možnosť jednoduchej zmeny systému prideľovania farieb.

Text popisu lístkov musí byť vždy čitateľný. Popis lístkov zostáva identický s popisom prednej strany na fyzickej verzii tabule. Informácie o histórií posunov zo zadnej strany lístku sa budú zobrazovať ako tooltip.

4.3.1.4 F04 – Nastavenie tímovej tabule

Postup práce na Todo a Bug sa v jednotlivých týmoch môže líšiť. Preto je potrebné, aby si každý tím mohol nastaviť vlastné pravidlá vývoja. Tieto pravidlá sú zachytené podmnožinou stavov, ktoré tím používa (z celkovej množiny stavov v NET Geniu) a ich postupnosťou.

Postupnosť stavov sa môže líšiť pre rôzne typy lístkov.

4.3.1.5 F05 – Podpora presúvania lístkov z hľadiska požadovanej tímovej postupnosti stavov

Na základe pozorovaného problému chybných presunov lístkov po tabuli opísaného v kapitole 4.1 je potrebné, aby program tímovej tabule užívateľovi radil vhodné destinácie pri pohybe lístku po tabuli podla nastavených pravidiel.

Každý posun je následok odvedeného úkonu. Niektoré úkony, ako code review alebo testovanie, môžu mať výsledok pozitívny, alebo negatívny – „Našla sa chyba, opraviť!“, alebo „Všetko v poriadku, lístok môže ísť ďalej.“ V týchto situáciách sú destinácie pre pozitívny výsledok vyznačené farbou zelenou a pre negatívny farbou červenou.

Tímová tabula podporuje aj posun viacerých lístkov naraz, čo je použité pri merge do vetve grandmaster.

4.3.1.6 F06 – Generovanie merge message

Program vygeneruje súhrnnú správu lístkoch, ktoré sú namergované v tímovej master vetve.

4.3.2 Nefunkčné požiadavky

4.3.2.1 N01 – Modifikovateľnosť zdroju tímových nastavení

Pri implementácii je potrebné zaručiť jednoduchú zmenu zdroju dát nastavení do NET Genia.

4.3.2.2 N02 – Výkonnostné požiadavky

Program beží ako desktopová aplikácia. Čerpá dáta z databázy NET Genia, fungujúcej na technológií SQL Server.

NET Genium je vyťažená webova aplikácia, preto je potrebné, aby ju program tímových tabúľ zaťažoval minimálne. Všetky dáta, okrem Todo a Bug, môže z databázy čítať iba raz za svoj beh. Aktualizáciu dát Todo a Bug ovláda užívateľ tlačidlom „Update“.

Program číta dáta Todo a Bug vždy iba pri vyvolaní aktualizácie. Dáta sú následne lokálne filtrované do jednotlivých častí digitálnej tabule.

Záťaž 150 aktívnych užívateľov tímovej tabule môže spôsobiť zníženie rýchlosti odozvy NET Genia maximálne o 100 ms v priemere zo meraných 1000 akcií.

Užívateľské rozhranie tímovej tabule nesmie mat pri akejkoľvek operácií odozvu dlhšiu ako 1 s. Pri nečinnosti nesmie využívať hardwarove zdroje (aktívne čakať).

4.3.2.3 N03 – Bezpečnostné požiadavky

Programu tímovej tabule na prístup k dátam do NET Genia využije prihlasovací token užívateľa Windowsu [19]. Týmto spôsobom bude aplikácia prístupná iba pre užívateľov, ktorý z firemných

doménových účtov. Zároveň v sebe program nebude mať žiadne citlivé údaje, ktoré by bolo potrebné šifrovať.

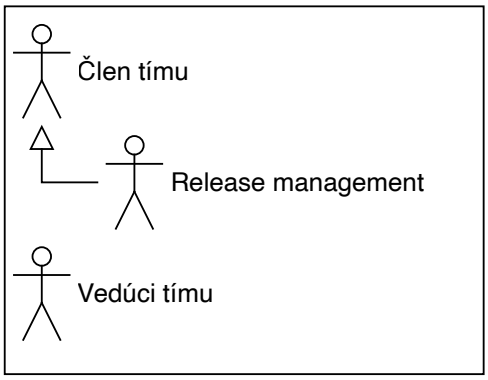
4.3.2.4 N05 – Čítanie dát z NET Genia

Program musí byť schopný čítať dáta o Todo, Bugs, užívateloch a tímoch z NET Genia.

4.3.2.5 N06 – Zápis dát do NET Genia s generovaním korektnej histórie

Program musí generovať korektnú históriu, tak ako keby boli zmeny vykonané v NET Geniu.

4.4 Aktéri



Obr. 4.1 Hierarchia aktérov modelu prípadov užitia

4.4.1 Člen tímu

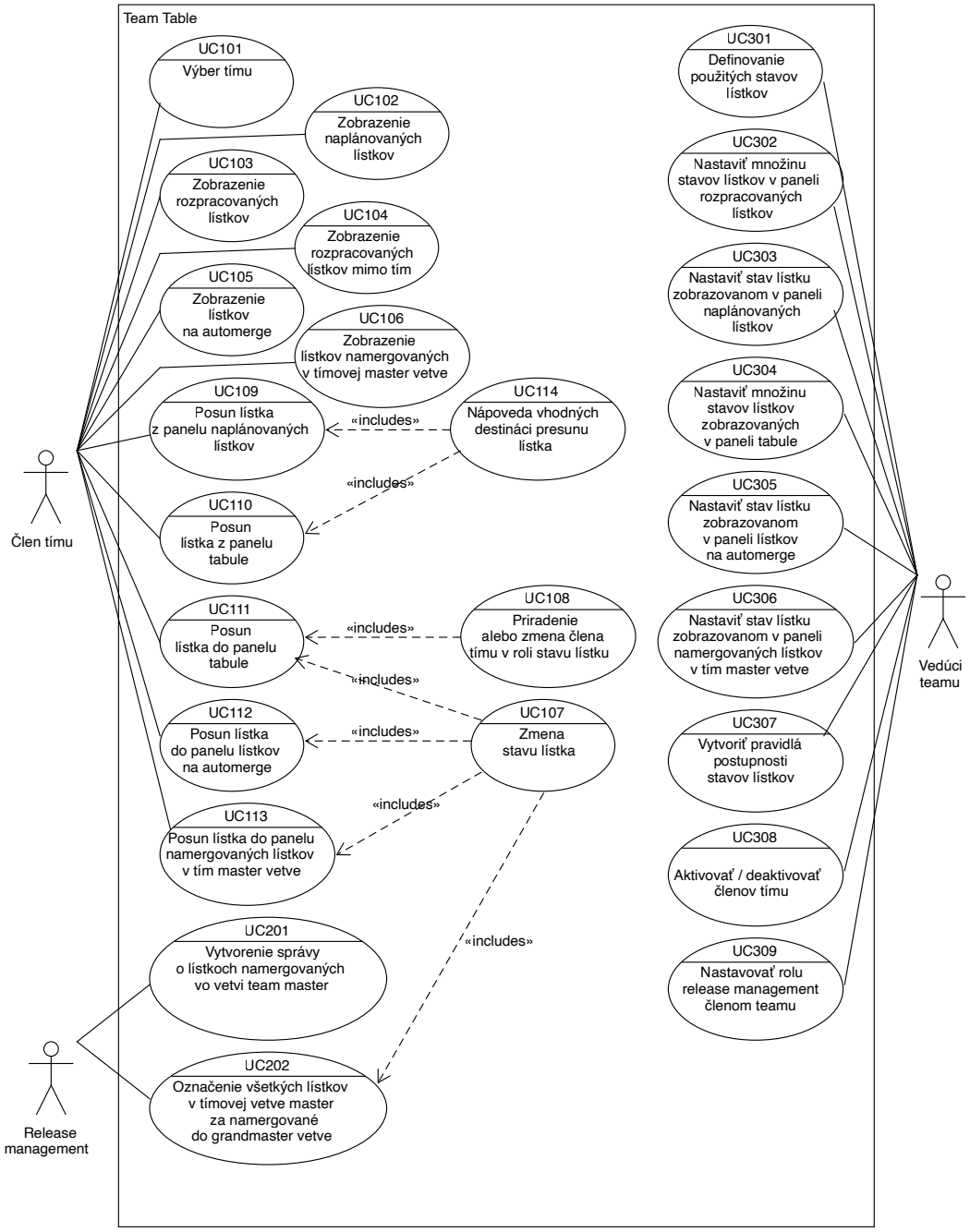
Člen teamu je užívateľ NET Genia, ktorý je priradený aspoň do jedného tímu. Je to zamestnanec, ktorý sa podieľa na vývoji a pri svojej činnosti posúva lístky po tabuli.

4.4.2 Release management

Je podrola člena tímu, ktorý vykonáva merge timovej vetvi do grandmaster vetvi. Typicky to je SCRUM Master tímu. V tejto roli môžu byť iba členovia tímu.

4.4.3 Vedúci tímu

Vedúci tímu sa stará o nastavenie tímovej tabule. Typicky túto rolu zastáva hlavný programátor alebo SCRUM Master tímu.



Obr. 4.2 Diagram prípadov užitia

4.5 Prípady užitia

4.5.1 Člen tímu

4.5.1.1 UC101 – Výber tímu

Člen tímu vidí zoznam všetkých tímov, ktorých je členom. Má možnosť prepínať ktorý tím je aktívny. Tím z najnižším číslom je predvolený ako aktívny. Vždy musí byť aktívny práve jeden tím.

4.5.1.2 UC102 – Zobrazenie naplánovaných lístkov

Paneli naplánovaných lístkov zobrazí všetky tímové lístky, ktorých stav odpovedá nastavenému v kapitole 4.5.3.3.

Lístky sú zobrazené podla hierarchie v stromovom usporiadaní zobrazenom na obrázku 2.3.

4.5.1.3 UC103 – Zobrazenie rozpracovaných lístkov

Panel rozpracovaných lístkov zobrazí všetky tímové lístky, ktorých stav patrí do množiny nastavenej v kapitole 4.5.3.2. Na tomto paneli sú lístky zobrazené podla hierarchie v stromovom usporiadaní zobrazenom na obrázku 2.3.

Panel tabule zobrazí všetky lístky, ktorých stav patrí do množiny nastavenej v kapitole 4.5.3.4, a zároveň na lístku práve teraz pracuje aktívny člen tímu.

4.5.1.4 UC104 – Zobrazenie rozpracovaných lístkov mimo tím

Panel rozpracovaných lístkov mimo tím zobrazí všetky tímové lístky, ktorých stav patrí do množiny nastavenej v kapitole 4.5.3.4, a na lístku práve teraz pracuje užívateľ, ktorý nie je aktívny člen tímu.

Pri lístkoch mimo tím musia byť zobrazené údaje o stave a osobe, ktorá na nom pracuje (podla roli nastavenej v stave lístka). Lístky sú zobrazené ako zoznam bez ohľadu na ich hierarchické vzťahy.

4.5.1.5 UC105 – Zobrazenie lístkov na automerge

Panel lístkov na automerge zobrazí všetky tímové lístky, ktorých stav odpovedá nastavenému v kapitole 4.5.3.5. Lístky sú zobrazené podla hierarchie v stromovom usporiadaní zobrazenom na obrázku 2.3.

4.5.1.6 UC106 – Zobrazenie lístkov namergovaných v tímovej master vetve

Panel lístkov namergovaných v tímovej master vetve zobrazí všetky tímové lístky ktorých stav odpovedá nastavenému v kapitole 4.5.3.6. Lístky sú zobrazené podla hierarchie v stromovom usporiadaní, zobrazenom na obrázku 2.3.

4.5.1.7 UC107 – Zmena stavu lístka

Posun lístka musí upraviť dáta v NET Geniu a vygenerovať históriu tak, ako keby bola zmena vykonaná priamo v NET Geniu. Výnimkou je, ak destinácia a počiatočná poloha odpovedá rovnakému stavu.

4.5.1.8 UC108 – Priradenie alebo zmena člena tímu v roli stavu lístku

Posun lístka do panelu tabule nastavuje člena tímu podla riadku. Člen je priradený do role prislúchajúcej stavu stĺpcu.

Príklad:

Bunka panelu tabule, do ktorej bol lístok posunutý:

- Riadok - Peter Liptai
- Stĺpec - InWork - odpovedajúca rola je ”Programmer”

V NET Geniu sa pri posune do tejto bunky nastaví lístku atribút „Programmer“ užívateľ Peter Liptai.

4.5.1.9 UC109 – Posun lístka z panelu naplánovaných lístkov

Lístky zobrazené na paneli musia byť uchopiteľné pre posun na tabuli.

4.5.1.10 UC110 – Posun lístka z panelu tabule

Lístky zobrazené na paneli musia byť uchopiteľné pre posun na tabuli.

4.5.1.11 UC111 – Posun lístka do panelu tabule

Lístky je možné umiestňovať do voľných buniek tohto panelu. Pri posune panel napovedá vhodné polohy, kap. 4.3.1.5.

Po posune sa nastavuje stav lístku podla stĺpca, kap. 4.5.1.7, a rola stavu lístka podla riadku, kap. 4.5.1.8.

4.5.1.12 UC112 – Posun lístka do panelu lístkov na automerge

Lístky je možné umiestňovať do tohto panelu. V prípade, ak je stav pridelený panelu vhodný ako nasledujúci stav lístku, tak sa panel zvýrazní.

4.5.1.13 UC113 – Posun lístka do panelu namergovaných lístkov v tím master vetve

Lístky je možné umiestňovať do tohto panelu. V prípade, ak je stav pridelený panelu vhodný ako nasledujúci stav lístku, panel sa zvýrazní.

4.5.1.14 UC114 – Nápoveda vhodných destináci presunu lístka

Na základe nastavenej tímovej postupnosti stavov lístkov vieme pre lístok odporučiť vhodné destinácie (bunky panelu tabule / ostatné panely) pri presune lístka. Vhodné destinácie sú zvýraznené:

- v paneli tabule zafarbením vhodných buniek,
- v ostatných paneloch zafarbením hlavičky panelu.

4.5.2 Release management

4.5.2.1 UC201 – Vytvorenie správy o lístkoch namergovaných vo vetvi team master

Funkcia vytvoriť správu popisujúcu všetky lístky v paneli namergovaných lístkov v team master vetve. V prípade, ak sa v paneli nachádza lístok, ktorý má priradené ďalšie lístky, správa obsahuje iba koreňový lístok.

