

Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Ilkovičova 2, 812 19 Bratislava

Tímový projekt – Stratosféricky balón

Dokumentácia riadenia

Tím 10

Cvičiaci/Vedúci: Ing. Michal Valiček.

Akademický rok: 2016/2017

Autori:

Bc. Tomáš Urban

Bc. Martin Oravský

Bc. Márius Rak

Bc. Jakub Findura

Bc. Maroš Frkáň

Bc. Ján Pánis

Bc. Dominik Pisarovič

Obsah

Zoznam obrázkov	5
1. Úvod	6
2. Predstavenie tímu	6
2.1 Členovia tímu	6
3. Manažérské nástroje	7
3.1 Trello	7
3.2 Burndown for Trello.....	7
3.3 FB chat	7
3.4 Slack	8
3.5 Google Drive, Github, MS Office 365	8
4. Sumarizácie a retrospektívky šprintov	8
4.1 Šprint 1	8
4.1.1 Úlohy	9
4.1.2 Retrospektíva.....	9
4.2 Šprint 2	10
4.2.1 Úlohy	10
4.2.2 Retrospektíva.....	12
4.3 Šprint 3	14
4.3.1 Úlohy	14
4.3.2 Retrospektíva.....	16
4.4 Šprint 4	18
4.4.1 Úlohy	18
4.4.2 Retrospektíva.....	19
4.5 Zápisu zo stretnutí	20
4.5.1 1. stretnutie – zoznamovacie stretnutie.....	20
4.5.2 2. stretnutie – úvodné stretnutie.....	23
4.5.3 3. stretnutie - začiatok 1. šprintu	25
4.5.4 4. stretnutie – priebeh 2. šprintu	28
4.5.5 5. stretnutie - koniec 1. šprintu, začiatok 2. šprintu.....	30
4.5.6 6. stretnutie – priebeh 2. šprintu	32
4.5.7 7. stretnutie – koniec 2. šprintu, začiatok 3. šprintu	34
4.5.8 8. stretnutie – priebeh 3. šprintu	35
4.5.9 9. stretnutie – koniec 3. šprintu, začiatok 4. šprintu	36
4.5.10 10. stretnutie – priebeh 4. šprintu	38

4.5.11	11. stretnutie – priebeh 4. šprintu	39
4.5.12	12. stretnutie – koniec 4. šprintu	41
5	Globálna retrospektíva	42
6	Preberacie protokoly.....	44
7	Export úloh	45
8	Metodiky	45
8.1.	Metodika dokumentácie	45
8.1.1.	Dedikácia metodiky.....	45
8.1.2.	Formátovanie textu.....	45
8.1.3.	Metodika dokumentovania práce na projekte.....	46
8.1.4.	Metodika písania zápisníc stretnutí	46
8.1.5.	Metodika spisovania retrospektív	46
8.2.	Metodika konfigurácie softvérového systému.....	47
8.2.1	Dedikácia metodiky	47
8.2.2	Súvisiace metodiky.....	47
8.2.3	Roly	47
8.2.4	Procesy	47
8.3.	Metodika riadenia	49
8.4	Metodika archivácie verzií HW.....	51
8.4.1	Úvod	51
8.4.2	Fritzing	51
8.4.3	Procesy verziovania.....	52
8.5	Metodika verzií zdrojového kódu.....	58
8.5.1	Roly	58
8.5.2	Súvisiace metodiky.....	58
8.5.3	Zoznam pojmov a skratiek.....	58
8.5.4	Procesy	59
8.5.5	Pravidlá písania	65
8.6	Metodika rizík	65
8.6.1	Rozdelenie rizík	66
8.6.2	Ciele metodiky rizík	66
8.6.3	Postup identifikácie rizík	66
8.7	Metodika chýb	67
8.7.1	Úvod	67
8.7.2	Dedikácia metodiky	67
8.7.3	Zoznam nadväzujúcich metodík a dokumentov	67
8.7.4	Roly	68

8.7.5 Manažérské procesy	68
8.8 Metodika testovania.....	71
8.8.1 Úvod	71
8.8.2 Procesy	71
A. Vykonávanie jednotkových testov	72
B. Systémové a akceptačné testy	73
9 Uplatňovanie metodiky riadenia.....	74
9.1. Metodika pri prvom šprinte.....	74
9.2. Metodika pri druhom šprinte	75
9.3. Metodika pri tretím šprinte	77
9.4. Metodika pri štvrtom šprinte	79
10 Webové sídlo projektu.....	80
11 Bibliografia.....	81

Zoznam obrázkov

Obrázok 3 Testovacie zapojenie	55
Obrázok 4 Screenshot z programu Fritzing	55
Obrázok 5 Screenshot z programu Fritzing	56
Obrázok 6 Screenshot z programu Fritzing	56
Obrázok 7 Screenshot z programu Fritzing	56
Obrázok 8 Screenshot z programu Fritzing	57
Obrázok 9 Screenshot z programu Fritzing	57
Obrázok 10 GitHub.com - Vytvorenie nového repozitára	59
Obrázok 11 GitHub.com - formulár pre vytvorenie repozitára	59
Obrázok 12 GitHub.com - vytvorenie novej vety	60
Obrázok 13 SourceTree - Program pre správu git	61
Obrázok 14 GitHub.com - odkaz na vzdialený repozitár	61
Obrázok 15 SourceTree - Pripojenie vzdialeného repozitára	62
Obrázok 16 SourceTree - Rozhranie pre vytvorenie nového commitu	62
Obrázok 17 SourceTree - vytvorenie novej vety	63
Obrázok 18 SourceTree - spojenie vetyl (merge)	64
Obrázok 1 Burndown chart 2. sprintu	76
Obrázok 2 Burndown chart 3. sprintu	78

1. Úvod

Oblast' stratosférických letov je mimoriadne zaujímavá a sama o sebe značne špecifická. Táto práca obsahuje dokumentáciu k projektu, ktorým by sme chceli nadviazať na úspešné vypustenie stratosférického balóna v máji 2016. V nasledujúcich častiach sa budeme venovať popisu funkčnosti nášho tímu, jeho metodikám, manažérskym úlohám, dokumentovaniu našich šprintov v rámci scrumu ako aj retrospektívam v rámci tímovej práce.

2. Predstavenie tímu

Náš tím tvorí súdržnú agilnú skupinu, ktorej silnou stránkou je široká škála rozmanitých vedomostí, pokrývajúca rozsiahlu oblast' informatiky a informačných technológií. Jeho členovia svojimi rozsiahlymi vedomosťami a svojím vzájomným sa dopĺňaním vytvárajú perfektnú pôdu pre rast tohto projektu a pre vynikajúce výsledky. Náš tím pozostáva zo 7 členov:

Bc. Tomáš Urban

Bc. Martin Oravský

Bc. Mário Rak

Bc. Jakub Findura

Bc. Maroš Frkáň

Bc. Ján Pánis

Bc. Dominik Pisarovič

2.1 Členovia tímu

Ján Pánis je expert na nízko-prúd, drobnú elektroniku a fyziku. Dominik Pisarovič má zameranie na navigáciu dronov, rozpoznávanie a spracovanie obrazu. Maroš Frkáň je špecialista na strojové učenie a konvolučné neurónové siete. Mário Rak má vysoké skúsenosti s webovými systémami a drobnou elektronikou. Jakub Findura je odborník v oblasti Unity, webových technológií a android aplikácií. Martin Oravský je expert na siete, servery, Linux a grafiku. Tomáš Urban má rozsiahle znalosti v obore drobnej elektroniky (Arduino, Raspberry Pi...) a v sietiach.

Sme súdržná agilná skupina, ktorej silnou stránkou je široká škála rozmanitých vedomostí, pokrývajúca rozsiahlu oblast' informatiky a informačných technológií. V tíme pôsobia Ján a Tomáš, ktorí vedia zabezpečiť analýzu, návrh a implementáciu HW modulu. Pri tvorbe tohto modulu uplatnia svoje vedomosti, ktoré okrem iného nadobudli vo svojich

bakalárskych prácach s názvami: Batéria s rýchlym nabíjaním a Inteligentný dom. Martin vie zabezpečiť prenos komunikácie HW modulu do našej technologickej základne a nainštalovanie, konfiguráciu a následnú administráciu webservera. Skúsenosti získal v súčasnom zamestnaní ako aj v bakalárskej práci, ktorá sa zaoberala protokolom TCP v bezdrôtových systémoch. Dominik vie pracovať so súradnicami a zabezpečiť lokalizáciu HW modulu, jeho navigáciu, spracovanie obrazu a identifikáciu objektov v okolí. Dominikov prínos v tejto problematike pramení v jeho bakalárskej práci venovanej rozoznávaniu objektov a navigácií dronov. Maroš vie sofistikované spracovávať údaj a podávať efektívne výsledky. Využitím strojového učenia môžeme dosiahnuť objavovanie znalostí z rôznych typov dát. Márius môže zabezpečiť sprostredkúvanie dát z HW modulu do webového rozhrania. Webovým technológiám sa aktívne a kommerčne venuje už niekoľko rokov. Vo svojej bakalárskej práci sa venoval HW a spracovaniu dát. Jakub dokáže sprostredkúvať údaje z HW používateľovi pomocou mobilného zariadenia.

3. Manažérské nástroje

Z hľadiska manažmentu našej práce sme sa postupne venovali rôznym nástrojom na správu a koordináciu pokroku v rámci nášho tímového projektu.

3.1 Trello

Trello je vynikajúci webový nástroj určený na manažment projektov. Jeho využívanie je jednoduché a veľmi efektívne. V našom projekte sme sa vďaka Trellu dokázali výborne skoordinovať a tak markantne znásobiť efektivitu nášho scrumu.

3.2 Burndown for Trello

Vzhľadom k tomu, že freeware licencia Trella priamo nepodporuje generovanie Burndown chartov, sme boli nútení nájsť si inú aplikáciu. Burndown for Trello podľa možností uspokojivo splnil naše požiadavky.

3.3 FB chat

V začiatkoch našej práce, kde bolo potrebné rýchlo a efektívne sa zosynchronizovať a začať spolupracovať od prvého momentu, bol FB chat jediným a uspokojivým riešením. Jeho

velkou výhodou bola rýchla odozva všetkých členov a ich aktívne zapájanie sa do debát. Avšak kvôli neprehľadnosti správ sme boli nútení nájsť vhodnú alternatívu.

3.4 Slack

Slack nás oslovil v priebehu druhého týždňa spolupráce a dokonale uspokojil naše potreby a viac než dostačujúco nahradil dosiaľ využívaný FB chat. Prehľadnosť našej komunikácie sa enormne zlepšila, čo sa odrazilo na zvýšenej aktivite a vyššej efektivite spolupráce a plnenia na seba nadvážujúcich taskov.

3.5 Google Drive, Github, MS Office 365

Neodmysliteľnou súčasťou našej spolupráce je zdieľanie výsledkov našej práce v reálnom čase za účelom zamedzenia duplicity prác na našich dokumentoch. Pre tento účel sme zo začiatku používali spoločný účet na Google Drive a Githube. Koncom druhého sprintu sme spisovanie dokumentácie presunuli do zdieľaného MS Wordu, ktorý nám poskytoval lepšie možnosti formátovania spoločného textu.

4. Sumarizácie a retrospektívy šprintov

Táto kapitola obsahuje už ukončené šprinty, ich výsledky, zápisu zo stretnutí, priebeh spracovania úloh ako aj retrospektívy zhrnuté pri ich ukončení.

4.1 Šprint 1

Prvý šprint sa niesol v takom tréningovom duchu, vzhľadom k prvému pokusu o implementáciu scrumu do riešenia nášho projektu. V snahe naučiť sa efektívne využívať scrum sme urobili mnoho chýb, avšak s dôrazom na empirický prístup k tejto práci sme sa v mnohom poučili a posunuli tak efektivitu našej práce o množstvo krokov dopredu.

4.1.1 Úlohy

Úloha	Zodpovedná osoba	Stav
Dekompozícia user stories	Všetci	OK
Pozriť ako funguje trello so slackom	Všetci	OK
Zapísať si svoje estimates	Všetci	OK
Flask python	Martin Oravský	OK
Analýza grantov na ministerstve(ESA)	Martin Oravský	OK
Motivácia, výber projektu	Všetci	OK
Spísanie úvodu a celej kostry do dokumentácie	Dominik Pisarovič	OK
Zlepenie analýzy do jednej dokumentácie	Dominik Pisarovič	OK
Odkazy, prístupy a pdfka na jednu kartu	Dominik Pisarovič	OK
Analýza UML-RT	Maroš Frkáň	OK
História vypúšťania balónov	Maroš Frkáň, Jakub Findura	OK
Názov a plagát tímu	Ján Pánis	OK
Kalendár tímu spolu s termínmi	Ján Pánis	OK
Prepojenie Github, Trello, Slack	Márius Rak	OK
Preštudovať dôkladne funkčnosť trella	Márius Rak	OK
Analýza a výber používaných nástrojov	Márius Rak	OK

4.1.2 Retrospektíva

V závere šprintu číslo jedna sme sa začali zamýšľať nad našou interpretáciou a samotným využívaním scrumu v našom projekte. Šprint ako taký bol v celku úspešný,

dokončili sme veľké množstvo úloh, rozchodili sme základné prvky hardvéru potrebného pre nás projekt a taktiež aj analýza nabrala celkom slušný vzhľad.

Avšak scrum ako taký u nás celkom nefungoval. Zle sme si definovali user stories a následne aj jednotlivé tasky, robili sme veci do radu a nebrali do úvahy hodnotenie user stories.

1. Čo bolo dobré

- Splnených okolo 35 jednotlivých taskov rozličnej náročnosti
- Rozsiahle spracovanie analýzy
- Rozchodenie základných prvkov hardvéru
- Objavenie chybnej adaptácie scrumu na nás projekt a návrh reštrukturalizácie

2. Čo bolo zlé

- Komplikácie v práci v tíme, nezosúladený tím
- Nedôsledné definovanie DoD pre jednotlivé tasky
- Riešenie taskov ALAP počas celého šprintu
- Nedostatočná komunikácia v tíme

4.2 Šprint 2

Druhý šprint z pohľadu využívania scrumu bol omnoho lepší. Tentoraz sa niesol v duchu preúčaní sa a zžívaní sa so scrumom a jeho upravovaním a aplikovaním na nás projekt.

4.2.1 Úlohy

Absolvovaných 50 story pointov.

Úloha	Zodpovedná osoba	Stav
Zistiť ktorí členovia potrebujú Git repozitáre	Martin Oravský	OK
Vytvoriť Git repozitáre pre Jančího	Martin Oravský	OK
Vytvoriť Git pre Tomáša	Martin Oravský	OK

Analyzovať ako sa dá posielat' arduino kód z IDE na Git	Martin Oravský	OK
Štúdium Git	Martin Oravský	OK
Štúdium continuous integration	Martin Oravský	OK
Nastaviť CI pre tímový web	Martin Oravský	OK
Nastaviť CI pre live mapu balóna	Martin Oravský	OK
Analýza FB API	Maroš Frkáň	OK
Vytvoriť funkčný WIP skript pre postovanie FB statusov	Maroš Frkáň	OK
Vyskúšať spam obmedzenia pre FB statusy	Maroš Frkáň	OK
Analýza merania času na mikrokontroléri Arduino	Ján Pánis	OK
Prepočítanie času na mikrokontroléri a poslanie údajov do PC	Ján Pánis	OK
Zobrazenie času v PC	Ján Pánis	OK
Získať info o vedení tímu od Jančiho	Márius Rak	OK
Spísat' ako sme pristupovali k scrumu	Márius Rak	OK
Doštudovať si scrum	Márius Rak	OK
Spísat' pravidlá scrumu pre násť tím	Márius Rak	OK
Vysvetliť scrum členom	Márius Rak	OK
Naštýlovať web	Jakub Findura	OK
Implementovať frontend javascript	Jakub Findura	OK
Implementovať backend	Jakub Findura	OK
Naštýlovať mapu	Jakub Findura	OK
Vykresľovanie predpokladanej trasy	Jakub Findura	OK

Analýza spôsobov merania teploty	Ján Pánis	OK
Výber vhodného senzoru a jeho HW implementácia	Ján Pánis	OK
Prepojenie senzoru s mikrokontrolérom a výpočet hodnoty teploty	Ján Pánis	OK
Preposielanie a zobrazovanie hodnôt na počítači v čase	Ján Pánis	OK
Vypracovať prihlášku	Dominik Pisarovič	OK
Revidovať a odovzdať prihlášku	Dominik Pisarovič	OK
Získať informácie o práci ostatných	Dominik Pisarovič	OK
Spísať zápisnice	Dominik Pisarovič	OK
Spísať retrospektívny	Dominik Pisarovič	OK
Nájdenie vhodných knižníc pre použitie s GSM shieldom	Tomáš Urban	OK
Zakúpenie SIM kariet	Tomáš Urban	OK
Zapojenie a implementácia základnej komunikácie pomocou GSM	Tomáš Urban	OK

4.2.2 Retrospektíva

- **Jano**

- spravil všetko čo mal, možno niečo naviac
- časy v celku odhadol celkom relatívne fajn, nečakal že spisanie analýzy mu zaberie tak dlho
- zanalyzoval meranie teploty
- zanalyzoval meranie času
- vie preposielat údaje z gps do PC
- odporúča zapisovať analýzu a dokumentáciu k aktuálnej práci priebežne a to ešte skôr ako sa pustíme do ďalšieho tasku

- **Michal**

- bol v SOSE
- napísal grant
- antény a arduino vezmeme zo starého modulu
- príjmač máme univerzálny
- CHCE
 - analýzu zvýšenej spoločnosti
 - Ošetrenie výpadku signálu a znovunabehnutie do ďalšieho releasu

- **Dominik**

- tiež stihol všetko čo mal
- celkom to aj rýchlo ubehlo a tasky odsýpalí
- mohol by sa zlepšiť v nerobení taskov ALAP.

- **Maroš**

- zanalyzoval facebook API
- vytvoril funkčný WIP skript pre postovanie FB statusov
- spravil všetko tak ako si naplánoval
- potivo priebežne spracoval aj dokumentáciu svojej práce

- **Jakub**

- Stihol všetky vytiahnuté tasky
- potrebuje zlepšiť spracovanie dokumentácie a analýzy

- **Martin**

- Pracoval na continuous integration
- stihol všetko
- chce zapracovať na priebežnom dokumentovaní svojej práce

- **Tomáš**

- Podarilo sa spraviť všetky technické veci
- Prvé funkcionality fungujú správne
- nestihol dokumentáciu, dobehne ju
- Chce sa zlepšiť v nedokladaní vecí na nedeleň večer, chce robiť veci skôr

- **Márius**

- Veľa vecí robil na poslednú chvíľu
- niektoré veci robil ešte počas stretnutia

- Stihol všetko čo mal naplánované
- zle vybratie podporných úloh týkajúcich sa aktuálneho šprintu, ktoré je možné robiť až po ukončení šprintu

3. Čo bolo dobré

- Úspešné splnenie všetkých user stories
- Lepšie fungovanie v tíme
- Zlepšené scrumovanie

4. Čo bolo zlé

- Riešenie úloh ALAP
- Nesprávne definovanie DoD
- Nedopĺňanie dokumentácie priebežne, resp. vôbec
- Slabé dokumentovanie plnených úloh

4.3 Šprint 3

Tretí šprint sme začali s veľkým nadšením zo požadovaného dostatočného chápania scrumu ako takého a jeho aplikácie na nás projekt. Vzhľadom k tomu že sme už mali vlastné know-how, stretnutia odsýpali šikovne a celkovo naša práca sa zlepšila.

4.3.1 Úlohy

Úloha	Zodpovedná osoba	Stav
Zistiť ktorí členovia potrebujú Git repozitáre	Martin Oravský	OK
Vytvoriť Git repozitáre pre Jančiho	Martin Oravský	OK
Premysliť štruktúru dokumentácie na webe	Martin Oravský	In queue
Nahodiť novú štruktúru	Martin Oravský	In queue
Analýza dostupných možností posielania fotografií zo stratosféry	Martin Oravský	OK

Rozhodnúť podľa analýzy posielania foto, či ich budeme posielat'	Martin Oravský	OK
Vytvoriť finálnu FB skupinu	Maroš Frkáň	OK
Vytvorenie Twitter účtu a skriptu pre automatické posielanie obsahu pre naše vypustenie balónu	Maroš Frkáň	In progress
Ošetrenie možných náhodných chybových súradníc	Ján Pánis	In queue
Implementácia odosielania údajov z PC	Ján Pánis	In queue
HW implementácia zakúpeného GPS shieldu	Ján Pánis	In progress
Implementácia programu pre Arduino	Ján Pánis	In progress
Návrh schémy posielaných údajov + API	Jakub Findura, Ján Pánis	In progress
Preskúmať možnosti konverzie JSONu do PDF	Márius Rak	In queue
Set-up githubu	Márius Rak	In queue
Vytvoriť GUI pre vkladanie JSONu a získavanie PDF	Márius Rak	In queue
Parsovanie JSONu na potrebné dátá a dátové štruktúry	Márius Rak	In queue
Grafika pre vyexportované PDF v HTML	Márius Rak	In queue
Konverzia grafiky do PDF	Márius Rak	In queue
Set-up nástroja pre bežné používanie	Márius Rak	In queue
Návrh architektúry servera	Jakub Findura	In progress
Implementácia API servera	Jakub Findura	In progress

Implementácia databázovej vrstvy	Jakub Findura	In queue
Implementácia logiky servera	Jakub Findura	In queue
Prepísat' texty na webe	Dominik Pisarovič	In queue
Preveriť pravost', správnosť a pravdivosť textov	Dominik Pisarovič	In queue
Zozdielať dokumentáciu v jednom Word dokumente	Dominik Pisarovič	OK
Zabezpečiť úspešný prístup všetkých členov	Dominik Pisarovič	OK
Spísať zápisnice	Dominik Pisarovič	OK
Spísať retrospektívny	Dominik Pisarovič	OK
Zohnať a zabezpečiť vloženie výstupov úloh do dokumentácie	Dominik Pisarovič	In progress
Návrh spôsobu príjmania SMS	Tomáš Urban	In progress
Zohnať potrebné zariadenia	Tomáš Urban	In progress
Implementácia	Tomáš Urban	In queue
Testovanie	Tomáš Urban	In queue

4.3.2 Retrospektíva

S výsledkom šprintu sme boli celkovo spokojný aj napriek nižšej velocity(42), čo bolo ale spôsobené odovzdávaním dokumentácie k prvému kontrolnému bodu. Prešli sme si splnené úlohy a s výnimkou dvoch user stories sa nám podarilo všetko úspešne dokončiť.

- **Tomáš**

- prehodnotil spracovanie s modemom,
- nestihol dokončiť ale zdokumentoval čo zatial spravil, zvyšok presúvame do ďalšieho šprintu, pretože tento návrh realizovať nejdeme,

- zanalyzuje nejakú inú možnosť

- **Maroš**

- spravil všetko čo mal
- twitter, facebook vytvoril skupiny
- Márius by mohol prepojiť s webom
- skript hodil na server, no musí to prebrať ešte s Kubom, nedá sa vyznať v Kubovom kóde

- **Jano**

- mal dva tasky, oboje stihol
- posielanie dát na server funguje, cíti limitácie arduina z hľadiska výkonu pri viacero premenných, je potrebne prejsť na MEGU – väčší výkon
- dátá posielame na nás server, Kubo nemá doladene v jeho kóde aby sa to zobrazovalo na serveri
- GPS súradnice už tiež posielala na server, posielala to Json formát, ošetruje už náhodné chybné súradnice
- Spísal k tomu dokumentáciu

- **Márius**

- nedostal sa k niektorým taskom, ale pracuje na tom nástroji na export z Trella
- aktuálne potrebuje dátá z jsonu uložiť do nejakých premenných, nástroj vyzerá že bude funkčný
- Jano ho konštruktívne skritizoval za nedokončenie jednoduchých taskov, Márius má zamakať do budúcnia

- **Dominik**

- Pracovalo sa mu celkom dobre
- stihol všetko, celkom načas, nie na poslednú chvíľu
- úspešne zakončil spisovanie dokumentácie pre prvý kontrolný bod a odovzdal do aisu

- **Martin**

- začal v predstihu, nie v utorok ale už v nedeľu, stihol všetko čo mal, a je spokojný
- mal by spisovať dokumentáciu priebežne a do nabudúca skúsi dodržať začiatok ďalšieho tasku až po spísaní dokumentácie k predošlým
- zlyhalo to preto lebo si neboli istý či to má spisovať keď to aj tak do dokumentácie nebudeme dávať keď to ani robiť nepôjdeme, spíše to ale, pretože do analýzy je to vhodné

- **Michal**

- single point of failure – ak nám zlyhá jeden spôsob komunikácie, nemôže to znefunkčniť celý modul

5. Čo bolo dobré

- Úspešné odovzdanie dokumentácie k prvému kontrolnému bodu
- Zrýchlenie a zefektívnenie stretnutí

- Zlepšenie priebežného plnenia úloh od začiatku šprintu
6. Čo bolo zlé
- Nedokončenie všetkých user stories
 - Nižšia velocity
 - Slabé dokumentovanie plnených úloh

4.4 Šprint 4

4.4.1 Úlohy

Úloha	Zodpovedná osoba	Stav
Metodika dokumentácie	Dominik Pisarovič	OK
Chcem majáčik an jemné dohľadanie	Dominik Pisarovič	OK
analýza aktuálnych riešení	Dominik Pisarovič	OK
Špecifikácia požiadaviek	Dominik Pisarovič	OK
Návrh riešenia	Dominik Pisarovič	OK
Zakúpenie jednotlivých súčiastok	Dominik Pisarovič	OK
Implementácia	Dominik Pisarovič	OK
Testovanie	Dominik Pisarovič	OK
Metodika testovania	Jakub Findura	OK
Metodika verziovania	Jakub Findura	OK
Zobrazenie poslaných dát na onepage stránke	Jakub Findura	OK
Prepojenie databázy servera so stránkou	Jakub Findura	OK
Metodika archivácie verzií HW	Ján Pánis	OK
Ukladanie zálohy na pamäťovú kartu	Ján Pánis	OK
Analýza ukladania dát na pamäťovku + výber zariadenia	Ján Pánis	OK
HW implementácia	Ján Pánis	OK
Implementácia programu na Arduine	Ján Pánis	OK
Metodika rizík	Maroš Frkáň	OK
Chceme prijať odoslanú SMS z GSM modulu	Tomáš Urban, Maroš Frkáň	OK
Zohnať potrebné zariadenia	Tomáš Urban, Maroš Frkáň	OK
Implementácia	Tomáš Urban, Maroš Frkáň	OK
Testovanie	Tomáš Urban, Maroš Frkáň	OK
Metodika chýb	Tomáš Urban	OK
Analýza UML-RT	Martin Oravský	OK
Continuous integration	Martin Oravský	OK
Fixnúť CI	Martin Oravský	OK
Metodika riadenia	Márius Rak	OK
Export dát o backlogu z Trella	Márius Rak	OK
Vykreslovanie pekných exportov z trella	Márius Rak	OK

Vytvoriť GUI pre vkladanie JSONu a získavanie PDF	Márius Rak	OK
Parsovanie JSONu na potrebné dátá a dátové štruktúry	Márius Rak	OK
Grafika pre vyexportované PDF v HTML	Márius Rak	OK
Konverzia grafiky do PDF	Márius Rak	OK
Set-up nástroja pre bežné používanie	Márius Rak	OK
Upraviť rozhranie pre nahrávanie čiastkových súborov	Márius Rak	OK
Dokumentácia k nástroju	Márius Rak	OK

4.4.2 Retrospektíva

- **Tomáš**

- relatívne v čas všetko stihol
- potivo priebežne spracoval aj dokumentáciu svojej práce
- stihol by možno ešte jeden task, v budúcnu sa vyskúša viac využiť

- **Maroš**

- k metodike sa dostal trochu neskôr
- Stihol všetky vytiahnuté tasky
- nevedel základy androidu a z tohto dôvodu mu spracovanie tasku trvalo dlhšie, lebo sa musel priúčať základom
- jeho oblúbená farba je modrá

- **Jano**

- je so svojou prácou spokojný
- mimo svojich taskov sa mu nepáčilo že sa snažil dotláčať do práce ostatných členov, čím bral prácu Máriusovi
- stretol mnoho komplikácií so súbežnou pracou s arduinom a príslušnými shieldami

- **Jakub**

- metodiky mohli byť hotové skôr, robil ich na poslednú chvíľu ale spravil jednu navyše
- ostatné tasky spravil rýchlo a skoro, do ďalšieho šprintu si tiež plánuje vybrať náročnejšie resp. viac taskov
- do budúcnu by ocenil robiť testy, aby nebolo potrebné sa venovať nepodstatným veciam

- **Márius**

- spravil všetko tak ako si naplánoval
- mal rôznorodé tasky a je spokojný s ich plnením na čas
- zaťažený sa cíti byť v rámci normy, tak akurát

- **Dominik**

- chce zapracovať na priebežnom dokumentovaní svojej práce
- Spokojný, stihol všetky tasky

- testovanie majáčika preukázalo potrebu nahradenia silnejšími súčiastkami

- **Martin**

- spravil všetky svoje tasky úspešne
- zlepšil sa v nerobení taskov ALAP
- so sprintom je celkom spokojný
- do ďalšieho sprintu chce zvýšiť svoju produktivitu

- **Michal**

- neboli prítomný

7. Čo bolo dobré

- Stihli sme všetky tasky
- Dokončili sme úspešne dokumentáciu k druhému kontrolnému bodu
- Získali sme grant
- Úspešne sme spojazdnili zobrazovanie polohy na našej onepage stránke

8. Čo bolo zlé

- Návrat k riešeniu úloh ALAP
- Slabá interakcia členov pri riešení taskov
- Zlyhanie, resp. vynechanie daily scrumu z časových príčin

4.5 Zápis zo stretnutí

V tejto kapitole uvádzame zápis z našich týždenných stretnutí, monitorujúce ich priebeh.

4.5.1 1. stretnutie – zoznamovacie stretnutie

Čas: 27.9.2016 14:30

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Findura

- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Stav plnenia úloh

Úloha	Zodpovedná osoba	Stav
Motivácia, výber projektu	Celý tím	Hotovo
Názov tímu	Ján Pánis	Hotovo
PLagát Tímu	Ján Pánis	Hotovo

Priebeh a výsledky stretnutia

Na tomto stretnutí sme sa zoznámili s našim vedúcim Ing. Michalom Valíčkom. Oboznámili sme sa z predchádzajúcimi výsledkami letu do stratosféry našej fakulty a požiadavkami nášho servisného modulu pre potenciálne nasledujúce lety. Rozobrali sme si túto problematiku a ujasnili si naše predstavy čo nás čaká v tomto projekte.

Pokračovali sme zvolením vedúceho nášho tímu (Ján Pánis) a rozdelením prvých úloh ktoré nás čakajú najbližší týždeň, ktoré sa predovšetkým zameriavajú na spracovanie analýzy danej problematiky.

Úlohy do ďalšieho stretnutia

Úloha	Zodpovedná osoba	Termín
Termíny, tímový kalendár	Ján Pánis	4.10. 2016
Analýza a výber používaneho nástroja	Márius Rak	4.10. 2016
Preskúmanie vypúšťania balónov(najmä v Anglicku, príp. NASA) Ako a hlavne prečo, resp. Načo?! ...výstup: kus beletrie	Maroš Frkáň, Jakub Findura	4.10. 2016
Zápis, zformovanie dokumentácie	Dominik Pisarovič	4.10. 2016
Web	Márius Rak	4.10. 2016
Analýza GPS senzor, teplotný senzor, radiácia, tlak	Ján Pánis, Dominik Pisarovič	4.10. 2016
Analýza platforemi(spotreba, výkon,...), vývojových prostredí(dohodnuté AVR)	Tomáš Urban	4.10. 2016
Analýza preposielania údajov,Analýza vypúšťania(senzor,ventil,...)	Martin Oravský	4.10. 2016

4.5.2 2. stretnutie – úvodné stretnutie

Čas: 3.10.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Findura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

Na tomto stretnutí máme v pláne prerokovať následné body:

- Výsledky analýzy
- Plnenie úloh
- Meranie veličínn naším servisným modulom
- Naplánovanie prerozdelenie nových úloh
- Naplánovanie User stories

Stav plnenia úloh

Úloha	Zodpovedná osoba	Stav
Termíny, tímový kalendár	Ján Pánis	OK
Analýza a výber používaného nástroja	Márius Rak	OK
Preskúmanie vypúšťania balónov(najmä v Anglicku, príp. NASA)	Maroš Frkáň, Jakub Findura	OK
Zápis, zformovanie dokumentácie	Dominik Pisarovič	OK

Web	Márius Rak	In progress
Analýza GPS senzor, teplotný senzor, radiácia, tlak	Ján Pánis, Dominik Pisarovič	In progress
Analýza platforiem(spotreba, výkon,...), vývojových prostredí(dohodnuté AVR)	Tomáš Urban	OK
Analýza preposielania údajov ,Analýza vypúšťania(senzor,ventil,...)	Martin Oravský	In progress

Priebeh stretnutia

Na druhom stretnutí sme sa stretli znova v plnom počte. V úvode sme si prešli splnené úlohy a náš celkový progres. Odprezentovali sme vedúcemu nás zvolený nástroj na delenie úloh, Trello, a jeho funkcie. Ďalej sme sa venovali výsledkom analýzy, kde sme prehodnotili naše ďalšie smerovanie, najmä špecifikáciu nášho zamerania na meranie rôznych veličín ako teplota, tlak, radiácia, vlhkosť a iné.

V druhej časti stretnutia sme sa venovali brainstormingu. Vytvorili sme si štyri user story, ktorým sa chceme venovať počas nášho projektu a následne sme popísali nové úlohy a ich prerozdelenie, čím sme ukončili naše stretnutie a úspešne naplnili jeho plán.