Príklad správy:

```
Team A provedl merge do grandmasteru:

User Story:
US: 5805 Joint (Steel) - Design; Weld Check
US: 5601 Joint (Steel) - Design; Bearing
US: 5906 Joint (Steel) - Components - Fin Plate

Testing Task:
TT: 85 New AT: tst\Steel\Joint\Components\FinPlate\Test
```

4.5.2.2 UC202 – Označenie všetkých lístkov v tímovej vetve master za namergované do grandmaster vetve

Nastavenie stavu všetkým lístkom, ktoré sú v paneli namergovaných do tímovej vývojovej vetvi, na stav odpovedajúci lístku namegovaného do vývojovej vetve grandmaster.

4.5.3 Vedúci teamu

4.5.3.1 UC301 – Definovanie použitých stavov lístkov

Definovanie **stavov lístkov**. Stav lístku je objekt, ktorý má identifikačné číslo, nepovinné väzby na Todo stavy a Bug stavy z NET Genia.

Stav lístku môže obsahovať odkaz na jeden z atribútov lístka:

- Programmer – Atribúte je odkaz na užívateľa NET Genia – programátora, ktorý ma lístok implementuje, stará sa o jeho merge a opravuje nedostatky zistené pri code review a testovaní.
- Tester – Atribúte je odkaz na užívateľa NET Genia – testera, ktorý testuje implementáciu.
- Reviewer – Atribúte je odkaz na užívateľa NET Genia – programátora, ktorý vykonáva code review implementácie.

Takýto odkaz nazývame **rola stavu lístku**.

Stav lístku musí mať väzbu aspoň na jeden stav (Todo alebo Bug stav).

Toto nastavenie sa vťahuje vždy k jednému tímu.

4.5.3.2 UC302 – Nastaviť množinu stavov lístkov v paneli rozpracovaných lístkov

Určenie podmnožiny definovaných stavov lístkov. Označuje stavy tímových lístkov, ktoré sa zobrazujú v paneli rozpracovaných lístkov.

Toto nastavenie sa vťahuje vždy k jednému tímu.

4.5.3.3 UC303 – Nastaviť stav lístku zobrazovanom v paneli naplánovaných lístkov

Určenie jedného stavu z definovaných stavov lístkov. Označuje stav tímových lístkov, ktoré sa zobrazujú v paneli naplánovaných lístkov.

Toto nastavenie sa vťahuje vždy k jednému tímu.

4.5.3.4 UC304 – Nastaviť množinu stavov lístkov zobrazovaných v paneli tabule

Určenie usporiadanej podmnožiny definovaných stavov lístkov. Podla určenej množiny sú vytvorené stĺpce panelu tabule. Poradie stavov lístkov určí poradie stĺpcov.

Každý stav lístku použitý na paneli tabule musí mať definované:

- Bug stav
- Todo stav
- Rolu tímového stavu

Toto nastavenie sa vťahuje vždy k jednému tímu.

4.5.3.5 UC305 – Nastaviť stav lístku zobrazovanom v paneli lístkov na automerge

Určenie jedného stavu z definovaných stavov lístkov. Označuje stav tímových lístkov, ktoré sa zobrazujú v paneli lístkov na automerge.

Toto nastavenie sa vťahuje vždy k jednému tímu.

4.5.3.6 UC307 – Nastaviť stav lístku zobrazovanom v paneli namergovaných lístkov v tím master vetve

Určenie jedného stavu z definovaných tímových stavov. Označuje stav tímových lístkov, ktoré sa zobrazujú v paneli namergovaných lístkov v tím master vetve.

Toto nastavenie sa vťahuje vždy k jednému tímu.

4.5.3.7 UC307 – Vytvoriť pravidlá postupnosti stavov lístkov

Pravidlá postupnosti stavov sa skladajú z usporiadaných dvojíc stavov lístkov, filtra podla typov lístkov, pre ktoré platia a informáciu či ide o pokrok v práci, alebo krok späť.

Množina takýchto pravidiel charakterizuje postupnosť práce v tíme. Príklad pravidla z obr. 2.3:

„In Testing“ -> „Auto Merge to GM“, filter: iba pre Global Bugs, pokrok

Toto nastavenie sa vťahuje vždy k jednému tímu.

4.5.3.8 UC308 – Aktivovať / deaktivovať členov tímu

Užívateľov NET Genia, ktorý sú pridelený do tímu, je možné vyradiť z tímovej tabule.

Toto nastavenie je vhodné použiť, ak je člen tímu neaktívny vo vývoji a zaberal by na paneli tabule prázdny riadok.

Toto nastavenie sa vťahuje vždy k jednému tímu.

4.5.3.9 UC309 – Nastavovať rolu release management členom teamu

Prideľovanie role členom tímu. Implementácia tohto prípadu užitia bude realizovaná, až po presune nastavení do NET Genia.

Toto nastavenie sa vťahuje vždy k jednému tímu.

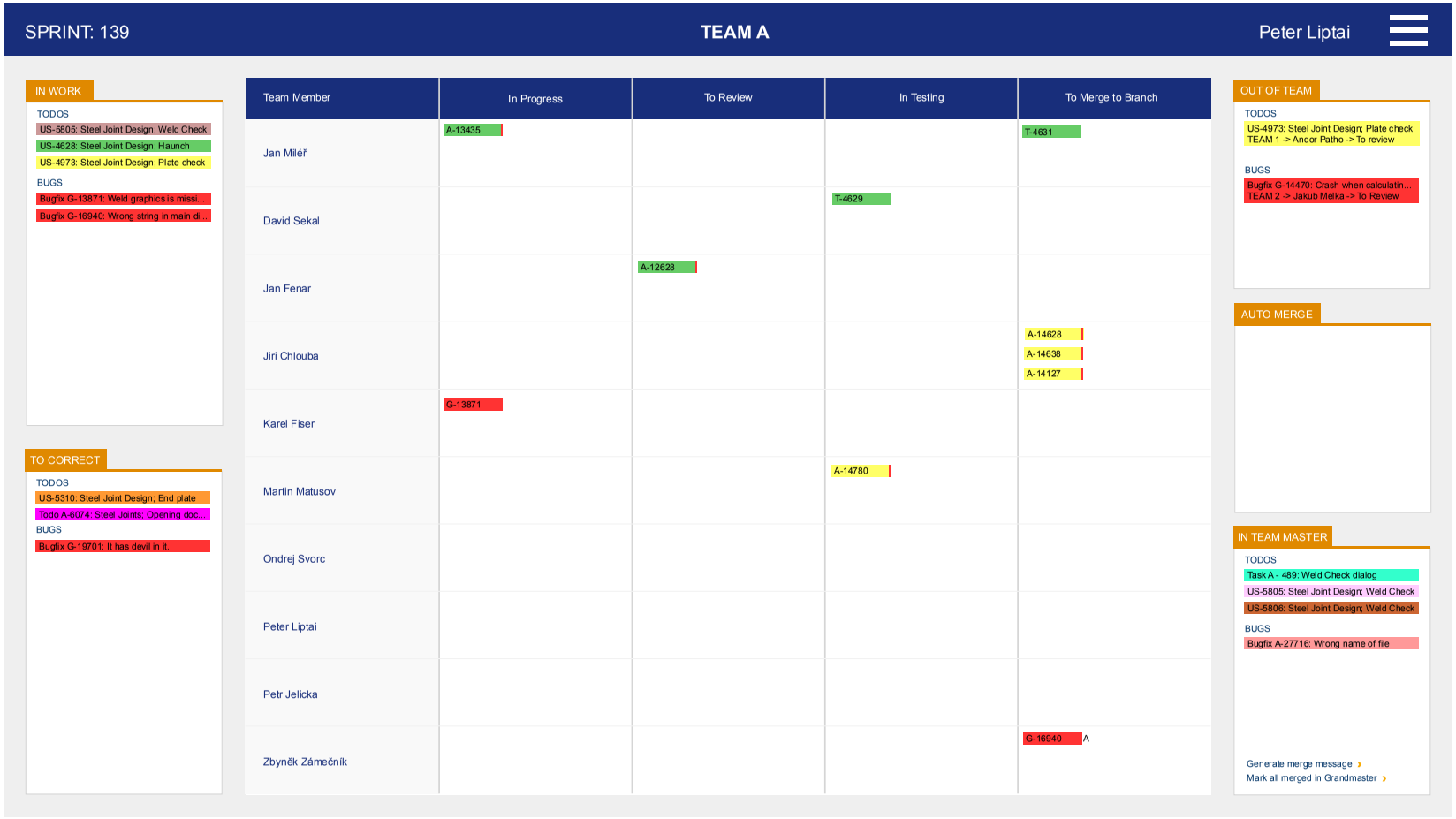
4.5.4 Pokrytie požiadaviek

Pokrytie funkčných požiadaviek s prípadmi užitia je uvedené v tabuľke 4.1.

Tabuľka 4.1 Prehľad realizácie požiadavkov prípadmi užitia

	F01	F02	F03	F04	F05	F06
Člen tímu						
UC101	✓	✓				
UC102			✓			
UC103			✓			
UC104			✓			
UC105			✓			
UC106			✓			
UC107					✓	
UC108					✓	
UC109					✓	
UC110					✓	
UC111					✓	
UC112					✓	
UC113					✓	
UC114					✓	
Release management						
UC201					✓	✓
UC202					✓	
Vedúci tímu						
UC301			✓	✓	✓	
UC302			✓	✓	✓	
UC303			✓	✓	✓	
UC304			✓	✓	✓	
UC305			✓	✓	✓	
UC306			✓	✓	✓	
UC307				✓	✓	
UC308				✓		
UC309				✓	✓	

4.6 Návrh obrazoviek



Obr. 4.3 Návrh obrazovky - tímová tabula

4.6.1 Tímová tabuľa

Návrh obrazovky tímovej tabuľe je ukázaný na obrázku 4.3. Stromové zoznamy sa zostavujú podla hierarchie lístkov obr. 2.3. Hĺbka stromu je maximálne jedna.

4.6.1.1 UI01 – Menu

Menu obsahuje:

- Prepínanie aktívneho tímu aplikácie. Vždy musí byť aktívny práve jeden tím.
- Meno užívateľa, ktorý program tímovej tabule používa.
- Tlačítko na vytvorenie merge message.
- Tlačítko na merge lístkov z tímovej master vetve do grandmaster vetve.

4.6.1.2 UI02 – Panel rozpracovaných lístkov

Stromový zoznam všetkých lístkov, v ktorých je nastavený tím, je zhodný s aktívnym tímom aplikácie a stav lístka patrí do množiny aktívnych stavov. Tieto stavy sa nastavujú v konfiguračnom súbore tímu. Meno elementu v konfiguračnom súbore je InWorkMappings.

4.6.1.3 UI03 – Panel naplánovaných lístkov

Stromový zoznam všetkých lístkov, v ktorých je nastavený tím, je zhodný z aktívnym tímom aplikácie. Stav lístkov odpovedá úlohám a chybám, na ktorých sa má zažať pracovať v súčastnom, alebo budúcom sprinte. Tento stav sa nastavuje v konfiguračnom súbore tímu. Meno elementu v konfiguračnom súbore je ToCorrectMapping.

4.6.1.4 UI04 – Panel tabule

Panel tabule má formu tabuľky. Jednotlivé bunky tabuľky sú lístky, alebo prázdne miesta. Riadky určujú člena tímu, ktorý na lístku pracuje. Stĺpce určujú stav, v ktorom sa lístok nachádza. Na tabuli sa môžu nachádzať lístky, ktoré majú nastavený iný tím ako aktívny tím aplikácie. Toto je bežné, ak robíme review alebo testing pre iný tím.

4.6.1.5 UI05 – Panel rozpracovaných lístkov mimo tím

Zoznam lístkov, ktoré majú nastavený tím rovnaký ako aktívny tím aplikácie a zároveň na nich pracuje užívateľ, ktorý nie je členom aktívneho tímu aplikácie. Príklad: Ak je lístok v stave „ToReview“ a atribút „Reviewer“ je nastavený na užívateľa NET Genia, ktorý nie je člen tímu, lístok sa objavý v tomto paneli. Pod lístkom sa zobrazí informácia o stave lístka a meno užívateľa, ktorý má lístok pridelený.

4.6.1.6 UI06 – Panel lístkov na automerge

Stromový zoznam všetkých lístkov, v ktorých je nastavený tím, je zhodný s aktívnym tímom aplikácie a nachádzajú sa v stave na automerge. Tento stav sa nastavuje v konfiguračnom súbore tímu. Meno elementu v konfiguračnom súbore je AutoMergeMapping.

4.6.1.7 UI07 – Panel namergovaných lístkov v tímovej master vetve

Stromový zoznam všetkých lístkov, v ktorých je nastavený tím, je zhodný s aktívnym tímom aplikácie a nachádzajú sa v stave namergované v tím master vetve. Tento stav sa nastavuje v konfiguračnom súbore tímu. Meno elementu v konfiguračnom súbore je InTeamMasterMapping.

4.6.2 Nastavenia

V súčasnom riešení sú nastavenia tabule bez užívateľského rozhrania nastaviteľné pomocou konfiguračných XML súboroch. Tento formát bol vybraný z dôvodu že je jednoducho človekom čitateľný a manuálne upravovateľný.

Po očakávanej zmene nastavení do NET Genia budú nastaviteľné pomocou komponenty NET Genia – „Editačný formulár“ [16].

4.6.3 Pokrytie prípadov užitia prvkami užívateľského rozhrania

V tejto kapitole sú uvedené pokrytia prípadov užitia jednotlivými prvkami užívateľského rozhrania.

Tabuľka 4.2 Prehľad realizácie požiadavkov prípadmi užitia

	UI01	UI02	UI03	UI04	UI05	UI06	UI07
Člen tímu							
UC101	✓						
UC102			✓				
UC103		✓		✓			
UC104		✓			✓		
UC105						✓	
UC106							✓
UC107			✓	✓		✓	
UC108				✓			
UC109			✓	✓		✓	
UC110				✓			
UC111				✓			
UC112						✓	
UC113							✓
UC114			✓	✓		✓	✓
Release management							
UC201	✓						✓
UC202	✓						✓
Vedúci tímu							
Prípady užitia budú v budúcnosti pokryté užívateľským rozhraním NET Genia. Teraz nemajú užívateľské rozhranie. Sú pokryté nastavovaním konfiguračných XML súborov.							

4.7 Doménový model

Na obrázku 4.4 je zobrazený analytický doménový model. Entity doménového modelu sú rozdelené farebnosťou do troch skupín. Červenou farbou sú vyznačené entity, ktoré reprezentujú entity existujúce v NET Geniu. Entity vyznačené zelenou farbou slúžia na konfiguráciu programu. Ostatné nezvýraznené entity sú použité na zachytenie logiky programu.

Popis hlavných tried programu:

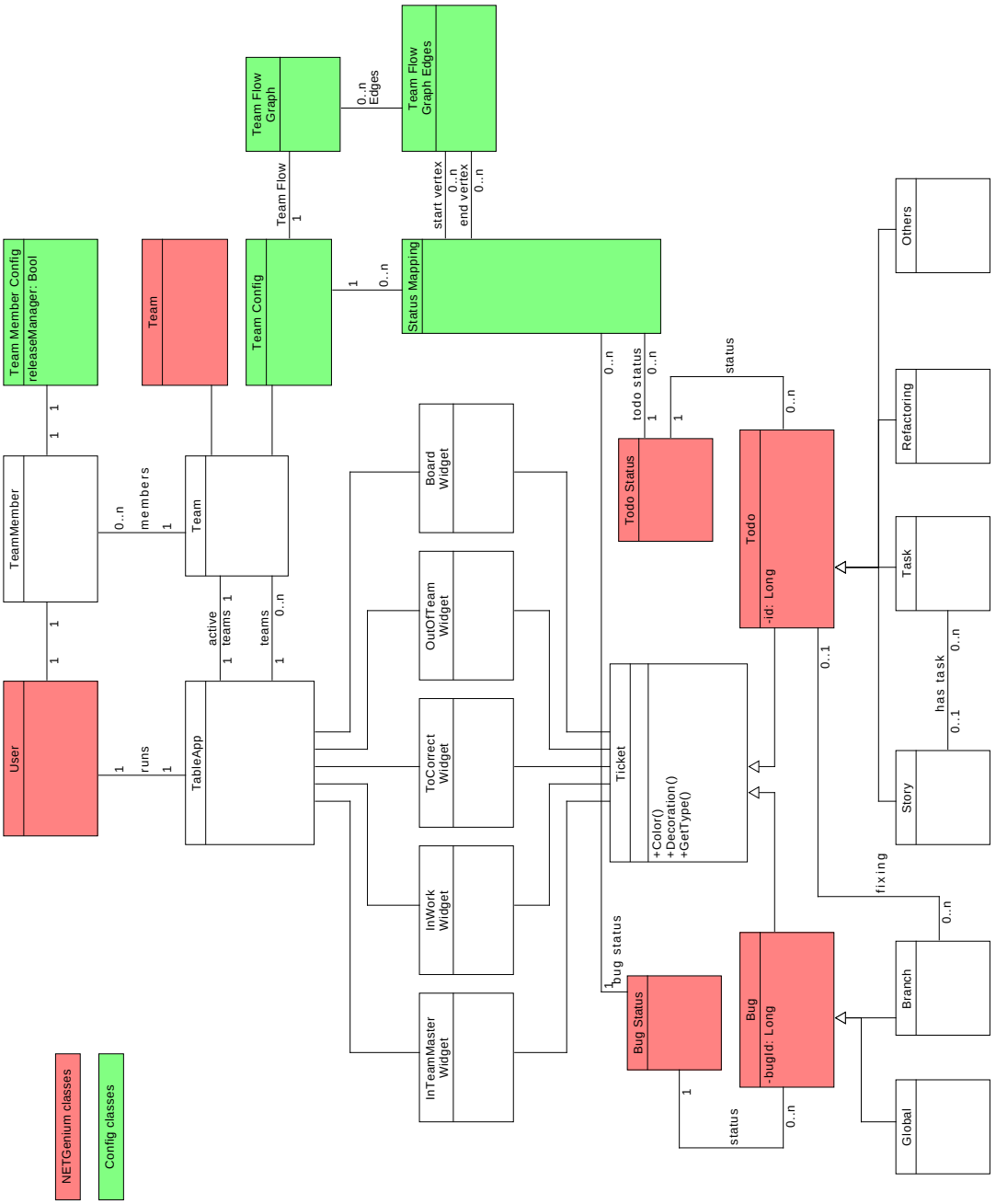
Ticket je entita, ktorá zovšeobecňuje entity NET Genia Todo a Bug. Táto entita má metódy na získanie všetkých potrebných informácií na vykreslenie na tabuli.

TableApp riadi chod programu. Spúšťa aktualizácie dát a panelov užívateľského rozhrania. Má informáciu o užívateľovi NET Genia ktorý program používa a o tímoch ktorých je členom. Vždy musí byť aktívny práve jeden tím z množiny tímov ktorých je užívateľ členom.

InTeamMaster, InWork, ToCorrect, OutOfTeam a Board Widget sú entity ktoré riadia panely užívateľského rozhrania. Každá entita z nich vlastní vlastnú množinu entít typu Ticket ktorú zobrazuje.

Team entita existuje ako reprezentácia entity tímu v NET Geniu a zároveň ako entita ktorá má informácie o členoch tímu, nastaveniach tímu a väzbe na entitu tímu v NET Geniu.

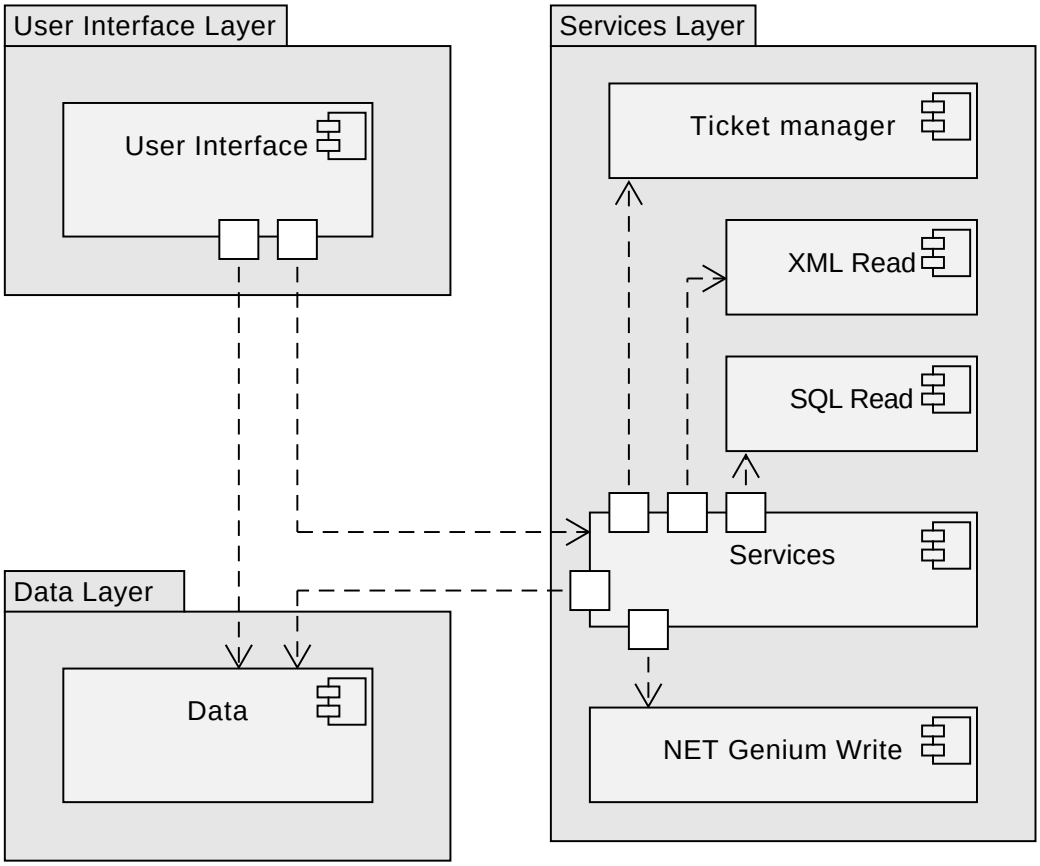
Team Member reprezentuje užívateľa z pohľadu tabule. Má jeho nastavenia a väzbu na entitu užívatela v NET Geniu.



Obr. 4.4 Doménový model

4.8 Návrh architektúry

Program bude realizovaný pomocou relaxovanej trojvrstvovej architektúry.



Obr. 4.5 Architektúra programu

4.8.1 Prezenčná vrstva – User Interface Layer

Vrstva nesie zodpovednosť za užívateľské rozhranie. Táto vrstva využíva Model/View architektúru [5].

4.8.2 Dátová vrstva – Data Layer

Dátová vrstva nesie zodpovednosť za logiku domény. Logika je zabezpečená vhodnou štruktúrou dedičnosti a funkčnosťou dátových tried.

4.8.3 Vrstva technických služieb – Service Layer

Vrstva zaisťuje prístup k dátam. Realizuje čítanie z databázy NET Genia, zápis dát do NET Genia a čítanie konfiguračných súborov.

- Services – Sada tried, ktorá slúži na čítanie dát. Triedy sú realizované podla návrhového vzoru singleton. Data tied sa pri prvom prístupe inicializujú a pri ďalších prístupoch sa použijú zapamätané dáta. Vďaka čomu vieme zabezpečiť výkonnostné požiadavky definované v kapitole 4.3.2.2. Metódy realizujúce prístup ku konfiguračným XML súborom majú predpísané rozhranie. Predpísané rozhranie znižuje previazanosť a tým minimalizuje zmeny pri prechode konfigurácie z XML súborov do NET Génia.
- XML Read – Pomocná trieda na čítanie dát z XML súborov
- SQL Read – Pomocná trieda na čítanie dát z databáze NET Genia
- NET Genium Write – Trieda na zápis dát do NET Genia. Zápis prebieha v druhom – pracovnom vlákne, aby sme dodržali rýchlu odozvu definovanú vo výkonnostných požiadavkoch v kapitole 4.3.2.2.
- Ticket manager – Tieda zodpovedná za výrobu lístkov na tabuli. Výroba lístkov je navrhnutá podla návrhového vzoru **továreň** opísaného v knihe [15], kapitola „Factory Method“.

4.9 Použité technológie

4.9.1 Programovací jazyk

Na implementáciu bol zvolený jazyk C++ a framework Qt. Tieto nástroje boli zvolené z dôvodu skúsenosti autora. Nástroje sú tak tiež používane vo firme, ktorá bude program tímových tabúľ používať. Program bude teda jednoducho rozšíriteľný a upravovateľný aj inými zamestnancami.

V neposlednej rade sa jedná o objektovo orientovaný jazyk. Doména tímovej tabule je mimoriadne hodná na realizáciu pomocou objektového prístupu.

Framework Qt má vhodné nástroje na realizáciu jednotlivých panelov popísaných v kapitole 4.6.

4.9.1.1 NET Genium Wrapper

Zápis dát musí byť realizovaný pomocou dynamickej knižnice **NETGeniumConnection.dll** podla užívateľskej príručky [16]. Knižnica je skompilovaná pre .NET Framework 4.5.2, ktorý je potrebný na jej obsluhu.

Obsluhu NETGeniumConnection.dll knižnice bude realizovať procedurálny program napísaný v programovacom jazyku **C#**. Tento program bude asynchrónne volaný z hlavného programu pomocou tiedy QProcess [10]. Program dostane vstupy ako argumenty pri spustení. Výstup zapíše na štandardný výstup, ktorý hlavný program sleduje (Observer Pattern [15]).

4.9.2 Spôsob ukladania dát

Program funguje ako rozšírenie NET Genia. Rozhodovanie o spôsobu ukladania dát bolo určené už existujúcim riešením.

Dáta budú čerpané z existujúcej SQL Server databázy NET Genia. Čítanie dát bude zabezpečovať tieda QSqlDatabase [11].

V súčastnosti sa jedná o prototyp programu, z tohoto dôvodu sú nastavenia aplikácie ukladané iba lokálne v XML formáte [20], sú verzované a zdielané pomocou GITu. Konfiguračné dáta budú presunuté do NET Genia. Po presune sa k nim bude pristupovať rovnako ako k ostatným dátam.

Zápis dát do NET Genia rieši C# program popísaný vyššie 4.9.1.1.

TODO – UML sequence graf zápisu dát do NET Genia (volanie a spätná väzba).

Kapitola 5
Implementácia

Program bude realizovaný ako desktopová aplikácia, čo vyhovuje spôsobu práce vo firme. Každý zamestnanec pracuje na počítači s operačným systémom Windows, do ktorého je prihlásený pomocou doménového účtu. Tento doménový účet je použitý na komunikáciu s NET Geniom tak, že odpadá potreba autentifikácie.

V prípade práce z domova sa k počítaču v práci zamestanci pripájajú pomocou Remote Desktop Protocol [3]. Ten umožní používanie programu tímovej tabule počas práce z domova tak, ako keď sú zamestnanci v fyzicky v práci.

5.1 Graf postupnosti stavov

Na zachytenie postupnosti vyžadovanej tímovými pravidlami je vytvorený orientovaný graf.

5.1.1 Vrcholy grafu

Množiny stavov ktoré môže Todo a Bug nadobúdať je rozličná. Avšak, je medzi nimi možné nájsť stavy ktoré majú rovnaký význam. Pravé preto sme zaviedli pojem **stav lístku**, ktorý je v kóde reprezentovaný triedou **StatusMapping**, viz výpis kódu 5.1. Vrcholy grafu postupnosti stavov sú reprezentované inštanciami triedy **StatusMapping**.

StatusMapping môže mať väzbu na Todo stav a Bug stav. Aspoň jedna väzba je povinná. Ďalej trieda môže obsahovať odkaz na atribút označujúci užívateľa NET Genia, ktorý sa v stave lístkom zaoberá.

5.1.2 Hrany grafu

Hrany grafu spájajú vrcholy reprezentované triedou **StatusMapping**. Každá hrana má informáciu či sa jedná o pokrok, alebo krok späť v premennej **m_progress**.

Hrany grafu sú určené iba pre vybrané typy lístkov. Typy Todo sa určujú v premennej **m_todoEdge** a Bug v **m_bugEdge**. Každý možný typ má pridelený bit (vlajku), ktorý ho zapína, viz kód 5.2.

Príklad: Ak chceme, aby hrana platila pre všetky lístky, musí platiť:

- m_todoEdge = 0xF = 0b1111
- m_bugEdge = 0x3 = 0b0011

Hrana grafu obsahuje iba koncový vrchol **m_endVertex**. Počiatočný vrchol je kľúč vyhľadávania v množine všetkých hrán.