Nové úlohy

Pozvánka michal@valicek.sk - >trello,slack, google drive	Jakub Findura, Márius Rak
Iný teplotný senzor, pozrieť sa na DHT22, Viac senzorov, priemerovanie, vyššia presnosť	Ján Pánis
GPS modul	Ján Pánis
GSM modul	Tomáš Urban
Senzor vlhkosti	Dominik Pisarovič
Rádio vysielače	Martin Oravský, Márius Rak
Odkazy, prístupy, linky na jedno miesto	Dominik Pisarovič

Brainstorming – dekompozícia user stories (Google Doc)	Dominik Pisarovič
Facebook Api	Jakub Findura
Preštudovať dôkladne funkcia funkciu Trella	Márius Rak
Preštudovať staré tímaky	Maroš Frkáň
Správa webu	Márius Rak, Martin Oravský
Spracovať výsledky analýzy do jedného dokumentu	Dominik Pisarovič

4.5.3 3. stretnutie - začiatok 1. šprintu

Čas: 10.10.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Findura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Plnenie úloh, nové úlohy
- Prediskutovanie brainstormingu
- Príprava 1. šprintu

Stav plnenia úloh

Úloha	Zodpovedná osoba	Stav
Analýza GPS senzor, teplotný senzor, radiácia, tlak	Ján Pánis, Dominik Pisarovič	OK
Analýza preposielania údajov ,Analýza vypúšťania(senzor,ventil,...)	Martin Oravský	OK
Pozvánka michal@valicek.sk ->trello,slack, google drive	Jakub Findura, Mária Rak	OK
Iný teplotný senzor, pozrieť sa na DHT22, Viac senzorov, priemerovanie, vyššia presnosť	Ján Pánis	OK
GPS modul	Ján Pánis	OK
GSM modul	Tomáš Urban	OK
Senzor vlhkosti	Dominik Pisarovič	OK
Rádio vysielače	Martin Oravský, Mária Rak	OK
Odkazy, prístupy, linky na jedno miesto	Dominik Pisarovič	OK
Brainstorming – dekompozícia user stories (Google Doc)	Dominik Pisarovič	OK
Facebook Api	Jakub Findura	In progress
Preštudovať dôkladne funkcionality Trella	Mária Rak	In progress
Preštudovať staré tímaky	Maroš Frkáň	OK
Správa webu	Mária Rak, Martin Oravský	OK

Spracovať výsledky analýzy do jedného dokumentu	Dominik Pisarovič	In progress
---	-------------------	-------------

Priebeh stretnutia

Na treťom stretnutí sme sa venovali podľa plánu splneným úlohám, brainstormingu, príprave prvého šprintu a novým úlohám. Prerokovali sme vybratý a zakúpený GPS modul, GSM modul a výsledky analýzy preposielania údajov. Pri výbere teplotného senzora sme sa dostali k problému, vzhladom na veľkú plánovanú výšku a nízky rozsah teplotných senzorov aktuálne na trhu. Prerokovali sme výber rádio vysielačov, pokračovanie brainstormingu na dekompozíciu user stories. Ďalej sme sa venovali veľkú pozornosť brainstormingu o návrhu našej Facebookovej stránky, ktorú využijeme na propagáciu nášho tímového projektu počas letu balóna.

Prerozdelili sme si úlohy, medzi ktorými sme uznali ako najdôležitejšiu realizáciu návrhu architektúry, z ktorej sa ďalej odrazíme do ďalších dôležitých úloh. Zamysleli sme sa nad hlavnou funkcionalitou nášho prototypu. Do prvého releasu sme vybrali pre nás prototyp meranie teploty, akcelerácie, stránku s vizualizáciou polohy nášho modulu, prípadne podľa možností doplnené aj o posielanie sms správ a funkčného GPS modulu.

Nové úlohy

Ondrej zawiň - vypustal balony, na "svetelektron" popisuje svoje projekty vypustania, teraz pracuje v sose - treba preskumat	Dominik Pisarovič
Prepojenie Github Trello Slack	Márius Rak
Zohnať doménu	Martin Oravský
Navrhnuť architektúru, prototyp, onepage, spracovanie správ, archivovanie, stránka na vizualizáciu polohy	Maroš Frkáň, Jakub Findura
Získavanie informácií z teplotného senzora, preposielanie	Ján Pánis

Tvorenie brainstormingu	Dominik Pisarovič, Martin Oravský
Dokumentácia	Dominik Pisarovič
Preliezť fóra o vypúšťaní	Tomáš Urban
Aktualizácia fotiek	Martin Oravský
Beletria	Dominik Pisarovič
Granty na ministerstve (ESA)	Martin Oravský
Legislatíva, povolenia (výška, veľkosť)	Martin Oravský

4.5.4 4. stretnutie – priebeh 2. šprintu

Čas: 17.10.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Findura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Revízia plnenia úloh a rozdelenie nových
- Diskusia o výbere kamery využitej na našom servisnom module
- Granty
- Účasť na TP Cupe

Stav plnenia úloh

Úloha	Zodpovedná osoba	Stav
Facebook Api	Jakub Findura	OK
Preštudovať dôkladne funkciu Trella	Márius Rak	OK
Spracovať výsledky analýzy do jedného dokumentu	Dominik Pisarovič	OK
Ondrej zawiň - vypustal balony, na "svetelektron" popisuje svoje projekty vypustania, teraz pracuje v sose - treba preskumat	Dominik Pisarovič	OK
Prepojenie Github Trello Slack	Márius Rak	OK
Zohnat' doménu	Martin Oravský	OK
Navrhnuť architektúru, prototyp, onepage, spracovanie správ, archivovanie, stránka na vizualizáciu polohy	Maroš Frkáň, Jakub Findura	OK
Získavanie informácií z teplotného senzora, preposielanie	Ján Pánis	OK
Tvorenie brainstormingu	Dominik Pisarovič, Martin Oravský	In progress
Dokumentácia	Dominik Pisarovič	In progress
Preliezť fóra o vypúšťaní	Tomáš Urban	OK
Aktualizácia fotiek	Martin Oravský	OK
Beletria	Dominik Pisarovič	In progress
Granty na ministerstve (ESA)	Martin Oravský	OK
Legislatíva, povolenia (výška, veľkosť)	Martin Oravský	Neúspešne

Priebeh stretnutia

V tomto stretnutí sme podľa plánu prebrali tri podstatné témy a rozobrali si ďalšie napredovanie. Na začiatok sme si zrevidovali plnenie úloh a potom sme pokračovali grantami. Žiadosti o granty realizuje násť vedúci s termínom do 10.11. Rozsiahla časť našej diskusie sa venovala aj výberu kamery, ktorá bude osadená k servisnému modulu. Zjednotili sme sa na verzii využitia GoPro kamery.

Ďalším predmetom nášho stretnutia bolo doriešenie potrebnosti stretnutia sa s ľuďmi so SOSY. Toto stretnutie je dôležité pre získanie viac informácií a tiež získanie lepšieho pohľadu do tejto problematiky. Jedným z dôležitých bodov tohto stretnutia bude diskusia o využití rádiového modulu a rádiového prenosu.

Na záver sme prediskutovali našu účasť na TP Cupe. Jednohlasne sme sa rozhodli zúčastniť sa tejto súťaže a umiestniť sa na čo najlepšej pozícii.

Nové úlohy

Stretnutie so SOSA	Ján Pánis
Trello export nech to vyzerá jak má	Márius Rak
Prihlaska na tp cup	Dominik Pisarovič
Scrum nastudovať	Márius Rak
Flask python	Martin Oravský, Jakub Findura
Architektúra firmwéru	Ján Pánis
Analýza UML-RT	Maroš Frkáň
Premysliť rádio, pred pýtaním zo SOSY	Tomáš Urban

4.5.5 5. stretnutie - koniec 1. šprintu, začiatok 2. šprintu

Čas: 24.10.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak

- Bc. Jakub Findura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Retrospektíva z prvého šprintu
- Revízia úloh z prvého šprintu
- Reštruktualizácia nášho SCRUMu
- Návrh User Stories do šprintu č. 2

Stav plnenia úloh

Úloha	Zodpovedná osoba	Stav
Tvorenie brainstormingu	Dominik Pisarovič, Martin Oravský	OK
Dokumentácia, beletria	Dominik Pisarovič	In progress
Stretnutie so SOSA	Ján Pánis	In progress
Trello export nech to vyzerá jak má	Márius Rak	In progress
Prihlaska na tp cup	Dominik Pisarovič	OK
Scrum nastudovať	Márius Rak	OK
Flask python	Martin Oravský, Jakub Findura	OK
Architektúra firmwéru	Ján Pánis	OK
Analýza UML-RT	Maroš Frkáň	In progress

Premyslieť rádio, pred pýtaním zo SOSY	Tomáš Urban	OK
---	-------------	----

Priebeh stretnutia

Na piatom stretnutí sme sa znova zišli v plnom počte. Stanovili sme si plán stretnutia spomenutý vyššie a následne sme presne podľa neho postupovali.

Ako prvý bod sme rozobrali úspešnosť prvého šprintu, našu spokojnosť a retrospektívne sme si rozobrali jeho priebeh. V ďalšom bode sme si zrevidovali plnenie úloh a našu efektivitu. V tejto časti sa potvrdilo naše podozrenie, že náš spôsob scrumovania nie je celkom v poriadku. Veľký čas sme teda venovali reštrukturalizácii fungovania nášho tímu a rozobrali si ďalší postup.

Ako úvod do ďalšieho šprintu sme si predefinovali nové User stories a prerobili štruktúru delenia Epics, User stories, jednotlivých taskov a celkovo celý product backlog.

Nové Epici:

- Zber údajov
- Dohľadanie balóna
- Prístup k dátam
- Propagácia
- Všeobecné a podporné úlohy
- TP Cup
- Dokumentácie

4.5.6 6. stretnutie – priebeh 2. šprintu

Čas: 27.10.2016 19:00-21:00

Miesto: Nová sála UPC

Zúčastnení členovia

- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Fiňdura

- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Doplnenie User stories
- Rozdelenie na tasky
- Ohodnotenie user stories
- Vybranie user stories do šprintu č.2
- Rozdelenie úloh

Priebeh stretnutia

Stretnutie sme realizovali v náhradnom termíne kvôli sviatkom. Zišli sme sa v plnom počte okrem vedúceho projektu. Ako prvé sme doplnili User stories podľa dokončeného elektronického brainstormingu. Následne sme si zahrali SCRUM poker a ohodnotili naše user stories, v závislosti od referenčnej úlohy „Vypracovanie a odovzdanie prihlášky na TP Cup“

Následne sme podľa spoločnej diskusie povytáhovali požadované user stories do šprintu č.2 a každý sme si vytiahli úlohy, ktorým sa budeme venovať v danom šprinte.

Vybraté User stories do šprintu č. 2

• Chcem doručiť dátá z modulu pomocou GSM na zem	SP 13
• Chcem propagovať balón na sociálnych sietiach	SP 5
• Chcem zobrazovať polohu balónu a zaujímavé informácie na webe	SP 8
• Chceme odovzdať dokumentácie k šprintu 2	SP 5
• Chceme sa naučiť fungovať ako tím	SP 5
• Chceme pripraviť dokumentáciu/metodiku k doterajším šprintom	SP 5
• Continuous integration	SP 8
• Pripraviť Git pre časti projektu	SP 2
• Meranie času	SP 3
• Chcem merať teplotu vo vnútri servisného modulu	SP 5
• Chceme podať prihlášku na TP Cup	SP 1

4.5.7 7. stretnutie – koniec 2. šprintu, začiatok 3. šprintu

Čas: 8.11.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Findura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Šprint review
- Retrospektíva
- Doplnenie user stories do backlogu
- Príprava tretieho šprintu

Priebeh stretnutia

Na úvod stretnutia sme si prešli sprint review a retrospektívnu druhého šprintu. Zhodnotili sme nás prístup, splnené úlohy, získané story pointy. Zamysleli sme sa nad efektivitou nášho empirického prístupu k metóde scrumu a dohodli sme sa na realizácii daily scrumu v sobotu o 16tej hodine. Do backlogu sme po žiadúcej debate doplnili nové user stories a následne sme si z nich povytáhovali user stories do 3. šprintu, rozdelili si ich na tasky a podelili.

Vytiahnuté user stories do 3. šprintu:

- | | |
|---|------|
| • Chceme odovzdať dokumentáciu k stretnutiam v 2. šprinte | 5 SP |
| • Chceme upraviť obsah na webe | 2 SP |
| • Chceme vedieť či môžeme posielat' foto zo stratosféry | 5 SP |
| • Chceme propagovať balón na sociálnych sieťach | 2 SP |
| • Chceme prehľadné exporty z trela | 8 SP |

• Prehodnotiť štruktúru dokumentácie na webe	1 SP
• Chceme prijať odoslanú SMS z GSM modulu	8 SP
• Chceme vedieť aktuálne GPS súradnice	8 SP
• Chceme spolupracovať cez MS Office	1 SP
• Posielanie dát na server	13 SP

4.5.8 8. stretnutie – priebeh 3. šprintu

Čas: 15.11.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Findura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Stav plnenia úloh
- Daily scrum
- Odovzdávanie dokumentácie k prvému kontrolnému bodu

Stav dokončenia user stories

User story	Zodpovedná osoba	Stav
PÚ7: Vykreslovanie pekných exportov z Trella (SP8)	Márius Rak	In progress
PÚ9: Spolupráca cez MS Office (SP1)	Dominik Pisarovič	Done
PG2: Propagácia balónu na sociálnych sieťach(SP2)	Maroš Frkáň	In progress
PD4: Príjmanie odoslanej SMS z GSM modulu(SP8)	Tomáš Urban	In progress

DB3: Chceme vidieť aktuálne GPS súradnice(SP8)	Ján Pánis	Done
PD3: Posielanie dát na server(SP13)	Jakub Findura	In progress
PG5: Chceme vedieť či môžeme posielat' fotky zo stratosféry(SP5)	Martin Oravský	Done
PÚ11: Chceme upraviť obsah na webe(SP2)	Dominik Pisarovič	To do
PÚ10: Chceme odovzdať dokumentácie k druhému šprintu(SP5)	Dominik Pisarovič	In progress
PÚ8: Prehodnotiť štruktúru dokumentácie na webe(SP1)	Martin Oravský	In progress

Priebeh stretnutia

Stretnutie prebiehalo podľa plánu, najskôr sme si prešli plnenie úloh, pridali úlohy Martinovi, ktorý už bol so svojimi hotový a následne sme si prešli daily scrumom. Tiež sme podstatnú časť stretnutia venovali dokončeniu dokumentácie k prvému kontrolnému bodu a jej odovzdaniu.

Spísali sme si postrehy na ktoré nemôžme zabudnúť:

do metodiky riadenia:

- k jenkinsu - metodika našej implementácie - každý člen má k dispozícii nejaký nástroj na CI a ten využívame
- Martin má pridelené napísať 2-3 strany dokumentácie k jenkinsu
- doplniť metodiku k pridelovaniu úloh
- doplniť metodiku písania dokumentácie
- CI - git, jenkins - Martin - 2-4 strany
- do soboty aktualizovať web - nahrať všetky dokumenty zápisnice atď.

4.5.9 9. stretnutie – koniec 3. šprintu, začiatok 4. šprintu

Čas: 22.11.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak

- Bc. Jakub Findura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Šprint review
- Retrospektíva
- Doplnenie user stories do backlogu
- Príprava štvrtého šprintu

Stav plnenia úloh

User story	Zodpovedná osoba	Stav
PÚ7: Vykresľovanie pekných exportov z Trella (SP8)	Márius Rak	In progress
PÚ9: Spolupráca cez MS Office (SP1)	Dominik Pisarovič	Done
PG2: Propagácia balónu na sociálnych sietiach(SP2)	Maroš Frkáň	Done
PD4: Príjmanie odoslanej SMS z GSM modulu(SP8)	Tomáš Urban	In progress
DB3: Chceme vidieť aktuálne GPS súradnice(SP8)	Ján Pánis	Done
PD3: Posielanie dát na server(SP13)	Jakub Findura	Done
PG5: Chceme vedieť či môžeme posielat' fotky zo stratosféry(SP5)	Martin Oravský	Done
PÚ11: Chceme upraviť obsah na webe(SP2)	Dominik Pisarovič	Done
PÚ10: Chceme odovzdať dokumentácie k druhému šprintu(SP5)	Dominik Pisarovič	Done
PÚ8: Prehodnotiť štruktúru dokumentácie na webe(SP1)	Martin Oravský	Done

Priebeh stretnutia

Na stretnutí sme si postupne prešli sprint review s retrospektívou. S výsledkom šprintu sme boli celkovo spokojný aj napriek nižšej velocity(42), čo bolo ale spôsobené odovzdávaním dokumentácie k prvému kontrolnému bodu. Prešli sme si splnené úlohy a s výnimkou dvoch user stories sa nám podarilo všetko úspešne dokončiť. Spolu s produkt ownerom sme si prešli obsah product backlogu a doplnili potrebné user stories, ktoré sme si počas šprintu uvedomili a označili za potrebné. Ďalej sme postupovali ohodnocovaním jednotlivých user stories a ich tahaním do štvrtého šprintu. Po úspešnej príprave štvrtého šprintu sme uzavreli výnimomačne rýchle ale efektívne stretnutie.

4.5.10 10. stretnutie – priebeh 4. šprintu

Čas: 29.11.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Fiňdura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Stav plnenia úloh
- Daily scrum

Stav plnenia user stories

User story	Zodpovedná osoba	Stav
Fixnutie CI	Martin Oravský	Done
Chceme prijať odoslanú SMS z GSM	Maroš Frkáň, Tomáš Urban	In progress
Ukladanie zálohy na pamäťovku	Ján Pánis	In progress

Zobrazovanie poslaných dát na onepage stránke	Jakub Findura	In progress
Vykresľovanie exportov z Trella	Márius Rak	To Do
Chceme pripraviť dokumentáciu/metodiku k sprintom	Márius Rak	In progress
Analýza UML-RT	Martin Oravský	To Do
Chceme majáčik na jemné dohľadanie balónu	Dominik Pisarovič	In progress
Spísanie metodík	Márius Rak	In progress

Priebeh stretnutia

Stretnutie prebiehalo podľa plánu, najskôr sme si prešli plnenie úloh a následne sme si prešli daily scrumom. Po prvom týždni sa nám podarilo jedno User story už uzavrieť a zvyšné rozpracovať, pričom mnohé z nich sa už blížia k dokončeniu.

- Jano
 - mal spraviť metodiku, ale nezačal ešte
 - údaje už priebežne nejaké ukladá na pamäťovku ale analýze sa zatiaľ nevenuje a taktiež žiadne dokumentovanie svojej práce zatiaľ nespísal
 - postreh – treba pri testovaní spoľahlivosti skúsiť vytiahnut' pamäťovku a strčiť ju naspäť
- Márius
 - pustil sa do podporných úloh
 - väčšina je v štádiu 90% hotovo
- Jakub
 - začal robiť task so zobrazením na mape a z časti to už aj funguje
- Dominik
 - Začal analyzovať aktuálne riešenie v oblasti využitia majáčika na drobné dohľadanie
 - prešiel aktuálne riešenia v rádioamatérskom orientačnom behu
- Martin
 - spravil CI, pokračovať ide ďalej na analýze UML-RT
- Tomáš, Maroš
 - zohnali telefón, skúšali vytvorenú apkú
 - Potrebujú prediskutovať či ked' budeme prijímať smsky či to musí ísť cez notebook alebo to stačí poslať rovno na server, zhodli sme sa že stačí rovno na server
 - už to majú aj celkom hotové

4.5.11 11. stretnutie – priebeh 4. šprintu

Čas: 6.12.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Findura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Stav plnenia úloh
- Daily scrum

Stav dokončenia user stories

User story	Zodpovedná osoba	Stav
Fixnutie CI	Martin Oravský	Done
Chceme prijať odoslanú SMS z GSM	Maroš Frkáň, Tomáš Urban	Done
Ukladanie zálohy na pamäťovku	Ján Pánis	Done
Zobrazovanie poslaných dát na onepage stránke	Jakub Findura	Done
Vykresľovanie exportov z Trella	Márius Rak	In progress
Chceme pripraviť dokumentáciu/metodiku k sprintom	Márius Rak	In progress
Analýza UML-RT	Martin Oravský	In progress
Chceme majáčik na jemné dohľadanie balónu	Dominik Pisarovič	In progress
Spisanie metodík	Márius Rak	In progress

Priebeh stretnutia

Stretnutie prebiehalo podľa plánu, najskôr sme si prešli plnenie úloh a následne sme si prešli daily scrumom. Po druhom týždni sa nám podarilo už viac User stories uzavrieť a zvyšné rozpracovať, pričom mnohé z nich sa už blížia k dokončeniu.

- Jakub
 - bavil sa so serverom, ukladanie a vykresľovanie dát už funguje
 - nedostatok dokumentácie
 - dokumentovanie kódu in progress
- Maroš
 - aplikácia je plne funkčná
- Márius
 - vykresľovanie úloh z trella už funguje do pekných tabuľiek
- Ján
 - ja som spravil ukladanie dát na kartu, funguje to s tým že zdokumentované to veľmi nemá, zatiaľ len po 4 body ku každému
 - metodiku má viac menej spracovanú
- Dominik
 - spravil som metodiku
 - zakúpil vysielač/príjmač, prepojenie s arduinom je in progress
- Tomáš
 - gsm je hotové a metodiku tiež dokončil
- Martin
 - metodiku spravil a poslal
 - Analýza UML-RT sa značne skomplikovala náročnosťou dostupných 300 stranových dokumentov

4.5.12 12. stretnutie – koniec 4. šprintu

Čas: 13.12.2016 13:00-15:00

Miesto: Funtoro laboratórium FIIT

Zúčastnení členovia

- Ing. Michal Valíček
- Bc. Tomáš Urban
- Bc. Martin Oravský
- Bc. Márius Rak
- Bc. Jakub Findura
- Bc. Maroš Frkáň
- Bc. Ján Pánis
- Bc. Dominik Pisarovič, zapisovateľ

Plán stretnutia

- Šprint review
- Retrospektíva

Stav dokončenia user stories

User story	Zodpovedná osoba	Stav
Fixnutie CI	Martin Oravský	Done
Chceme prijať odoslanú SMS z GSM	Maroš Frkáň, Tomáš Urban	Done
Ukladanie zálohy na pamäťovku	Ján Pánis	Done
Zobrazovanie poslaných dát na onepage stránke	Jakub Findura	Done
Vykresľovanie exportov z Trella	Márius Rak	Done
Chceme pripraviť dokumentáciu/metodiku k šprintom	Márius Rak	Done
Analýza UML-RT	Martin Oravský	Done
Chceme majáčik na jemné dohľadanie balónu	Dominik Pisarovič	Done
Spísanie metodík	Márius Rak	Done

Priebeh stretnutia

Na stretnutí sme si postupne prešli sprint review s retrospektívou. S výsledkom šprintu sme boli celkovo spokojný, pertože sa nám podarilo dokončiť všetky user story v rámci tohto šprintu, čo bolo ale spôsobené dlhším šprintom. Prešli sme si splnené úlohy, zhodnotili šprint, zamysleli sa nad zmenou do budúcnia. Pripravili sme sa na prezentáciu riadenia v rámci predmetu Manažment v softvérovom inžinierstve a tiež sme si rekapitulovali zvyšné úlohy potrebné na finalizáciu dokumentu a odovzdanie práce k druhému kontrolnému bodu.

5 Globálna retrospektíva

Prvý šprint sa niesol v takom tréningovom duchu, vzhľadom k prvému pokusu o implementáciu scrumu do riešenia nášho projektu. V snahe naučiť sa efektívne využívať

scrum sme urobili mnoho chýb, avšak s dôrazom na empirický prístup k tejto práci sme sa v mnohom poučili a posunuli tak efektivitu našej práce o množstvo krokov dopredu.

V závere šprintu číslo jedna sme sa začali zamýšľať nad našou interpretáciou a samotným využívaním scrumu v našom projekte. Šprint ako taký bol v celku úspešný, dokončili sme veľké množstvo úloh, rozchodili sme základné prvky hardvéru potrebného pre nás projekt a taktiež aj analýza nabrala celkom slušný vzhl'ad.

Avšak scrum ako taký u nás celkom nefungoval. Zle sme si definovali user stories a následne aj jednotlivé tasky, robili sme veci do radu a nebrali do úvahy hodnotenie user stories.

Druhý šprint z pohľadu využívania scrumu bol omnoho lepší. Tentoraz sa niesol v duchu preúčaní sa a zžívaní sa so scrumom a jeho upravovaním a aplikovaním na nás projekt. Podarilo sa nám splniť všetky úlohy a tak stihnuť úspešne 50 story pointov dokončiť. Fungovanie v tíme a spolupráca sa tiež zlepšili podobne ako aj samotné scrumovanie.

Tretí šprint sme začali s veľkým nadšením zo požadovaného dostatočného chápania scrumu ako takého a jeho aplikácie na nás projekt. Vzhľadom k tomu že sme už mali vlastné know-how, stretnutia odsýpali šikovne a celkovo naša práca sa zlepšila. V čase prvého kontrolného bodu a 1. odovzdávania sme však boli v strede tretieho šprintu a tak jeho výsledky ešte nie sú definitívne.

Počas tohto zimného smestra sme spravili kus práce na našom projekte. Z hladiska analýzy sme pokryli takmer všetky oblasti potrebné na správne pochopenie problému a následne jeho vhodnú implementáciu. Nás prototyp taktiež nabral pekné rozmery a jeho funkcia sa rozšírila a priniesla pekné výsledky. Propagáciu sme už dotiahli tiež na veľmi peknú úroveň.

6 Preberacie protokoly

Odobzdávajúci subjekt: Tím č. 10 – StratosFIIT, Balooooooooon

Preberajúci subjekt: Ing. Michal Valiček

Predmet prebratia:

- Dokumentácia riadenia
- Dokumentácia inžinierskeho diela
- Prototypu

Poznámky: Prototyp poskytuje doposiaľ možnosť zasielania informácií o GPS polohe pomocou SMS správy a jej vykreslovanie na našej webpage spolu s ostatnými potrebnými informáciami.

Preberajúci subjekt podpisom potvrzuje prevzatie vyššie uvedených častí dokumentácie a prototypu

V Bratislave

.....
Dátum

.....
Podpis

7 Export úloh

Priebežné výsledky spracovaných úloh sme evidovali pomocou nástroja Trello. Tento nástroj poskytoval možnosť exportu spracovaných úloh, ktorý následne Márius mierne upravil pre lepší a prehľadnejší vzhľad. Všetky exporty našich úloh sú zverejnené na tímovej stránke <http://balooooooon.tk/#documentation> v sekcii „Exporty z Trella“.

8 Metodiky

8.1. Metodika dokumentácie

Metodika dokumentácie je jednou z najpodstatnejších metodík využívaných na tímových projektoch. Vzhľadom na spoluprácu veľkého množstva ľudí na jednom dokumente, je náročné udržať konzistentnosť.

8.1.1. Dedikácia metodiky

Táto metodika sa zameriava na manažment dokumentovania celého životného cyklu nášho tímového projektu. Zaoberá sa manažmentom dokumentovania práce na projekte(analýzy, návrhu architektúry, implementácie,...), spisovania zápisníc zo stretnutí ako aj spisovania jednotlivých retrospektív z ukončených sprintov. Primárnym cieľom tejto metodiky je dodržanie konzistentnosti, uniformity, vecnosti a pravosti projektovej dokumentácie nášho tímového projektu.

8.1.2. Formátovanie textu

Základným prvkom konzistentnosti textu ako takého je jednotné formátovanie textu v rámci celej dokumentácie. Titulná strana podľa vzoru uloženom na Google Drive. Štruktúru nadpisov využívame túto:

1. Nadpis prvej úrovne

– Times New Roman, 20pt, Bold, vždy na novej strane

1.1. Nadpis druhej úrovne

– Times New Roman, 16pt, Bold,

1.1.1. Nadpis tretej úrovne

- Times New Roman, 12pt, Bold,

Celkový text formátujeme:

- Times New Roman, 12pt
- Zarovnanie podľa okrajov
- Riadkovanie 1,5
- Strany číslované v pravom dolnom rohu
- Popisy pod obrázkami, nad tabuľkami

8.1.3. Metodika dokumentovania práce na projekte

Práca na projekte (analýza, návrh architektúry, implementácia,...) sa dokumentuje priebežne. Každý člen počas svojej práce priebežne zapisuje každú svoju činnosť a jej výsledky podľa vzoru formátovania. Spísané ucelené celky následne vloží do share-ovaného MS Wordu 2016 do príslušnej časti(analýza, návrh architektúry, implementácia,...). Po vložení preverí dodržanie konzistentnosti textu manažér dokumentácie, v prípade potreby upraví formátovanie alebo posunie prispomienky autorovi.

8.1.4. Metodika písania zápisníc stretnutí

Pri spisovaní zápisníc zo stretnutí sa zapisovateľ drží vytvorennej šablóny uloženej na Google Drive. Pravidlá formátovania sú identické ako pri spisovaní dokumentovania práce. Po ukončení zápisu, revízie a uzavretí zápisnice ju manažér dokumentácie prekontroluje z hľadiska konzistentnosti a zverejní na tímovom webe.

8.1.5. Metodika spisovania retrospektív

Pri spisovaní retrospektív z jednotlivých šprintov sa zapisovateľ drží vytvorennej šablóny uloženej na Google Drive. Pravidlá formátovania sú identické ako pri spisovaní dokumentovania práce. Po ukončení zápisu retrospektívy, jej revízie a uzavretí spisovania retrospektívy ju manažér dokumentácie prekontroluje z hľadiska konzistentnosti a zverejní na tímovom webe.

8.2. Metodika konfigurácie softvérového systému

Metodika určuje presne vymedzené postupy a praktiky súvisiace s konfiguráciou softvérového systému.

8.2.1 Dedičnosť metodiky

Metodika je určená všetkým členom tímu podieľajúcim sa na konfigurácii niektorého zo softvérových systémov. Je potrebné všetky postupy uvedené v nej dodržiavať presne a bezpodmienečne.

8.2.2 Súvisiace metodiky

- Metodika testovania
- Metodika verzií

8.2.3 Roly

V procesoch konfigurácie softvérového systému vystupujú nasledovné roly:

1. Vývojár - Úlohou vývojára je konfigurovať softvérový systém po dokončení vývoja istej fázy projektu.
2. Správca konfiguračného nástroja - Správca konfiguračného nástroja je zodpovedný za korektné nastavenie nástrojov kontinuálnej integrácie.

8.2.4 Procesy

8.2.4.1 Konfigurácia nástroja na kontinuálnu integráciu

V tomto procese vystupuje správca konfiguračného nástroja a vývojár.

- Vstupy – požiadavka na konfiguráciu kontinuálnej integrácie pre projekt
- Výstupy – nakonfigurovaná kontinuálna integrácia
- Popis:

V tomto procese vývojár oznámi potrebu konfigurácie nástroja na kontinuálnu integráciu pre jeho aktuálne implementovanú časť projektu. Správca následne nakonfiguruje nástroj podľa jeho požiadaviek. Pokyny, ktorými je potrebné sa riadiť pri konfigurácii kontinuálnej integrácie, sú nasledovné:

- Prihlásenie sa do systému jenkins na adresu baloooooon.tk:8080/jenkins svojimi prihlásovacími údajmi
- Kliknutie na New Item
- Vyplníme názov projektu a zvolíme možnosť Freestyle project, potvrďme kliknutím na OK

- V časti configure nakonfigurujeme nasledovné veci:
 - zaškrtnutie Github project, vyplnenie URL k Github repozitáru
 - zaškrtnutie Custom Workspace v časti General /Advanced/ a vyplnenie cesty k projektu na tímovom serveri.
 - zaškrtnutie Git v časti Source Code Management a vyplnenie Repository URL
 - zaškrtnutie „Build when a change is pushed to GitHub“ v časti Build triggers
 - konfiguráciu uložíme a potvrdíme

8.2.4.2 Konfigurácia softvérového systému

V tomto procese vystupuje vývojár.

- Vstupy – implementovaná časť projektu
- Výstupy – výstup z procesu buildovania projektu
- Popis:

V tomto procese vývojár zadá do systému požiadavku na buildovanie projektu. Nástroj na kontinuálnu integráciu túto požiadavku akceptuje a spúšťa build projektu. V prípade úspešného buildu je vývojár o tom informovaný. V tomto procese platí pravidlo, aby sa nahrané zmeny z lokálneho repozitára automaticky stiahli do lokálneho repozitára na serveri. Toto pravidlo je umožnené dosiahnuť pomocou konfigurácie nástroja spolu so službou GitHub podľa procesu Konfigurácia softvérového systému. Build je vykonávaný približne 1x do týždňa, čiže počas jednoho šprintu sú vykonávané približne 2 buildy softvérového systému a spravidla je vykonávaný bezprostredne pred scrum meetingom. Podrobnejšie pokyny pre vykonanie konfigurácie projektu sú:

- Prihlásenie sa do systému jenkins na adresu baloooooon.tk:8080/jenkins svojimi prihlásovacími údajmi
- kliknutie na zvolený projekt v časti All na hlavnej stránke
- kliknutie na „Build Now“ v ľavej časti stránky

V prípade úspešného dokončenia buildu je proces ukončený, v prípade neúspešného buildu kontrolujeme Console Output daného buildu a fixujeme

8.3. Metodika riadenia

Štruktúrovaný opis princípov

- Metodika práce v tíme je postavená na metodike SCRUM, popísané sú zmenené/upravené časti metodiky
- Metodika SCRUM je upravená vzhľadom na špecifická, požiadavky a obmedzenia vyplývajúce z organizácie predmetu a samotného projektu
- Sprint trvá 2 týždne, začína a končí každý druhý utorok, ak to okolnosti dovoľujú
- Výstup práce v šprinte nie je hotový a nasaditeľný produkt. Namiesto toho ide o pokrok v prototype, alebo neskôr finálnom produkte, pri podporných úlohách môže ísť o finálny produkt.
- Nekoná sa daily scrum. Namiesto neho sa raz za maximálne 3 dni tím stretne na spoločnej internetovej chatovacej službe a prezentuje prácu od posledného stretnutia a po najbližšie plánované stretnutie
- Každý task má vlastný definition of done. Obsahuje špecifikáciu, čo bolo/bude skontrolované a čo je možné s produkтом/prototypom robiť po dokončení user story
- Scrum master je tiež členom developer team
- Tím si zakladá na empirizme a metóde pokus - omyl za účelom rozvoja tímu ale aj jednotlivých členov a ich skúsenosti so SCRUMom

Vysvetlenie princípov

Cieľová metodika sa zakladá na princípoch metodiky SCRUM. Zrejme ale bude potrebné metodiku SCRUM upraviť pre špecifické potreby, ktoré vznikajú z formy a podstaty samotného produktu, ako aj prostredia a podmienok tímu, teda fungovanie v škole, zriedkavejšie stretnutia a skutočnosť, že členovia nemajú skúsenosti so scrumom ani s niektorými oblastami vývoja vyvíjaného produktu.

V zásade sa metodika práce v tíme 10 riadi metodikou SCRUM podľa príručky. Časti metodiky, ktoré sú ovplyvnené podmienkami práce alebo charakteristikou projektu a teda sa líšia od metodiky SCRUM sú popísané nižšie.

Kvôli povahе vyvíjaného produktu je vhodnejší vodopádový prístup. Z toho dôvodu neplatí, že výstupom každého šprintu je hotová iterácia, nasaditeľná do produkcie. Je ale snahou tímu, aby každá iterácia predstavovala nejaký ucelený krok ku kompletnému finálnemu produktu. V ideálnom prípade, je výstupom šprintu aspoň prototyp, ktorý je obohatený o nový funkčný hardvér.