Výpis kódu 5.1 Trieda StatusMapping

```
class StatusMapping : public AbstractObject
{
    ...
    enum ERole
    {
        E_ROLE__BEGIN,
        E_ROLE_NONE = E_ROLE__BEGIN,
        E_ROLE_PROGRAMMER,
        E_ROLE_REVIEWER,
        E_ROLE_TESTER,
        E_ROLE__END
    };
    ...
private:
    const QSharedPointer<const BugStatus> m_bugStatus;
    const QSharedPointer<const TodoStatus> m_todoStatus;
    ERole m_role;
};
```

Výpis kódu 5.2 Atribúty hrany grafu postupnosti stavov

```
enum ETodoEdgeFlag
{
    E_TODO_EDGE_FLAG_STORY = 0x1,              //< ng_todotype = 1
    E_TODO_EDGE_FLAG_TASK = 0x2,               //< ng_todotype = 2
    E_TODO_EDGE_FLAG_OTHERS = 0x4,             //< ng_todotype = 3
    E_TODO_EDGE_FLAG_REFACTORING = 0x8         //< ng_todotype = 4
};
enum EBugEdgeFlag
{
    E_BUG_EDGE_FLAG_BRANCH = 0x1,         //< ng_bugtype = 1 branch
    E_BUG_EDGE_FLAG_GRANDMASTER = 0x2     //< ng_bugtype = 2 grandmaster
};
..
// start mapping is key in graph
StatusMappingConstPtr m_endVertex;
bool m_progress;
TodoEdgeFlags m_todoEdge;
BugEdgeFlags m_bugEdge;
```

Výpis kódu 5.3 Dátový typ grafu postupnosti stavov

```
typedef QMultiMap<StatusMappingConstPtr, TeamFlowGraphEdgeConstPtr>
TeamFlowGraph;
```

5.1.3 Dátová štruktúra grafu

Graf je uložený do kontajneru typu QMultiMap. Kľúč je počiatočný vrchol, obsah sú jednotlivé hrany, ktoré obsahujú koncový vrchol hrany.

Kontajner QMultiMap bol zvolený z dôvodu rýchleho vyhľadávaniu podla vrcholu, v ktorom sa nachádzame. Kontajner je potomok kontajneru QMap ktorý používa algoritmus čierno červeného stromu, ktorý má rýchlosť vyhľadávania $\mathcal{O}(n \log n)$ [4]. Rýchli prístup je potrebný pri posune lístkov. V momente „zdvihnutia“ lístku je potrebné podla jeho stavu nájsť vhodné destinácie. Keďže sa jedná o užívateľské rozhranie, je potrebné zabezpečiť požiadavky stanovené analýzou v kapitole 4.3.2.2.

5.2 Farby lístkov

Farby boli vygenerované pomocou webovej aplikácie Mokole.com [17]. Nastavenia použité na vygenerovanie farieb:

- **colors:** 60
- **minimum luminosity:** 1%
- **maximum luminosity:** 99%
- **maximum loops:** 15 000

Zo zoznamu vygenerovaných farieb bola odstránená červená farba, kód farby RGB modeli [21] – **#ff0000**. Táto farba je použitá pre lístky typu Global Bug.

Ak nastane problém, že počet farieb nebude dostatočný, je nevhodné zvyšovať ich počet, pretože by to spôsobilo zníženie rozdielu medzi jednotlivými farbami za rozlíšiteľnosti bežným okom.

Tento problém vyriešime tak, že sa farba lístku bude skladať z dvoch rôznych farieb. Z množiny 59 farieb získame variáciou bez opakovania $\frac{59!}{(59-2)!} = 3422$ usporiadaných dvojíc farieb (vynecháme usporiadané dvojice dvoch rovnakých farieb, lebo by sa vizuálne nelíšil od monochromatického riešenia).

Čitateľnosť textu na lístkoch zabezpečuje prispôsobovanie farby textu podla farby pozadia. Ak svietivosť farby pozadia v RGB modeli je väčšia ako 127, text sa zobrazí čiernou farbou. Naopak ak je svietivosť menšia lebo rovná ako 127, text sa zobrazí farbou bielou.

Svietivosť farby zapísanej pomocou RGB modelu sa vypočíta podla vzťahu: [18]
$0.3R + 0.59G + 0.11B$

- **R** – množstvo červenej farby (0-255)
- **G** – množstvo zelenej farby (0-255)
- **B** – množstvo modrej farby (0-255)

Farby lístkov nie sú perzistentné. Pri každom spustení programu sa farby lístkom znovu prideľujú. Algoritmus prideľovania farieb sa snaží prideliť lístku vždy rovnakú farbu. Avšak ak pri štarte programu máme inú množinu lístkov, je možné, že aj farby lístkov budú odlišné.

Pri aktualizácií dát tlačítkom „Update“ nám už zobrazené lístky nemenia farbu, nové lístky dostanú nepoužité farby. Farby lístkov, ktoré boli pri aktualizácií dát odstránené z tabule, sú označené znovu ako nepoužité.

Pri bežnej pracovnej komunikácií sa nikdy na lístok neodkazujeme farbou, ale číslom, alebo popisom. Preto je súčasný algoritmus prideľovania farieb dostačujúci.

5.3 Synchronizácia zmien do NETGenia

K zápisu dát do NET Genia je odporúčané podla príručky [16] použiť knižnicu „NETGeniumConnection.dll“. Na prácu z knihovňou je potrebný .NET Framework verzie 4.5.2.

Obecne je možné kompilovať a spúšťať C# kód v rámci C++ programu pri použití MSVC kompilátoru [2] zo zapnutým Common Language Runtime [1].

Bohužiaľ pri použití Qt frameworku je CLR nepodporované.

Qt docs – často kladené otázky [14]:
„Is there any way to compile the Qt source code and Qt applications with /clr flag?
We don't support building Qt or Qt applications with CLR directly. To integrate existing components you can use the Active Qt http://doc.qt.io/qt-5/activeqt.html framework, which allows integration of Active X and COM object in Qt applications.“

Z tohto dôvodu obsluhu knihovni zabezpečuje C# program „NETGeniumWrapper“. Pri každom zápise dát do NET Genia je program spustený. Dáta na zápis program dostane ako argumenty pri spustení. Ak počas zápisu nastane chyba, program vypíše chybovú hlášku na štandardný výstup. Program sa po vykonaní zápisu ukončí.

Tento prístup má zdržanie rádovo v sekundách, preto je program spúšťaný asynchrónne. Pri posune lístka sa predpokladá, že zápis prebehne v poriadku. Dáta lístku sa lokálne modifikujú a ich zobrazenie v paneloch sa aktualizuje. V prípade ak zápis neprebehne v poriadku (typicky do 2 sekúnd), dostaneme informáciu z programu NETGeniumWrapper. Po obdržaní chybovej hlášky ju zobrazíme užívateľovi a dáta aktualizujeme z NET Genia. Tento prístup nám vráti lokálne zmeny, ktoré neboli zapísané do NET Genia. Užívatelské rozhranie bude zobrazovať stav ktorý je aktuálne v NET Geniu.

5.4 Logovanie

Súčasťou programu je funkcionalita logovania – zapisovania dôležitých a neočakávaných situácii, ktoré nastali pri behu programu. Funkcionalitu logovania využívajú všetky vrstvy programu.

Na logovanie je využité štandardné riešenie zabudovane v Qt [13]. Program počas behu zapisuje správy podľa úrovni dôležitosti do zásobníkov: **qDebug(), qInfo(), qWarning(), qCritical() a qFatal()**. Spracovanie týchto správ zabezpečuje funkcia messageOutput, viz kód 5.4. Funkcia zapisuje správy do súboru log.txt. V prípade ak program nemá právo zapisovať do tohto súboru, správy sú vypísané na štandardný chybový výstup – stderr. Správy dôležitosti Fatal sú vypísané aj v informačnom okne užívateľského rozhrania.

Výpis kódu 5.4 Funkcia MessageHandler::messageOutput

```
void MessageHandler::messageOutput(QtMsgType type,
                                   const QMessageLogContext& context,
                                   const QString& msg)
{
    const char* file = context.file ? context.file : "";
    const char* function = context.function ? context.function : "";
    QString message = QString(": %1 (%2:%3, %4)\n")
    .arg(msg)
    .arg(file)
    .arg(context.line)
    .arg(function);

    switch (type)
    {
        case QtDebugMsg:    message.prepend("Debug");    break;
        case QtInfoMsg:     message.prepend("Info");     break;
        case QtWarningMsg:  message.prepend("Warning");  break;
        case QtCriticalMsg: message.prepend("Critical"); break;
        case QtFatalMsg:    message.prepend("Fatal");    break;
    }

    QFile logFile("log.txt");
    if (logFile.open(QIODevice::WriteOnly | QIODevice::Append))
    {
        logFile.write(message.toLocal8Bit());
        logFile.close();
    }
    else
    {
        std::cerr << message.toStdString() << std::endl;
    }

    // for fatal error we terminate app,
    // the message will be also shown in message box
    if (type == QtFatalMsg)
    {
        QMessageBox msgBox;
        msgBox.setText(msg);
        msgBox.exec();
    }
}
```

Kapitola 6

Test použiteľnosti

Test použiteľnosti má odhaliť chyby pri návrhu užívateľského rozhrania. Test sa nezaoberá nastaveniami tímovej tabule. Nastavenia v konfiguračných XML súboroch budú presunuté do NET Genia, z tohto dôvodu ich nie je potrebné testovať.

6.1 Metodika testovania

6.1.1 Plánovanie

6.1.1.1 Účastníci testu

Účastníci testu sú členovia SCRUM tímu – Team A. Tým bol zvolený z dôvodu, že autor je jeho členom. Program je dodávaný spolu s konfiguráciou práve podľa metodiky vývoja v tomto tíme.

Členovia tímu majú rôzne role v tíme:

- programátor,
- tester,
- analytik.

Rôzne role účastníkov pomôžu vybudovať komplexnejší pohľad na program. Je veľká výhoda, že autor pozná účastníkov testu. Bude jednoduchšie dosiahnuť hlbšieho porozumenia.

6.1.1.2 Úlohy

Účastníci poznajú prácu s fyzickou tabuľou v kancelárií, ktorú artefakt nahradzuje. Ich úlohou bude snažiť sa v artefakte urobiť všetky úkony, ktoré by robili na tabuli.

6.1.1.3 Prostredie

Vzhľadom na súčasnú situáciu – vládou nariadená práca z domova – je nemožné zabezpečiť kontrolované a monitorované prostredie testovania. Na druhú stranu, artefakt bude testovaný presne tak, ako bude používaný po nasadení – cez vzdialenú plochu, bez možnosti osobnej komunikácie.

6.1.1.4 Zber dát

Zber dát bude zaistený cez zdielanými formulár medzi autorom a jednotlivými účastníkmi testu. Tieto formuláre budú čitateľné iba autorom a účastníkom.

Vo formulári budú pred pripravené bežné scenáre použitia. Každý scenár bude mať sériu otázok zameranú na hlbšie pochopenie myšlienok účastníka testu. Formulár je dostupný v prílohe práci B.1. Účastníci budú týždeň bežne pracovať a pri použití artefaktu zapíšu ich poznatky.

6.1.1.5 Príprava účastníkov

Účastníci budú informovaný ako si program do počítaču nahrať a krátky popis funkčnosti bez detailných opisov. Návod sa nachádza na začiatku formuláru v prílohe B.1.

6.1.1.6 Spracovanie dát

Jednotlivé formuláre budú spracované a na ich základe sa uskutoční krátky rozhovor s účastníkom testu. Spracovanie formuláru a rozhovor bude zameraný na hľadanie príčiny nepoužitia jednotlivých scenárov. Pokus o hlbšie pochopenie problémov s prácou s artefaktom.

6.2 Zhodnotenie výsledkov a návrh zmien

Výsledky testu používateľnosti rozdelíme do kategórií, ktoré sú rozobrané v nasledujúcich kapitolách. Prehľad všetkých odpovedí na formulár testu používateľnosti je v prílohe B.2.

V tejto kapitole budú postupne prebrané jednotlivé zistené problémy. Popis problému bude nasledovať návrh riešenia daného problému a odhad časovej náročnosti zmeny.

6.2.1 Užívateľské rozhranie

Všetky panely boli účastníkmi testu identifikované takmer okamžite. Identifikácia panelu, UI05 – Panel rozpracovaných lístkov mimo tím, bola pre užívateľov najzložitejšia. Aj napriek zložitosti väčšina užívateľov panel našlo takmer okamžite. Možná príčina zvýšenej zložitosti bola, že panel počas testovania neobsahoval žiaden lístok a tým pútal menej pozornosti.

Problémy zistené v užívateľskom rozhraní:

- **Problém:** Rozdieli medzi použitými farbami sú nedostatočné. Odlišné farby vizuálne splývajú do jednej.

  Návrh zmeny: Zníženie počtu farieb z 59. Odstránenie farieb, ktoré sú nerozoznateľné od ostatných použitých farieb.

  Odhad časovej náročnosti: 1 hodina

- **Problém:** Nedostatočné oddelenie lístkov typu Bug od lístkov typu Todo v paneloch užívateľského rozhrania, ktoré listy zobrazujú ako zoznam, alebo stromový zoznam.

  Návrh zmeny: Lístky v paneloch, ktoré zobrazujú lístky formou zoznamu, alebo stromového zoznamu, rozdeliť nadpismi Todo a Bug do dvoch skupín. Nadpis sa zobrazí ak panel zobrazuje aspoň jeden lístok daného typu.

  Odhad časovej náročnosti: 8 hodín

- **Problém:** Na tabuli chýba informácia o roli zamestnanca vo vývoji (položka Job Role z NET Genia).

  Návrh zmeny: Pridať pod meno užívateľa informáciu o jeho roli.

  Odhad časovej náročnosti: 2 hodiny

- **Problém:** Chýba možnosť paneli užívateľského rozhrania skladať vedľa seba v stĺpcoch na jednej strane.

  Návrh zmeny: Použitá trieda okna aplikácie QMainWindow [8] toto správanie nepodporuje. Dočasné riešenie je povoliť pre QDockWidgets [7] plávajúce prostredie panelov. Táto možnosť sa zapína pomocou **QDockWidget::DockWidgetFloatable**. Plávajúce prostredie zabezpečí, že jednotlivé paneli je možné z hlavného okna vytiahnuť a umiestniť ich ľubovolne na obrazovke.

  Odhad časovej náročnosti: 1 hodina

- **Problém:** Je zložité nájsť, ako sa zapínajú vypnuté paneli.

  Návrh zmeny: Pridanie položky „View“ do UI01 – Menu, ktorá bude obsahovať tlačítka ovládajúce zapnutie a vypnutie jednotlivých panelov.

  Odhad časovej náročnosti: 4 hodiny

- **Problém:** Rozmiestnenie panelov na tabuli sa neukladá. Pri každom spustení programu sú paneli rozmiestnené pôvodne.

  Návrh zmeny: Ukladať stav a polohu jednotlivých panelov užívateľského rozhrania. Pri spustení programu paneli nastaviť podľa uloženého stavu, ak existuje.

  Odhad časovej náročnosti: 8 hodín

- **Problém:** Pri vyššom počte lístkov rovnakej farby (3 a viac) je zdĺhavé hľadať lístky z UI02 – Panel rozpracovaných lístkov na UI04 – Panel tabule a naopak.

  Návrh zmeny: Pri vybraní lístka sa lístok zvýrazní aj na ostatných paneloch, ak sa na nich nachádza. **Odhad časovej náročnosti:** 6 hodín

- **Problém:** Pri zmene aktívneho panelu, v neaktívnych paneloch zostáva výber lístku. Tento výber zmení farbu lístku, takže znemožňuje identifikáciu lístku podľa farby.

  Návrh zmeny: Pri deaktiváci panelu (užívateľ aktivuje iný panel kliknutím do neho) sa zruší výber lístkov v tomto paneli.

  Odhad časovej náročnosti: 1 hodina

- **Problém:** V UI04 – Panel tabule počet riadkov neodpovedá počtu členov tímu a počet stĺpcov neodpovedá množstvu použitých stavov. Výber lístkov, ktoré sú v druhom a ďalších riadkoch člena tímu, alebo v druhom stĺpci stavu nezvýrazňuje korektne záhlavie.

  Návrh zmeny: Odstrániť deliace línie medzi stĺpcami rovnakého stavu a medzi riadkami rovnakého člena tímu. Pri výbere lístku na UI04 – Panel tabule zvýrazniť príslušné záhlavie.

  Odhad časovej náročnosti: 8 hodín

6.2.2 Aktualizácia programu

- **Problém:** Aktualizácia programu vyžaduje kopírovanie novej verzie programu do vlastného počítaču. Tento proces je pre užívateľovu príliš zdĺhavý. Užívatelia musia byť upozorňovaný na novú verziu.

  Návrh zmeny: Súbory, do ktorých program zapisuje, budú presunuté zo zložky programu do zložky Dokumenty prihláseného používateľa. Užívatelia budú môcť program spúšťať zo zdieľaného disku. Aktualizácie budú prebiehať prepísaním súborov programu na tomto zdieľanom disku.

  Odhad časovej náročnosti: 2 hodiny

6.2.3 Existujúce funkcionality

6.2.3.1 Posunutie lístka

- **Problém:** Odporúčané destinácie presunu lístka splývajú s farbami lístkov.

  Návrh zmeny: Odporúčané destinácie vyznačiť šrafovaním – „Qt::BDiagPattern“ [6].

  Odhad časovej náročnosti: 1 hodina

- **Problém:** Zobrazenie odporúčanej destinácie lístku by sa malo zobraziť už pri zodvihnutí lístka, nie až keď s ním pohneme.

  Návrh zmeny: Zobraziť destináciu lístku pri stlačení ľavého tlačidla myši na lístok. Prestať zobrať destináciu pri pustení tohto tlačidla.

  Odhad časovej náročnosti: 8 hodín

- **Problém:** Nie je možné nastavovať lístkom stavy, ktoré nie sú na tabuli.

  Návrh zmeny: Pridanie kontextového menu vyvolaného kliknutím ľavého tlačidla myši na lístok. Menu bude obsahovať možnosť zmeniť stav na akýkoľvek z používaných v NET Geniu.

  Odhad časovej náročnosti: 8 hodín

- **Problém:** Presun nenastavuje posunutému lístku údaje „Modified On“ a „Modified By“ v NET Geniu. Údaje sú potrebné pre náhľady z NET Genia, keďže užívatelia preferujú zoradenie podľa najaktuálnejších lístkov.

  Návrh zmeny: Nastavovať spomenuté údaje pri každom posune.

  Odhad časovej náročnosti: 1 hodina

6.2.3.2 História lístku

- **Problém:** Užívatelia očakávali, že sa v tooltipe zobrazí okrem histórie aj popis lístkov. Popis je vo väčšine lístkov moc dlhý a nezmestí sa celí na lístok v paneloch.

  Návrh zmeny: Celí popis lístka bude zobrazený v tooltipe nad históriou lístku.

  Odhad časovej náročnosti: 4 hodiny

6.2.4 Nové funkcionality

6.2.4.1 Automatické generovanie správy pri merge do vetvi grandmaster

Z testu vyplynulo, že táto funkcionalita nie je potrebná. Osoba zodpovedná za túto činnosť v tíme označila funkcionalitu ako nepotrebnú.

6.2.4.2 Presun všetkých lístkov z panelu namergovaných lístkov v tímovej master vetve do stavu ”Tested”.

Táto funkcionalita je potrebná, avšak bude nahradená funkcionalitou kontextového menu, ktorý umožní zmeniť stav na akýkoľvek z množiny stavov v NET Geniu. Funkcionalita je opísaná v kapitole 6.2.3.1 – Problém „Nie je možné nastavovať lístkom stavy, ktoré nie sú na tabuli“.

6.2.4.3 Pridať k lístkom hypertextový odkaz do NET Genia

- **Problém:** Program tímová tabuľa neobsahuje a ani nemá obsahovať všetky funkčnosti NET Genia. Pri potrebe práce na lístku v NET Geniu je potrebné si zapísať číslo lístka a podla neho lístok nájsť v NET Geniu.

  Návrh zmeny: Lístky budú mať dostupné na dvojklik a v kontextovom menu odkaz na „editačný formulár“ Todo alebo Bug podľa typu lístka.

  Odhad časovej náročnosti: 8 hodín

6.2.4.4 Aktualizácia dát

- **Problém:** Pri práci s tímovou tabulou je potrebné aktualizovať lístky na tabuli.

  Návrh zmeny: Aktualizácia dát bude prebiehať automaticky. Frekvencia aktualizácií sa bude líšiť v závislosti či program používame, alebo ho máme spustený v pozadí. Pri aktívnom používaní programu sa dáta aktualizujú každých 30 sekúnd. Ak je program v pozadí, aktualizácia prebehne raz za 5 minút. Informácie o čase poslednej aktualizácie budú zobrazené v UI01 – Menu.

  Odhad časovej náročnosti: 8 hodín

6.2.4.5 Notifikácie zmien

- **Problém:** Užívateľ nie je upozornený na lístky, ktoré sú mu priradené.

  Návrh zmeny: Ak sa pri aktualizácií dát zmení množina lístkov, ktoré má prihlásený užívateľ vo svojom riadku na UI04 – Panel tabule, tak dostane notifikáciu v podobe „QMessageBox“ [9], ktorý bude obsahovať informácie o lístkoch, ktoré sa zmenili a osobu, ktorá zmenu vykonala. Upozornenia si bude užívateľ môcť vypnúť vo svojich nastaveniach.

  Odhad časovej náročnosti: 10 hodín

Kapitola 7
Záver

Podarilo sa urobiť analýzu, ktorá zachytáva potreby všetkých tímov. Bolo zistené, že jednotlivé tímy využívajú iné stavy Todo a Bug v NET Geniu na zachytenie stavov lístkov na tabuli a majú inú postupnosť vývoja. V niektorých tímoch sa ako prvé po implementácií robí code review, v iných testing.

Z analýzy sa podarilo vytvoriť návrh riešenia, ktorý zistené odlišnosti umožňuje nastavovať pre jednotlivé tímy pomocou konfiguračných súborov. Návrh zabezpečuje jednoduché presunutie konfigurácie z konfiguračných súborov do informačného portálu NET Genium.

Na implementáciu programu boli použité vhodné nástroje Qt a C++. Výsledkom je robustný program, ktorý je jednoducho rozšíriteľný.

Program bol nasadený a otestovaný. Výsledky testu poukázali na možnosti zlepšenia v grafickom spracovaní užívateľského rozhrania. Boli navrhnuté jednoduché zmeny farebnosti a grafiky, ktoré zlepšia prehľadnosť programu a znížia tak kognitívnu zátaž užívateľov. Z testu ďalej vyplynula zmena prípadov užitia Release Managementu. Prípad užitia UC201 – Vytvorenie správy o lístkoch namergovaných vo vetvi team master 4.5.2.1 bol označený release managementom pri testovaní ako nepotrebný. Prípad užitia UC202 – Označenie všetkých lístkov v tímovej vetve master za namergované do grandmaster vetve popísaný v kapitole 4.5.2.2 bol nahradený kontextovým menu umožňujúcim presun popísaný v kapitole 4.3.1.5.

Test ukázal na potrebu notifikačného systému, ktorý bol popísaný v kapitole 6.2.4.5.

Pre všetky zistené nedostatky a potrebné rozšírenia programu bolo navrhnuté riešenie a určená odhadovaná časová náročnosť implementácie. Žiadny z nedostatkov nie je v rozpore s riešením a bude jednoducho implementovaný.

Program je nasadený, členovia tímu ho s obľubou používajú a uľahčuje im každodennú prácu.

Příloha A

Príručka programátora

A.1 Zdrojový kód

Zdrojový kód programu je dostupný v GitLab úložisku `https://gitlab.fit.cvut.cz/liptape1/teamtable.git`.

Git pre windows je dostupný na `https://git-scm.com/download/win`. Na prácu z kódom odporúčame úložisko naklonovať pomocou git príkazu clone.

Výpis kódu A.1 Git príkaz na vytvorenie klonu zdrojového kódu

```
git clone https://gitlab.com/gitlab-tests/sample-project.git
```

Po vykonaní git clone je vytvorený priečinok ktorý je kópiou úložiska. Tento priečinok nazývame repozitár. V tejto kapitole sa budeme na daný priečinok odkazovať pomocou: `<repository-dir>`.

A.2 Nastavenie vývojového prostredia

Nastavenie vývojového prostredia je ukázané na čistej inštalácií operačného systému.

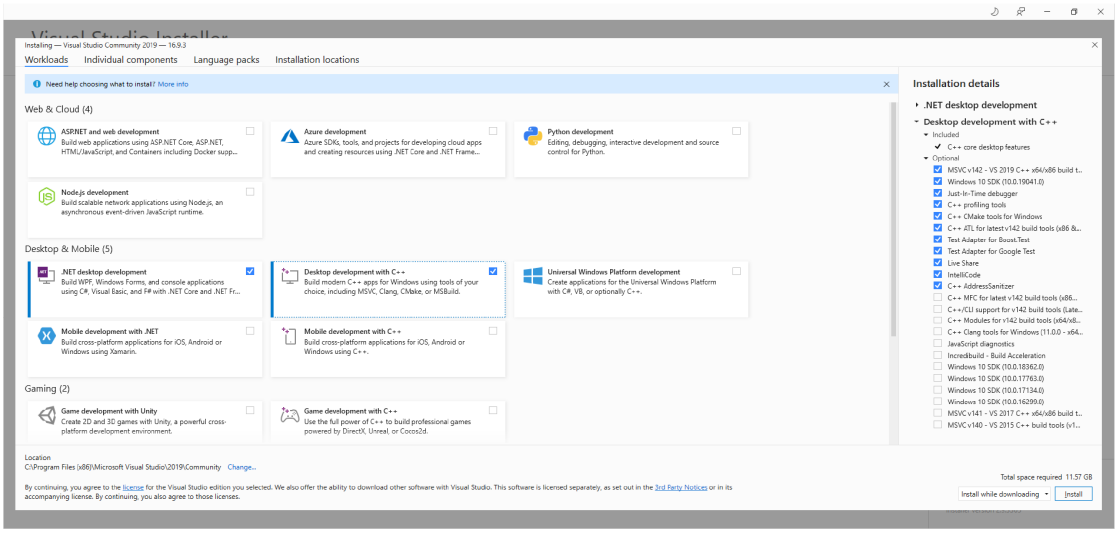
```
Edition          Windows 10 Home
Version          20H2
Installed on     4/7/2021
OS build         19042.631
Experience       Windows Feature Experience Pack 120.2212.31.0
```

A.2.1 Microsoft Visual Studio 2019 Comunity

Program je dostupný stiahnutie `https://visualstudio.microsoft.com/vs/community/`.

A.2.1.1 Inštalácia

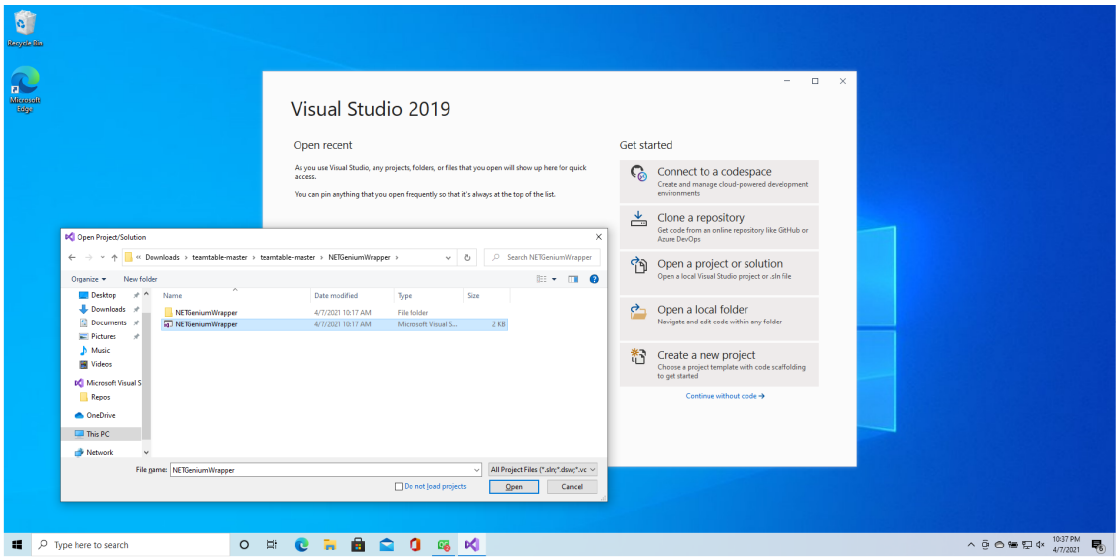
Inštalátor stiahneme a spustíme. Pri inštalácií zvolíme balíčky „.NET desktop development“ a „Desktop development with C++“, tak ako je ukázané na obrázku A.1



Obr. A.1 Visual Studio - inštalácia

A.2.1.2 Nastavenie projektu

Nainštalované Visual Studio spustíme, na úvodnej obrazovke vyberieme možnosť „Open a project or solution“. Vyberieme súbor `<repository-dir>/NETGeniumWrapper/NETGeniumWrapper.sln` podla obrázku A.2



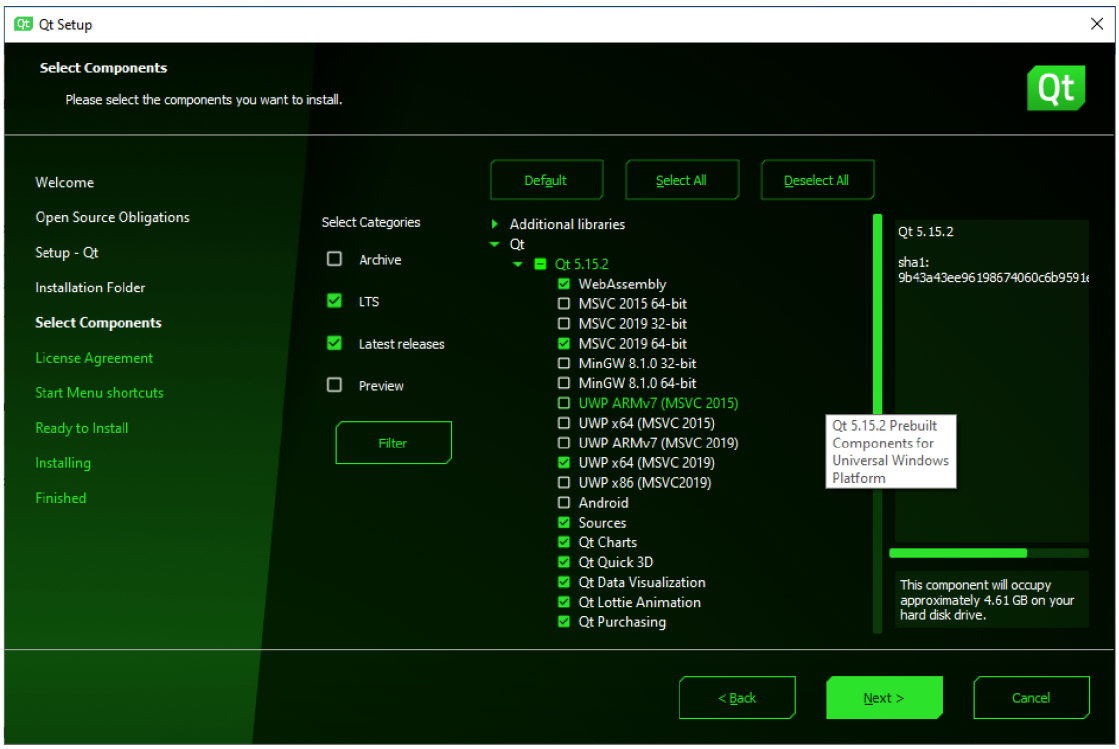
Obr. A.2 Visual Studio - nastavenie

A.2.2 Qt

A.2.2.1 Inštalácia

Qt nainštalujeme pomocou inštalátoru „qt-unified-windows-x86-4.0.1-1-online“ dostupného z `https://www.qt.io/download`. Pri inštalácií vyberieme balíčky Qt 5.15.2 podla obrázku A.3

a balíčky Qt Creatoru podla obrázku A.4.