Vzhľadom na fakt, že pracovné kapacity tímu sú obmedzené a množstvo práce je primerané pre plný počet členov tímu, nie je možné, aby bol jeden člen tímu iba scrum master. Z toho dôvodu je

jeden z členov scrum master a zároveň aj člen developer teamu. Tento člen plní na scrum eventoch rolu a úlohy scrum mastra a zároveň ich kombinuje s úlohami člena developer teamu. Metodika SCRUM nezakazuje takýto prístup. Tento člen ale má na starosti najmä podporné úlohy tímu, úlohy týkajúce sa behu tímu a priebehu práce. Vzhľadom na skutočnosť, že tento člen, rovnaké ako ostatní nemá skúsenosť so SCRUMom, tiež sa musí vzdelávať v tejto oblasti, vykonať nejakú prácu pre tím aby naštudoval a pripravil SCRUMové podmienky pre tím a teda jeho prínos do tímu je na rovnakej úrovni ako iného člena developer teamu.

Šprint v tíme trvá 2 týždne najmä kvôli organizácii predmetu. Tieto 2 týždne sú primerané množstvo času na analýzu potrebných častí a oblastí vývoja na produkte, návrh implementácie HW, prípadne SW a následne aspoň na jednoduchú implementáciu do prototypu. V neskorších fázach vývoja to bude dostatočné množstvo času na vylepšenie implementácie, zabezpečenie robustnosti systému a prípadne aj prerobenie prototypovej implementácie do implementácie na finálnom produkte.

Tím je obmedzený skutočnosťou, že sa nestretáva každý deň. Metodika SCRUM je silne závislá na denných stretnutiach kedy sa má konať daily scrum každé ráno. Tento scrum event je preto nemožné dodržať podľa jeho pôvodného popisu avšak jeho prínos do fungovania tímu je nesporný. Preto je nutné zaviesť v tíme aktivitu, ktorá aspoň sčasti nahradí tento event a poskytne tímu takmer to isté. Za týmto účelom sa tím stretne pravidelne stretnutie na spoločnej internetovej komunikačnej službe vo vopred dohodnutý a pravidelný čas, kde každý člen upovedomí ostatných členov tímu o jeho postupe práce tak ako by to urobil na daily scrume avšak nezhrnie prácu za posledných a nasledujúcich 24 hodín ale za obdobie od posledného stretnutia po najbližšie plánované stretnutie. Toto stretnutie sa uskutoční raz za najviac 3 dni.

Ked'že cvičiaci na predmete, ktorý je zároveň product owner by mal prvom rade dohliadať na priebeh práce, scrum master ma za úlohu vytvárať, upravovať a dohliadať na product backlog. Product backlog vzniká na základe konzultácie a s celým tímom a cvičiacim, resp. product ownerom a developer teamom. Po dokončení spoločnej práci na product backlogu ho product owner iba skontroluje.

Vzhľadom na povahu produktu, je problém stanoviť jednu spoločnú definition of done. Preto tím pri vytváraní user story popíše túto definíciu pre daný user story a pripojí ju k nemu. Táto definícia nie je popisom tasku ale definuje, aké kontroly práce prebehli a čo je možné s výsledkom práce robiť. Napríklad definuje, že bolo skontrolované schematické zapojenie HW, implementácia do prototypu a po dokončení je možné vykonávať konkrétnu činnosť s prototypom.

Popísaný prístup a metodika sa zhodujú s hlavnou myšlienkovou SCRUM - empirizmom.

Tím 10 sa učí tejto metodike a každý získava skúsenosti, na základe ktorých, sa metodika práce vyvíja a obmieňa, preto aj v prípade, že niektorý z postupov alebo princípov nie je správny, je

dôležité, že je v tíme špecifikovaný a tím sa ho drží aby sa ukázalo či funguje alebo nie a do akej miery je kontraproduktívny, resp. škodlivý. Takýto spôsob posúva a rozvíja tím 10. [1]

8.4 Metodika archivácie verzií HW

8.4.1 Úvod

V tomto dokumente sa nachádza metodika archivácie verzií HW, ktorá je určená projektu zaoberajúcemu sa vypúšťaniu stratosferických balónov s názvom Stratos FIIT.

Hardvér je neoddeliteľnou súčasťou nášho projektu, bez ktorého by jeho realizácia nebola možná. Tak ako je bežné a prirodzené verziovať softvér pomocou metodiky na báze webových služieb ako je Github/Bitbucket, je toto nevyhnutné robíť aj pre verziovanie hardvéru. Tento proces je však diametrálnie odlišný a preto je opísaný v tomto dokumente s názvom Metodika archivácie verzií HW. V prípade akýchkoľvek nejasností ma kontaktujte: panis1994@gmail.com.

8.4.2 Fritzing

Fritzing je softvér s otvoreným zdrojovým kódom, ktorého hlavnou úlohou je zjednodušíť návrh elektronických zariadení. Keďže sa v ňom dajú jednoducho vytvárať návrhové schémy, je taktiež užitočný pri uchovávaní týchto schém a následnom zálohovaní/verziovaní. Program nevyžaduje inštaláciu a je možné ho prevziať s oficiálnej stránky <http://fritzing.org/download/>.

Obsahuje veľké množstvo modulov, mikrokontrolérov, mikropočítačov, súčiastok a iných komponentov využívajúcich sa na vytváranie HW. Pokial' sa potrebná súčiastka v programe nenachádza, je možné ju importovať do programu. Keďže ide o open-source program existuje veľká komunita ľudí, ktorá vytvára nové súčiastky a následne ich všetky zdieľajú na Githube. Osobne sa mi ešte nestalo, že by som nejakú súčiastku nenašiel. Avšak v prípade, že by sa tak stalo, spoločnosť Fritzing vytvorila jednoduchý návod pomocou, ktorého je možné zstrojiť akúkoľvek súčiastku (návod: <http://fritzing.org/learning/tutorials/creating-custom-parts/>).

8.4.3 Procesy verziovania

Pri verziovaní HW existuje viacero procesov, ktoré treba v správnom poradí vykonať a až po vykonaní posledného kroku, môžeme prehlásiť, že daná verzia/prototyp je správne zdokumentovaný a jeho spätá rekonštrukcia nebude zložitá.

Vychádzame z predpokladu, že úspešne prebehlo týchto 5 predchádzajúcich fáz:

1. Analýza požiadaviek
2. Analýza dostupných možností (správnosť, dostupnosť HW)
3. HW implementácia
4. SW implementácia
5. Testovanie prototypu

Následne prichádzajú na rad 2 základné procesy, ktorých gro je spísané v tejto metodike. Tieto procesy sú: 1. Návrh schémy zapojenia a 2. Archivácia schémy, zdrojových kódov a potrebných náležitostí.

8.4.3.1 Návrh schémy zapojenia

Ako som písal v kapitole 2, na zstrojenie využívame nástroj Fritzing, v ktorom využívame výhradne záložku „Schéma“ z dôvodu, že pomocou schémy je najjednoduchšia spätná rekonštrukcia daného produktu/verzie.

1. Pridanie potrebných komponentov na pracovnú plochu

Postup:

1. Na pravej strane programu vidíme modul „Súčiastky“, v ktorom sú všetky súčiastky kategorizované podľa výrobcu.
2. V prípade, že by ste súčiastku podľa kategórii výrobcov nenašli, môžete využiť možnosť prehľadávať medzi všetkými (piktogram lupa).
3. Do polička v kategórii „prehľadávať všetky“ zadajte názov súčiastky.
4. V prípade, že by ste ju nenašli postupujte podľa návodu v kapitole 8.4.2.
5. V prípade, že ste našli potrebnú súčiastku, presuňte ju na pracovnú plochu.
6. Proces opakujte dokým nepridáte všetky potrebné súčiastky.
7. Na záver všetky súčiastky vhodne pomenujte.

2. Prepojenie komponentov

Postup:

1. V prvom rade je potrebné zvoliť vhodné rozloženie a orientáciu komponentov vzhľadom na počty a vzdialenosťi všetkých prepojení.
2. Vytvárajte vhodné prepojenia podľa prvotnej schémy/podľa reálneho prototypu.
3. Prepojenie vytvoríte kliknutím na koncový bod súčiastky/uzla, podržaním ľavého tlačidla myši a následného prepojenia s druhým komponentom/uzlom.
4. Pri vytváraní týchto spojení dodržujte tieto pravidlá:
 - a. Prepojenia by mali byť štvorcového charakteru
 - b. Križovanie spojov neznamená ich fyzické prepojenie
 - c. V prípade kríženia spojov s uzlom sa neguje pravidlo b.
5. Proces opakujte kým nezostrojíte všetky prepojenia
6. Skontrolujte, či sú všetky prepojenia správne

8.4.3.2 Archivácia schémy a zdrojového kódu

Archivácia aktuálnej verzie obnáša archivovanie schémy i zdrojového kódu.

1. Vytvorenie schémy vhodnej na archiváciu

1. Archivácia schémy, ktorú možno neskôr upravovať:
 - a. Kliknúť na „Súbor“
 - b. Zvoliť „Uložit“
 - c. Ako názov súboru zadat „schéma_fri“
 - d. Ako formát vybrať „Fritzing (*.fzz)“
 - e. Kliknúť na tlačidlo „Uložit“
2. Archivácia schémy, ktorú možno zobraziť bez potrebného SW:
 - a. Kliknúť na „Súbor“
 - b. Zvoliť „Exportovať“
 - c. Zvoliť „ako obrázok“
 - d. Zvoliť „PNG...“
 - e. Ako názov súboru zadat „schéma_pic“
 - f. Kliknúť na tlačidlo „Uložit“

2. Vytvorenie súboru zdrojového kódu vhodného na archiváciu

Na prototypovanie softvéru/firmvéru využívame webovú službu Github, avšak ako záloha kódu k aktuálnemu HW je lepšie si daný kód uložiť aj do súboru s názvom „code.c“ (v prípade, že zdrojový kód je v jazyku C).

3. Archivácia predpripravených súborov na zdieľané úložisko

Pre uchovanie verzií HW používame zdieľané úložisko (Drive) od spoločnosti Google na adrese: https://drive.google.com/drive/folders/0B44hCCThq4F_QUxqYnJNRF NYSE0.

V adresári „Verziovanie HW“ sa nachádzajú 3 hlavné priečinky s názvami hlavných častí, ku ktorým sú pridelené zodpovedné osoby. Tieto priečinky sú:

1. Dominik Pisarovič – Maják
2. Ján Pánis – Servisný modul
3. Tomáš Urban – GSM

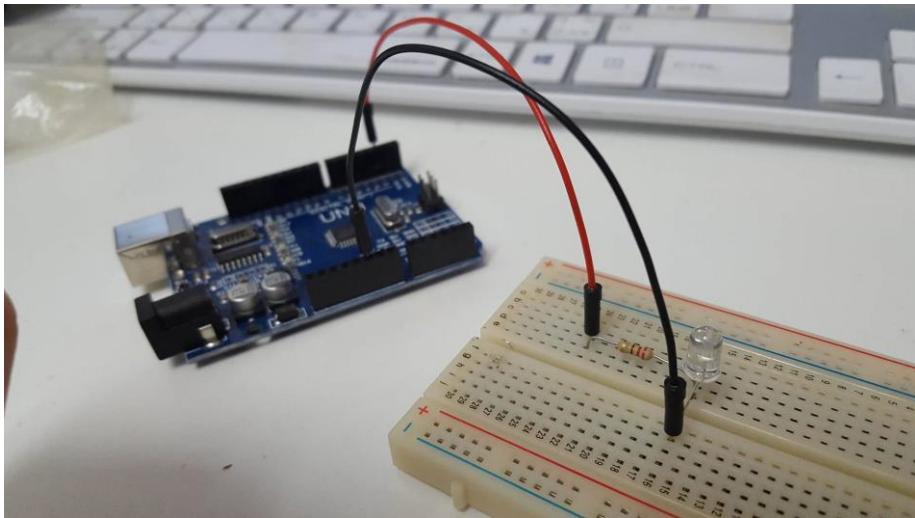
Pri archivácií postupujte nasledovne:

1. Zvoľte vhodný priečinok, do ktorého daná verzia patrí.
2. Vhodne pomenujte názov prototypu tak, aby vystihoval hlavné funkcionality pridanej hodnoty.
3. V prípade, že je nemožné jednou vetou názvu opísť pridanú funkcionalitu, pridajte do priečinka textový súbor, ktorý by opisoval všetky funkcie.
4. Do priečinka vložte nasledovné súbory:
 - a. „schéma_fri“
 - b. „schéma_pic“
 - c. „code.c“.
5. Verzia je archivovaná a v budúcnosti, ak bude treba bude možné ju zrekonštruovať.

8.4.4. Obrázkový návod na vytváranie schémy v programe Fritzing

Ako ukážkový príklad sme si zvolili návrh schémy založenej na mikrokontroléry Arduino UNO, červenej ledky a jedného rezistora.

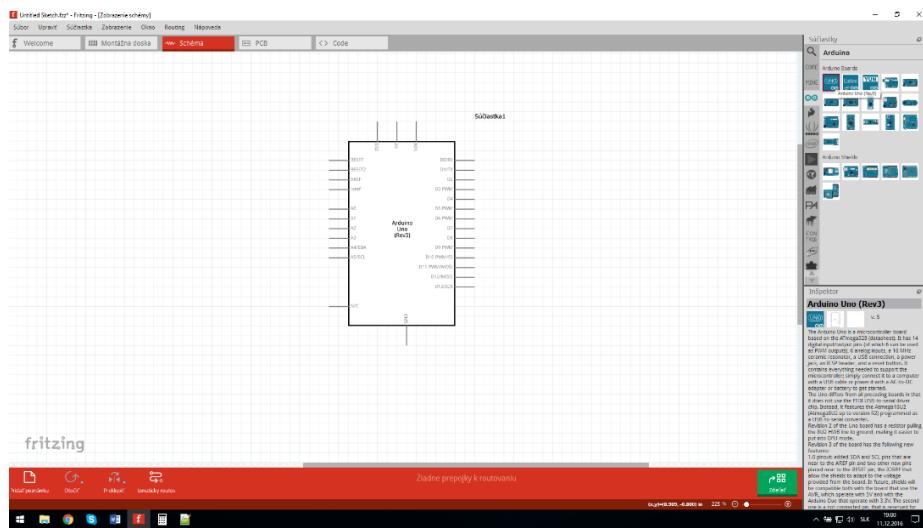
Obrázok aktuálnej verzie vyzerá nasledovne:



Obrázok 1 Testovacie zapojenie

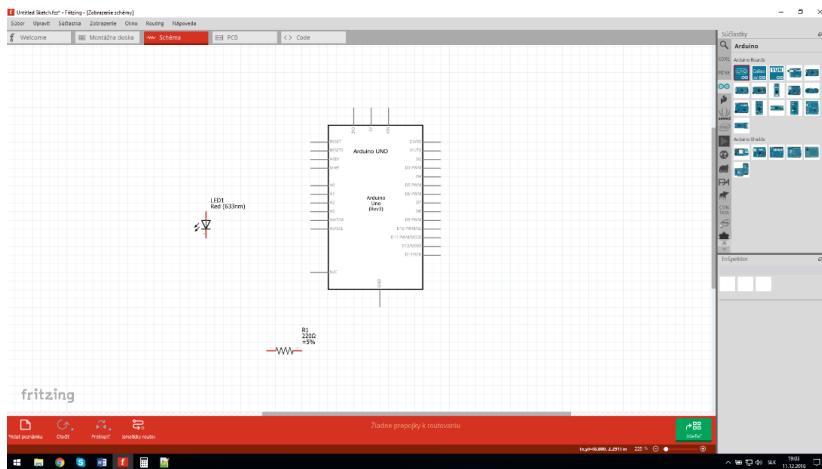
Postup:

1. Otvoríme si program Fritzing a vyberieme správny komponent



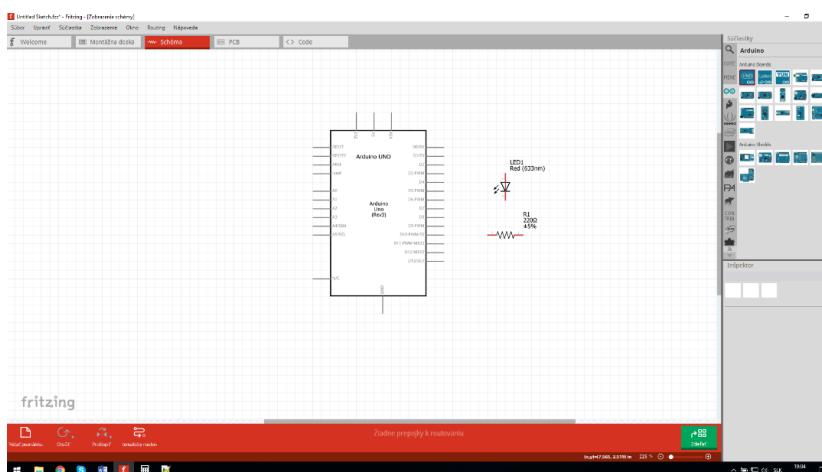
Obrázok 2 Screenshot z programu Fritzing

2. Proces opakujeme, kým na pracovnej ploche nemáme všetky potrebné komponenty



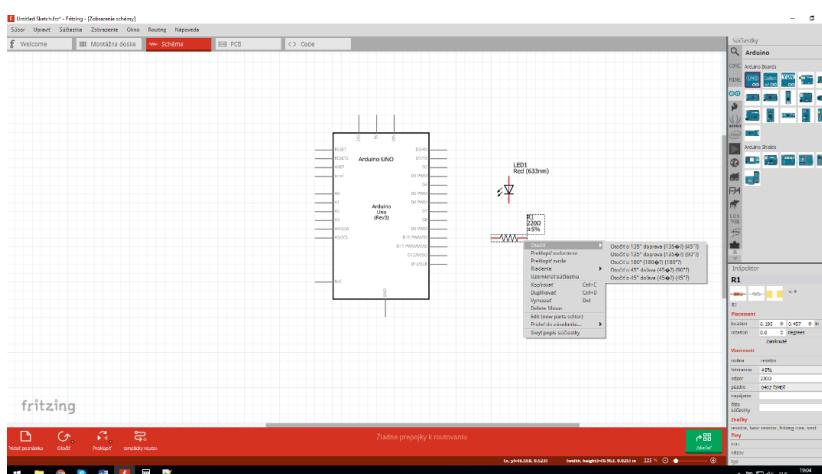
Obrázok 3 Screenshot z programu Fritzing

3. Zvolíme správne rozloženie komponentov



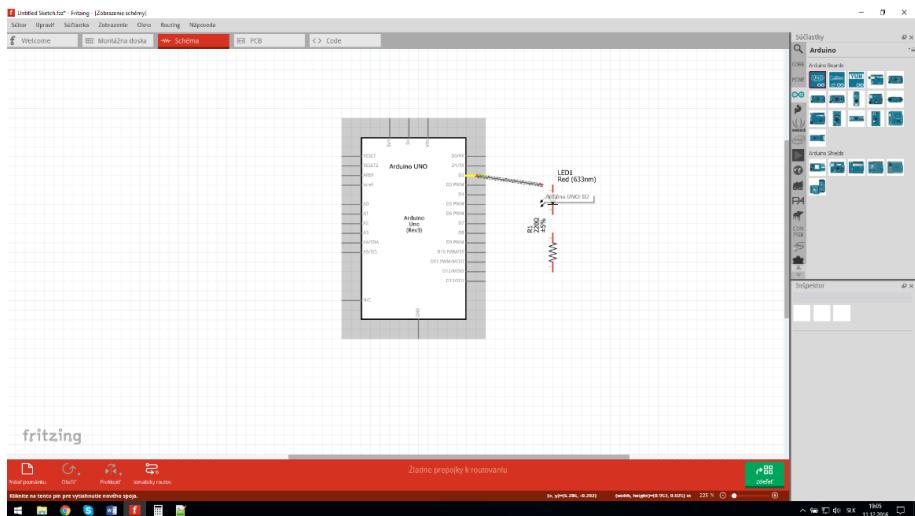
Obrázok 4 Screenshot z programu Fritzing

4. Zvolíme správnu orientáciu komponentov



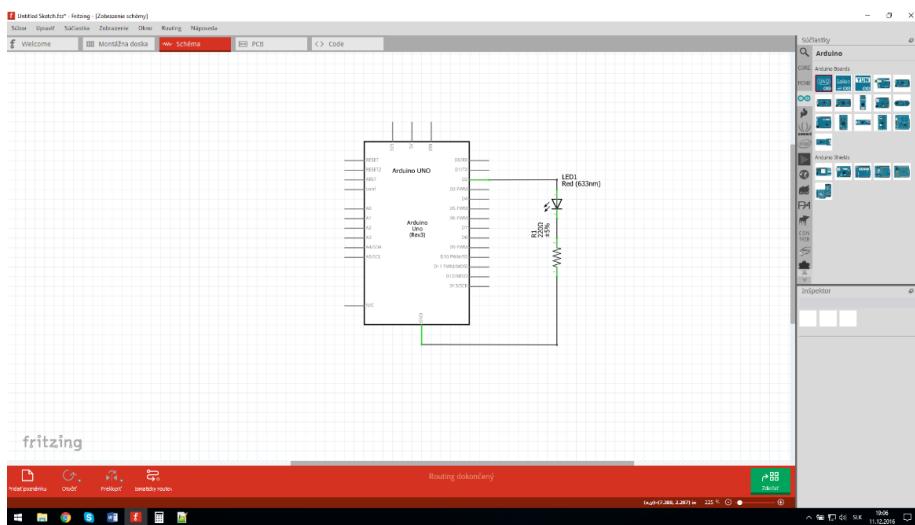
Obrázok 5 Screenshot z programu Fritzing

5. Začíname vytvárať prepojenia medzi komponentami



Obrázok 6 Screenshot z programu Fritzing

6. Pri správnom sa riadení všetkými pravidlami by mala výsledná schéma vyzeráť nasledovne:



Obrázok 7 Screenshot z programu Fritzing

8.5 Metodika verzií zdrojového kódu

Táto metodika je určená členom tímu, ktorí v rámci svojich úloh tvoria zdrojový kód niektornej zo súčasti systému. V rámci nášho projektu existuje viacero samostatných celkov systému, ktoré sú odlišné z pohľadu zdrojového kódu. Preto aj v rámci správy zdrojového kódu sú jednotlivé súčasti logicky oddelené.

Pre zdieľanie zdrojového kódu v tíme a uchovávanie jednotlivých verzií využívame systém git. Tento systém nám poskytuje služba GitHub. Pre každú samostatnú časť systému je vytvorený samostatný repozitár.

8.5.1 Roly

Správca repozitára – Člen tímu, ktorý spravuje daný repozitár. Zabezpečuje jeho vytvorenie, správu vetiev, pull requestov, kontroluje vykonávanie code review.

Vývojár – člen tímu, ktorý implementoval daný zdrojov kód. Spravuje svoj lokálny repozitár a odosiela vytvorený zdrojový kód na vzdialený repozitár.

Kontrolór kódu – člen tímu, ktorý vykonáva kontrolu kódu. Kontrolór v rámci kontroly kódu, kód nevytvára ani neopravuje.

8.5.2 Súvisiace metodiky

Metodika riadenia

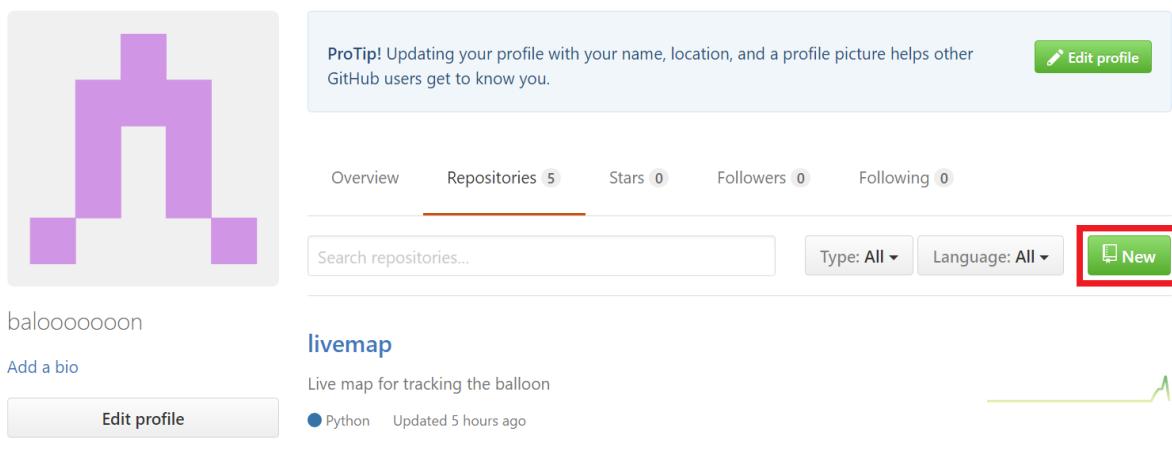
8.5.3 Zoznam pojmov a skratiek

Git	systém na správu a verziovanie zdrojového kódu
Branch	Vetva, možnosť robiť nezávislé zmeny zdrojového kódu.
Pull Request	Požiadavka na zlúčenie nového zdrojového kódu, ktorý musí prejsť kontrolou
Commit	Súbor zmien zdrojového kódu v repozitári
Merge	Spojenie dvoch vetiev
Code review	Prehliadka kódu, slúži na kontrolu zdrojového kódu, či je vhodný na pridanie do funkčnej vetvy.
Pull	Prevzatie zmien ostatných členov tímu zo vzdialého repozitára.
Push	Odoslanie lokálnych zmien na vzdialený repozitár.

8.5.4 Procesy

8.5.4.1 Vytvorenie repozitára

Nový repozitár je možné vytvoriť vo webovom rozhraní serveru GitHub. Názov repozitára by mal jedno(dvoj)slovne vystihovať danú logickú časť nášho systému. Pri vytváraní je vhodné ihneď vytvoriť súbor README a .gitignore, ktorý vynechá súbory, ktoré je nepotrebné alebo nežiaduce do repozitára ukladať.



Obrázok 8 GitHub.com - Vytvorenie nového repozitára

Create a new repository

A repository contains all the files for your project, including the revision history.

The screenshot shows the 'Create a new repository' form. It includes fields for 'Owner' (set to 'baloooooon'), 'Repository name' (set to 'new_repository'), a 'Description (optional)' field with a placeholder, and a toggle between 'Public' and 'Private' visibility (set to 'Public'). There's also a checked checkbox for 'Initialize this repository with a README' and dropdowns for 'Add .gitignore' (set to 'Python') and 'Add a license' (set to 'None'). A large green 'Create repository' button is at the bottom.

Owner: baloooooon

Repository name: new_repository

Description (optional):

Public: Anyone can see this repository. You choose who can commit.

Private: You choose who can see and commit to this repository.

Initialize this repository with a README

Add .gitignore: Python

Add a license: None

Create repository

Obrázok 9 GitHub.com - formulár pre vytvorenie repozitára

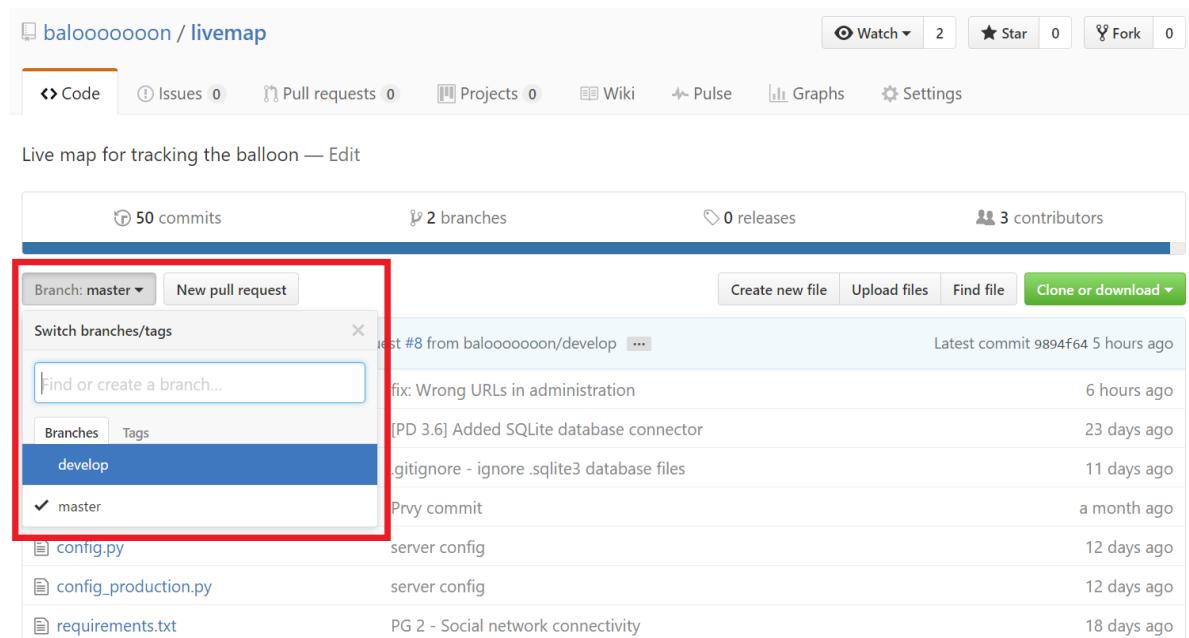
8.5.4.2 Vytvorenie vetvy

Každý repozitár obsahuje minimálne dve základné vetvy.

Master – Táto vetva obsahuje otestovaný a funkčný kód, ktorý môže byť nasadený do prototypu. Do tejto vetvy nie je možné priamo commitovať, je nutné použiť tzv. Pull Request. Pull request v tejto vetve musí prejsť cez Code Review aspoň jedným ďalším členom tímu.

Develop – V tejto vetve je zdrojový kód vo fáze vývoja. Zdrojový kód v tejto vetve by mal byť funkčný a spustiteľný. Autor commitu je zodpovedný za zdrojový kód a jeho otestovanie pred vykonaním commitu.

V rámci vzdialeného repozitára nie je nutné vytvárať novú vetvu pre každú úlohu/feature, ktorú člen tímu implementuje. Pokiaľ na menšej úlohe pracuje člen tímu sám, môže tieto vetvy používať v lokálnom repozitári.

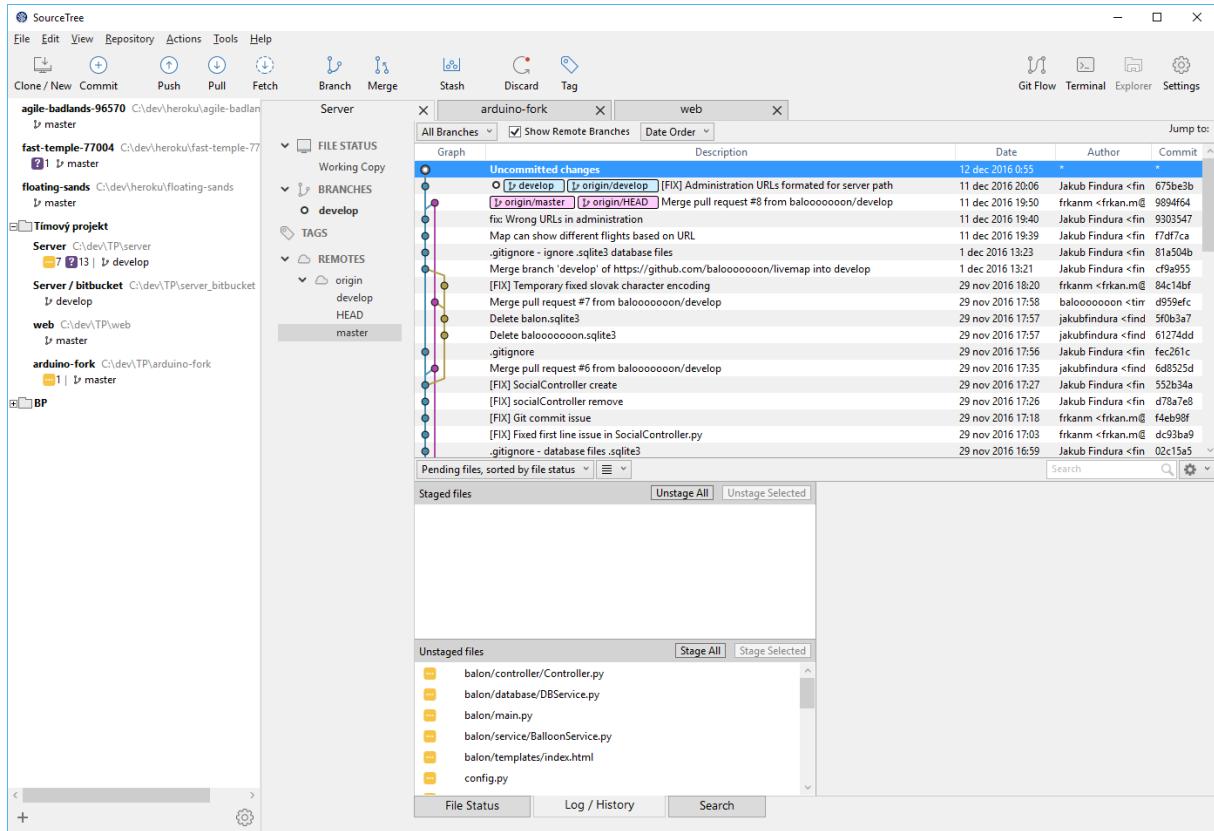


Obrázok 10 GitHub.com - vytvorenie novej vetvy

8.5.4.3 Vytvorenie lokálneho repozitára

Pre správu verzií na lokálom počítači je nutné mať nainštalovaný klientský program pre správu git. Odporúčaný je program SourceTree. Tiež je možné použiť git priamo v konzole.

Git init



Obrázok 11 SourceTree - Program pre správu git

8.5.4.4 Prepojenie vzdialeného repozitára

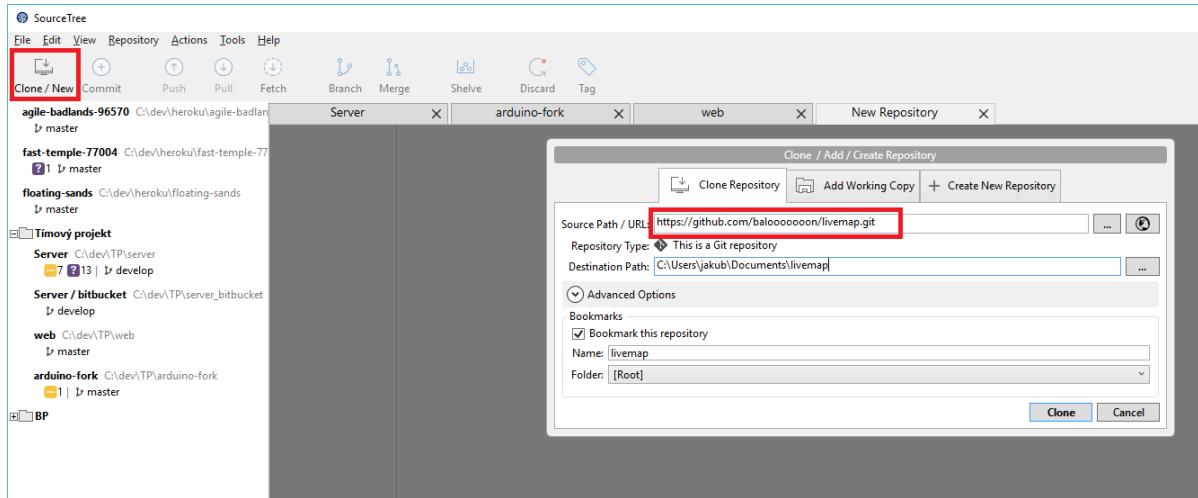
Aby bolo možné zdrojový kód zdieľať so vzdialeným repozitárom, je nutné vytvoriť lokálny repozitár a pridať odkaz na vzdialený repozitár.

```
git remote add [name] [git_url]
git fetch [name]
git checkout develop
```

The screenshot shows a GitHub repository page for 'baloooooon/livemap'. At the top, there are statistics: 50 commits, 2 branches, 0 releases, and 3 contributors. Below that, there are buttons for 'Create new file', 'Upload files', 'Find file', and a prominent green 'Clone or download' button. A modal window titled 'Clone with HTTPS' is open, displaying the URL 'https://github.com/baloooooon/livemap.git' and options to 'Open in Desktop' or 'Download ZIP'. The main part of the page lists repository files and their last commit times.