Obr. A.3 Qt – nastavenie inštalovaných balíčkov Qt 5.15.2

A.2.2.2 Nastavenie projektu

Spustíme nainštalovaný Qt Creator. Vyberieme možnosť „Projects Open“ a vyberieme `<repository-dir>/src/TeamTable.pro`. Výsledok vidíme na obrázku A.5.

Následne projekt nastavíme podľa obrázku A.6. Pri výbere „Build directory:“ je potrebné zvolit cestu
`<repository-dir>/build_release` pre zostavenie na vydanie a cestu
`<repository-dir>/build_debug` pre zostavenie na vývoj.

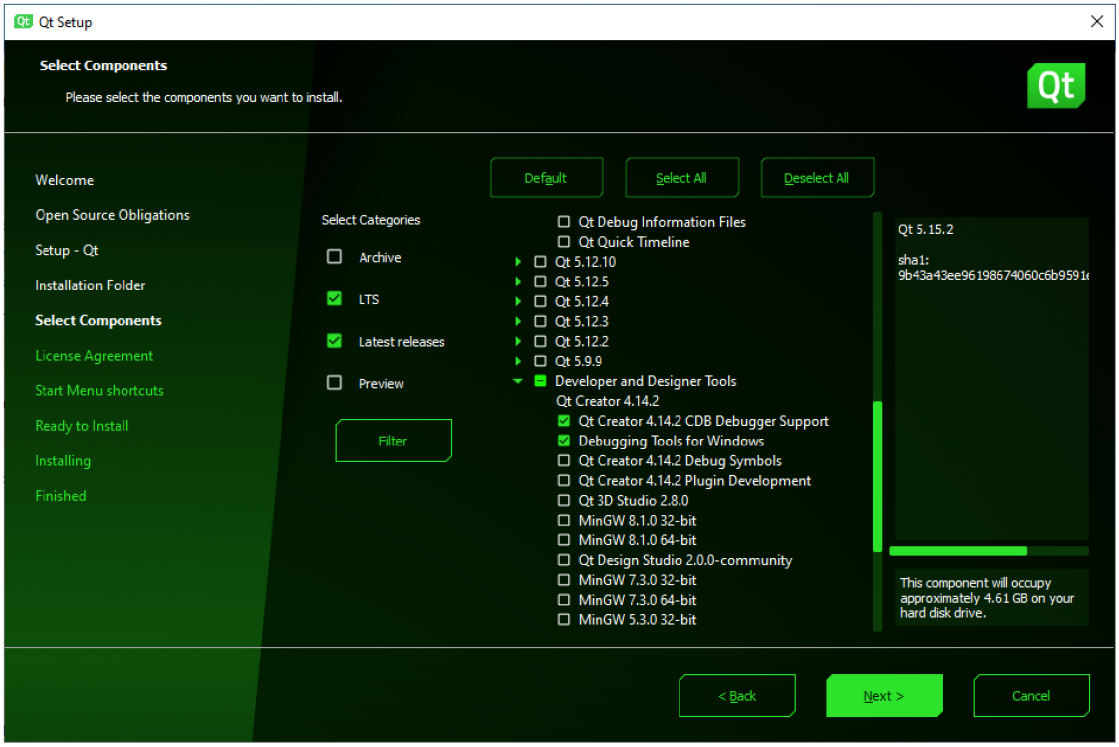
A.2.3 Microsoft ODBC Driver Management

Na fungovanie programu je potrebné mať nainštalovaný Microsoft ODBC Driver Management, ktorý slúži na komunikáciu z SQL Server databázou. Program je dostupny na:
`https://docs.microsoft.com/en-us/sql/connect/odbc/download-odbc-driver-for-sql-server?view=sql-server-ver15`.

A.3 Kompilácia pre užívateľa

Pre vydanie programu potrebujeme zostaviť oba programy – NETGeniumWrapper vo Visual Studiu a TeamTable v Qt Creatore. Oba programi zostavíme na vydanie – „Release build“.

Po zostavení oboch programov je potrebné k zostavenému Qt programu v priečinku `<repository-dir>/build_release/release` pridať Qt knižnice aby bol program spustiteľný. O pridanie potrebných knižníc sa stará nástroj **windeployqt** [12].



Obr. A.4 Qt – nastavenie inštalovaných balíčkov Qt Creator

Ďalší krok je pridanie konfiguračných súborov a programu NETGeniumWrapper. Toto je možné realizovať pomocou PowerShell použitím príkazov A.2.

Výpis kódu A.2 PowerShell príkazy na vytvorenie kompilácie pre užívateľa

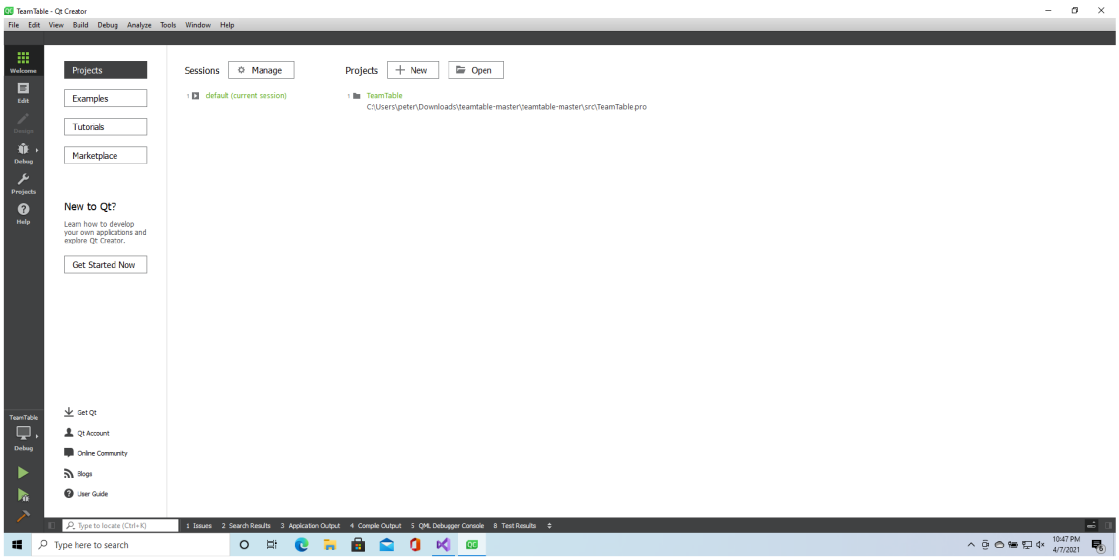
```
cd <repository-dir>
windeployqt .\build_release\release\ --release
cp .\config\ .\build_release\ -r
cp .\NETGeniumWrapper\NETGeniumWrapper\bin\Release
   .\build_release\NETGeniumWrapper\NETGeniumWrapper\bin\Release -r
```

Po splnení všetkých krokov je možné program tímovej tabule spustiť pomocou `<repository-dir>\build_release\release\TeamTable.exe`.
Ak chceme program nasadiť stačí skopírovať priečinok `<repository-dir>/build_release`.

A.4 Potrebné práva na používanie programu

Na používanie programu tímovej tabuľe je potrebné, aby užívateľ mal v databáze **netgenium** právomoc:

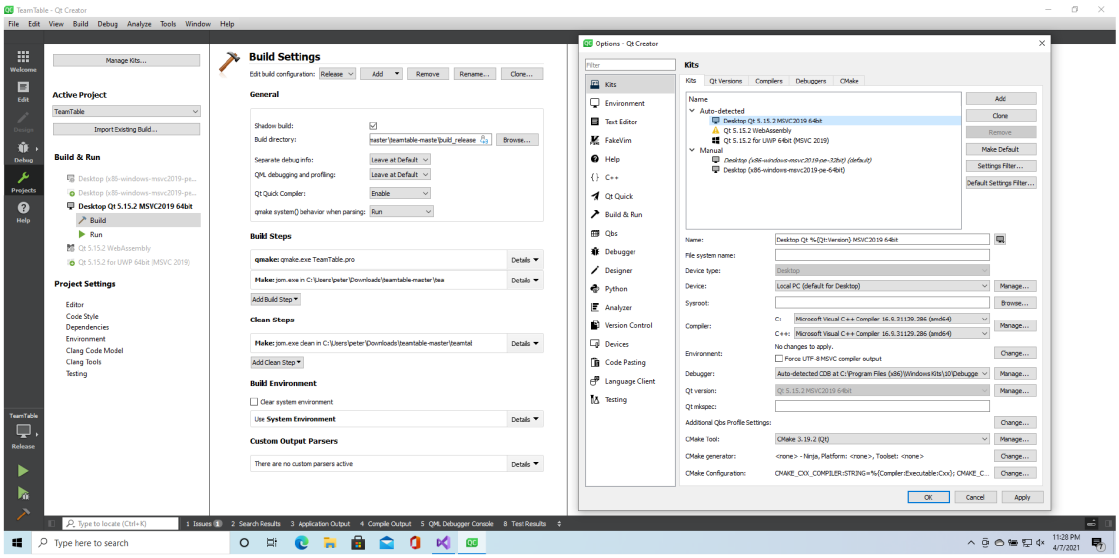
- **READ** do tabuliek:
  - ng_team
  - susers
  - ng_bugstatus0
  - ng_todostatus0



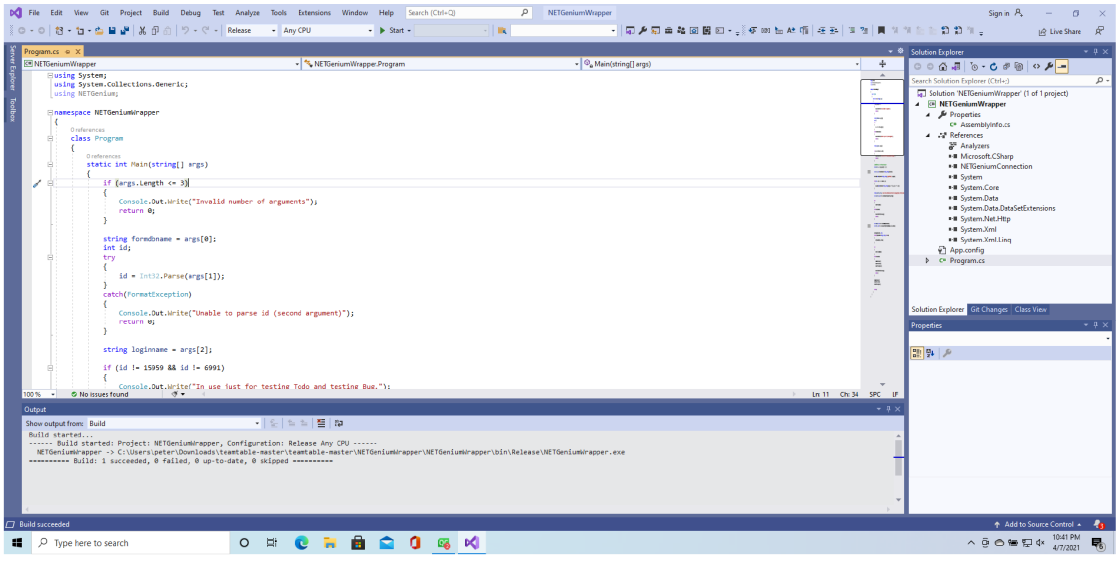
Obr. A.5 Qt – pridanie nového projektu

- ng_bug1
- ng_tododevelopmentii
- ng_bug1_history
- ng_tododevelopmentii_history

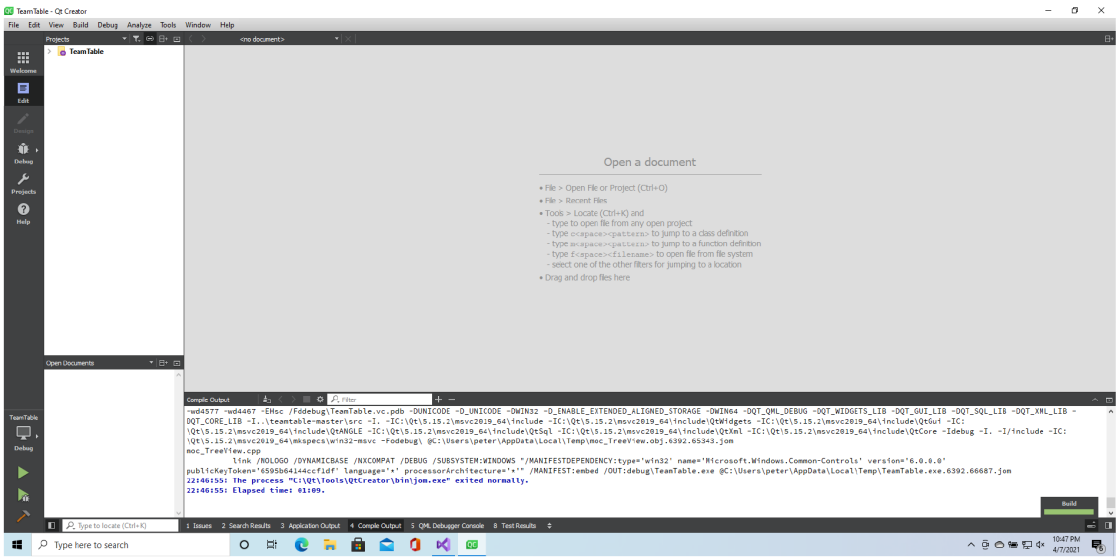
- **CREATE** do tabuliek:
  - ng_bug1_history
  - ng_tododevelopmentii_history
- UPDATE do tabuliek:
  - ng_bug1
  - ng_tododevelopmentii



Obr. A.6 Qt – pridanie nového projektu



Obr. A.7 Visual Studio - zostavenie



Obr. A.8 Qt - zostavenie

Příloha B

Test používateľnosti

B.1 Formulár

Team Table - Test používateľnosti

V prvom rade Vám ďakujem že ste si našli čas na vyplnenie tohto testu. Jeho cieľom je odhaliť nedostatky užívateľského rozhrania programu Team Table ktorý má nahradiť fyzickú tabulu umiestnenú v kancelárií.