File	Last Commit
balon	fix: Wrong URLs in administration
docs	[PD 3.6] Added SQLite database connector
.gitignore	.gitignore - ignore .sqlite3 database files
balon.wsgi	Prvy commit
config.py	server config
config_production.py	server config
requirements.txt	PG 2 - Social network connectivity

Obrázok 12 GitHub.com - odkaz na vzdialený repozitár



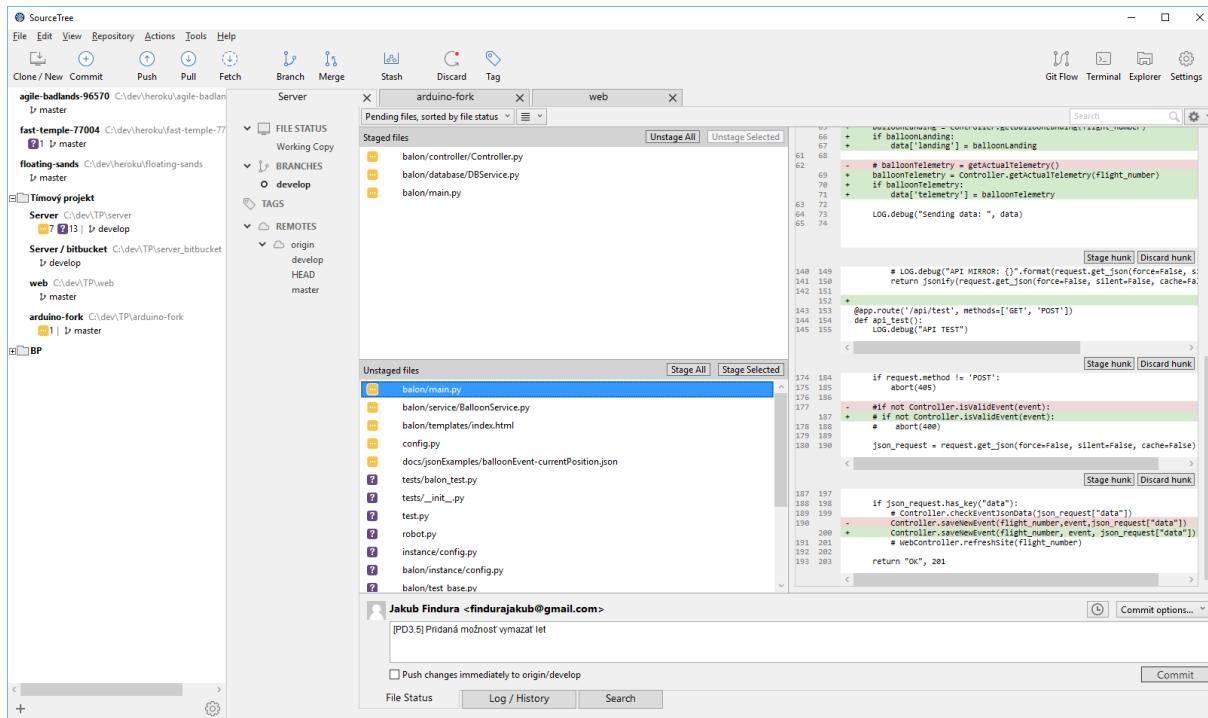
Obrázok 13 SourceTree - Pripojenie vzdialeného repozitára

8.5.4.5 Vytvorenie commitu

Prvým krokom je označenie zmenených súborov alebo častí súborov, ktoré majú byť súčasťou commitu. Každý commit obsahuje správu, ktorá popisuje, čo je predmetom daného commitu.

Git add .

Git commit -m „message“



Obrázok 14 SourceTree - Rozhranie pre vytvorenie nového commitu

8.5.4.6 Synchronizácia so vzdialeným repozitárom

Vytvorený commit sa nachádza len v lokálnom repozitári. Pre odoslanie všetkých lokálnych zmien je potrebné vykonať príkaz *push*. Pred vykonaním príkazu *push* je nutné vykonať príkaz *pull*, ktorý stiahne všetky vykonané zmeny vo vzdialom repozitári ostatných členov tímu. Pokiaľ by nastal konflikt v prípade, že dva členovia urobili zmeny v rovnakej časti kódu, je nutné ho vyriešiť. Vhodná je externá aplikácia DiffMerge od SourceGear.

Git pull

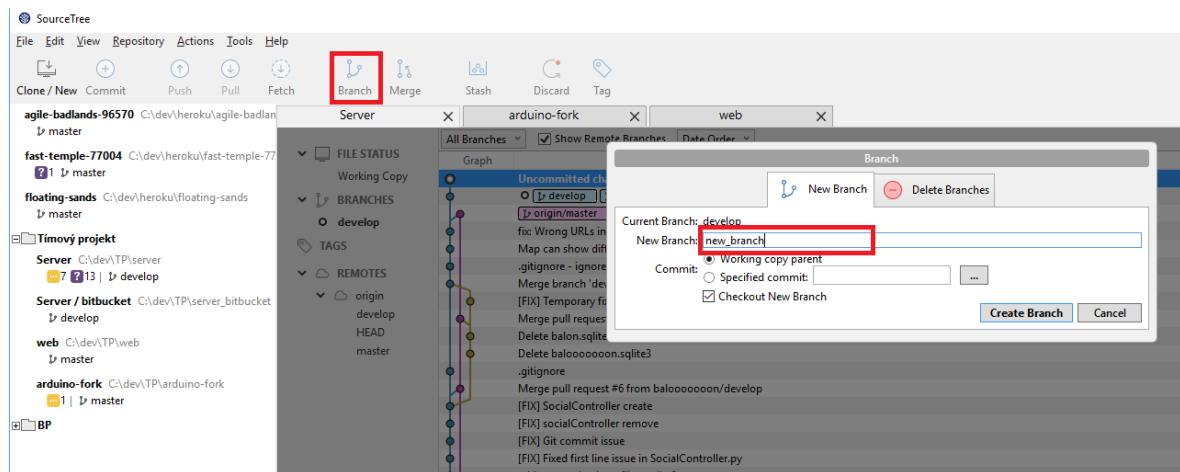
Git push

8.5.4.7 Vytvorenie novej vetvy

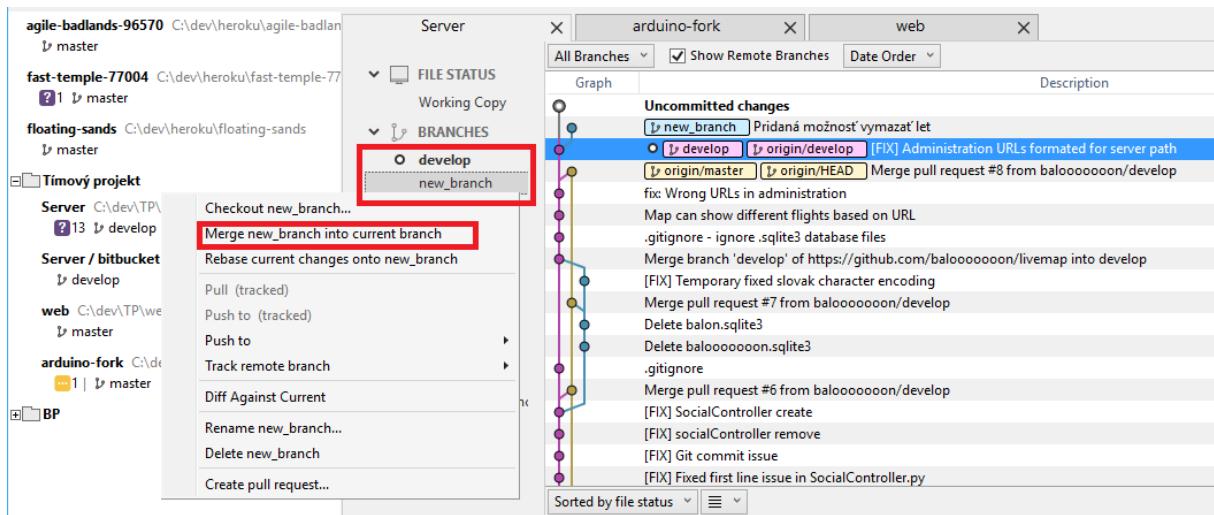
Pokiaľ je vyvíjaná nová funkcia, ktorá výrazne zasahuje do už existujúceho kódu, prípadne je predpoklad, že nakoniec bude zahodená, je vhodné vytvoriť novú vetvu. Pre správu vетiev existujú dva hlavné príkazy

Branch – vytvorenie novej vetvy

Merge – spojenie dvoch vетiev



Obrázok 15 SourceTree - vytvorenie novej vetvy



Obrázok 16 SourceTree - spojenie vetiev (merge)

8.5.5 Pravidlá písania

8.5.5.1 Názvy vetiev

Názvy vetiev by mali jasne vystihovať, čo je ich primárnym zameraním. Názov vetvy by sa mal skladať z prefixu a jedného, prípadne dvoch vystihujúcich slov.

Prefixy:

feature_ pre samostatné časti, ktoré sa neviažu na konkrétnu priradené úlohu.
pd-1-6_ pre funkcie, ktoré sa týkajú konkrétnej úlohy pridelenej členovi tímu alebo celého user story. Prefix vychádza zo značenia úloh, ktoré je opísané v Metodike riadenia.

Príklad:

feature_database
db-1_lokalizacia
pg-3-6_mapa

8.5.5.2 Commit message

Správa pri commite by mala krátkym textom vystihovať zmeny, ktoré boli v commitie vykonané. Správa môže obsahovať prefix, ktorý obsahuje typ commitu (napríklad fix) alebo číslo úlohy, ak je viazaný priamo na úlohu.

Príklad:

[FIX] Opravené cesty k statickým súborom
[PD3.6] Zobrazovanie trasy letu na mape

8.6 Metodika rizík

Manažment rizík je systematický proces, v ktorom sa riziko identifikuje, analyzuje a definuje optimálny spôsob jeho zvládnutia pri minimálnych nákladových aspektoch a rešpektovaní systémových cieľov subjektu. Úlohou rizikového manažmentu je predovšetkým dosiahnutie maximálnej bezpečnosti a ochrany majetku vypracovaním optimálnej stratégie riadenia rizík ako hlavných nositeľov možných budúcich škôd.

Táto metodika je určená pre členov študentský tímový projekt, ktorého cieľom je vyvinúť servisný modul najmä pre dohliadanie stratosférického balónu po úspešnom experimentálnom vypustení do atmosféry.

8.6.1 Rozdelenie rizík

Rozdelenie rizík v tíme

- funkcionálne – riziká týkajúce sa funkcií a služieb servisného modulu
- manažérske – riziká týkajúce sa tímového riadenia a jednotlivcov v tíme

8.6.2 Ciele metodiky rizík

Cieľom tejto metodiky je identifikovať čo najviac rizík. Samozrejme, povaha nášho projektu nám neumožňuje identifikáciu všetkých rizík nakoľko po vypustení balónu nemáme kontrolu nad balónom a nedokážeme priebežne dodať funkčnosť modulu umiestneného v balóne.

Preto sa snažíme využiť viaceru možných spôsobov ako dohliadať náš modul v prípade, že jeden spôsob zlyhá.

8.6.3 Postup identifikácie rizík

1. identifikovať, charakterizovať riziká
2. posúdiť dopad rizík na kritické časti modulu alebo fungovanie tímu pri obsluhe a vývoji servisného modulu
3. stanovenie rizika (očakávané pravdepodobnosti a následky konkrétnych typov rizík)
4. identifikovať spôsoby, ako zmierniť dopad rizík
5. uprednostniť opatrenia na zníženie rizík podľa vopred pripravených stratégíí

8.6.4 Identifikované rizika a ich dopad

Tabuľka 1 Tabuľka rizík identifikovaných v projekte

Názov	Pravdepodobnosť	Miera Dopadu (1min, 10max)	Risk Response	Dopad	Ošetrenie
Vypadnutie člena tímu	0,9	8	Reduce	Členovia tímu si musia rozdeliť povinnosti a zodpovednosť vypadnutého člena tímu.	Pri vyvádzaní projektu sa členovia tímu musia oboznámiť s prácou a programami

					ostatných členov tímu
Absencia člena tímu na dobu max. 1 sprintu	0,4	2	Accept	Spomalenie vývoja hardvérovej alebo softvérovej časti projektu.	Absencia závažne neovplyvní vývoj projektu nakoľko sa snažíme robiť tasky v časovom predstihu.
Výpadok rádiovej komunikácie medzi modulom a notebook prijímačom	0,3	9	Reduce	V prípade zlyhania komunikácie nemôžeme poskytovať reálne informácie o lete na soc. sieťach alebo live mape.	Servisný modul bude obsahovať okrem rádiového spojenia aj telefónny modul služiaci na odosielanie SMS s aktuálnymi hodnotami balóna
Vysoká odchýlka GPS súradníc, ktoré sa zapisujú do live mapy na webe.	0,9	5	Reduce	Live mapa bude vykreslovať nepresnú trasu, ktorá nemusí byť reálna.	Pred odoslaním údajov na server musí prebehnúť validácia údajov vzhľadom na predošlé odoslané hodnoty.

V prípade, že identifikujeme ďalšie riziká je nutné doplniť identifikované riziká do tohto dokumentu. Pravdepodobnosti, ošetrenia a zvyšné atribúty rizika je vhodné prebrať na tímovom stretnutí s celým tímom, prípadne ak sa nejedná o závažné riziko sa môže riziko prediskutovať na tímových komunikačných kanáloch ako napr. Slack.

8.7 Metodika chýb

8.7.1 Úvod

Metodika určuje presne vymedzené postupy a praktiky súvisiace s manažmentom chýb pri riešení projektu StratosFIIT.

8.7.2 Dedičnosť metodiky

Metodika je určená všetkým členom tímu podieľajúcim sa na niektorom z krokov, ktoré zahŕňa manažment chýb pri práci na projekte StratosFIIT. Je potrebné všetky postupy uvedené v metodike dodržiavať presne.

8.7.3 Zoznam nadvážujúcich metodík a dokumentov

S metodikou manažmentu chýb súvisia aj tieto metodiky:

- Metodika plánovania
- Metodika pre priradovanie úloh
- Metodika testovania

8.7.4 Roly

V procesoch, ktoré obsahuje metodika manažmentu chýb, vystupujú nasledujúci účastníci produkčných tímov: **ohlasovateľ chyby, programátor, tester, projektový manažér**.

Predstaviteľia týchto rolí majú nasledujúce úlohy:

8.7.4.1 Ohlasovateľ chyby:

- S chybou sa stretne ako prvý a identifikuje ju.
- Rovnakú chybu vyhľadá v informačnom systéme, ak sa tam chyba nenachádza, pridá opisanú chybu.

8.7.4.2 Programátor:

- Zodpovedný za implementačnú časť opravenia chyby.

8.7.4.3 Tester:

- Zodpovedný za otestovanie opravenej chyby.
- Uzatvára chybu, ak je úspešne opravená.

8.7.5 Manažérske procesy

Manažérske procesy pri manažmente chýb sú späť s jednotlivými stavmi chyby v rámci životného cyklu chyby.

8.7.5.1 Proces č.1: Nahlásenie chyby

Proces nahlásenia chyby opisuje postup účastníka procesu, ktorý sa stretáva s chybou, identifikuje ju, vytvorí opis chyby a nahlási chybu do určeného miesta, konkrétnie sekcia Chyby v IS Trello.

Vstupný stav: -
Výstupný stav: Nová
Účastníci procesu: Ohlasovateľ chyby

Kroky procesu:

1. Ohlasovateľ chyby identifikuje chybu.
2. Opis chyby vyhľadá v sekcií Trella na to určenej.
 - a. V prípade, že opis chyby v Trele nenájde, chybu nahlási do IS – vytvorí opis chyby v stĺpci pre nové chyby.
 - b. V prípade, že sa rovnaká chyba už nachádzala v IS, ohlasovateľ chyby môže doplniť nové, resp. chýbajúce informácie do opisu chyby.
3. Uvedie čo najpresnejší opis kde, kedy a za akých okolností sa chyba vyskytla a čo spôsobila, resp. čo znemožnila.

8.7.5.2 Proces č.2: Priradenie chyby členovi tímu

Tento proces opisuje postup priradenia chyby na riešenie ďalším účastníkom v rámci manažmentu chýb.

Vstupný stav: Nová
Výstupný stav: Opravovaná
Účastníci procesu: Programátor, Projektový manažér

Kroky procesu:

1. Projektový manažér pre chybu so stavom Nová určí osobu zodpovednú za opravenie chyby. Rovnako určí aj osobu zodpovednú za otestovanie tejto chyby, zvolené osoby okrem priradenia danej chyby členom tímu, tieto osoby uvedie aj do popisu chyby.
2. Chybu priradí určenej osobe a presunie ju do stĺpca **Opravované**.

8.7.5.3 Proces č.4: Opravenie chyby

Tento proces opisuje postup účastníka procesu – programátora, ktorý implementuje opravenie chyby.

Vstupný stav: Opravovaná
Výstupný stav: Opravená .
Účastníci procesu: Programátor

Kroky procesu:

1. Na základe popisu chyby sa snaží docieliť jej opravenie.
2. Po úspešnom opravení chyby do popisu chyby doplní stručný popis v čom spočíva vykonaná oprava.
3. Zmení stav chyby na **Opravená**.

8.7.5.4 Proces č.5: Testovanie

Proces testovanie opisuje postup testera pri testovaní chyby, opravenej programátorom.

Vstupný stav: **Opravená**

Výstupný stav: **Uzavretá**

Účastníci procesu: **Tester**

Kroky procesu:

1. Vytvorí test a otestuje opravenie chyby – podrobnejší postup vid'. Metodika testovania
 - a. Ak chyba pretrváva, vráti chybu na ďalšie opravenie programátorovi – presunie do stĺpca **Opravované**.
 - b. Ak chyba nepretrváva, zmení stav chyby na Uzavretá a chybu presunie do stĺpca **uzavreté**.

8.8 Metodika testovania

8.8.1 Úvod

Táto metodika je určená členom tímu, ktorí v rámci svojich úloh tvoria zdrojový kód niektoré zo súčasťí systému alebo sú zodpovední za vykonávanie testov. V rámci tímu existuje niekoľko úrovní testovania: jednotkové testy, systémové testy, akceptačné testy. Väčšina testov súvisiaca s hardvérovými súčasťami systému sú nefunkcionálne testy.

8.8.1.1 Roly

Tester – Člen tímu, ktorý je zodpovedný za vykonávanie testov. Je zodpovedný za vykonanie testu, jeho zdokumentovanie a v prípade chyby jej nahlásenie. Postup je v metodike chýb.

Vývojár – člen tímu, ktorý vytvára zdrojový kód. Je zodpovedný za tvorbu jednotkových testov k danému kódu.

8.8.1.2 Súvisiace metodiky

- Metodika riadenia
- Metodika chýb

8.8.2 Procesy

8.8.2.1 Stratosferický modul

Testovaniu servisného modulu stratosférického balóna je venovaná väčšia pozornosť, keďže prípadnú chybu (softvérovú alebo hardvérovú) nie je možné počas letu opraviť a môže dôjsť k neúspešnému ukončeniu letu.

A. Funkcionálne testovanie

Najdôležitejšie je testovanie zdrojového kódu, ktorý nie je závislý od komunikácie s hardvérom. Testovanie je formou jednotkových testov. Za tvorbu a vykonávanie týchto testov je zodpovedný člen tímu, ktorý implementuje danú funkciu.

B. Nefunkcionálne testovanie

V rámci nefunkcionálneho testovania je nutné testovať výnimočné prípady, ktoré môžu nastat' pri lete, hlavne v súvislosti s hardvérom. Nefunkcionálne testy sú vykonávané na hotovom prototype. Dôležité oblasti testovania sú komunikácia modulu s pozemným strediskom, spoľahlivosť modulu v extrémnych podmienkach (nárazy, teplota), výdrž batérií.

8.8.2.2 Serverová aplikácia

Testovanie servera je formou jednotkových testov. Za tvorbu testov je zodpovedný člen tímu, ktorý implementuje danú funkcionality priradenú v úlohe. Pred uzavretím úlohy je potrebné aby všetky testy prebehli úspešne.

Súčasťou nasadenia novej produkčnej verzie je vykonanie všetkých funkčných testov členom tímu zodpovedným za nasadenie a členom tímu zodpovedným za prehliadku tímu.

A. Vykonávanie jednotkových testov

Framework Flask, ktorý je použitý v serverovej aplikácii, poskytuje moduly pre tvorbu testov a ich vykonávanie.

```
# balon/balon_test.py
import unittest
from balon import main, db

class BalonTestCase(unittest.TestCase):

    def setUp():
        ...

    def tearDown():
        ...

    def test_metoda_sub(self):
        a = 3
        b = 3
        assertEquals(a,b)
```

Ukážka 1 Vzorový súbor pre jednotkové testy

Test spustíme príkazom **python balon_test.py**

```
C:\dev\TP\server>python balon_test.py
2016-12-13 01:01:48,105 - Balon Logger [DEBUG] __init__.<module>(): Database
Path: C:\dev\TP\server\balooooooon.sqlite3
2016-12-13 01:01:49,589 - Balon Logger [DEBUG] __init__.<module>(): Starting
flask app __init__.py
main.<module>(): Starting flask app main.py

....
-----
Ran 5 tests in 0.143s
OK
```

Ukážka 2 Výpis po úspešnom vykonaní všetkých testov

B. Systémové a akceptačné testy

Pred odovzdaním prototypu je vykonané systémové a akceptačné testovanie. Systémové testovanie vykonávajú zodpovední členovia tímu. Cieľom je overiť funkčnosť a správny beh webovej aplikácie. Akceptačné testy sú vykonávané spolu so zákazníkom. Cieľom je overiť, či implementovaný systém spĺňa všetky zadané požiadavky.

8.8.2.3 Dokumentovanie testovania

Úspešné výsledky funkcionálnych testov nie je potrebné dokumentovať. Systémové testy sú vykonávané spolu s prehliadkou kódu, ktorého schválenie znamená aj úspešné vykonanie testov. Nefunkcionálne testy sú dokumentované v rámci dokumentácie inžinierskeho diela.

Pokiaľ je objavená chyba alebo iný nedostatok, je potrebné ju zdokumentovať. Procesy manažmentu chýb opisuje metodika chýb.

9 Uplatňovanie metodiky riadenia

9.1. Metodika pri prvom šprinte

Jednoduchý popis

Prvý šprint neprebiehal podľa metodiky SCRUM. Tím sa zoznamoval s projektom, sčasti aj členovia tímu medzi sebou a na využitie SCRUMu neostal takmer priestor. Avšak vzhľadom na snahu o dodržiavanie tejto metodiky sme z nej využili zopár princípov, ktoré nám boli v danom čase známe.

Tím poznal rolu product owner, scrum master a implicitne existoval aj develeopment team. Tím očakával od product ownera zadania, resp. úlohy, či požiadavky pre projekt, ktoré by sa dali považovať za SCRUMové user stories. Tieto požiadavky mali byť neskôr prepísané alebo rozčlenené na množstvo úloh. Avšak prínos product ownera do tímu nespíňal charakteristiku role. Tím navrhoval funkcionálne i nefunkcionálne požiadavky na produkt a product owner iba schvaľoval alebo upravoval požiadavky. product owner ale požadoval to, čomu by sa dalo povedať SCRUMové Epicy.

Po zapísaní úloh nasledoval planning poker, ktorý bol ale znova vykonaný nesprávne, keďže ohodnocované boli epic(y(vtedy nesprávne nazývané user stories), nebola stanovená referencia pre story points, pridelené hodnoty neboli nikam zapísané a v konečnom dôsledku táto aktivita nemala pre tím prínos.

Nasledovalo pridelovanie úloh členom developement teamu. Metodika SCRUM hovorí, že každý člen tímu si má úlohy voliť aktívne a z vlastnej iniciatívy. Avšak v tíme bolo potrebné aby scrum master pridelil úlohy jednotlivým členom.

V tíme bola snaha o vytvorenie sprint backlogu. Ten ale neboli úplne pochopený. Všeobecne bolo povedomie o troch stĺpcach “To-Do”, “In progress”, “Done”. Nesprávne ale vznikli viaceré tabule a to pre každý epic (vtedy user story) jedna. Toto rozdelenie vyplynulo z predstavy, že na user stories budeme pracovať počas celého vývoja produktu. Vzhľadom na toto nesprávne rozdelenie nemal význam ani burndown chart a tým pádom nebolo možné ani určiť velocity.

V priebehu šprintu vznikla snaha o anonymnú formu brainstormingu pomocou nástrojov dostupných na webe, ktorej výstup mali byť epic(y, user stories a tasky pre ďalší šprint. Táto snaha ale bola väčšinou členov ignorovaná. Stretnutie počas šprintu bolo naplnené obsahom, ktorý patrí do sprint review a obsahom príbuzným obsahu retrospektívy. Na tomto stretnutí uprostred šprintu sme zhodnotili prácu na taskoch a ich dokončenie a pridelili si nové úlohy.

Na stretnutí na záver šprintu už bolo v tíme povedomie o praktikách a metodikách SCRUMu a preto sa aspoň spravil sprint review a retrospektíva. Tu každý člen tímu oboznámil kolegov o stave jeho taskov a následne kriticky ohodnotil svoju prácu počas šprintu. Tieto dva evenenty boli avšak veľmi

úzko spojené a zmiešané. Značným problémom počas šprintu bolo, že scrum master sa venoval práci, ktorú mali mať na starosti členovia developement teamu.

Správne princípy

- Zapísanie úloh, pre sprint
- Rozdelenie úloh medzi členov
- Informovanie členov tímu o priebehu úloh pomocou 3-stĺpcového systému
- Empirizmus

Nesprávne princípy

- Veľmi veľké rozdiely oproti SCRUM metodológií
- Nepochopenie úloh jednotlivých scrumových rolí
- Nepochopenie scrum artifacts
- Nepochopenie scrum events
- Veľmi slabá iniciatíva a vysoká pasivita u členov tímu

9.2. Metodika pri druhom šprinte

Jednoduchý popis

Vzhľadom na to, že jedna z úloh bola podrobnejšie preskúmať SCRUM, tímu 10 sa podarilo priblížiť sa fungovaniu podľa tejto metodiky. Veľkým prínosom pre tím bolo aj cvičenie z MIS, kde nás cvičiaci Ing. Srba vysvetlil ponúkol riešenia tímu ako pracovať bližšie k metodike SCRUM aj aké a ako používať nástroje. Na základe toho vznikla aj metodika popísaná vyššie.

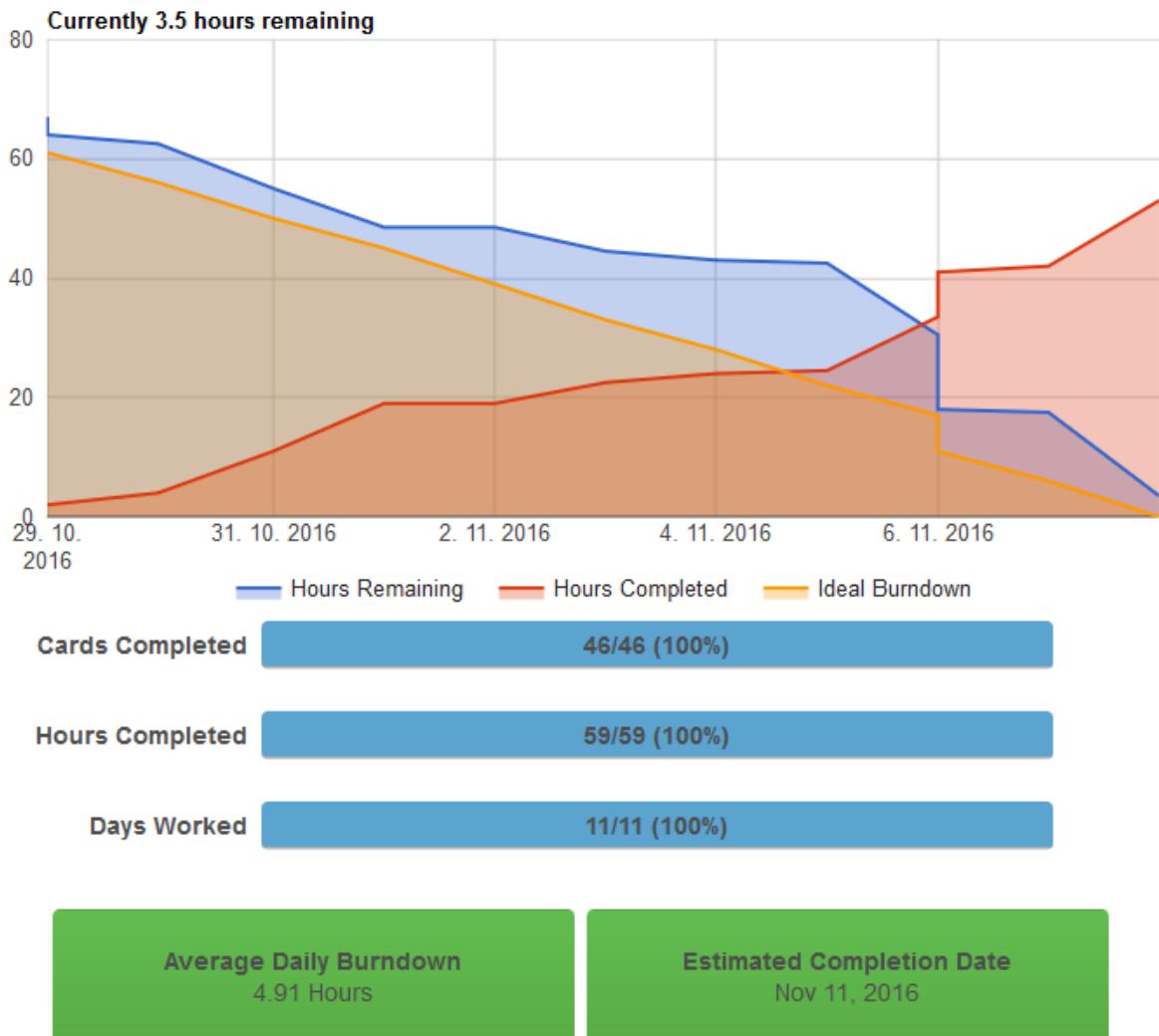
Po ukončení prvého šprintu tím okamžite uskutočnil sprint planning. Ten pozostával z vytvorenia user stories. Po ich vytvorení začal tím pracovať na rozdeľovaní úloh do taskov avšak podlhom stretnutí boli členovia vyčerpaní a stretnutie sa ukončilo s nie celkom jasným záverom.

Tím sa ale stretol o 2 dni, kedy dokončili sprint planning. Na tomto stretnutí tím prebral finálnu formu user stories. Uskutočnil sa planning poker a dohodol sa konkrétny postup práce na úlohach. Následne sa zvolili úlohy z product backlogu do sprint backlogu a každý z členov si vybral úlohy, ktoré mu vyhovovali najviac a zároveň každý z členov mal pridelené úlohy ohodnotené spolu rovnakým počtom story points. Pri plánovaní bolo ale úplne vyniechané určenie sprint goal z dôvodu, že tím o potrebe jeho definovania nevedel. V priebehu ďalších dvoch dní, každý z členov pridal do sprint backlogu úlohy ku každému z jemu pridelených user stories. Nasledujúce dni všetci členovia pracovali na svojich úlohach čo bolo viditeľné na zmenách v sprint backlogu.

Na záver šprintu sa tím znova stretol v pravidelnom čase. Prebehol sprint review, kde každý člen informoval o svojej práci. Každému členovi sa podarilo aspoň začať prácu na všetkých úlohách

a veľké množstvo sa podarilo aj dokončiť. Medzi nedokončenými úlohami ostali aj také, ktoré boli podporné úlohy, týkali sa samotného sprintu a preto nebolo možné ich počas sprintu splniť. Počas sprintu sa zmenili odhady časov práce na úlohách a tie boli zapísané do nástroja na správu sprintu. Tentokrát zo sprintu vznikol aj burndown chart, ktorý ale stále nie je správny, kvôli zmenám v odhadoch, avšak stále má klesajúcu tendenciu a výsledok je pozitívny.

Velocity sprintu bola 60 story points.



Obrázok 17 Burndown chart 2. šprintu

Nasledovala retrospektíva, kde každý z členov development teamu v krátkosti ohodnotil a skritizoval svoju prácu. Väčšina hodnotení bola v zásade neutrálna ale korektná.

Úlohy a user stories, ktoré sa nepodarilo dokončiť v šprinte boli presunuté späť do product backlogu.

Správne princípy

- Vykonanie sprint planningu

- Správny product backlog
- Planning poker
- Vznikol burndown chart s iba klesajúcou tendenciou
- Získali sme hodnotu velocity
- Práca so sprint backlogom a product backlogom

Nesprávne princípy

- Rozdelenie sprint planningu
- Vynechanie sprint goal
- Časovo nevhodne zvolené úlohy
- Zapísanie zmenených odhadov

9.3. Metodika pri tret'om šprinte

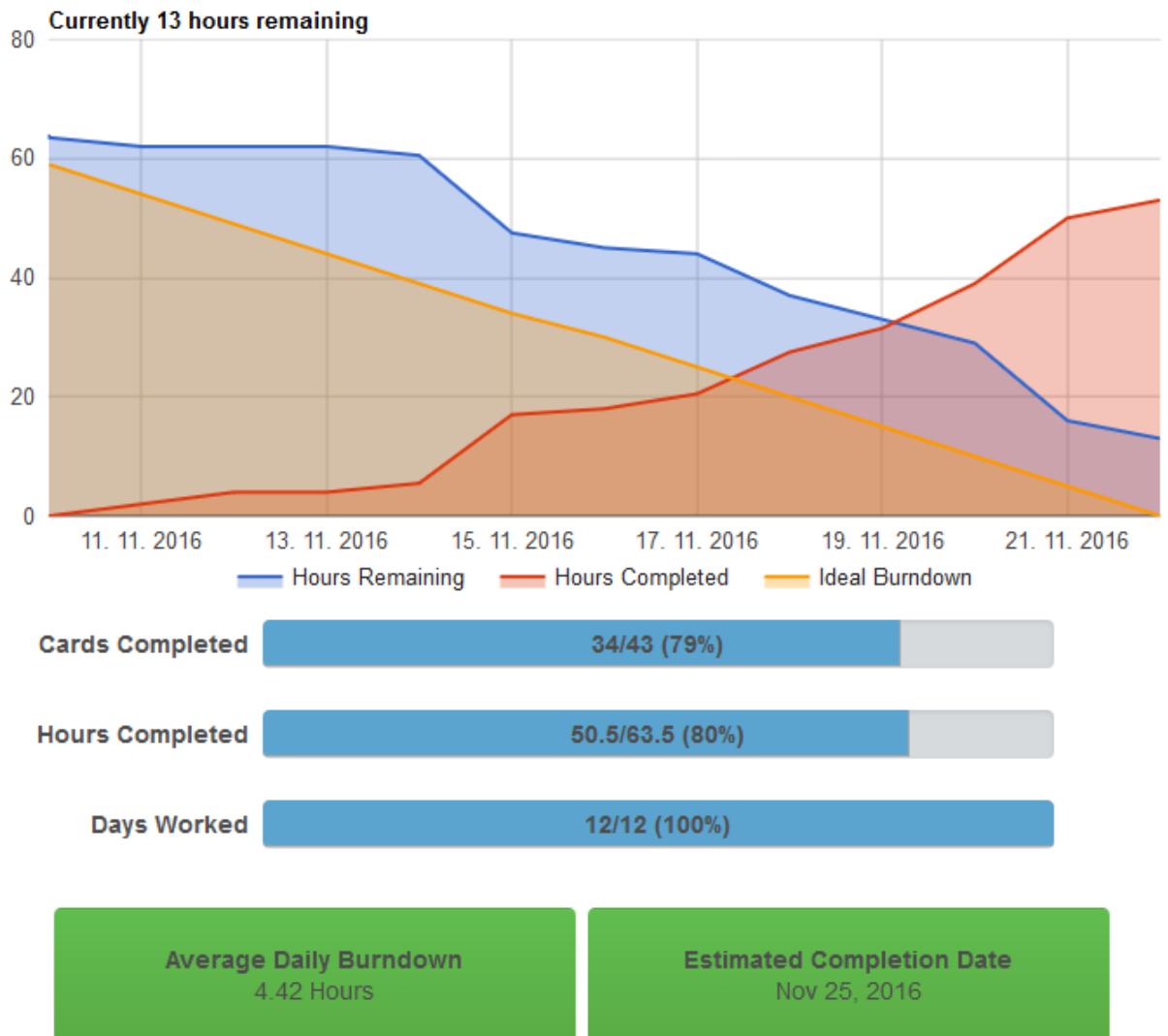
Jednoduchý popis

Po uzavretí druhého šprintu bol ihneď doplnený product backlog o user stories, ktoré sa tímu podarilo v projekte identifikovať. Vybrané user stories boli ohodnotené technikou planning poker. Hodnotili sa príbehy o ktorých bolo zrejmé, že sa dostanú do 3. šprintu.

Následne boli príbehy presunuté do novo vytvoreného sprint backlogu. Tu si každý člen zvolil príbeh, o ktorý mal záujem, zapísal k nemu definition of done a prípadne rozdelil na tasky.

Tím sa dohodol na termíne modifikovaného „daily scrum“, ktorý sa mal uskutočniť v najbližšiu sobotu a to tým spôsobom, že do 16:00 každý z členov informuje pomocou internetovej komunikačnej služby o svojom postupe na príbehoch šprintu. Každý člen mal uviesť na čom už pracoval a na čom plánuje pracovať do ďalšieho stretnutia.

Záver šprintu prebehol štandardne cez sprint review a retrospektívou. Niektoré z úloh sa nepodarilo dokončiť a niektoré ani začať, avšak väčšina bola k záveru šprintu hotová. Velocity bola približne 37.



Obrázok 18 Burndown chart 3. šprintu

Správne princípy

- Správna práca s user stories
- Využitie modifikovaného daily scrumu

Nesprávne princípy

- Nedokončenie úloh

9.4. Metodika pri štvrtom šprinte

Jednoduchý popis

Pri začiatku štvrtého šprintu už bolo možno o tíme povedať, že je zabehnutý. Každý vedel čo sa deje, čo má robiť, čo sa od neho očakáva a čo chceme na stretnutí dosiahnuť. V zásade nebolo pri začiatku čo vytknúť a ani nebola oblasť v ktorej by nastalo výrazné zlepšenie. Vzhľadom na organizáciu predmetu bol šprint predĺžený na 3 týždne.

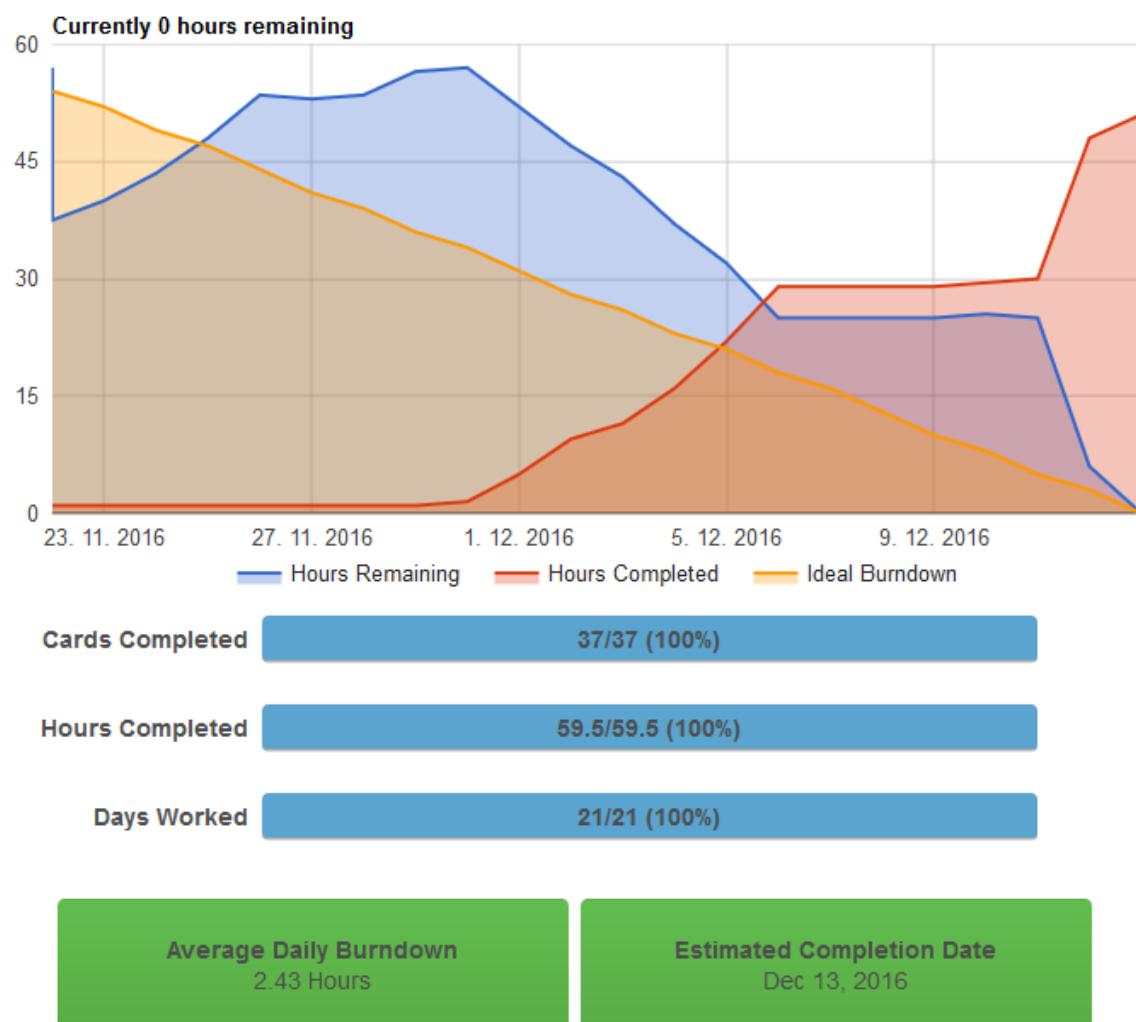
V priebehu šprintu ale pribudli do šprint backlogu tasky a k nim aj časové odhady čo spôsobilo, že burndown chart sa nedržal optimálnej línie ale stúpal. Neskôr tím úplne pozastavil prácu na projekte z dôvodu vyťaženosť inde.

Správne princípy

- Držanie sa princípov SCRUMu

Nesprávne princípy

- Vynechanie niekoľko dni práce



10 Webové sídlo projektu

Aby sme sprístupnili progres na tvorbe projektu verejnosti, prezentujeme projekt na webovej stránke <http://labss2.fii.tstuba.sk/TeamProject/2016/team10is-si/>. Aby sme zjednodušili URL webovej stránky, zakúpili sme doménu baloooooon.tk, ktorá je nastavená tak, aby zobrazovala ten istý obsah. Webová stránka je umiestnená na serveri poskytnutom fakultou. Na webovom sídle projektu sa nachádzajú:

- základné informácie o stratosférickom balóne
- vlastnosti servisného modelu, ktorý bude výsledkom projektu
- kalendár udalostí, ktoré sa udiali v rámci tvorby projektu
- dokumenty týkajúce sa projektu – zápisnice, exporty z programu Trello, retrospektívy a šprinty
- zoznam členov tímu
- kontakt na členov tímu

Tento web pravidelne aktualizujeme novými informáciami a dokumentami. Po každom tímovom stretnutí zobrazíme na webe zápisnicu z tohto stretnutia vo formáte pdf, rovnako ako aj exporty z programu Trello a ďalšie informácie.

11 Bibliografia

- [1] K. Schwaber a J. Sutherland, The Scrum Guide, 2016.

Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Ilkovičova 2, 812 19 Bratislava

Tímový projekt – Stratosférický balón

Dokumentácia k inžinierskemu dielu

Tím 10

Cvičiaci/Vedúci: Ing. Michal Valiček.

Akademický rok: 2016/2017

Autori:

Bc. Tomáš Urban
Bc. Martin Oravský
Bc. Márius Rak
Bc. Jakub Findura
Bc. Maroš Frkáň
Bc. Ján Pánis
Bc. Dominik Pisarovič

Obsah

Zoznam obrázkov	5
Zoznam tabuliek	5
1 Úvod.....	6
1.1 Ciele projektu	6
1.2 Globálne ciele pre zimný semester	6
1.3 Stratosférický balón	6
1.3.1 Náklad	7
1.3.2 Legislatíva a bezpečnosť	8
1.4 História vypúšťania experimentálnych balónov	9
1.5 Súčasné prístupy	10
1.5.1 Nasa	10
1.5.2 Balón Universum.....	11
1.5.3 Lietajúci experiment BU60-1	12
2 Celkový pohľad na systém.....	12
2.1 Architektúra	12
3 Moduly systému a integrácia	14
3.1 Platformy a komponenty	14
3.1.1 Platforma Raspberry Pi	14
3.1.2 Prototypovacia platforma Arduino.....	16
3.1.3 Intel Galileo.....	19
3.1.4 SparkFun GPS zaznamenávací štít.....	20
3.1.5 Meranie teploty	21
3.1.6 Meranie času	23
3.1.7 Preposielanie údajov zo stratosférického balóna do aplikácie	25
3.1.8 Vypúšťací ventil na postupné znižovanie tlaku v stratosférickom balóne	26

3.2	Server.....	28
3.2.1	Služby na správu zdrojového kódu	28
3.2.2	Aplikačný server	30
3.2.3	Databázové úložisko	30
3.3	Návrh	31
3.3.1	Architektúra servera	31
3.3.2	Webová stránka	32
3.4	Implementácia 1. prototypu.....	37
3.4.1	Zaznamenávacia časť	37
3.4.2	Preposielacia časť.....	38
3.4.3	Backup pamäťový modul	40
3.4.4	Prijímanie SMS - problém.....	41
3.4.5	Prijatie SMS správy v Android aplikácií.....	42
3.4.6	Odoslanie HTTP requestu v Android aplikácií	43
3.4.7	Majáčik na dohľadanie	44
3.4.8	Spracúvajúca časť.....	45
3.4.9	Zobrazovacia časť	47
3.5	Propagácia na Facebooku	49
3.5.1	Prehľad	49
3.5.2	Prístupové tokeny	49
3.5.3	Získanie prístupových Facebook tokenov	49
3.5.4	Integrácia v aplikácií	50
3.5.5	Grafový API prieskumník	51
3.6	Propagácia na Twitteri.....	51
3.6.1	Prehľad	51
3.6.2	Význam Twitteru.....	52
3.6.3	Prístup aplikácie pomocou REST API.....	52

3.6.4	Vytvorenie aplikácie pre Twitter API	53
3.6.5	Limitovanie prístupu k Twitter API	54
3.6.6	Knižnice pre prístup k Twitter API	54
3.7	Testovanie.....	55
3.7.1	Zobrazovanie údajov na stránke v reálnom čase.....	55
3.7.2	Spracovanie údajov na serveri.....	55
4	Technická dokumentácia.....	Chyba! Záložka nie je definovaná.
4.1	daco.....	Chyba! Záložka nie je definovaná.
5	Bibliografia	56

Zoznam obrázkov

Obrázok 1 Študentský amatérsky stratosférický balón vypustený v americkej púšti Mojave.	10
Obrázok 2 Balón vypustený kanadskou vesmírnou agentúrou z roku 2011.	11
Obrázok 3 Model architektúry projektu	13
Obrázok 4 Raspberry Pi	14
Obrázok 5 Rozloženie pinov na Raspberry Pi	15
Obrázok 6 Mikrokontrolér Arduino Uno [6]	17
Obrázok 7 Mikrokontrolér Arduino Mega 2560 [7]	19
Obrázok 8 SparkFun GPS zaznamenávací štít [8]	20
Obrázok 9 Teplotný senzor DS18B20 [14]	23
Obrázok 10 Graf závislosti teploty od odporu (KTY81-120) [15]	23
Obrázok 11 Schéma zapojenia RTC modulu [17]	24
Obrázok 12 Porovnanie presnosti „Arduino stopiek“ a stopiek	25
Obrázok 13 Rádiový tracker	26
Obrázok 14 Vypúšťací ventil	27
Obrázok 15 Grafické rozhranie Jenkins	30
Obrázok 16 Navrhovaná viacvrstvová architektúra servera	32
Obrázok 17 Navrhovaná podoba webovej stránky	33
Obrázok 18 Schéma GSM modulu	34
Obrázok 19 Zjednodušený model zaznamenávacej a preposielcej časti	39
Obrázok 20 Prototyp obsahujúci backup pamäťový modul	41
Obrázok 21 Príjanie SMS v android systéme pomocou broadcast príjmačov	42
Obrázok 22 Schéma RF transmitting module 433Mhz [22]	44
Obrázok 23 Diagram znázorňujúci interakciu medzi adminom, aplikáciou a Facebook Graph API	50
Obrázok 24 Grafový API priezkumník	51
Obrázok 25 Generovanie prístupových tokenov na dev.twitter.com	54

Zoznam tabuľiek

Tabuľka 1 Sumarizačná tabuľka mikrokontroléra Arduino Uno [6]	17
Tabuľka 2 Sumarizačná tabuľka mikrokontroléra Arduino Mega 2560 [7]	18

1 Úvod

Oblast' stratosférických letov je mimoriadne zaujímavá a sama o sebe značne špecifická. Táto práca obsahuje dokumentáciu k projektu, ktorým by sme chceli nadviazať na úspešné vypustenie stratosférického balóna v máji 2016. V nasledujúcich častiach popisujeme ciele, ktoré plánujeme dosiahnuť v rámci vývoja nášho servisného modulu, ktorý je hlavnou časťou našej práce. Ďalej popisujeme história vypúšťania balónov do stratosféry, ich využitie a následne aj náš návrh riešenia. Tento návrh opisuje všetky moduly nášho systému, od analýzy, cez návrh až po implementáciu. Taktiež popisujeme nami zvolený spôsob propagácie nášho projektu, ako aj jeho výsledkov.

1.1 Ciele projektu

Cieľom nášho projektu je plne funkčný a spoľahlivý servisný modul umožňujúci pravidelné vypúšťanie balónov s experimentami rôznych projektov našej fakulty. Tento servisný modul je špeciálne zariadenie, ktoré pracuje v špecifických podmienkach ako je radiačné pozadie a extrémne zmeny teplôt. Tiež jeho neoddeliteľnou súčasťou je schopnosť zasielať informácie o svojej polohe za účelom úspešného dohľadania balóna po dopade naspäť na zem.

1.2 Globálne ciele pre zimný semester

Počas zimného semestra máme v pláne zamerat' sa v prvom rade na kvalitnú analýzu problému, z ktorej následne budeme vychádzať. Správne pochopenie problému je nesmierne dôležité pre dobré nasmerovanie projektu. Po ukončení analýzy sa zameriame na prípravu hardvéru a jeho vzájomnú kompatibilitu a overenie požadovanej funkčnosti. Popri vývoji softvéru a jeho testovaní, by sme sa tiež chceli venovať špecifikácií metodík, pre efektívny a plynulý kolaboratívny postup v našej práci. Spolu s navrhnutím architektúry sa plánujeme taktiež venovať implementácii prototypu s obmedzenou funkcionalitou. Prototyp nášho servisného modulu sice nebude spĺňať všetky požiadavky umožňujúce let do stratosféry, avšak zabezpečia základ pre zachytávanie a preposielanie dát a tiež propagáciu s využitím základných informácií z modulu.

1.3 Stratosférický balón

Stratosférické balóny sú balóny vypúšťané do atmosféry za rôznym účelom. Ich cieľom je dosiahnuť stratosféru, vrstvy zemskej atmosféry, ktorá sa nachádza vo výške približne 10 až

50 km nad zemským povrhom. Pod balónom je upevnený náklad, ktorý je potrebné vyniesť balónom do stratosféry. Väčšinou sa jedná a meteorologické prístroje, vedecké experimenty, rôzne senzory či kamery alebo fotoaparáty.

Balón neobsahuje žiadny aktívny pohon. Jeho jedinou hybnou silou je nižšia hustota plynu vo vnútri balónu oproti okolitej atmosfére. Vďaka tomu sa balón pôsobením vztlakovej sily dokáže vznášať a vystúpiť do požadovanej výšky. Pretože s narastajúcou výškou klesá tlak atmosféry, plyn v balóne sa rozpína, čím sa zväčšuje priemer balóna. Keď je vnútorný tlak väčší ako pružnosť balónu, dôjde k jeho roztrhnutiu, často označovanému anglickým výrazom „burst“. Po roztrhnutí nasleduje voľný pád nákladu, ku ktorému je pripojený menší padák, ktorý spomalí finálny pád a riziko poškodenia nákladu pri dopade.

Na to aby bol úspešne splnený cieľ vypustenia balónu, musí celá konštrukcia obsahovať niekoľko komponentov. Hlavným je samozrejme balón naplnený potrebným plnom. Ďalšou časťou je náklad, ktorý je v ochrannom obale, ktorý ho izoluje od nepriaznivých podmienok vo veľkých výškach atmosféry. Obsahom nákladu sú prístroje a experimenty, ktoré je potrebné vyniesť a obslužná elektronika, ktorá sa stará o lokalizáciu balóna a odosielanie jeho polohy aby bolo možné nájsť náklad po dopade.

1.3.1 Náklad

Pod pojmom náklad môžeme rozumieť dve veci. V prvom prípade sa jedná o všetko zariadenie vrátane ochranného obalu, ktoré je pod balónom zavesené. V užšom význame môžeme pod nákladom chápať zariadenia uložené v priestore, ktorý poskytneme iným v našom balóne.

Zásadným obmedzením je hmotnosť, keďže balóny majú obmedzené množstvo hmotnosti, ktorú dokážu vyniesť do atmosféry. Hlavným faktorom ovplyvňujúcim hmotnosť nákladu je veľkosť balóna. Čím je balón väčší, tým má väčší vztlak a dokáže vyniesť väčšiu hmotnosť. Hmotnosť nákladu tiež ovplyvňuje parametre samotného letu, konkrétnie výšku stúpania a nadmorskú výšku, pri ktorej sa balón roztrhne. Preto je prioritou čo najviac minimalizovať hmotnosť nákladu, čím urýchlimo vzostup a zvýšime maximálnu dosiahnutelnú nadmorskú výšku a použitie menšieho balóna zníži náklady na jeho vypustenie.

Cely náklad je uložený v nádobe z polystyrénu, ktorá poskytuje hlavne ochranu pred poveternostnými vplyvmi a tepelnú ochranu elektroniky, pričom môže byť obalená v ďalších materiáloch zabezpečujúcich ochranu pred UV žiarením alebo radiáciou. Vo vnútri sa nachádzajú obslužné zariadenia pre riadenie letu a zariadenia alebo predmety potrebné pre splnenie účelu letu balóna.

Primárnym účelom servisného modulu je lokalizácia balóna a odosielanie jeho polohy riadiacemu stredisku pre monitorovanie pohybu a jeho vyzdvihnutie po dopade na zem. Servisný modul obsahuje riadiacu jednotku, GPS modul na určenie polohy, polohové senzory ako akcelerometer, kompas či gyroskop a rôzne ďalšie senzory pre meranie teploty, tlaku alebo vlhkosti. Odosielanie telemetrie riadiacemu stredisku je zabezpečené cez rádiové vlny alebo cez GSM siete pri nižších výškach v dosahu pozemnej infraštruktúry operátora mobilných sietí. Nezanedbateľnou časťou aj z pohľadu hmotnosti je zdroj elektrickej energie v podobe batérie prípadne v kombinácii so solárnymi článkami.

Vynášaný náklad môže byť v princípe čokoľvek, čo je v rámci rozmerových a hmotnostných limitov. Najčastejšie to bývajú meteorologické prístroje, vedecké experimenty, fotoaparáty a iné zariadenia.

1.3.2 Legislatíva a bezpečnosť

Pred vypustením balóna by sme sa mali zamyslieť nad bezpečnosťou a rizikami spojenými s vypustením balóna do voľného priestoru. Pretože balón nie je možné v druhej väčšine riadiť pri vzstupe ani pri páde, môže potenciálne ohrozit majetok a zdravie iných ľudí.

Počas manipulácie je potrebné si dávať pozor na citlivú elektroniku, ktorá by sa mohla poškodiť. Pokial bude balón naplnený vodíkom, je obzvlášť potrebná vysoká obozretnosť kvôli vysokej výbušnosti.

Pred vypustením je potrebné sa oboznámiť s legislatívou, ktorá sa zaoberá letovou prevádzkou a v prípade potreby kontaktovať dané úrady. Všetky dopravné prostriedky v letovom priestore musia byť oboznámené s prítomnosťou balóna aby nedošlo ku kolízií prípadne vážnejšiemu neštastiu. Pri dopade je potrebné myslieť na miesto dopadu, tak aby nedošlo k poškodeniu majetku alebo poraneniu iných osôb. K tomu slúžia rôzne softvérové nástroje na výpočet predpokladanej trasy letu a čiastočnému naplánovaniu miesta dopadu.

V rámci Slovenskej republiky vypúšťanie balónov, resp. akýchkoľvek predmetov, do atmosféry reguluje zákon č. 143/1998 Zb., Zákon o civilnom letectve. Zákon nehovorí priamo o balónoch, no riadi a obmedzuje využívanie vzdušného priestoru Slovenskej republiky s ohľadom na bezpečnosť. Relevantné sú pre nás hlavne §5 až §8. Okrem toho je vydávaná Letecká informačná príručka (AIP) [1] odborom manažérstva leteckých informácií (AIM), ktorý patrí pod štátny podnik Letové prevádzkové služby Slovenskej republiky. Táto príručka je určená hlavne pre riadenie leteckej dopravy nad územím Slovenskej republiky, avšak určuje

aj pravidlá pre vypúšťanie stratosférických balónov. Okrem tejto príručky vydáva aj rôzne civilné predpisy týkajúce sa okrem iného aj vypúšťania balónov.

Stratosférickým balónom sa venuje predpis **L2 Pravidlá lietania** vychádzajúci z Dohovoru o medzinárodnom civilnom letectve, konkrétnie Oddiel 3, kapitola 1,

SERA.3140 Neobsadené voľné balóny

„Lety neobsadených voľných balónov sa vykonávajú spôsobom, ktorým sa minimalizuje nebezpečenstvo pre osoby, majetok alebo pre ostatné lietadlá a ktorý je v súlade s podmienkami stanovenými v dodatku 2.“ [2]

Ďalej je neobsadeným voľným balónom, ich klasifikáciu a pravidlám venovaný Dodatok 4 predpisu L2. [3] V rámci Európskej únie je tento predpis s miernymi zmenami prijatý v nariadení č. 923/2012. Neobsadeným voľným balónom je venovaný dodatok 2. Z predpisov okrem iného vyplýva, že na prevádzkovanie stratosférických balónov je potrebné povolenie leteckého úradu.

1.4 História vypúšťania experimentálnych balónov

Vo francúzsku v roku 1783 sa prvý krát verejne vypustil balón naplnený vodíkom pred zrakmi 300 000 ľudí. Jacques Charles, francúzsky profesor fyziky spoločne s bratmi Robertovcami, ktorí pracovali ako renomovaní výrobcovia vybavenia určeného pre fyzikálne experimenty. Balón s názvom Charlière, ktorý dosahoval veľkých rozmerov bol napúštaný vodíkom po dobu 5 dní. Startovacím miestom bol verejný park Champ de Mars, ktorý sa nachádza nedaleko Eiffelovej veže. Expanzia plynu v balóne spôsobila jeho roztrhnutie po 45 minútach od štartu a balón sa dostal do výšky 20km nad Paríž.

Ďalším z významných použití, ktoré boli zdokumentované bol meteorologický balón, ktorý zaznamenal francúzsky meteorológ Leon TEISSERENC de Bort. Aktívne začal vypúšťať meteorologickej balóny už v roku 1896. Práca tohto meteorológa významne pomohla pri objavení a najmä bližšom preskúmaní troposféry a stratosféry. Jedná sa o jedinečné vrstvy v zemskej atmosfére, ktoré už v dnešnej dobe môžu skúmať hodikto z nás práve vypustením vlastných meteorologických balónov. Vzhľadom na to, že práca vedca Leon Teisserenc de Bort bola tak inštrumentálna, bol poctený tým, že bol po ňom pomenovaný kráter na Mesiaci a rovnako aj na Marse.

Začiatkom 20. storočia, meteorológ a geofyzik menom Alfreda Wegener použitím meteorologického balónu vykonal experimenty, ktoré ho viedli k objaveniu významnej teórie

pohybu kontinentov, niekedy aj často označovanou ako kontinentálny drift. Práca, ktorá skúmala túto teóriu bola zverejnená už v roku 1912. Táto teória sa stretla s veľkým odporom a akceptovaná bola až v roku 1960, čo je viac ako 30 rokov po jeho smrti. Taktiež, podobne ako de Bort bol poctený tým, že na Mesiaci a na Marse boli po ňom pomenované dva krátery.

Ďalším dôležitým vedcom bol americký vedec James Van Allen, ktorý v 50. rokoch 20. storočia vykonával rôzne balónové experimenty. Tieto experimenty viedli k objaveniu radiačných pásov v okolí Zeme. Tieto pásy niekedy označujeme aj ako Van Allenove radiačné pásy. Časopis Time ocenil Van Allena ako muža roka „Man of the Year“ v roku 1960 práve kvôli jeho vedeckým zásluhám.



Obrázok 1 Študentský amatérsky stratosférický balón vypustený v americkej púšti Mojave.

1.5 Súčasné prístupy

1.5.1 Nasa

Veľké bezpilotné balóny, plnené najmä héliom, poskytujú americkej vládnej agentúre NASA nenhoditeľné možnosti ako vyniesť rôzne vybavenie do kozmického priestoru. Unikátne vlastnosti takýchto programov sú kľúčové pre vývoj nových technológií. Mnoho dôležitých objavení a technológií ale aj náklady (angl. payload) pre sofistikovanejšie vesmírne lety boli práve výsledkom experimentov v ktorých boli podklady získané zo stratosférických balónov. Konkrétnie sú to napr. pozorovania Röntgenového a gama žiarenia, kozmického žiarenia alebo podklady pre odbor infračervenej astronómie. Novo vyvinuté prístupy sú zamierané na tenko vrstvové stratosférické balóny určené pre dosiahnutie vysokej nadmorskej výšky a ich dlhšie pôsobenie v atmosfére.

V roku 2009 boli vykonané viaceré dlhotrvajúce experimenty, ktoré boli prevádzkované v Antarktíde počas letného obdobia na južnej pologuli. Tieto experimenty boli výsledkom spolupráce organizácie NASA (National Aeronautics and Space Administration) a organizácie NSF (National Science Foundation). NSF zabezpečila komunikáciu a logistiku a NASA zabezpečila satelitnú komunikáciu. Unikátne vlastnosti atmosféry nad Antarktidou umožňujú vedcom vypustiť balón z vedeckej stanice McMurdo a následne ho z takmer rovnakého miesta obnoviť aj po uplynutí niekoľkých týždňov. Balóny vypustené v Antarktíde sú dlhotrvajúce, kvôli javu nazývanému polárny vír (angl. polar vortex). Tento vír spôsobuje stále, veľké nízkotlakové správanie v atmosfére, najmä kvôli nízkym teplotám. Konštantné denné svetlo v Antarktíde znamená, že sa teplota prechodom zo dňa na noc nemení. Tento fakt pomáha balónu zostať na v rovnej výške po dlhý časový interval.



Obrázok 2 Balón vypustený kanadskou vesmírnou agentúrou z roku 2011.

1.5.2 Balón Universum

Popri svojich študentských aktivít sa na Žilinskej univerzite jeden zo študentov Ondrej Závodský so svojím tímom vypustil 25.9.2010 jeden z prvých stratosférických balónov na Slovensku. Vlastný balón si zakúpil v Prahe, následne ho nafúkali tak aby bol schopný nadvihnuť 21 flášu s vodou, pripojili modul a vypustili. Pri prvom vypustení sa stretli

s množstvom komplikácií. Prvý vysielač odišiel z dôvodu odtrhnutia antény už pri vzletnutí. Druhý GPS modul zamrzol pri páde 9km nad zemou a preposielal rovnaké info, ktoré po dopade prestalo vysielať z dôvodu uvoľnenia batérií [4].

Celkový let im trval 2 hodiny a 20 minút, z čoho 1 hodina 38 bol čas stúpania a 42 minút čas klesania. Balón dokopy prešiel vzdialenosť 85 km s maximálnou dosiahnutou rýchlosťou 90km/h. Maximálna dosiahnutá výška bola 25 946m a minimálna teplota dosiahla -33,29°C.

V ďalších riešeniacach sa autor chystá zameriť na rôzne otázky ako zabránenie rotácie bud' gyroskopom alebo pomocnými krídelkami za účelom stabilizácie fotoaparátu, vyriešiť riziko pádu balóna do vody, vysielanie súradníc pomocou APRS, prípadne solárny panel alebo riadený pád pomocou kĺzavého padáku. [5]

Z jeho riešenia sme sa v mnohom poučili a jeho spísané skúsenosti využili pri implementácii nášho riešenia.

1.5.3 Lietajúci experiment BU60-1

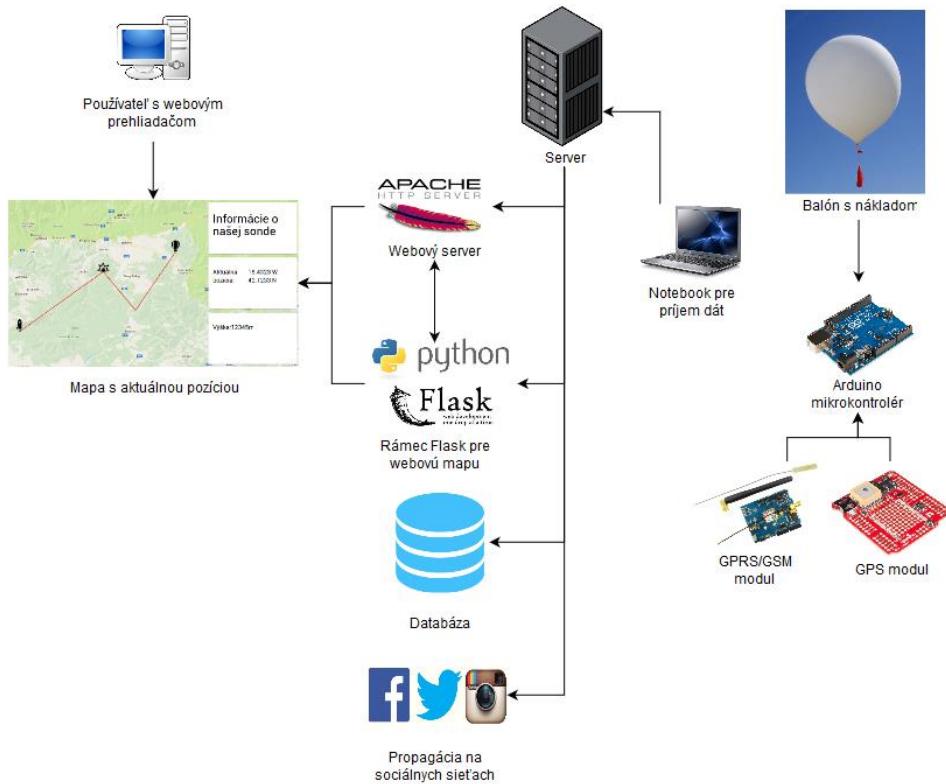
Samotný BU60-1 balón vážil 34,37kg a dosahoval dĺžku 74,5m a 53,7m po obvode. Celková váha balónu pred štartom dosahovala 39,77kg s padákom vážiacim 0,8kg spoločne s pozorovacími prístrojmi, ktorých váha zaťažila balón o 4,6kg. Prístrojmi, ktoré zaznamenávali servisné informácie o experimente boli: dve ITV kamery pre tvorbu fotiek monitorujúce nafúknutie balóna a GPS modul pre presné meranie výšky balónu.

Balón bol vypustený v ranných hodinách 23. mája 2002 o 6:35. Dosahované rýchlosť stúpania balónu dosahovala 260m za minútu a dosiahla doteraz najvyššiu zaznamenanú výšku a to 53km o necelé 4 hodiny v 10:07. Tento úspech pokoril americký rekord z 1972, ktorý dosiahol hranicu 51,8km.