Test je možné rozpracovať a vrátiť sa k nemu viac krát. Tento postup je doporučovaný. Test bude prebiehať jeden pracovný týždeň v termíne od 12.4. do 16.4. Pri každej zmene je posledné formulár odoslať pomocou tlačítka "Submit" na poslednej strane. Pre neskoršie upravovanie formuláru si uložte URL po odoslaní. (Odporúčam vytvoriť si záložku). Toto URL nezdieľajte, lebo by vaši kolegovia prepisovali vaše odpovede.

Počast tohto týždna, vždy v situácií kedy by ste pracovali z fyzickou tabulou v kancelárii použite program Team Table a prácu vykonajte v ňom.

Ak vám čokoľvek nebude fungovať alebo objavíte chybu (v programe alebo tomto formulári) tak ma neváhajte kontaktovať. Počas tohto týždna budú nájdené chyby opravené, tak aby neprekážali pri testovaní.

Po vyplnení testov prebehne krátky videohovor zameraný na hlbšie pochopenie vyplnených problémov. Vyplnený dotazník bude pridaný ako príloha do bakalárskej práce. Pri dotazníku nebude uvedené vaše meno, ani emailová adresa, iba pracovná pozícia.

Peter Liptai

Orientácia v paneloch užívateľského rozhrania

V nasledujúcich bodoch prosím vyplňte ako rýchlo sa vám podarilo nájsť jednotlivé paneli

1. Menu

Mark only one oval.

	1	2	3	4	5	
Panel som videl okamžiťe	◯	◯	◯	◯	◯	Trvalo mi to dlhú dobu alebo som ho vôbec nenašiel

2. Panel rozpracovaných lístkov

Mark only one oval.

	1	2	3	4	5	
Panel som videl okamžiťe	◯	◯	◯	◯	◯	Trvalo mi to dlhú dobu alebo som ho vôbec nenašiel

3. Panel naplánovaných lístkov

Mark only one oval.

	1	2	3	4	5	
Panel som videl okamžiťe	◯	◯	◯	◯	◯	Trvalo mi to dlhú dobu alebo som ho vôbec nenašiel

4. Panel tabule

Mark only one oval.

	1	2	3	4	5	
Panel som videl okamžiťe	◯	◯	◯	◯	◯	Trvalo mi to dlhú dobu alebo som ho vôbec nenašiel

5. Panel rozpracovaných lístkov mimo tím

Mark only one oval.

	1	2	3	4	5	
Panel som videl okamžiťe	◯	◯	◯	◯	◯	Trvalo mi to dlhú dobu alebo som ho vôbec nenašiel

6. Panel lístkov na automerge

Mark only one oval.

	1	2	3	4	5	
Panel som videl okamžiťe	◯	◯	◯	◯	◯	Trvalo mi to dlhú dobu alebo som ho vôbec nenašiel

7. Panel namergovaných lístkov v tímovej master vetve

Mark only one oval.

	1	2	3	4	5	
Panel som videl okamžiťe	◯	◯	◯	◯	◯	Trvalo mi to dlhú dobu alebo som ho vôbec nenašiel

8. Poznámky a postrehy. Ak vás čokoľvek v užívateľskom rozhraní prekvapilo, zaujalo alebo nahnevalo napíšte to sem.

Posunutie lístka

Otázky v tomto odseku budú zamerané na testovanie funkcionality posúvanie lístka. Posúvanie lístka je analogické ako posúvanie lístka po fyzickej tabuli.

9. Použili ste túto funkcionalitu?

Mark only one oval.

- ◯ Áno
- ◯ Nie

10. Všimli ste si odporúčané pozície počas posúvania lístku?

Mark only one oval.

	1	2	3	4	5	
Áno, odporúčané pozície som si všimol okamžite	◯	◯	◯	◯	◯	Nepostrehol som že by mi program niečo odporúčal

11. Rozumiete prečo sú odporúčané pozície sfarbené červenou a zelenou farbou? Váše vysvetlenie, alebo návrh na zmenu prosím napíšte.

História lístku

Na fyzickej tabuli zapisujeme históriu lístkov zo zadu. V program Team Table sa vám zobrazí ako Tool Tip ak podržíte myš na lístku

12. Použili ste túto funkcionalitu?

Mark only one oval.

- Áno
- Nie

13. Myslíte že forma zobrazenia histórie je prehľadná?

Mark only one oval.

	1	2	3	4	5	
Orientácia v histórií bola jednoduchá						V histórí som sa strácal, nájsť potrebnú informáciu bolo zložité

14. Nápady a postrehy na zmenu histórie lístku

Ďalšie funkcionality

Označte váš názor na potrebnosť jednotlivých funkcionalít

15. Automatické generovanie správy pri merge do vetvi grandmaster .

Mark only one oval.

	1	2	3	4	5	
Potrebné						Nepotrebné

16. Presun všetkých lístkov z panelu namergovaných lístkov v tímovej master vetve do stavu "Tested".

Mark only one oval.

	1	2	3	4	5	
Potrebné	◯	◯	◯	◯	◯	Nepotrebné

17. Pridať k lístkom hypertextový odkaz na editačný formulár v NET Geniu pri dvojkliku na lístok.

Mark only one oval.

	1	2	3	4	5	
Potrebné	◯	◯	◯	◯	◯	Nepotrebné

18. Napadá vás niečo iné? Navrhnite ďalšie funkcionality programu.

Všeobecné postrehy

19. Postrehy a nápady

B.2 Výsledky

Team Table - Test používateľnosti

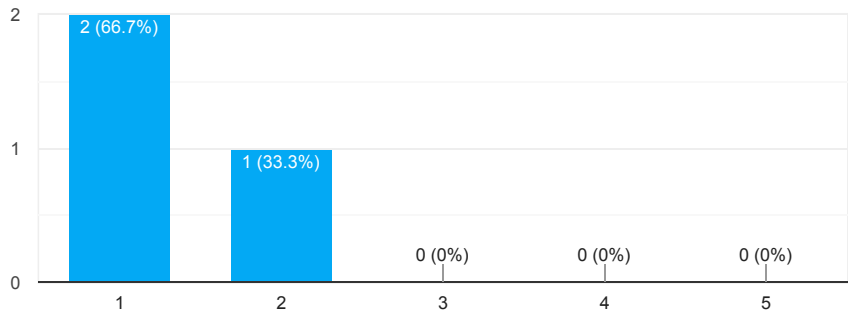
3 responses

Publish analytics

Orientácia v paneloch užívateľského rozhrania

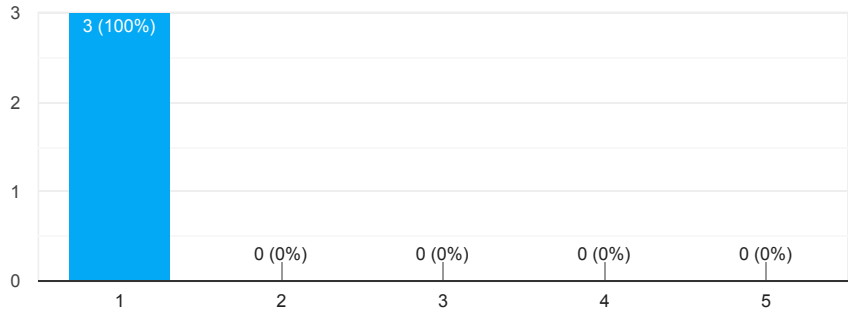
Menu

3 responses



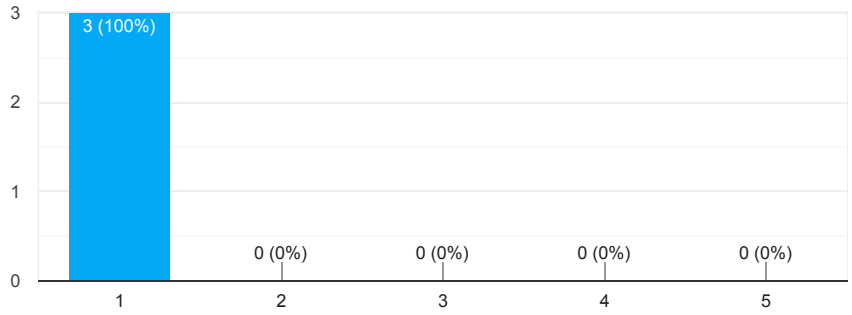
Panel rozpracovaných lístkov

3 responses



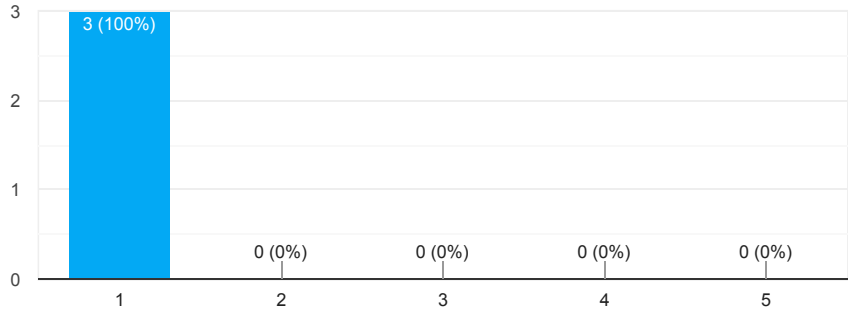
Panel naplánovaných lístkov

3 responses



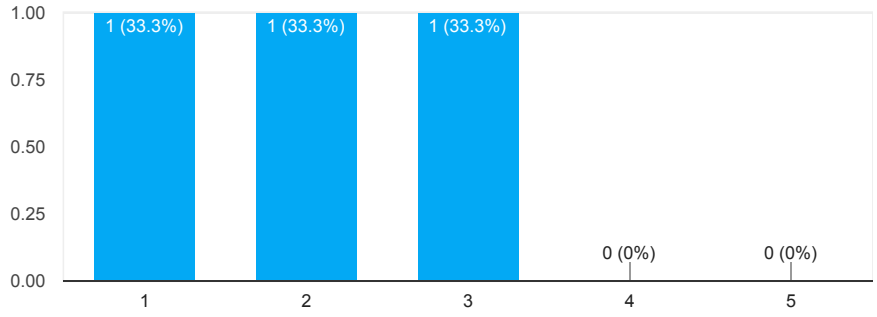
Panel tabule

3 responses



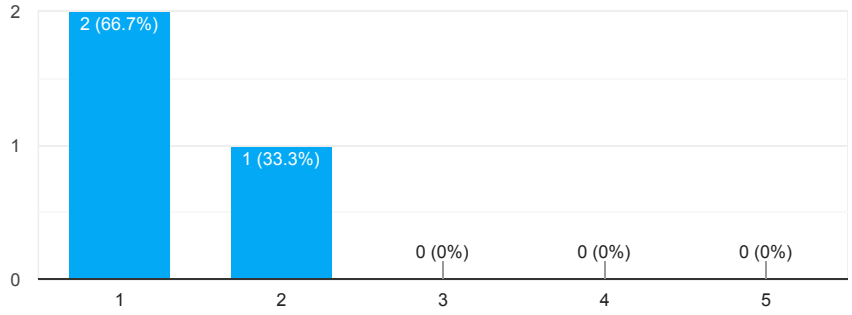
Panel rozpracovaných lístkov mimo tím

3 responses



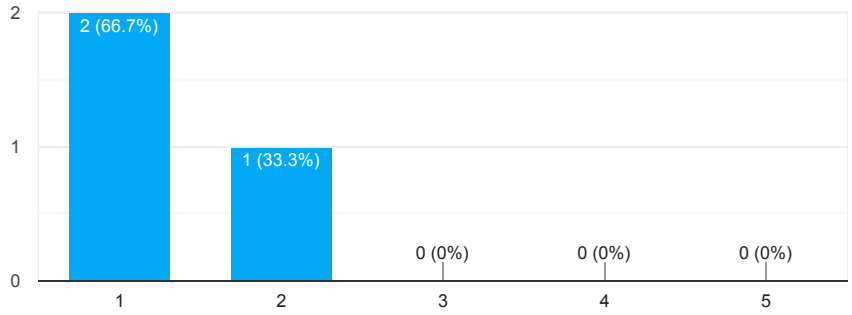
Panel lístkov na automerge

3 responses



Panel namergovaných lístkov v tímovej master vetve

3 responses



Poznámky a postrehy. Ak vás čokoľvek v užívateľskom rozhraní prekvapilo, zaujalo alebo nahnevalo napíšte to sem.

3 responses

Při výběru lístku v panelu by bylo hezké zvýraznit příslušný lístek na tabuli. Některé barvy user stories jsou podobné barvě pro bugy.

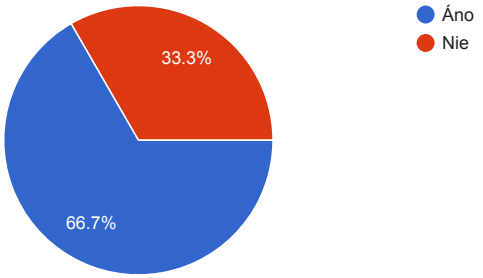
Je skvělé, že se dají panely v okně přesouvat.
Mohlo by to být ještě více benevolentní, do obecné tabulky. Například bych chtěl tři sloupce, vlevo přes celou výšku ToCorrect, uprostřed mít pod sebou tabuli a In Work tickets. To nejde.
Pak mi taky vadí, že když program zavřu, po otevření panely nedrží tak, jak jsem je minule poskládal.
Když nějaký panel zavřu, nevím, jak ho znova otevřít. Nakonec jsem na to přišel, takže to není až tak hrozné, ale menu Panely, kde by bylo totéž, co v kontextovém menu, by se hodilo.