2 Celkový pohľad na systém

2.1 Architektúra

V obrázku je stručne popísaná celková architektúra nášho servisného modulu.



Obrázok 3 Model architektúry projektu

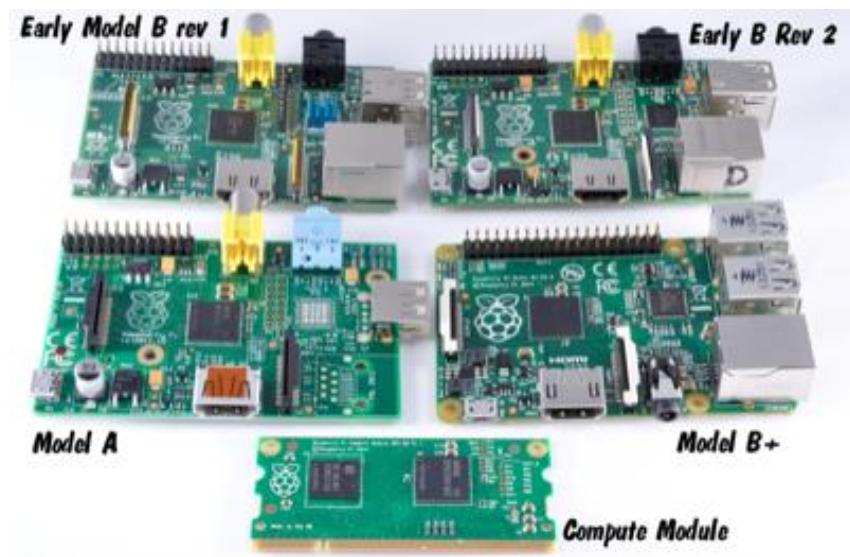
3 Moduly systému a integrácia

3.1 Platformy a komponenty

3.1.1 Platforma Raspberry Pi

Raspberry Pi je miniatúrny jednodoskový počítač vyvíjaný Anglickou spoločnosťou Raspberry Pi Foundation. Cieľom tvorcov Raspberry Pi je vývoj a predaj relatívne lacných jednodoskových počítačov s výkonom postačujúcim na fungovanie základných funkcií Linuxových operačných systémov.

Počítače Raspberry Pi používajú procesory s ARM architektúrou. V prípade prvej generácie ide o ARM1176JZF-S s taktovacou frekvenciou 700MHz. Aktuálna generácia obsahuje výkonnejší procesor ARM CortexA-7 s frekvenciou 900MHz.

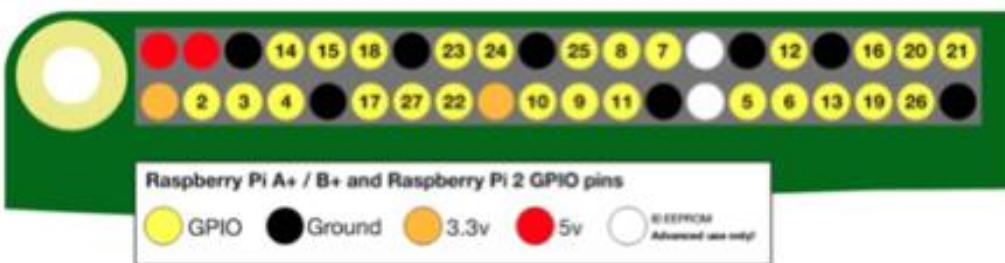


Obrázok 4Raspberry Pi

Na pripojenie rôznych štandardných aj neštandardných periférií slúži sériová linka GPIO. GPIO predstavuje 40 pinové vstupno-výstupné rozhranie. Z počtu 40 pinov 26 slúži ako GPIO a ostatné ako napájacie a uzemňovacie piny. Dva piny slúžia na prístup k EEPROM pamäti.

Pri použití GPIO pinu ako výstupu ho dokážeme softvérovo ovládať. Dokážeme ovládať hodnotu napäťia na výstupnom pine. Pri zapnutom stave je táto hodnota 3,3V a pri vypnutom 0V.

Pri použití GPIO pinu ako vstup na ňom dokážeme detegovať úroveň el. napäťia. Pri úrovni 0V Raspberry Pi deteguje vstup ako vypnutý a pri 3,3V ako zapnutý. Medzi týmito hodnotami Raspberry Pi deteguje tzv. „floating“ stav, kedy nie je možné presne určiť stav pinu. Riešením je používanie tzv. pull-up a pull-down rezistorov integrovaných na doske.



Obrázok 5 Rozloženie pinov na Raspberry Pi

Raspberry Pi je možné použiť v kombinácii s niekoľkými operačnými systémami. Oficiálne sú podporované tri linuxové distribúcie: Pidora(založené na distribúcii Fedora), Archlinux a Raspbian(Debian). Použitím týchto operačných systémov získavame plnohodnotný linuxový počítač.

Vďaka jeho univerzálnosti v kombinácii s nejakou z linuxových distribúcií má Raspberry Pi dôležité postavenie aj v odvetví IoT(Internet of Things). Ide však najmä o odvetvie rôznych domáčich projektov a nekomerčných riešení.

Jedným z vhodných programovacích jazykov pre vývoj programov pre Raspberry Pi je nepochybne C/C++. Vhodných prostredí na vývoj je hned' niekoľko, no medzi najpoužívanejšie patria NetBeans, Eclipse, XCode a podobne. Po väčšine závisí od osobných preferencií konkrétneho používateľa.

Raspberry Pi je napájané 5V jednosmerným napäťím. Výrobca na svojich stránkach odporúča zariadenie napájať zdrojom s kapacitou aspoň 1,8A v prípade druhej generácie zariadenia a 2,5A v prípade tretej. V prípade čistej dosky zariadenia, bez akýchkoľvek periférií, výrobca udáva spotrebu 330-400mA. Konkrétna energetická náročnosť však závisí od celého systému a na počte a type použitých periférií.

3.1.2 Prototypovacia platforma Arduino

Raspberry Pi nemá vstavaný A/D prevodník, preto sú na tento účel často používané iné mikrokontroléry ako napr. Arduino resp. kombinácia s nimi.

Arduino je open-source platforma pre vývoj rôznych prototypov a interaktívnych zapojení. Arduino dosky sú schopné čítať vstupy, vyhodnocovať akcie a udávať výstupy [6]. Platforma je založená na programovateľnom mikrokontroléri ATMega od spoločnosti Atmel a na trhu je dostupných niekoľko už skompletizovaných verzií.

Arduino Uno

Arduino Uno je jeden zo základných modelov tejto platformy a prvý z modelov s USB pripojením. Je založený na čipe ATMega328P.

Obsahuje:

- 14 digitálnych vstupno-výstupných pinov (6 z nich je možné použiť ako PWM [7] výstupy),
- 6 analógových vstupných pinov, [SEP]
- 16MHz oscilátor, [SEP]
- USB pripojenie, [SEP]
- napájací konektor(7-12VDC), [SEP]
- resetovacie tlačidlo. [SEP]

Na vytvorenie IP spojenia s inými zariadeniami je možné použiť Arduino Ethernet Shield. Na spoluprácu Arduina a Ethernet Shieldu je potrebné použiť vhodnú knižnicu. V prípade takýchto spojení dostávame veľmi silný nástroj na implementáciu aj zložitejších algoritmov použiteľných pri rôznych druhoch prototypov.

Na vytváranie programov pre Arduino je dostupný vývojové prostredie s rovnakým názvom. Programuje sa v jazyku založenom na C/C++. Pre programovanie sa využíva konvertor z USB na seriálový port [8].

Arduino Uno je napájané 5V jednosmerným napäťom. Odporúčané napätie je však 7-12V. Arduino má po zapnutí len približne 20-25mA. Vďaka takejto malej spotrebe sa Arduino

stáva jedným z vhodných kandidátov na použitie pre účely servisného modulu. Konkrétna energetická náročnosť však závisí od celého systému a na počte a type použitých periférií pripojených k Arduinu.

Sumarizačná tabuľka:

Mikrokontrolér	ATmega328
Operačné napätie	5 V
Vstupné napätie (doporučené)	7 - 12 V
Vstupné napätie (limity)	6 - 20 V
Digitálne V/V kolíky	14
Analógové vstupné kolíky	6
Prúd pre 5 V kolíky	40 mA
Prúd pre 3,3 V kolíky	50 mA
Flash pamäť	32 KB (0,5 KB je použitá pre program boot-ovania)
SRAM pamäť	2 KB
EEPROM pamäť	1 KB
Frekvencia kryštálu	16 MHz

Tabuľka 1 Sumarizačná tabuľka mikrokontroléra Arduino Uno [8].



Obrázok 6 Mikrokontrolér Arduino Uno [8]

Arduino MEGA

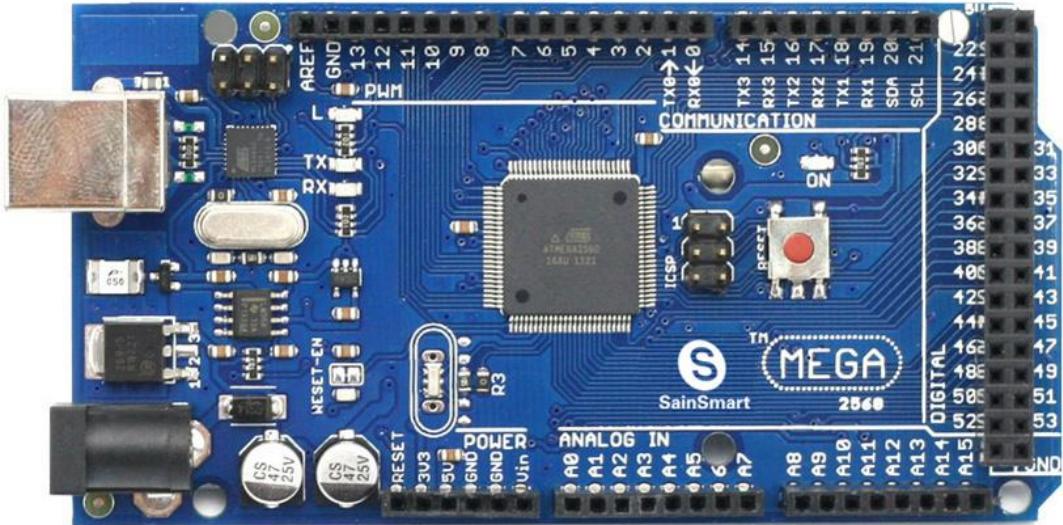
Arduino Mega 2560 je mikrokontrolér založený na čipe ATmega2560. Má 54 digitálnych vstupno/výstupných kolíkov (z ktorých 14 môže byť využitých ako PWM výstupy), 16 analógových vstupov a 4 UART. 16MHz keramický kryštál, USB pripojenie, napájací vstup a resetovacie tlačidlo. Obsahuje všetko, čo musí mikrokontrolér obsahovať. Napájať ho môžeme pomocou USB kábla, adaptéru ale aj batérie. Pre programovanie sa využíva konvertor z USB na seriálový port [9].

Mega je väčšia prototypovacia platforma, oproti klasickému Arduinu UNU, avšak je späťne kompatibilné so štítmi z menších mikrokontrolérov ako sú: UNO, Leonardo, Duemilanovo, Diecimila a iné...

Sumarizačná tabuľka:

Mikrokontrolér	ATmega2560
Operačné napätie	5 V
Vstupné napätie (doporučené)	7 - 9 V
Vstupné napätie (limity)	6 - 20 V
Digitálne V/V kolíky	54
Analógové vstupné kolíky	16
Prúd pre 5 V kolíky	40 mA
Prúd pre 3,3 V kolíky	50 mA
Flash pamäť	256 KB (8 KB je použitá pre program boot-ovania)
SRAM pamäť	8 KB
EEPROM pamäť	4 KB
Frekvencia kryštálu	16 MHz

Tabuľka 2 Sumarizačná tabuľka mikrokontroléra Arduino Mega 2560 [9]



Obrázok 7 Mikrokontrolér Arduino Mega 2560 [9]

3.1.3 Intel Galileo

Doska Intel Galileo Gen 2, momentálne najvýkonnejšia dosky so zbernicou Arduino. Srdcom vývojového kitu je výkonný 32-bitový procesor, presnejšie SoC (System on Chip), Intel Quark X1000a, ktorého inštrukčná súprava je kompatibilná s Pentiom. K dispozícii má 256 MB DDR3, 8 MB NOR flash a 8 kb EEPROM.

Popri štandardných digitálnych aj analógových vstupoch a výstupoch je k dispozícii aj slot mini-PCI Express, ktorý je možné použiť pridanie modulu Wi-Fi, 100Mbit ethernetový port, slot na kartu microSD alebo port. Na porovnanie, doska Arduino Ethernet má procesor ATmega328 16 MHz, 1 kb EEPROM, 2,5 kB RAM a 32 KB flash. Ako operačný systém možno využiť Linux (zostava Yocto 1.4 Poky), prípadne upravenú verziu Windows 8.1, ktorá je dostupná na stránke Microsoftu venovanej IoT. Windows 10 IoT Core pre túto dosku k dispozícii nie je.

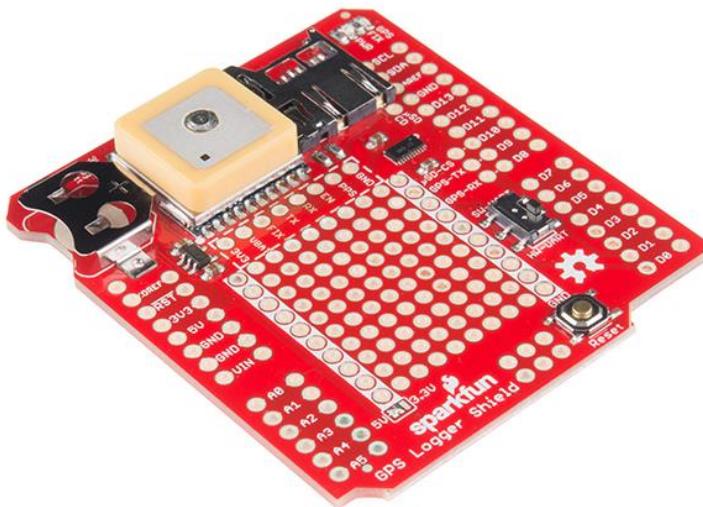
Vysoký výkon Galilea je pochopiteľne vykúpený nepomerne vyššou spotrebou, takže tento kit sa hodí tam, kde je dostať energie. V prípade použitia na účel sondy ktorá je súčasťou stratosférického balóna sme limitovaný hmotnosťou, a tým pádom aj kapacitou zdroja energie – batériou. V prípade neobmedzenej hmotnosti by bola táto platforma vďaka svojmu výkonu určite vhodnou voľbou.

3.1.4 SparkFun GPS zaznamenávací štít

Tento štít umožňuje Arduinu extra funkcionality. Sú to zaznamenávanie GPS súradníc, čo je pre nás kľúčovo dôležité, miesto pre microSD kartu (zaznamenávanie nameraných hodnôt na záložné médium), ale taktiež aj voľný priestor na usadenie ďalších senzorov alebo akčných členov [10].

Základom je čip GP3906-TLP (GPS prijímač), ktorého hlavnou výhodou je ultra nízka spotreba energie (okolo 20mA). Práve tento parameter je pre nás kľúčový. Taktiež jeho ďalšou nemenej podstatnou výhodou je práceschopnosť v rozmedzí teplôt od -30 do +85°C. Pracuje na 3,3V logike, avšak to nie je problém, keďže Arduino poskytuje aj napájaciu hodnotu napäťia 3,3V [11].

Ďalšia výhoda tohto štítu je, že je priamo predpripravený pre prototypovaciu platformu Arduino UNO, ktorá je späťne kompatibilná aj s platformou Arduino MEGA. To je pre nás veľmi výhodné z dôvodu, že nebudemusiet riešiť vývoj vlastných dosiek, do ktorých by bolo možné jednoducho osadiť daný, alebo podobný čip.



Obrázok 8SparkFun GPS zaznamenávací štít [10]

3.1.5 Meranie teploty

Meranie fyzikálnej veličiny teploty, nie je pre náš projekt až tak dôležité, avšak vďaka tomu vieme zistíť, aké sú okolité teploty v rozličných nadmorských výškach. Pomocou nameraných údajov, budeme sa v budúcnu vedieť lepšie pripraviť na okolité nežiadúce podmienky a využiť lepšie materiály, senzory alebo hardvér. Podľa analýzy sme zistili, že teplota v stratosfére väčšinou býva v rozmedzí od -60°C po 0°C , je tomu nutné prispôsobiť aj nás hardvér. Podľa funkcionality delíme teplotné senzory na:

1. **Tepломery so záporným tepelným súčiniteľom (NTC) – Termistory:** slovo termistor vychádza s kombinácie 2-ch slov (TERM-álne senzitívny rez-ISTOR). Je to špeciálny typ rezistora, ktorý predvídateľne mení svoj odpor na základe okolitej teploty. NTC termistor poskytuje vysokú odolnosť v nízkych teplotných podmienkach. Jeho efektívny pracovný rozsah je väčšinou od -50°C po 250°C (lísi sa od typu).
2. **Odporový teplotný detektor (RTD):** je tiež známy ako odporový teplomer. Meria teplotu na základe korelácií odporov medzi RTD prvkom a teplotou. RTD prvak je vyrobený s vysoko čistých, vodivých kovov ako sú: platina, med' alebo nikel. Ich hlavnou výhodou je, že ponúkajú pomerne presný lineárny výstup, ktorý je veľmi presný. Častokrát ich pracovný rozsah presahuje hranice -200°C až 600°C . Majú však jednu nevýhodu – cenu (sú extrémne drahé, v porovnaní s ostatnými).
3. **Termočlánok ako tepelný senzor:** tento typ tepelného senzoru sa skladá z dvoch rôznych kovov spojených v dvoch bodoch. Pomocou meniaceho sa napäťia medzi týmito dvomi bodmi sa dajú vypočítať zmeny teploty. Tieto články sú nelineárne a na prevod teploty sa častokrát využíva tabuľková kompenzácia hodnôt. Pri tomto spôsobe merania teploty je úplne normálna odchýlka merania 5°C . Ich hlavnou výhodou je schopnosť merať najširšie pásmo teplôt od -200°C až po 2000°C a aj preto sa stále využívajú.
4. **Tepelné senzory na báze polovodičov:** tento teplotný senzor je umiestnený na integrovanom obvode. Tieto senzory sú založené na dvoch diódach s tepelnou senzitivitou. Pomocou ich volt-ampérových charakteristik je možné s nich odčítavať a vypočítať teplotu. Ich výstup je lineárny avšak presnosť nie je veľmi uspokojivá (častokrát majú odchýlky od 1°C až po 5°C). Majú najpomalšie reakcie na zmeny teplôt (5 až 60 sekúnd), ale na druhú stranu sa s nimi veľmi jednoducho pracuje a častokrát (kedže pracujú na integrovanom obvode) dokonca vieme z nich odčítať číselnú hodnotu teploty [12] [13].

Pri analyzovaní a testovaní rôznych typov senzorov sme začínali pri tých najjednoduchších až po tie zložitejšie.

Ako prvý sme využívali teplotný senzor na báze polovodičov DS18B20. Pre jeho pomalé reakcie a nie veľmi vysokú presnosť sme ho zamietli. Práca s ním bola súčasťou jednoduchá, avšak namerané hodnoty nás nepresvedčili.

Ako druhý sme analyzovali o otestovali termistor KTY81-120 práca s ním bola so začiatku trochu komplikovaná, keďže sme museli zimplementovať funkciu, ktorá vedela jeho výstupy prevádzkať do číselných hodnôt, avšak po nájdení podobnej funkcie a patričných úpravách je práca s ním veľmi jednoduchá, presná a spoľahlivá. Zatiaľ ho považujeme za najlepší tepelný senzor, ktorý máme, aj keď jeho pracovné rozpätie činí od -55°C po 150°C. Minimálne môže a bude použitý na zaznamenávanie teploty vnútri v izolovanej sústave (predpokladáme, že teplota dnu by nemala prekročiť hodnoty od -10°C po 20°C).

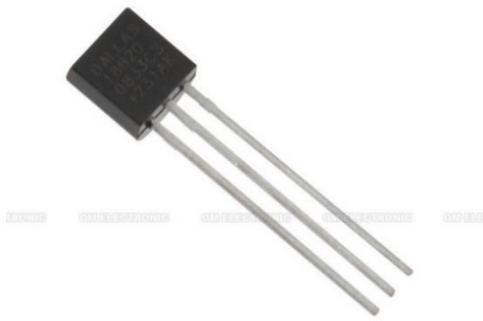
Aktuálne pracujeme na analýze a testovaní odporového teplotného detektora (RTD) Proffuse PT100-1020. Bol vybratý z dôvodu jeho presnosti, dostupnosti a schopnosti merať teploty v rozmedzí od -70°C po 500°C [14].

Tepelný senzor Dallas DS18B20

Číslicový digitálny senzor Dallas, najčastejšie sa vyskytujúci v púzdre TO-92 s tromi kolíkmi ponúka maximálne rozlíšenie 12 bitov, teda 0,0625°C. Operačný teplotný rozsah je -55°C až +125°C.

Každý tento senzor má unikátny 64 bitový identifikátor, vďaka ktorému je možné v sérií zapojiť aj viac ako jeden senzor [15].

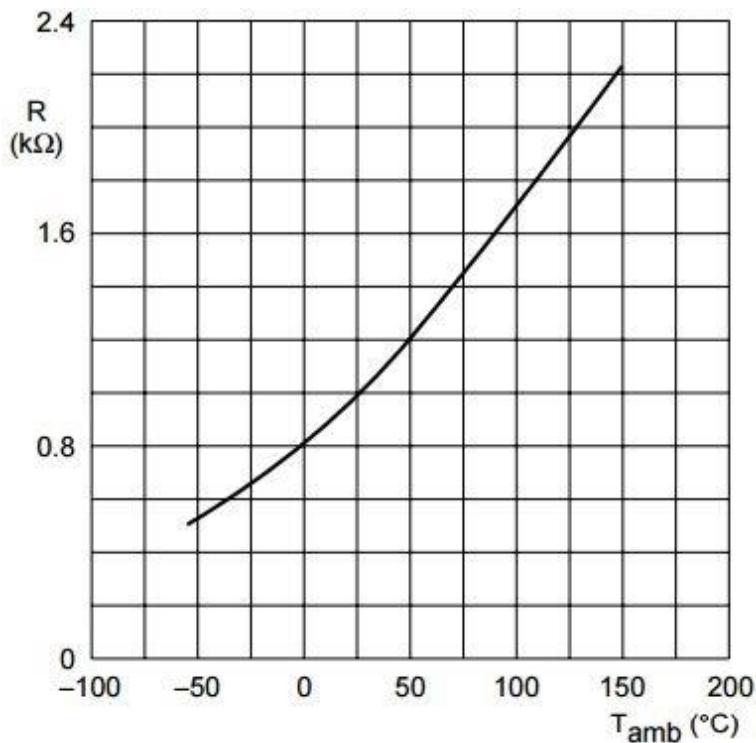
Základný princíp senzoru je nasledovný. V púzdre sa nachádzajú 2 oscilátory. Jeden s vysokým tepelným koeficientom, druhý s nízkym. Len čo sa na zbernicu vyskytne signál pre začatie merania teploty, oscilátory začnú pracovať. Po ich úspešnom splnení úlohy sa hodnota zapíše do 12 bitového registra a čaká na prečítanie hodnoty z nadradeného obvodu [7].



Obrázok 9 Teplotný senzor DS18B20 [16].

Termistor KTY81-120

Tieto teplotné senzory majú kladný teplotný súčiniteľ odporu a sú vhodné na použitie v meracích a kontrolných systémoch. Senzory sú zapuzdrené v tvrdenom plastovom SOD-70 balíku. Ich tolerancia činí 0,5%. Vyrába ich spoločnosť Philips Semiconductors [17].



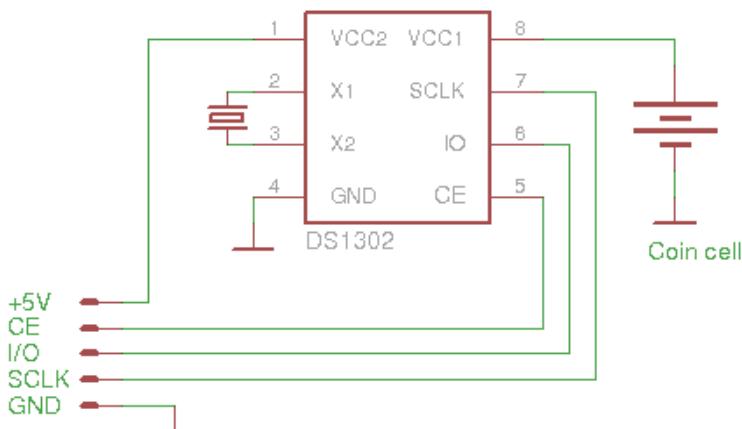
Obrázok 10 Graf závislosti teploty od odporu (KTY81-120) [17]

3.1.6 Meranie času

Meranie fyzikálnej veličiny času, je pre náš projekt kľúčové. Bez tejto veličiny by sme nevedeli v reálnom čase určiť, kedy a čo sa dialo s naším servisným modulom. Na mikrokontroléri Arduino UNO existuje viacero spôsobov zaznamenávania času.

Najjednoduchším princípom je merať čas pomocou nastavenia istej odozvy. Priebeh merania vyzeral takto: Inicializovali sme program, nastavili sme veličinu času na 0 a spustili sme opakujúcu sa slučku. Na konci každej slučky sme pozastavili činnosť procesora na 0,2 sekundy. A tak sme mohli po 5-tich zopakovaniach prehlásiť, že prešla 1 sekunda, atď... Pri jednoduchých programoch, ktoré nevyťažujú procesor, je táto metóda veľmi jednoduchá a spoľahlivá. Avšak, keď sme pridali do opakujúcej sa slučky viaceré volania metód, výpočet niektorých funkcií a čítania portov, bola táto metóda počítania času irelevantná, keďže bola veľmi nepresná. Pomocou stopiek sme odmerali 10 minút (600 sekúnd) priebehu programu, očakávali sme, že podobné hodnoty nám vráti aj naše Arduino, avšak vrátil hodnotu 7 minút a 23 sekúnd (443 sekúnd). To značilo, že namiesto 3000 cyklov sa vykonalo len 2215. Nepresnosť merania činila 26,17% a to bolo pre nás neprijateľné.

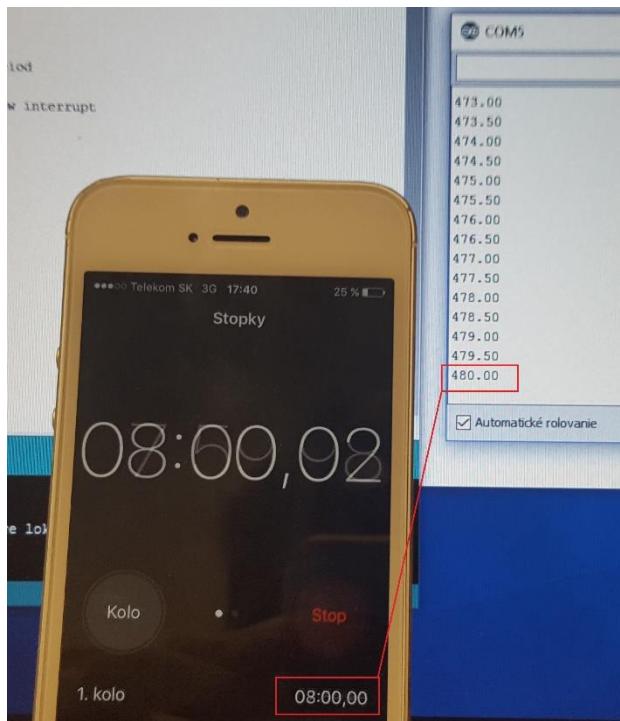
Ďalšou a asi najlepšou metódou merania času, je pridať k Arduinu integrovaný obvod RTC (Real-time clock – Hodiny reálneho času). Základom takmer každého RTC modulu je riadiaci čip (napríklad DS1302), kryštál a nabíjacia batéria alebo superkondenzátor. Schéma zapojenia väčšinou vyzerá asi takto [18].



Obrázok 11 Schéma zapojenia RTC modulu [19]

Aj keď táto metóda merania času je asi najlepšia a je vysoko pravdepodobné, že sa k nej nakoniec vrátíme, zistili sme že jednoduché a presné stopky sa dajú implementovať aj priamo na arduine s čipom ATmega 328p.

Na mikrokontroléri Arduino UNO je prítomný 16Mhz kryštál, ktorý kooperuje s procesorom ATmega 328p. Pomocou knižnice „TimerOne.h“ je možné jednoducho pracovať so všetkými hardvérovými časovačmi. Sú tri. Len čo sme implementovali jednoduchý program, otestovali sme ich. Časovače nás svojou vysokou presnosťou veľmi prekvapili. Stopli sme čas 8 minút a výsledky môžete vidieť aj sami v obrázku pod textom [20].



Obrázok 12Porovnanie presnosti „Arduino stopiek“ a stopiek

3.1.7 Preposielanie údajov zo stratosférického balóna do aplikácie

Vypúšťanie stratosférických balónov by bolo veľmi zložité a takmer zbytočné, pokiaľ by sme nemali k dispozícii aktuálne informácie o polohe balónu, jeho nadmorskej výške, vonkajšej resp. vnútornej teploty, tlaku a ďalších veličín. Vďaka zbieraniu takýchto informácií je potom dohľadanie balónu oveľa jednoduchšie. Existujú spôsoby, ktoré vzhľadom na nadmorskú výšku, v ktorej balón praskol, rýchlosť a smer vetra, hmotnosť balónu dokážu vypočítať, kde približne stratosférický balón dopadne. Jednou z možností je použiť webovú službu na <http://predict.habhub.org/>. V našom servisnom module bude použitý GPS modul, ktorý nás v každom čase dokáže informovať o súradničach, kde sa balón nachádza či už počas letu, alebo po dopade.

Tieto informácie bude samozrejme potrebné nejakým efektívnym a spoľahlivým spôsobom preniesť do našej aplikácie, ktorá bude tieto informácie zbierať a vyhodnocovať, aby boli lepšie čitateľné.

Ako funguje prenos údajov:

Aby mohol prenos údajov fungovať, potrebujeme tzv. tracker, siet' a internetové pripojenie. Tracker je jednoduchý prijímač GPS, ktorý prenáša pomocou rádiového prenosu prenáša tieto informácie na siet', ktorá je pripojená k internetu, takže informácie sú dostupné

všade na zemi. Sieť môže byť bud' pozemná (napríklad amatérske APRS stanice) alebo sieť založená na satelitných systémoch (napr. systém Iridium).

Dostupné možnosti prenosu údajov:

A. Telefónny tracker

Nie veľmi vhodná voľba, ktorá obsahuje množstvo nevýhod. Telefóny nie sú stavané na také nízke teploty, aké sa vyskytujú v takých nadmorských výškach, kam ideme balón vypustiť. Ďalšou nevýhodou je to, že servisný modul s telefónom môže dopadnúť v oblasti, kde nie je pokrytie telefónnej siete, tým pádom sa neprenesú údaje o polohe servisného modulu.

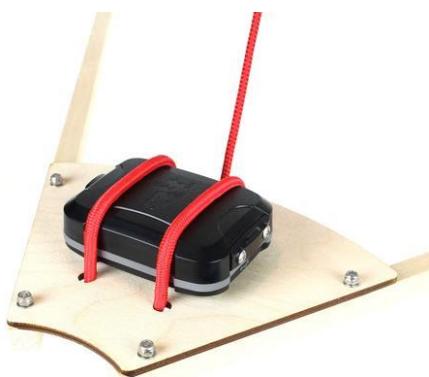
B. Satelitný tracker

Vhodnejšia voľba ako telefónny tracker. Na rozdiel od telefónov, satelitné trackery sa spoliehajú iba na satelitné siete, ktoré potom posielajú informácie ďalej na zem. Nevýhodou je, že anténa tohto trackeru musí byť vždy nasmerovaná na oblohu.

C. Rádiový tracker

V našom projekte zostavíme vlastný tracker pozostávajúci s GPS shieldu pre Arduino a rádiového prijímača a vysielača.

Jednotlivé modely rádiových vysielačov a prijímačov budú uvedené v neskoršej kapitole.



Obrázok 13 Rádiový tracker

3.1.8 Vypúšťací ventil na postupné znižovanie tlaku v stratosférickom balóne

Kedže so stúpajúcou výškou klesá tlak vzduchu, plyn v stratosférickom balóne sa rozpína, čím balón zväčšuje svoj objem až do bodu, kedy nepraskne. Aby sme s balónom dosiahli lepšie výsledky čo sa týka najvyššej dosiahnutej nadmorskej výšky, navrhli sme, aby

náš stratosférický balón používal systém na postupné vypúšťanie plynu pomocou na diaľku ovládaného ventilu. Tým sa zníží tlak v balóne a spolu s ním aj rýchlosť stúpania.

Takéto ventily môžu byť ovládané jednoduchým elektromagnetom. Bežne sú dostupné v dvoch variantoch – normálne zatvorené a normálne otvorené. Normálne zatvorené ventily sú zatvorené a držia tekutinu resp. vzduch v ventile, až kým elektromagnet nedosiahne požadovaný prúd a napätie, kedy sa ventil otvorí a pustí tekutinu ďalej. Normálne otvorený ventil je opakom normálne zatvoreného. Takýto ventil býva väčšinou ovládaný Arduinom.

Pre začiatok použijeme namiesto vypúšťacieho ventilu servomotor.



Obrázok 14 Vypúšťací ventil

3.2 Server

Zo špecifikácie nášho projektu vzniká potreba mať miesto, ktoré je dostupné z internetu. Na tomto mieste budú uložené dátá, ku ktorým budú mať prístup používatelia v rôznych roliach a externé časti nášho systému.

K serveru bude pristupovať náš zákazník, ktorý bude mať prístup k dátam, ktoré sú v reálnom čase prijímané zo sondy, odosielané a ukladané v databáze. Na našej webovej stránke bude verejnosti prístupná mapa s trasou letu balóna a základnou telemetriou, ktorá s aktualizuje v reálnom čase. Webová stránka bude slúžiť na propagáciu letu balóna a jeho lokalizáciu.

Dáta bude používať modul/podsystém na zdieľanie informácií n asociálnych sietiach, ktorý bude zverejňovať aktuálne informácie o balóne ako polohu, výšku balóna, teplotu okolia a podobne. Dáta na server bude cez aplikačné rozhranie odosielať riadiaci počítač letu, ktorý komunikuje priamo zo sondou pomocou GSM, rádia a iných technológií.

Je vhodné aby tieto funkcie boli združené na jednom mieste. Pre našu funkcionalitu musí server poskytovať webový server, databázové úložisko a aplikačný server. K tomu nám bude slúžiť server poskytnutý našou fakultou. Webový server zabezpečuje Apache HTTP server.

3.2.1 Služby na správu zdrojového kódu

Server poskytnutý fakultou sme sa rozhodli využiť naplno a okrem webového serveru Apache na ňom prevádzkujeme web tímového projektu a web na propagáciu letu balóna a jeho lokalizáciu. Obe tieto webové stránky pravidelne aktualizujeme, preto sme sa rozhodli používať verziovanie zdrojového kódu pomocou webovej služby GitHub a službu na kontinuálnu integráciu Jenkins.

Vždy, keď do jednej zo zmienených webových stránok pridáme nejakú funkcionalitu, každú verziu zmien ukladáme do určeného verejného repozitára s kódom. Vďaka tomuto postupu máme prehľad o vykonaných zmenách a ich prípadné vrátenie je jednoduchšie. Dokopy používame štyri github repozitáre:

1. Pre webovú stránku tímového projektu
2. Pre webovú stránku s aktuálnymi informáciami o balóne
3. 2 repozitáre pre kód určený pre mikropočítače

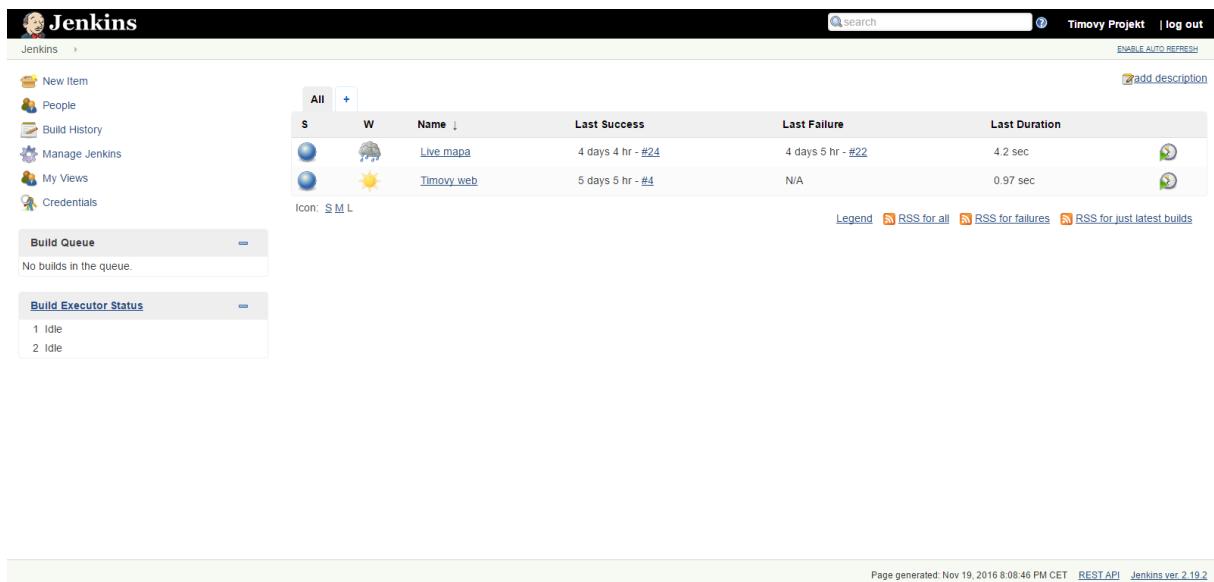
Prácu so zdrojovým kódom uľahčujú intuitívne príkazy služby git ako napríklad:

- `git add .`

- označenie aktuálneho priečinka ako „git repozitár“, v tomto priečinku sa budú sledovať zmeny zdrojového kódu
- git commit -m „sprava“
 - pomocou príkazu git commit sa pridajú zmeny do lokálneho repozitáru. Pomocou prepínaču „m“ dokážeme ku commitu pridať tzv. commit message, ktorý by mal všeobecne špecifikovať cieľ resp. obsah aktuálneho commitu
- git push
 - pomocou príkazu git push sa zmeny, ktoré boli vykonané v lokálnom repozitári, pošlú priamo na vzdialený server. K tomuto príkazu sa bežne pripája aj vetva zdrojového kódu, do ktorej ideme zmeny „pushnúť“, zvyčajne ide o hlavnú „master“ vetvu.
- git pull
 - pomocou príkazu git pull je možné do lokálneho repozitáru uložiť zdrojový kód zo vzdialého repozitáru

Ďalší prístup ku správe zdrojového kódu, ktorý sme sa rozhodli použiť je kontinuálna integrácia. Princípom tohto prístupu je priebežné sledovanie zmien zdrojového kódu, z ktorého je automaticky zostavovaný a testovaný projekt. [21]

Pre implementáciu tohto prístupu sme zvolili existujúci nástroj Jenkins, ktorý považujeme za najvhodnejší pre nás projekt. Na nasledujúcim obrázku vidíme aplikáčné rozhranie programu Jenkins. Poskytuje integráciu s už spomínanou službou GitHub, takže dokáže zmeny automaticky stiahovať a odosielat do potrebných repozitárov.



Obrázok 15 Grafické rozhranie Jenkins

3.2.2 Aplikačný server

Existuje viacero technológií, ktoré môžeme použiť pre náš server. Existuje viacero možností postavených na rôznych jazykoch. Niektoré frameworky poskytujú robustnú architektúru a komplexnú funkcionality, na druhej strane sú rýchle, jednoduché frameworky, ktoré je možné jednoducho upraviť podľa potrieb.

Zoznam uvažovaných frameworkov:

- Java: Spring, Stripes
- Python: Django, Flask, Pyramid
- Ruby: Ruby on Rails
- JavaScript: Node.js
- PHP: TYPO3, Yii, CodeIgniter

Zvolili sme si webový framework Flask postavený na jazyku Python. Tento framework je jednoduchý, nenáročný. Neobsahuje mnoho funkcionality, no je možné ho rozšíriť cez poskytované rozšírenia. Vyberali sme tak tiež podľa predchádzajúcich skúseností nášho tímu.

3.2.3 Databázové úložisko

V databáze budú uložené údaje, ktoré sú zaznamenávané počas letu balóna. Na základe povahy a štruktúry dát sme zvolil relačnú SQL databázu. Keďže frekvencia a početnosť dát nie je príliš veľká, zvolili sme databázu SQLite, ktorú je možné jednoducho implementovať do

našej aplikácie. Pokiaľ sa vyskytne neočakávaná potreba siahnuť po výkonnejšom databázovom systéme, môžeme zvoliť MySQL alebo PostgreSQL, ktoré sú podporované frameworkom Flask. S týmto variantom je vhodné počítať pri návrhu architektúry systému.

3.3 Návrh

3.3.1 Architektúra servera

Architektúra servera využíva princípy viacvrstvovej architektúry. Jednotlivé vrstvy majú definovanú funkciu a sú izolované v možnosti komunikovať len s najbližšími vrstvami systému.

Náš systém obsahuje tri základné vrstvy:

a. Vrstva riadenia

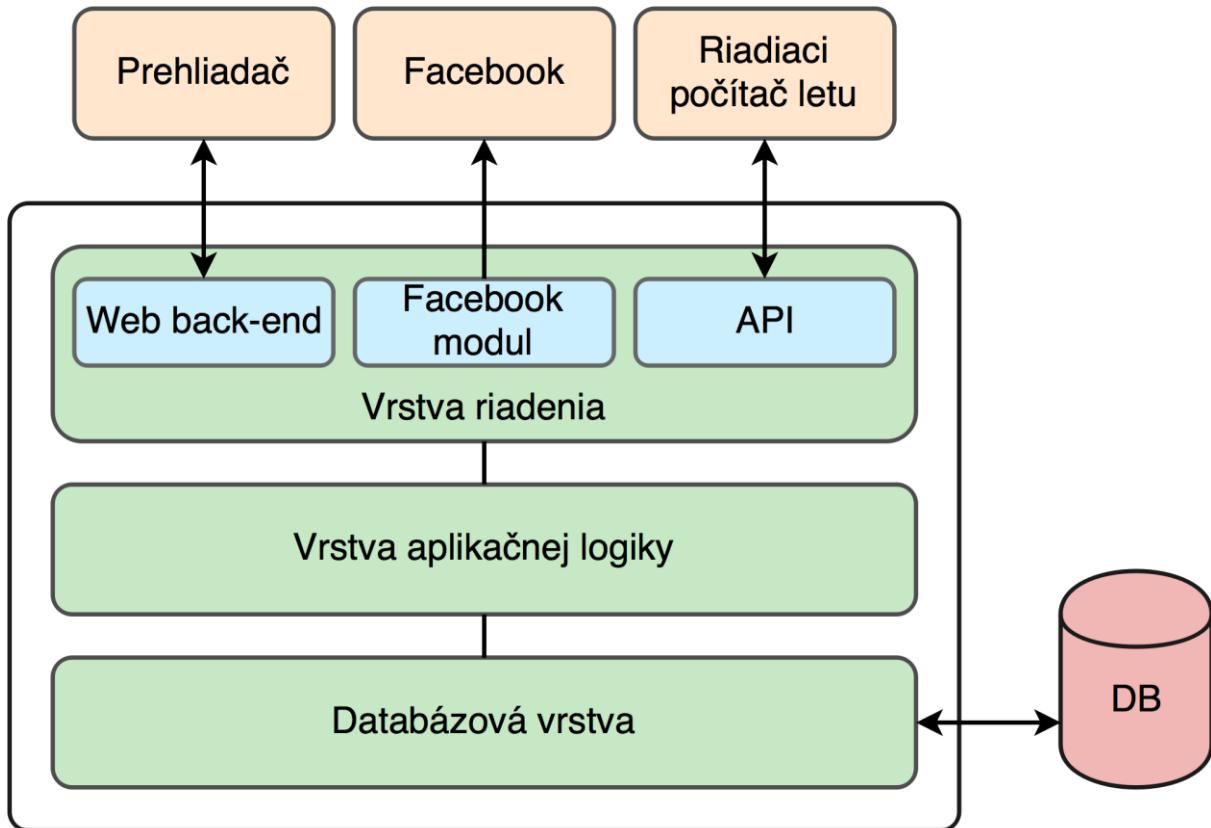
Vrstva riadenia obsahuje logiku, ktorá riadi prístup externých častí nášho systému. Stará sa o zobrazenie webovej stránky s trasou balóna v reálnom čase a odosielanie aktuálne pozície v reálnom čase do klientského prehliadača. Ďalej obsahuje modul, ktorý aktualizuje príspevky na sociálnych sieťach. Poslednou časťou je aplikačné rozhranie, ktoré využíva riadiaci počítač letu pre odosielanie údajov zo sondy. Poslednou časťou je používateľské rozhranie pre zákazníka, v ktorom si môže prezerať všetky údaje prijaté zo letu jeho sondy a možnosť ich exportovať.

b. Vrstva aplikačnej logiky

Vrstva aplikačnej logiky obsahuje hlavnú logiku nášho servera. Zabezpečuje komunikáciu jednotlivých modulov vrstvy riadenia s databázovou vrstvou.

c. Vrstva databázy

Databázová vrstva obsahuje logiku pre prístup k databáze. Databázový server je tak v prípade neočakávanej potreby väčšieho výkonu, spomenutej v časti Analýza, možné jednoducho vymeniť v tejto vrstve.



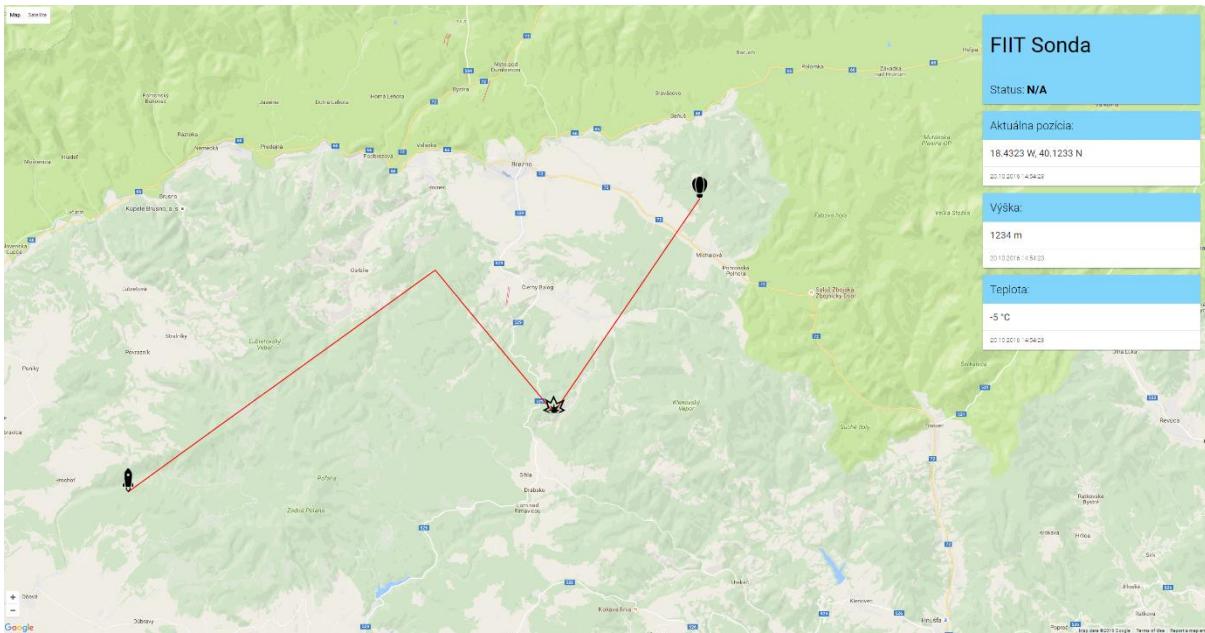
Obrázok 16 Navrhovaná viacvrstvová architektúra servera

3.3.2 Webová stránka

Hlavnou funkciou webovej stránky je zobrazovať na mape trasu letu balóna so základnými informáciami o stave. Webová stránka obsahuje dve hlavné časti: mapa a bočný panel.

Použité sú mapové podklady od spoločnosti Google, ktorá poskytuje možnosť zobraziť rozhranie s mapou na našej webovej stránke. Google takisto poskytuje aplikačné rozhranie, cez ktoré je možné do mapy pridávať vlastné prvky. Do mapy tak môžeme pridať body týkajúce sa letu balóna a čiary znázorňujúce trasu letu balóna.

Pretože pozícia balóna a ostatné údaje sa aktualizujú v reálnom čase, resp. hned' po validovaní hodnôt prijatých zo sondy, je potrebné aby si prehliadač klienta s webovou stránkou udržoval spojenie s našim serverom. Túto komunikáciu zabezpečuje protokol WebSocket, cez ktorý server odosiela prehliadaču aktuálne dátá, ktoré sú následne aktualizované na webovej stránke.



Obrázok 17 Navrhovaná podoba webovej stránky

3.3.3 Návrh komunikácie – GSM

Obvod s názvom A7 GPRS / GSM / GPS, je obvod, ktorý v našom servisnom module používame na komunikáciu s modulom pomocou mobilnej siete. Obvod používa najnovší GPRS / GPS / GSM čip A7. Obvod podporuje siete GSM / GPRS Quad-band (850/900/1800/1900). Tiež podporuje hlasové hovory, SMS, GPRS dátové služby a funkcie GPS. Obvod môže byť použitý na vytvorenie jednoduchého telefónu.

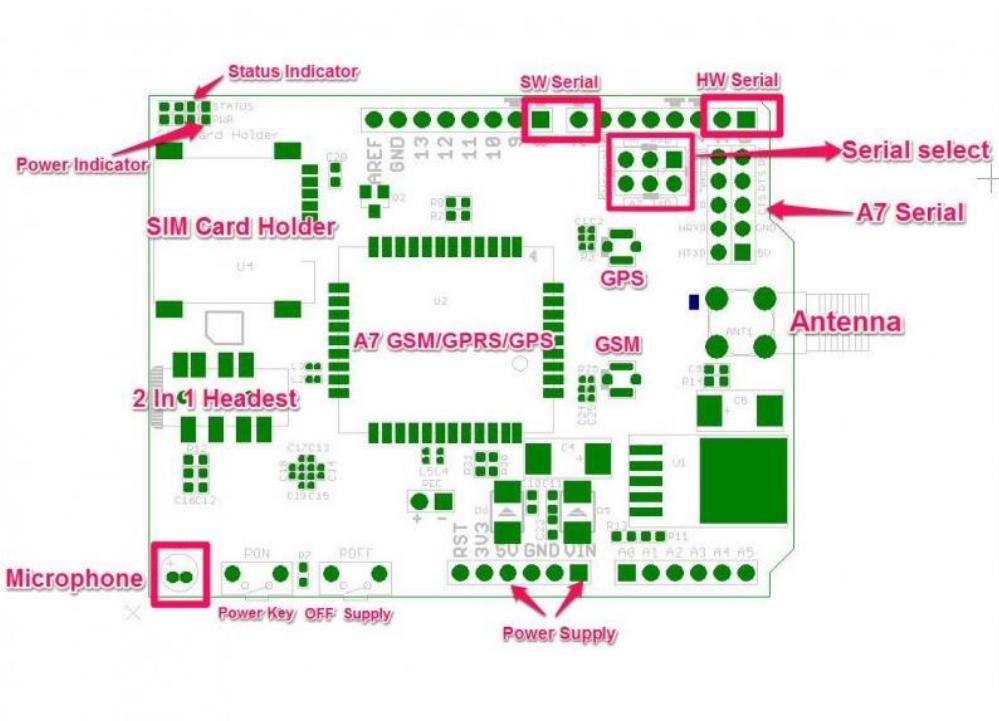
Modul je riadený AT príkazmi cez UART a podporuje 3,3V a 4,2V logické úrovne.

a. Technická špecifikácia

- Operating temperature -30 °C to + 80 °C;
- 1KG peak suction
- Low standby current
- Operating Voltage 3.3V-4.2V;
- Power voltage > 3.4V;
- Standby average current 3mA less;
- Support the GSM / GPRS four bands, including 850,900,1800,1900MHZ;
- Support China Mobile and China Unicom's 2G GSM network worldwide;
- GPRS Class 10;
- Sensitivity <-105;
- Support voice calls;
- Support SMS text messaging;
- Support GPRS data traffic, the maximum data rate, download 85.6Kbps, upload 42.8Kbps;
- Supports standard GSM07.07,07.05 AT commands and extended commands Ai-Thinker;
- Supports two serial ports, a serial port to download an AT command port;
- AT command supports the standard AT and TCP / IP command interface;
- Support digital audio and analog audio support for HR, FR, EFR, AMR speech coding;

- Support ROHS, FCC, CE, CTA certification;
- SMT 42PIN

b. Schéma:



Obrázok 18 Schéma GSM modulu

Power supply – Napájanie 5-9VDC z externého zdroja

Antenna interface – externá GPS a GSM anténa

Serial port select – Výber HW alebo SW sériového portu pomocou jumperov.

Hardware Serial - D0/D1 na Arduino/Crowduino

Software serial - D7/D8 na Arduino/Crowduino

Microphone – na uskutočnenie hovoru

Speaker - na uskutočnenie hovoru

Power key – zapínanie tlačidlo

OFF key – vypínacie tlačidlo

c. Využitie pinov na pripojenom Arduine

D0 - Nepoužitý v prípade použitia HW sériového portu

D1 – Nepoužitý v prípade použitia HW sériového portu

D2 - Nepoužitý

D3 - Nepoužitý

D4 - Nepoužitý

D5 - Nepoužitý

D6 - Nepoužitý

D7 – SW sériový port

D8 - SW sériový port

D9 – Využitý na SW ovládanie napájanie GSM shieldu

D10 - Nepoužitý

D11 - Nepoužitý

D12 - Nepoužitý

D13 - Nepoužitý

d. AT príkazy

Jednotlivé funkcie telefónu sa ovládajú zadávaním príslušných AT príkazov. Pre siet' GSM, európsky telekomunikačný a štandardizačný inštitút (ETSI) stanovil špecifické profily AT príkazov v norme (GSM 07.07). AT príkazy posielané do telefónu môžu byť zadávané v troch podobách:

- a) test AT príkazu, (či telefón príkazu rozumie) je AT+ príkaz =? CR
- b) načítanie nastavených hodnôt z telefónu AT+ príkaz ? CR
- c) zápis dát alebo hodnôt do telefónu AT+ príkaz = parameter CR

Skratka AT je začiatok príkazu, doplnenie podľa požadovaného povelu, sa zadáva iba v prípade, ak to požaduje príkaz pre nastavenie alebo zápis dát a je potvrdenie príkazu. Pri komunikácii z počítača je to ENTER a pri komunikácii z mikrokontroléra hodnota 0Dh.

Najjednoduchším AT príkazom je samotná dvojica znakov AT. Odpoveď mobilného telefónu na správne zadaný príkaz je OK. Zle zadané príkazy telefón ignoruje. Ak sú však v príkaze zadané nesprávne parametre, telefón odpovie ERROR. Pri tvorbe GSM komunikátora využívame najmä AT príkazy pre prácu s textovými správami sms.

e. Práca s textovými správami SMS

SMS správy v telefóne sa delia do troch skupín. Každej skupine môže byť priradený niektorý z pamäťových priestorov. Počet a kapacita pamäťových priestorov sú však dané typom telefónu. Každej z troch skupín sú priradené určité operácie pre prácu s textovými správami. Prvej skupine sú priradené operácie na čítanie a mazanie správ. Druhá skupina obsahuje operácie zapisovania a odosielania správ. Do tretej skupiny sú ukladané prijaté správy.

f. Odoslanie textovej správy

Odoslanie textovej správy v našom prípade prebieha pomocou sekvencie konkrétnych AT príkazov. Konkrétnie ide o príkazy AT+CMGF a AT+CMGS. AT+CMGF slúži na nastavenie módu posielania textových správ a AT+CMGS slúži na zvolenie telefónneho čísla na ktoré sa má správa odoslať. Každý AT príkaz musí byť potvrdený znakom CR, na čo GSM shield odpovedá správou „OK“. Na ukončenie odosielaného textu slúži znak CTRL+Z.

Konkrétna nami zostrojená funkcia na odoslanie SMS teda vyzerá nasledovne:

```
void SendSMS(String str){  
    swSerial.print("AT+CMGF=1"); //Odoslanie sms v textovom móde  
    delay(100);  
    swSerial.print(char(13)); //CR  
    delay(100);  
    swSerial.print("AT+CMGS=+421902744909;");  
    delay(100);  
    swSerial.print(char(13)); //CR  
    delay(100);  
    swSerial.print(str); //Samotný text správy  
    delay(100);  
    swSerial.print(char(26)); //CTRL+Z, ukončenie odosielania  
    delay(100);  
    swSerial.println();  
}
```

g. Prehľad niektorých AT príkazov podporovaných GSM shieldom

AT + CGATT = 1	Return OK, attached to the network
AT + CGACT = 1	Activate the network, then you can use the tcp/ip command
AT + CIPSTART = "TCP" "121.41.97.28", 60000	TCPIP server connection
AT + CIPCLOSE	Close TCP/IP connection
AT+CMGF=1	Send a text message
AT+CIPTCFG	Passthrough mode configuration
AT+CIPTMODE	Enter the passthrough mode

AT+GPS=1	Open GPS
AT+GPS=0	Close GPS
AT+AGPS=1	Open AGPS
AT+AGPS=0	Close AGPS

3.4 Implementácia 1. prototypu

3.4.1 Zaznamenávacia časť

V prvom prototype využívame viacero senzorov pomocou, ktorých meriame/počítame rôzne, pre nás dôležité fyzikálne veličiny. Tieto veličiny sú čas, teplota (2-ma senzormi), tlak a GPS súradnice. Ako sme písali vyššie v analýze na počítanie času aktuálne využívame 16MHz kryštál nachádzajúci sa na prototypovacej platforme Arduino UNO, ktorý kolaboruje s mikročipom ATMega328p. Pomocou neho vieme jednoducho časovať volané procedúry a na tomto časovaní je aj založená architektúra firmvéru (programu), na ktorom stále pracujeme. Časť programu sa nachádza pod odsekom.

```

void Time_function()
{
    timer2++;      // spocitavanie polperiod
}
void setup()
{
    Serial.begin(9600); // inicializacia seriovej linky
    Timer1.initialize(500000); // inicializacia casovaca 1 a nastavenie periody
    Timer1.attachInterrupt(Time_function); // volanie procedury pri pretečení nastaenej hodnoty
    bmp.begin(); // inicializacia bmp senzoru
}
void loop()
{
    timer = timer2 / 2; // pocitanie celych sekund
    if ((timer != oldTime) || firstTime) // podmienka vstupu do hlavnej casti programu
    {

```

```

firstTime = false;
oldTime = timer;
temp = kty(0); // volanie funkcie na vypocet teploty z analogoveho portu 0
bmp.getEvent(&event); // odcitanie hodnot zo senzorov
bmp.getTemperature(&temp2); //zapisanie teploty do premennej temp2
Print_function(); // volanie funkcie na vypis
}
}

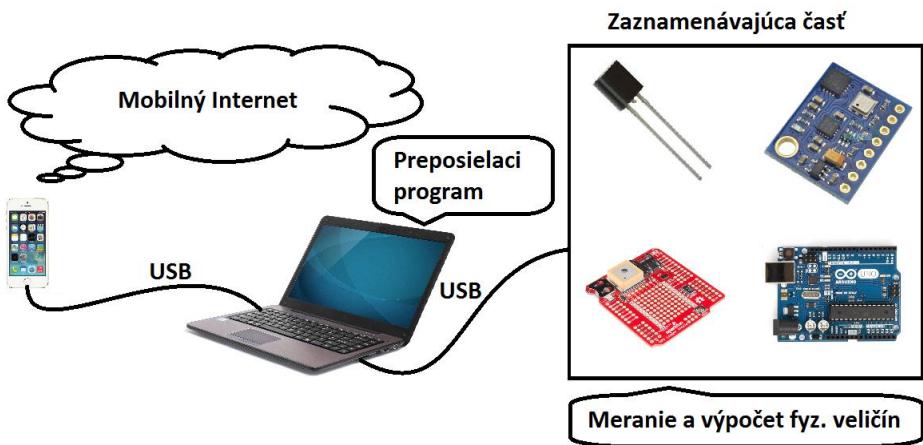
```

Na meranie vonkajšej teploty využívame termistor KTY81-120, pretože dokáže pracovať v podmienkach, ktoré sú pre nás kľúčové. Na meranie vnútornej teploty využívame teplotný senzor, ktorý sa nachádza na multi-senzore 10DOF IMU. Ide o čip BMP180, okrem merania teploty dokáže zaznamenávať aj tlak. Po dôkladnom hľadaní senzoru tlaku s vhodným meracím rozpätím sme nenašli žiadnený, ktorý by dokázal merať tlak vo výške nad 10km. Avšak na druhú stranu aj hodnoty do 10km môžu byť zaujímavé a preto aj túto fyzikálnu veličinu pomocou tohto multi-senzoru meriame.

V neposlednom rade meriame polohu pomocou GPS senzoru na Arduino štite od spoločnosti Sparkfun. Tento senzor nám z analýzy vyšiel ako najlepší avšak v ďalšej iterácii plánujeme do nášho servisného modulu zahrnúť ešte jeden sekundárny GPS modul (kvôli nepredvídateľným situáciám, ktoré by mohli nastat').

3.4.2 Preposielacia časť

V prvom prototype sme si model celého servisného modulu čiastočne zjednodušili, keďže sme vypustili komunikáciu cez rádiové spojenie. V skratke náš model je vyjadrený v obrázku pod odstavcom.



Obrázok 19 Zjednodušený model zaznamenávacej a preposielcej časti

Údaje sa namerajú a predspracujú na mikrokontoléry Arduino UNO. Následne sa pošlú na sériovú (USB) linku. Na tejto linke počítač a len čo zachytí odoslané dátu, spracuje ich do vopred dohodnutého formátu JSON. Následne pomocou knižnice „requests“ využitím metódy HTTP POST dát odošleme v správnom formáte na server, kde sa následne spracujú, respektívne zobrazia na našej stránke v mapke.

Pod odstavcom sa nachádza časť programu v jazyku Python, ktorý sa nachádza na počítači a prijaté dátá spracováva a odosiela na server:

```

while 1: // nekonecny cyklus pocuvania
    try:
        line=(ser.readline().decode("utf8"))           // pocuvanie udajov v spravnom formate
        if (line != ""):
            print(line)      // vypis prijateho riadku
            line_parsed = line.split(",")          // parsovanie prijateho suboru
            update['type'] = "updateMsg"         // vyskladanie struktury vhodnej pre json
            update['data']['time'] = line_parsed[0]
            update['data'][['location'][['x']] = line_parsed[4]
            update['data'][['location'][['y']] = line_parsed[5]
            update['data'][['location'][['z']] = line_parsed[6]
            update['data'][['temperature'][['in']] = line_parsed[3]
            update['data'][['temperature'][['out']] = line_parsed[2]
            update['data'][['isburst']] = line_parsed[7]
            json_data = json.dumps(update)     // parsovanie struktury do formatu JSON
            response = requests.post("http://posttestserver.com/post.php", json=json_data) // odosielanie na server

```

```

except ser.SerialTimeoutException:
    print('Data could not be read')
    time.sleep(0.1)

```

3.4.3 Backup pamäťový modul

Počas implementácie nášho 2.-hého prototypu sme sa stretli s otázkou zvýšenej spoločalivosti, ktorá je klúčovým bodom nášho servisného modulu. Údaje čo meriame sú základom nášho projektu (úspešného letu) a teda je dôležité ich ukladať na čo najviac miest.

Všetky údaje čo pošleme ukladáme do databázy nášho servera, avšak pokial' by nastala situácia, že by sa dátu nepodarilo odoslať alebo prijať, nemali by sme čo uložiť. Najjednoduchšou možnosťou ako predchádzať tomuto riziku je dátu po nameraní okamžite ukladať na nejaké médium. Rozhodli sme sa pre pamäťovú kartu, vzhľadom na možnosti, ktoré ponúka (veľká kapacita, nízke rozmer, odolnosť voči podmienkam a iné...).

Náš GPS Logger štít od spoločnosti Sparkfun obsahuje slot aj riadiacu jednotku k pamäťovej karte, avšak pri implementácii sa nám vyskytol závažný problém. Tento modul obsahuje zdieľanú operačnú jednotku aj pre GPS aj pre pamäťovú kartu. V praxi to znamená, že pokial' využívame štít pre meranie GPS súradníc, nie je možné v tom istom čase využívať aj pamäťový modul. Tento problém je možné vyriešiť nejakým softvérovým zámkom, avšak inicializácia GPS modulu zabere relatívne dlhý čas a preto je pre nás táto metóda aktuálne nepoužiteľná. V budúcnu však túto metódu opäť prehodnotíme a možno sa nakoniec knej vrátime.

Rozhodli sme sa preto pre externý pamäťový modul, ktorý náš problém vyriešil. Pred každým odosielaním, sa všetky dátu uložia na pamäťovú kartu, ktorá sa hned' po uložení dát zavrie a najbližšie sa otvorí až pri ďalšom zápisе. Vďaka tejto metóde neprídeme o žiadne dátu ani pri výpadku spojenia medzi modulom a serverom. Pod textom sa nachádza časť programu, ktorá zabezpečuje ukladanie dát na pamäťovú kartu a taktiež obrázok aktuálneho prototypu.

```

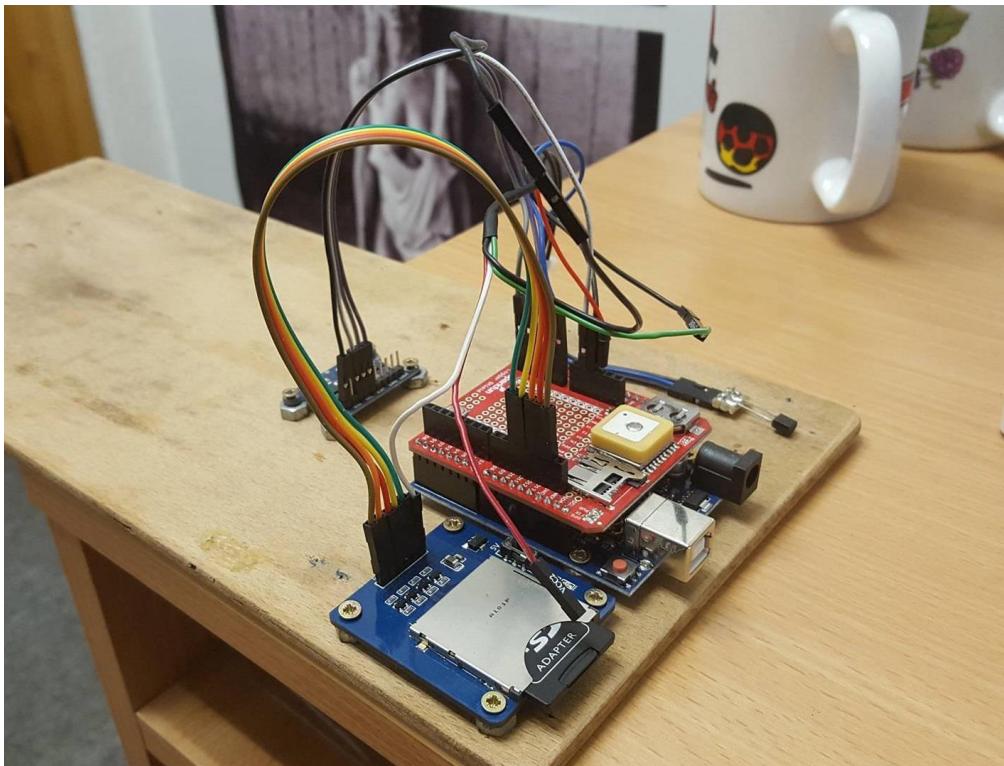
void logGPSData()
{
    createFile("test.txt");
    writeToFileFloat(timer);
    writeToFileFloat(aktLng);
    writeToFileFloat(aktLat);
    writeToFileFloat(aktAlt);
    writeToFileFloat(temp);
}

```

```

writeToFileFloat(temp2);
writeToFileFloat(event.pressure);
writeToFileFloat(urtime);
writeToFileFloat(tinyGPS.satellites.value());
file.println("\n");
closeFile();
}

```



Obrázok 20 Prototyp obsahujúci backup pamäťový modul

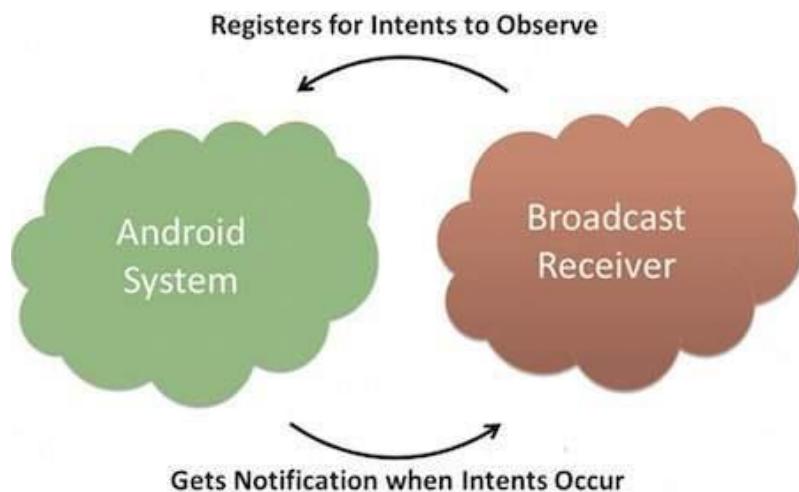
3.4.4 Prijímanie SMS - problém

Počas analýzy sme identifikovali riziko súvisiace s rádiovým spojením servisného modulu s naším prijímačom vo forme antény pripojenej k notebooku. Takéto spojenie nie je úplne spoľahlivé a hrozí výpadok spojenia čo má za následok absenciu dát v ďalších softvérových moduloch v našom projekte. Dopad takéhoto rizika vieme redukovať aplikovaním redundantného spôsobu odosielania GPS súradníc zo servisného modulu určeného pre dohľadanie a sledovanie pohybu balóna v atmosfére. Tento spôsob zahrňa pravidelné odosielanie SMS správ pomocou pridaného hardvérového GSM modulu do mikropočítača Arduino. Analýzou sme však narazili na problém prijatia SMS správy s GPS súradnicami a ich uloženie do databázy na serveri. Existujú viaceré spôsoby preposlania SMS

správy do notebooku avšak nepodarilo sa nám nájsť vhodný softvér, ktorý by tieto SMS jednoduchým spôsobom preposal formou HTTP requestu na náš hlavný server pomocou API implementovaného na serveri. Rozhodli sme sa tento problém vyriešiť vytvorením vlastnej jednoduchej Android aplikácie, ktorá prijatú SMS prepošle formou HTTP requestu na server.