Možná bych nějakou čarou oddělil bugy od US. Trošku mi to na první pohled splývá. V tabuli by možná mohla být označena role pracovníka (scrum master, tester, analytik, programátor, hlavní programátor), např. tooltipem po najetí na jméno.

Posunutie lístka

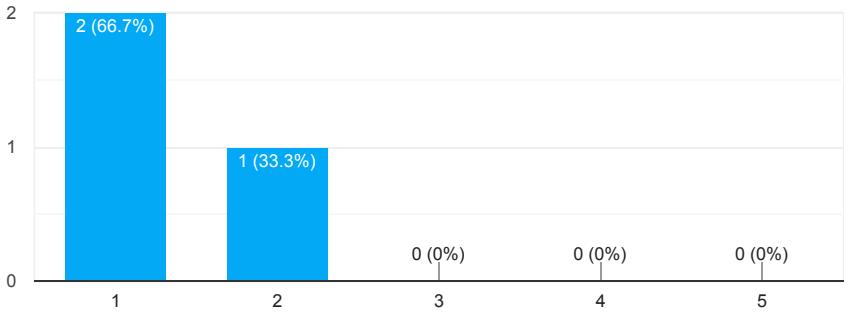
Použili ste túto funkcionalitu?

3 responses



Všimli ste si odporúčané pozície počas posúvania lístku?

3 responses



Rozumiete prečo sú odporúčané pozície sfarbené červenou a zelenou farbou? Váše vysvetlenie, alebo návrh na zmenu prosím napíšte.

3 responses

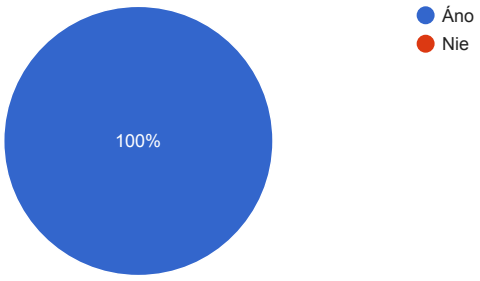
Ano, v pořádku.

Rozumím, že je to posun vpřed, pokud moje akce byla úspěšná, nebo posun zpět, pokud lístek vracím k opravení. Je to intuitivní a líbí se mi, že se zbarvují i panely okolo tabule.
Trochu mě mate, že to vypadá podobně jako barvy lístků. Červená jako červené bugy a zelená jako zelené US, respektive bugy na zelené US. U záhlaví panelů to nevadí.
Myslím, že by bylo lepší zvolit jiný design barevného zvýraznění, například tlustá barevná šipka uvnitř cílové buňky, šrafování, orámování.
Cílové políčko na tabuli (narozdíl od záhlaví panelů), se rozsvítí, až když při tažení opustím původní buňku. To je pozdě. Myslím, že by se to mohlo rozsvítit hned, jak stisknu tlačítko myši, klidně ještě než začnu táhnout. A když ho pustím, tak zase odbarvit.
Jako cíl by se dalo počítat i zabrané políčko jinou entitu, správně to určí sloupec i řádek a lístek se umístí na prázdné políčko.
Není mi jasné, jak nastavím další statusy, jak přesunu lístek z tabule pryč. U chyb máme možnost se zbavit lístku skrz další finální statusy, u TODO např. paused nebo tested.
Myslím, že by bylo dobré mít na tabuli u uživatelů i sloupec ToCorrect, kde by byly

História lístku

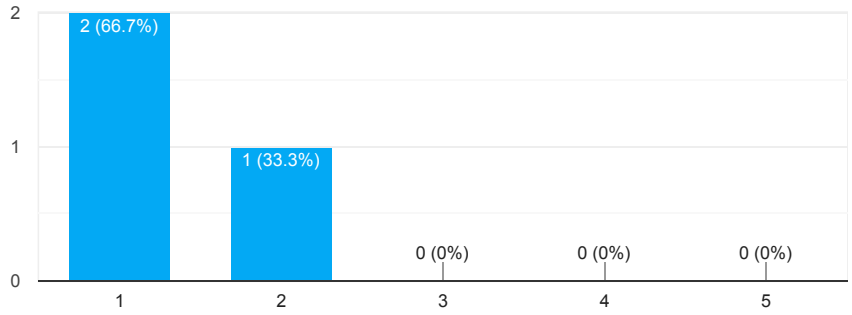
Použili ste túto funkcionalitu?

3 responses



Myslíte že forma zobrazenia histórie je prehľadná?

3 responses



Nápady a postrehy na zmenu histórie lístku

3 responses

Přístup k historii jsem nejdřív očekával pomocí pravého tlačítka myši a kontextového menu. Při výběru lístku by bylo hezké nakreslit jeho historii pomocí šipek, odkud kam se přesouval.

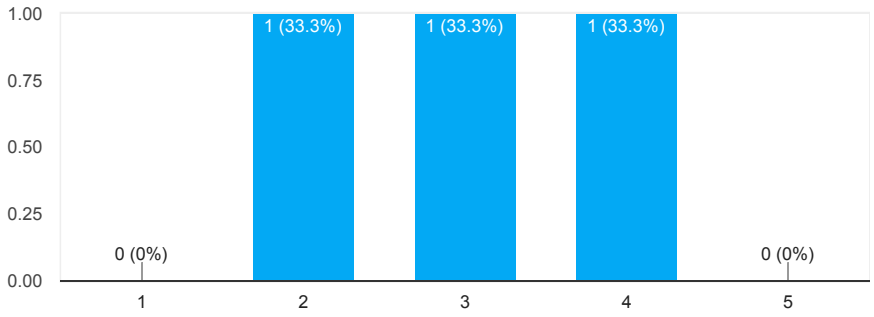
Historie je takto zobrazena přehledně.
V tooltipu bych zobrazoval i Subject.
V NG se ale do historie nevyplňuje datum Modified On a někdy ani Modified By. To způsobuje, že když mám v NG tabulku seřazenou podle Modified On, abych viděl poslední změněné entity, to co je změněno skrz tabuli je zahrabáno někde v historii. Tohle bych řekl, že je chyba.

To mi přijde hodně přehledné a užitečné. Na druhou stranu jsem trochu čekal, že tooltipu uvidím celý název úkolu na lístku. To mi přišlo jako celkem velký nedostatek. Názvy jsou dlouhé a roztahování panelů omezené. Nemám vlastně žádnou možnost přečíst si celý název. Dal bych to asi do toho tooltipu společně s tou historií.

Ďalšie funkcionality

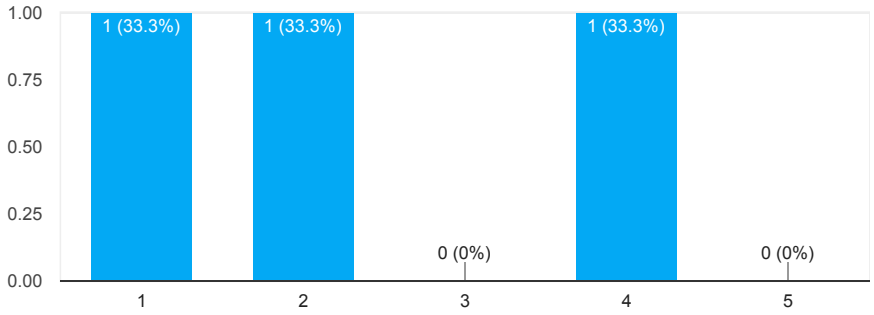
Automatické generovanie správy pri merge do vetvi grandmaster .

3 responses



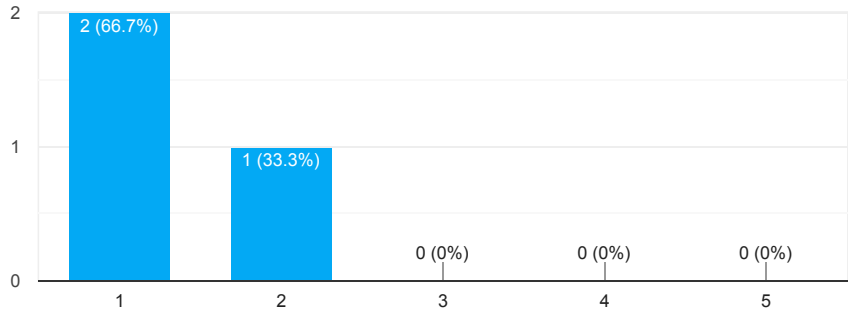
Presun všetkých lístkov z panelu namergovaných lístkov v tímovej master vetve do stavu "Tested".

3 responses



Pridať k lístkom hypertextový odkaz na editačný formulár v NET Geniu pri dvojkliku na lístok.

3 responses



Napadá vás niečo iné? Navrhnite ďalšie funkcionality programu.

3 responses

K dvojkliku by se už hodilo kontextové menu, kde by byla kromě odkazu i položka s historií.

Velkým nedostatkem je nutnost update. Když s tabulí nepracuji, může se sama updatovat každých pět minut. Pokud s ní pracuji (je aktivní okno, jezdím tam myší atd.) je to určitě třeba častěji, klidně každých deset nebo dvacet sekund. Při aktivním používání (jezdním myší po tabuli) bych se toho opravdu nebál. Samozřejmě je třeba dát pozor na synchronizaci, tedy aby nedošlo ke konfliktu při update nebo nějaké změně z mojí strany. Pak jsou chvíle, kdy je třeba update udělat okamžitě. Např. když přetáhnu lístek, je třeba udělat update co nejdřív tak, aby se mi správně načetla historie lístku s už provedenou změnou.

Graficky je tabule zpracována hezky, ještě bych vylepšil funkcionalitu spojených polí. Prakticky má tabule tolik sloupců, kolik je statusů a tolik řádků, kolik je lidí. V každé takové logické buňce je pak několik lístků. Chápu, že technicky je to zařízeno jinak, a je to dobře, že každý lístek má svoji fyzickou buňku, ale tuto myšlenku bych podpořil alespoň v zobrazení a chování.
Grid kreslit jen okolo logických buněk.
Když je aktivní buňka, ztuční mi to záhlaví (horizontální i vertikální), to ale funguje jen

Všeobecné postrehy

Postrehy a nápady

2 responses

Je třeba se vypořádat i se statusy, které nejsou ve work flow. Například chyba ve statusu ToDiscuss úplně zmizí z tabule. Dal by se na ně asi využít panel Out of Team (Out of workflow). Všechny statusy, co jsou v NG by měly být rozděleny na workflow statusy, to máme, cílové statusy (tyto lístky se nemají zobrazovat na tabuli) a ostatní (když někdo přidá nový, tak bude ostatní), které by na tabuli být vidět měli.

Předpokládám, že se u lidí přidávají řádky, když mají více lístků v jednom sloupci. Jinak jsem asi vše zmínil. Celkově se mi to opravdu líbí.

Literatúra

[1] Archiveddocs. /clr (common lanuguage runtime compilation). URL: https://docs.microsoft.com/en-us/cpp/build/reference/clr-common-language-runtime-compilation?view=msvc-160.

[2] Archiveddocs. Msvc compiler options. URL: https://docs.microsoft.com/en-us/cpp/build/reference/compiler-options?view=msvc-160.

[3] Archiveddocs. Remote desktop services overview. URL: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/hh831447(v=ws.11).

[4] IA Bahendwar, RP Bhardwaj, and PS Mundada. Amortized Complexity Analysis for Red-Black Trees and Splay Trees. *International Journal of Innovative Research in Computer Science & Technology*, 6(6):121–128, 2018. doi:10.21276/ijircst.2018.6.6.2.

[5] doc.qt.io. Model/view programming | qt widgets 5.15.3. URL: https://doc.qt.io/qt-5/model-view-programming.html.

[6] doc.qt.io. Qbrush class | qt widgets 5.15.3. URL: https://doc.qt.io/qt-5/qbrush.html.

[7] doc.qt.io. Qdockwidget class | qt widgets 5.15.3. URL: https://doc.qt.io/qt-5/qdockwidget.html.

[8] doc.qt.io. Qmainwindow class | qt widgets 5.15.3. URL: https://doc.qt.io/qt-5/qmainwindow.html.

[9] doc.qt.io. Qmessagebox class | qt widgets 5.15.3. URL: https://doc.qt.io/qt-5/qmessagebox.html.

[10] doc.qt.io. Qprocess class | qt widgets 5.15.3. URL: https://doc.qt.io/qt-5/qprocess.html.

[11] doc.qt.io. Qsqldatabase class | qt widgets 5.15.3. URL: https://doc.qt.io/qt-5/qsqldatabase.html.

[12] doc.qt.io. Qt for windows – deployment. URL: https://doc.qt.io/qt-5/windows-deployment.html.

[13] doc.qt.io. <qtglobal> - global qt declarations | qt core 5.15.3. URL: https://doc.qt.io/qt-5/qtglobal.html#qInstallMessageHandler.

[14] doc.qt.io. Technical faq. URL: https://wiki.qt.io/Technical_FAQ#Is_there_any_way_to_compile_the_Qt_source_code_and_Qt_applications_with_.2Fclr_flag.3F.

[15] Erich Gamma. *Design patterns: elements of reusable object-oriented software*. Addison-Wesley, 2013.

[16] Ing. Jan Kyral. *NET Genium Příručka administrátora*. NetGenium s.r.o., 7 2016.

[17] Mokole.com. Visually distinct colors generator. URL: https://mokole.com/palette.html.

[18] Scantips.com. More details about luminance in histograms. URL: https://www.scantips.com/lumin.html.

[19] VanMSFT. Choose an authentication mode - sql server. URL: https://docs.microsoft.com/en-us/sql/relational-databases/security/choose-an-authentication-mode?redirectedfrom=MSDN&view=sql-server-ver15#connecting-through-windows-authentication.

[20] w3schools.com. Xml introduction. URL: https://www.w3schools.com/xml/xml_whatis.asp.

[21] Wikipedia.org. Rgb - wikipedia. URL: https://sk.wikipedia.org/wiki/RGB.

Obsah priloženého médiá

- `readme.txt` — stručný popis obsahu média
- `exe` — adresár sa spustiteľnú formou implementácie
  - `release`
    - `TeamTable.exe` — spustiteľný súbor
- `src`
  - `impl` — zdrojové kódy implementácie
  - `thesis` — zdrojová forma práce vo formáte LaTeX
- `text` — text práce
  - `thesis.pdf` — text práce vo formáte PDF