3.4.5 Prijatie SMS správy v Android aplikácií

Pre prijatie SMS správy v aplikácii sme použili Android API, konkrétnie triedu BroadcastReceiver.



Obrázok 21 Príjanie SMS v android systéme pomocou broadcast príjmačov

```
public class SmsReceiver extends BroadcastReceiver {  
    private Bundle bundle;  
    private SmsMessage currentSMS;  
    private String message;  
    final String DELIMITER = ",";  
  
    @Override  
    public void onReceive(Context context, Intent intent) {  
        if (intent.getAction().equals("android.provider.Telephony.SMS_RECEIVED")) {  
            bundle = intent.getExtras();  
            if (bundle != null) {  
                Object[] pdu_Objects = (Object[]) bundle.get("pdus");  
                if (pdu_Objects != null) {  
                    for (Object aObject : pdu_Objects) {  
                        if (aObject instanceof SmsMessage) {  
                            currentSMS = (SmsMessage) aObject;  
                            message = currentSMS.getText();  
                        }  
                    }  
                }  
            }  
        }  
    }  
}
```

```
currentSMS = getIncomingMessage(aObject, bundle);
String senderNo = currentSMS.getDisplayOriginatingAddress();
message = currentSMS.getDisplayMessageBody();
```

Po prijatí SMS správy je nutné správu rozparsovať z csv formátu do formátu JSON. Pre vytvorenie JSON premennej sme použili json java knižnicu.

3.4.6 Odoslanie HTTP requestu v Android aplikácií

Pre odoslanie HTTP requestov sme použili knižnicu Volley. Táto sieťová knižnica pre Android aplikácie umožňuje odoslanie HTTP requestov jednoduchým spôsobom v čo najkratšom čase.

Výhody Volley knižnice

- Automatické plánovanie sietových požiadaviek
- Viacero súčasných sietových spojení
- Určenie priority pre požiadavky
- API prístup pre zrušenie odosielania
- Možnosť ošetrenia v prípade nedostupného servera

Volley je vhodnou knižnicou pre viaceré typy RPC (angl. Remote Procedure Call) operácií. Pri odosielaní je možné odosielať nielen reťazce znakov ale napr. aj obrázky a pre nás dôležité JSON objekty. Volley nie je vhodným riešením pri odosielaní veľkých súborov a pri veľkých prudových operáciách. Jadro tejto knižnice je vyvíjané v slobodnom repozitárii od spoločnosti Google.

Pre pridanie knižnice do projektového gradle súboru pridať závislosť na Volley zdrojové kódy.

```
dependencies {
    ...
    compile 'com.android.volley:volley:1.0.0'
}
```

V našom projekte stačí pomocou Volley knižnice odoslať pripravený JSON objekt na príslušné API na hlavnom serveri. Odpoved'ou serveru vieme overiť správne prijatie JSON objektu na serveri. Použitím návrhového vzoru observer je používateľ oboznámený o návratovom kóde serveru formou Toast notifikácie.

```

StringRequest stringRequest = new StringRequest(Request.Method.POST, URL, new
Response.Listener<String>() {
    @Override
    public void onResponse(String response) {
        Log.i("VOLLEY", response);
        ObservableObject.getInstance().updateValue(response);
    }
}

```

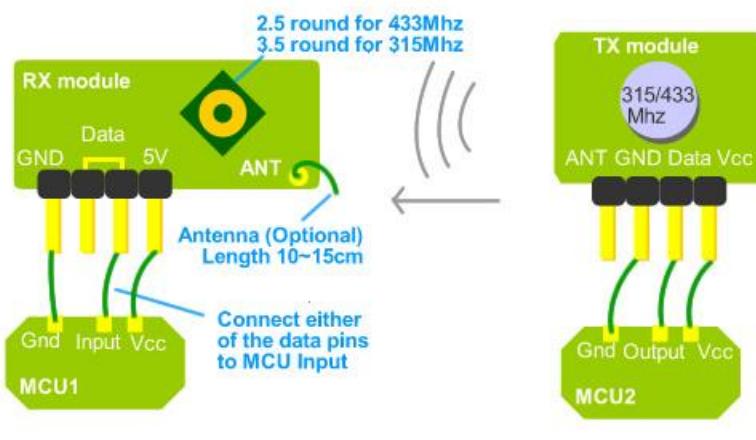
3.4.7 Majáčik na dohľadanie

Z dôvodu nedokonalej presnosti GPS súradníc, sme nútení vyriešiť alternatívne riešenie dohľadania balóna, resp. jeho presnejšie zameranie. Problém môže vzniknúť jednoducho napr. pri pade do hrachového poľa, kde aj pár metrov môže človeka pri hľadaní značne zmiast'. Práve preto v našom servisnom module implementujeme jednoduchý majáčik, ktorý pomocou smerovej antény bude jednoduchšie zamerat'.

Pri riešení tejto problematiky vychádzame hlavne z dostupných aktuálnych riešení, vyskytujúcich sa najmä v rádioamatérskych orientačných behoch. Rádioamatérsky orientačný beh je pretekársky šport, ktorý kombinuje určovanie smeru pomocou smerovej antény so schopnosťami orientovať sa v teréne. Cieľom je nájsť postupne 5 rôznych rádiových vysielačov v čo najkratšom čase. Toto by sme chceli využiť v našom servisnom module. K riešeniu teda potrebujeme tak ako v spomenutom športe drobný vysielač, ktorý sa primontuje k servisnému modulu a rádio prijímač, pomocou ktorého nás modul dohľadáme [22].

V našom prvom riešení využívame RF transmitting module 433Mhz spolu s dvomi Arduinami.

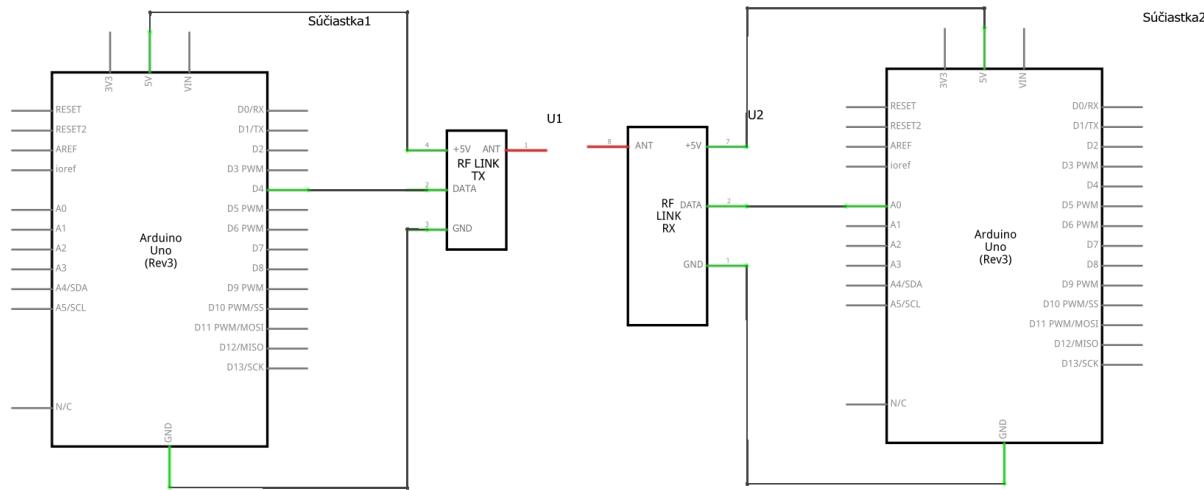
Špecifikácia RF modulu [23]:



Obrázok 22 Schéma RF transmitting module 433Mhz [23]

- Frekvencia: 433Mhz
 - Modulácia: ASK
 - Výstup príjmača: High - 1/2 Vcc, Low - 0.7v
 - Vstup vysielača: 3-12V (vyššie napätie = lepší dosah)
 - Dosah 40m v budove, 90m na otvorenom priestore

HW implementácia:



S výsledkom však nie sme spokojný, vzhľadom k tomu, že nie je možné dosiahnuť udávaný dosah komunikácie. Oproti 90m s ktorými sme rátali sme schopní komunikovať len na niekoľko centimetrov a teda dohľadanie týmto spôsobom by nebolo možné. Riešenie budeme musieť teda realizovať so silnejšími RF súčiastkami.

3.4.8 Spracúvajúca časť

Server je postavený na frameworku Flask, ktorý umožňuje jednoducho vytvárať webové aplikácie. Jednotlivé súbory sú rozdelené v štruktúre podľa pravidiel frameworku. Vrstvy podľa návrhu systému sú tiež v jednotlivých adresároch.

- /balon
 - o /templates – HTML súbory, ktoré sa zobrazujú na webovej stránke
 - o /static – statické súbory, ktoré sa týkajú webovej stránky ako štýly, JavaScript skripty a knižnice
 - o /controllers – vrstva riadenia
 - o /service – vrstva aplikačnej logiky
 - o /database – vrstva databáz

- /models – modely pre databázu
- main.py – hlavný súbor aplikácie
- config.py – konfiguračný súbor

Pomocou anotácií je možné jednotlivé funkcie mapovať na požadované URL adresy, pri ktorých sa vykonajú.

```
@app.route('/map')
def balloonDashboard():
    ...

@app.route('/admin/flight/<int:flight_id>/detail')
def flight_detail(flight_id):
    ...

@app.route('/api/flight/<int:flight_number>/telemetry', methods=['POST'])
def api_telemetry(flight_number):
```

Server prijíma a spracúva dátu, ktoré sú zo sondy posielané viacerými kanálmi, v našom prípade rádiové spojenie a GSM sieť. Aplikácie, ktoré tieto dátu zo servisného modulu prijímajú ich následne posielaju na webový server cez protokol HTTP. Definované sú dva typy správ, ktoré je možné odoslať na rozdielne URL.

Prvou je priebežné odosielanie parametrov počas letu, ktoré sú následne použité na vykreslovanie trasy letu a ďalších grafov v reálnom čase počas letu.

`/api/flight/<int:flight_number>/telemetry`

Druhá správa obsahuje udalosť, ktorá sa počas letu vyskytne. Jedná sa o štart letu, roztrhnutie balóna, a pristátie.

`/api/flight/<int:flight_number>/event/<string:event>`

HTTP požiadavka obsahuje všetky potrebné dátu vo formáte JSON. Podstatnou časťou sú jednotlivé hodnoty veličín, ktoré boli namerané.

Server umožňuje vytváranie nových letov, ku ktorému sa následne viažu odosielané údaje. Aby nebolo možné voľne pridať údaje k danému letu, je vytvorená zjednodušená autentizácia v podobe hash-u, ktorý musí každá správa obsahovať. Tento hash je vytvorený pri vytvorení letu. Po overení sú údaje uložené do databázy servera a zobrazené na webovej stránke.

Databázová vrstva

Databázová vrstva je najnižšia časť nášho systému, ktorá komunikuje s databázou. V databázovej vrstve je použitý plugin Flask SQLAlchemy, ktorý poskytuje možnosť ukladať dátu do SQLite databázy za využitia vzoru Active Record. Táto vrstva zjednodušuje prístup

k dátam a poskytuje funkcie pre jednoduché získavanie dát ako posledné parametre alebo filtrovanie parametrov podľa typu.

```
def saveParameter(parameter):
    db.session.add(parameter)
    db.session.commit()
    return parameter.id

def getParametersByFlight(flight_id)

def getParametersByKeyByFlight(key, value, flight_id)

def getParameterLastByFlight(key, value, flight_id)

def getParameterFirstByFlight(key, value, flight_id)
```

Pre jednotlivé dátá sú vytvorené modely, ktoré využívajú spomínaný Active Record a cez ktoré je možné jednoduché ukladanie a získavanie dát v podobe objektov bez ďalšej transformácie a vytvárania objektov pre použitie v aplikácii.

```
class Parameter(db.Model):
    id = db.Column(db.Integer, primary_key=True)

    type = db.Column(db.String(20))
    source = db.Column(db.String(20))

    valid = db.Column(db.Boolean)
    validated = db.Column(db.Boolean)

    time_received = db.Column(db.DateTime)
    time_created = db.Column(db.DateTime)

    flight_id = db.Column(db.Integer, db.ForeignKey('flight.id'))
    values = db.relationship('Value', backref="parameter", lazy='dynamic')
```

3.4.9 Zobrazovacia časť

Primárnu časťou zobrazovania je naša webová stránka, na ktorej sa nachádza mapa s trasou letu balóna. Na stránke sa zobrazujú údaje uložené v databáze servera. V rámci serverovej časti sme implementovali obsluhu požiadaviek klientskej časti na zobrazenie webovej stránky a odosielanie aktuálnych dát cez WebSocket. Na komunikáciu slúži knižnica Flask-Socket.IO, ktorá zabezpečuje spojenie.

Implementovali sme klientskú časť webovej stránky, ktorá zobrazuje mapu a údaje zo sondy, ktoré sa aktualizujú v reálnom čase. Túto aktualizáciu zabezpečuje na klientskej strane

JavaScript a protokol WebSocket, cez ktorý server posiela klientom aktuálne dátu. Toto spojenie zabezpečuje na klientskej strane knižnica Socket.IO. Po prijatí klient, tieto dátu spracuje a na stránke vykreslí aktuálnu trasu letu balóna a aktualizované hodnoty týkajúce sa letu (poloha balóna, výška, okolitá teplota).

```
socket.on('balloon_update', function(data) {  
    var message = JSON.parse(data);  
    switch(message.type) {  
        case "balloonEvent":  
            handleBalloonEvent(message);  
            break;  
        case "balloonPath":  
            handlePathUpdate(message);  
            break;  
        case "balloonTelemetry":  
            handleTelemetryUpdate(message);  
            break;  
    }  
});
```

Dáta medzi serverom a klientom sa prenášajú vo formáte JSON. Do tohto formátu je jednoduché prevádztať dátu na strane servera (Python) ako aj na strane klienta (JavaScript).

```
{  
    "type": "balloonEvent",  
    "created": 1477866660,  
    "data": {  
        "type": "currentPosition",  
        "eventTime": 1477867645,  
        "point": {  
            "time": 1477867645, // Optional  
            "lat": 49.548258,  
            "lng": 19.162854  
        },  
        "info": {  
            "title": "Reached 10000 m",  
            "text": "We have reached altitude of 10000 meters. YAY!!!!"  
        }  
    }  
}
```

Na zobrazenie mapy využívame mapy od spoločnosti Google, ktorý poskytuje taktiež rozsiahle API, ktorým je možné mapy upraviť podľa potreby. S dostupného API využívame objekty pre značku a čiaru, ktoré na mape znázorňujú trasu balóna a udalosti ako vzlet, roztrhnutie balóna alebo pristátie.

3.5 Propagácia na Facebooku

3.5.1 Prehľad

Stále viac a viac podnikov a spoločností si v dnešnej dobe vytvára vlastné stránky na Facebooku. Takéto stránky slúžia najmä na propagáciu svojich výrobkov a služieb. Taktiež môžu slúžiť na komunikáciu so zákazníkmi alebo na podávanie čerstvých informácií o novinkách týkajúcich sa fungovania spoločnosti alebo inštitúcie. Ak požadujeme vždy aktuálny obsah, veľké množstvo webových portálov a aplikácií sa spolieha na automatické odosielanie obsahu na ich príslušné Facebook stránky. Každý, kto niekedy hral hru na Facebooku alebo dokonca použil mobilnú aplikáciu Facebooku sa už stretol s používateľskými prístupovými tokenmi (angl. user-access tokens). Takéto tokeny umožňujú aplikáciám odosielať obsah pod používateľským účtom daného používateľa. Ale ako sa takýto obsah posiela pomocou aplikácie na konkrétnu facebook stránku? Na tento problém sa používajú prístupové tokeny.

3.5.2 Prístupové tokeny

Prístupové tokeny stránky umožňujú aplikácií vykonávať rôzne akcie na stránke ako keby ju vykonávala samotná stránka a nie ako bežný používateľ facebooku. Správcovia stránky majú možnosť zmeniť kontext účtu pod ktorým pridávajú obsah na vlastnú stránku. Pri následnom prezeraní stránky je daný obsah ktorý bol vytvorený (napr. status, komentár, pripomienka, udalosť...) odoslaný pod menom stránky a nie pod „osobným“ používateľským účtom správcu stránky. Prístupové tokeny stránky umožňujú ľubovoľnej aplikácií vytvárať obsah na stránke.

Tieto tokeny taktiež umožňujú aplikáciám publikovanie obsahu na Facebook stránku automaticky. To znamená že používateľ nemusí nijak zasahovať do procesu publikovania obsahu. Takúto automatizáciu dosiahneme využitím Facebook Graph API (angl. application programming interface). V prípade odosielania obsahu s tokenom pre používateľský účet a nie prístupovým tokenom stránky bude obsah odoslaný pod používateľským účtom používateľa a taktiež vzniká riziko úniku informácií z takéhoto osobného používateľského účtu. Takýto únik môže nastať v prípade ak ma aplikácia nezabezpečený prístup do databázy. Uložením prístupových tokenov sa znižuje pravdepodobnosť, že nastane takáto situácia. Ak by aj takáto situácia nastlala tokeny sú generované vždy len pre činnosť na stránke a na obmedzenú dobu platnosti.

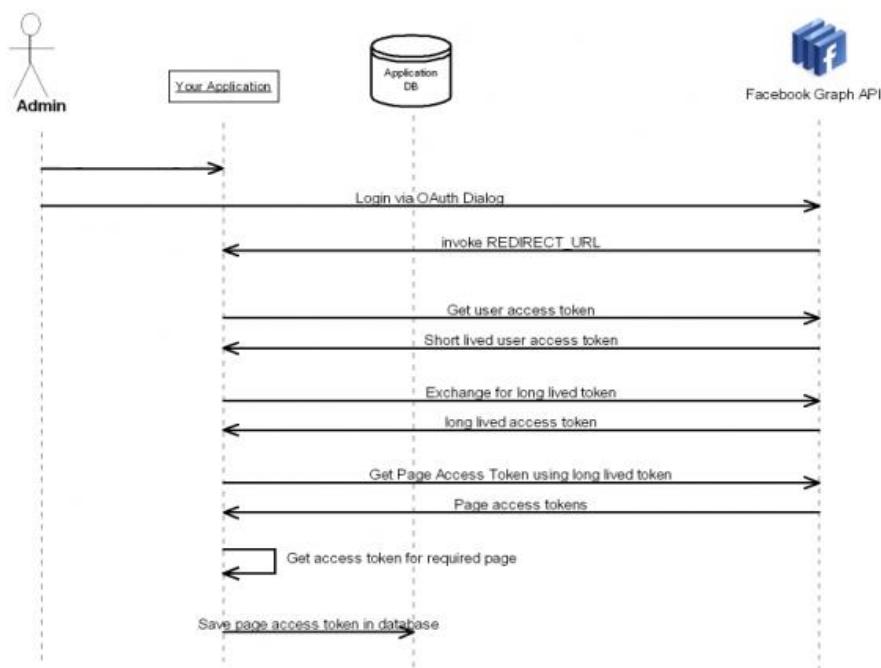
3.5.3 Získanie prístupových Facebook tokenov

Získanie Facebook prístupových tokenov sa skladá zo 4 krokov, ktoré musia byť vykonané v takomto poradí:

1. Zaregistrovanie Facebook aplikácie.
2. Získanie používateľského tokenu Facebook účtu, ktorý ma vhodné práva pre editovanie povolení pre Facebook stránku.
3. Výmena krátko žijúceho tokenu (angl. short lived token) za dlho žijúci token (angl.long lived token).
4. Použitie dlho žijúceho tokenu na získanie prístupového tokenu stránky.

3.5.4 Integrácia v aplikácii

Existuje viacero spôsobov ako implementovať tieto 4 kroky pre získanie prístupového tokenu stránky. Pre začiatok je nutné aby mala aplikácia prístup k Facebook aplikačnému ID (angl. Facebook application ID), aplikačnému tajomstvu (angl. application secret) a ID stránky pre ktorú požadujeme token.



Obrázok 23 Diagram znázorňujúci interakciu medzi adminom, aplikáciou a Facebook Graph API.

Názov grafové API (angl. graph API) je odvodený z idei sociálnych grafov. Takéto grafy reprezentujú rôzne informácie na Facebooku podľa toho z čoho sú zložené. Sú to:

- Uzly (angl. nodes) - jednoducho povedané sú to časti Facebooku ako napr. používateľ, fotka, stránka, komentár

- Hrany (angl. edges) - sú to spojenia medzi uzlami, môžu to byť napr. fotky na stránke alebo komentáre k fotke
 - Polia (angl. fields) - informácie o uzloch, napr. narodeniny používateľa, názov stránky
- Celé grafové API je založené na HTTP, takže s týmto rozhraním dokážeme pracovať v ktoromkoľvek programovacom jazyku pre ktorý je dostupná HTTP knižnica (napr. knižnice cURL, urllib...). Väčšina požiadaviek na Facebook Graph API vyžaduje použitie prístupových tokenov od aplikácie.

3.5.5 Grafový API prieskumník

Najjednoduchší spôsob ako pochopiť funkciu a otestovať Facebook Graph API je použiť Graph API prieskumníka (angl. explorer). Tento nízkoúrovňový nástroj pomáha pri vytváraní požiadaviek, či už na zobrazenie, pridanie alebo odstránenie dát z Facebook databáz.



Obrázok 24 Grafový API prieznamník

3.6 Propagácia na Twitteri

3.6.1 Prehľad

Twitter je sociálna platforma pre komunikáciu medzi ľuďmi pomocou odosielania krátkych textových správ. Tieto správy sa nazývajú tweety (angl. tweets) a sú limitované na maximálnu dĺžku 140 znakov. Táto vlastnosť umožňuje efektívne zobrazenie tweetov aj na mobilných zariadeniach, ktoré nemusia byť považované chytré (angl. smartphones). Taktiež tweety môžu byť odosielané z webových aplikácií, desktopových aplikácií a iných mobilných zariadení. Používatelia väčšinou odosielajú správy o vlastných aktivitách ako napr. kde sa nachádzajú, čo robia, nad čím práve premýšľajú, čo pozerajú atď. V tweetoch je taktiež možné

odoslať odkazy na stránky, obrázky alebo iné tweety. Používatelia Twitteru môžu sledovať obsah iných používateľov a tým si zobrazovať ich obsah na svojej Twitter stránke.

3.6.2 Význam Twitteru

Twitter sa v súčasnosti rýchlo rozrástol na efektívnu komunikáciu ako komunikovať s ostatnými a podávať najčerstvejšie informácie v čo najkratšom čase. Je jasné že Twitter účty niesú určené len pre jednotlivcov ale aj pre organizácie, firmy, skupiny atď. Efektívnosť a dosiahnutie používateľov je pre organizáciu primárnym cieľom. Čím väčší počet používateľov sleduje Twitter účet organizácie tým viac tweetov sa dostáva k používateľom.

Twitter však nie je len o odosielaní tweetov. Monitorovaním zmienok a haštagov (angl. hashtag) môže organizácia zistíť, čo ľudia hovoria o svojich skúsenostach súvisiacich s organizáciou. Reakciou na ľudí podľa ich Twitter používateľských mien docielime účinný spôsob, ako zaujať divákov a vyvolať v nich pocit, že ich názor je akceptovaný. Často sa nájdú ľudia ktorých obsah môže byť považovaný nevhodný a preto je nutné podať stážnosť príslušnému správcovskému oddeleniu Twitteru.

3.6.3 Prístup aplikácie pomocou REST API

Resttovské rozhranie poskytuje programový prístup k čítaniu a zapisovaniu dát k službe Twitter. Vytvorenie nového Twitter účtu, čítanie používateľského profilu a dát sledujúcich ľudí, a mnohé ďalšie. REST API identifikuje Twitter aplikácie a používateľa pomocou OAuth. Odpovede zo servera sú vo formáte JSON. JSON poskytuje štandard pre výmenu dát ktorý sa skladá z atribútov typu kľúč – hodnota.

Pre prístup k Twitter API je, podobne ako ku Facebook Graph API, potrebná autentifikácia. Aplikácie musia autorizovať všetky požiadavky pomocou protokolu OAuth 1.0a prípade je možné použiť autentifikáciu len pre aplikáciu (angl. Application only authentication). Autorizácia dovoľuje Twitteru chrániť dátá používateľov a zabraňuje podozrivému správaniu. Taktiež autorizácia poskytuje informácie pre vývojárske tímu Twitteru pre lepšiu analýzu používania API vývojarmi. Použitím týchto informácií vie Twitter lepšie optimalizovať ich API a rozširovať funkcie samotnej platformy.

Pre vykonávanie oprávnených volaní na Twitter API, aplikácia musí najprv získať prístupový token OAuth s menom používateľa služby Twitter. OAuth je autorizačný protokol, ktorý umožňuje používateľom schválenie požiadavky aplikácie pre žiadosť konáť v ich mene, bez nutnosti zdieľať svoje heslo. Spôsob, akým aplikácie získavajú tokeny a pristupujú k API

bude závisieť na riešenom prípade použitia aplikáciou. Najčastejšie prípady použitia sú znázornené v nasledujúcej tabuľke.

Prípad použitia	Názov prístupového rozhrania
Chceme poskytnúť tlačidlo "Prihlásiť pomocou Twitter účtu" na našej webovej stránke...	Sign in with Twitter
Chceme čítať a posielat Twitter dátá pod menom používateľov na našej webovej stránke...	3-legged OAuth
Máme mobilnú, desktopovú alebo vnorenú aplikáciu ktorá požaduje prístup k webovému prehliadaču...	PIN-based OAuth
Požadujeme prístup k API z nášho vlastného účtu...	Tokens from dev.twitter.com
Potrebuje prístup k výmene používateľských mien/hesiel a naša aplikácia bola schválená pre použitie procesom xAuth...	xAuth
Poskytujeme API ktoré posiela dátá pod menami Twitter používateľov...	OAuth Echo
Chceme posielat autorizované požiadavky pod menom samotnej aplikácie...	Application-only authentication

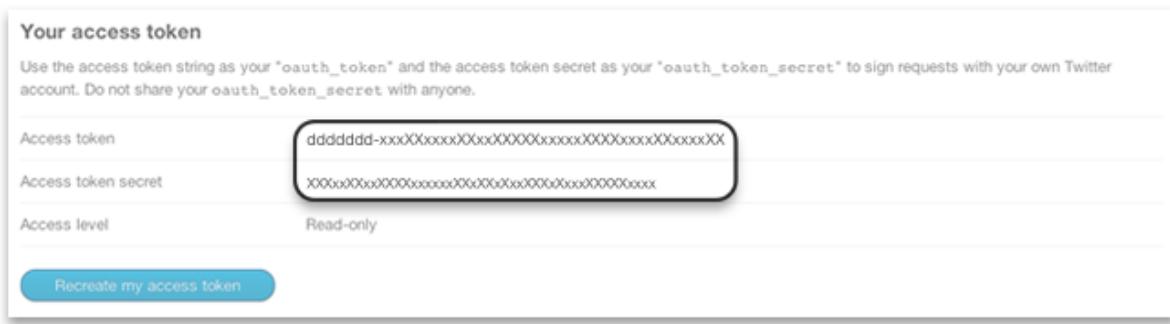
3.6.4 Vytvorenie aplikácie pre Twitter API

Vytvorenie novej aplikácie je možné na stránke apps.twitter.com. Po vytvorení prístupových tokenov a tokenov tajomstiev je možný prístup k Twitter API. Tokeny poskytované Twitterom sú generované ako tokeny bez exspirácie. Token sa však môže stať neplatným ak používateľ explicitne odmietne aplikáciu vo svojich nastaveniach prípadne ak ju Twitter administrátor označí ako suspendovanú z aplikácie. V prípade že je suspendovaná aplikácia, správca aplikácie bude oboznámený poznámkou na svojej aplikačnej správcovskej stránke.

Pri návrhu aplikácie je vhodné myslieť na situáciu, ktorá môže nastať v prípade, že sa prístupový token stane neprístupný. Takáto situácia môže nastať kedykoľvek a používateľ je núteneý sa znova autorizovať. Správne riešenie tohto problému značne odbremeneuje používateľa a zvyšuje používateľský komfort.

Mnoho používateľov verí aplikácii ktorá číta informácie o používateľovi, ale nie vždy títo používatelia nevyhnutne súhlasia s odsúhlásením práv pre posielanie nových statusov v ich mene. Aktualizácia informácií cez Twitter API - či už je to meno, pozícia alebo pridanie nového

tweetu - vyžaduje odoslanie HTTP POST požiadavky. Každá metóda API, ktorá vyžaduje HTTP POST je považovaná za spôsob zápisu a vyžaduje prístup k čítaniu a zápisu.



Obrázok 25 Generovanie prístupových tokenov na dev.twitter.com

V prípade ak proces OAuth, znie nad rámec integrácie aplikačného prepojenia pri návrhu aplikácie, je možné využitie webových zámerov (angl. web intents), ktoré nepotrebuju používať prístupové tokeny pre interakciu s Twitter API.

3.6.5 Limitovanie prístupu k Twitter API

Limitovanie množstva volaní Twitter API je primárne kontrolované pre jednotlivých používateľov aplikácie. Špecifickejšie sa jedná o limitovanie pre používateľský token. Znamená to, že ak volaná metóda dovoľuje 15 volaní pre používateľa v určitom časovom okne tak pre príslušný prístupový token je dovolených rovnako 15 volaní. Pri používaní autentizácie určenej len pre aplikáciu, limity pre volania API sa tykajú všetkých volaní v aplikácii spoločne. Toto limitovanie je považované ako separátne vzhl'adom na volania používateľov aplikácie.

Limity pre volania Twitter API sú nastavené na znovuobnovenie po 15 minútových intervaloch. Všetky volania vyžadujú autentizáciu takže neexistuje koncept neautorizovaných volaní a limitov. Volania, ktoré využívajú GET požiadavky sú limitované na 15 volaní každých 15 minút, pripadne 180 volaní každých 180 minút.

3.6.6 Knižnice pre prístup k Twitter API

Dostupná knižnica Python Oauth2 zabezpečuje implementačne náročnú časť podpisovania požiadaviek OAuth2 procesom. V nasledujúcom zdrojovom kóde je znázornená hlavná časť napojenia na Twitter API.

```

def oauth_req(url, key, secret, http_method="GET", post_body="",
http_headers=None):
    consumer = oauth2.Consumer(key=CONSUMER_KEY,
    secret=CONSUMER_SECRET)
    token = oauth2.Token(key=key, secret=secret)
    client = oauth2.Client(consumer, token)
    resp, content = client.request( url, method=http_method,
    body=post_body, headers=http_headers )
    return content

home_timeline = oauth_req(
'https://api.twitter.com/1.1/statuses/home_timeline.json', 'abcdefg',
'hijklmnop' )

```

3.7 Testovanie

3.7.1 Zobrazovanie údajov na stránke v reálnom čase

Pri testovaní odosielania údajov pomocou Web-socketov na stránku, kde sa trasa balónu vykresľuje v reálnom čase, sme zistili, že nás používaný Web server na serveri tímového projektu nepodporuje použitie socketov. Použitý server Apache spolu s pluginom mod_wsgi pre beh python aplikácie totiž neumožňuje vytvoriť a udržovať ďalšie asynchronné spojenie pre sockety. Je preto nutné použiť iný web server, ktorý nahradí Apache prípadne Apache musí tieto spojenia presmerovať na druhý webový server udržujúci web-socket spojenia.

3.7.2 Spracovanie údajov na serveri

V rámci testovanie sme odosielali údaje na server po dobu približne dvoch hodín. Frekvencia odosielania bola 1 sekunda, čo je sice extrémna hodnota, ktorá počas letu nebude realizovaná, no cieľom bolo vykonať záťažový test. Spracovanie dát a ich ukladanie bolo bez výraznej záťaže, no načítavanie údajov z databázy pre použitie na stránke sa ukázalo ako príliš pomalé. Načítavanie údajov trvá v priemere dve až tri minúty, čo je pre webovú stránku nepoužiteľné. Toto spomalenie je zapríčinené použitím SQLite databázy, ktorá je uložená v jednom súbore a pri každej požiadavke na databázu je súbor otváraný a postupne čítaný. Pri veľkom množstve dát (po teste mal súbor databázy približne 5MB) je tak prehľadávanie väčšieho množstva dát príliš pomalé. V ďalšej verzii bude potrebné nájsť a implementovať vhodnejší databázový systém.

4 Bibliografia

- [1] Letové prevádzkové služby Slovenskej republiky, š. p., „Letecká informačná príručka,“ 2016.
- [2] Európska komisia, „EUR-Lex, nariadenie č. 923/2012,“ 26 September 2012. [Online]. Available: <http://eur-lex.europa.eu/legal-content/SK/TXT/PDF/?uri=CELEX:32012R0923&qid=1479157226138&from=EN>.
- [3] Letecká informačná služba Slovenskej republiky, „Civilné predpisy, L2 Pravidlá lietania,“ Apríl 2009. [Online]. Available: <http://www.fabryatc.net/civilnepredpisy/L2.pdf>.
- [4] Zawin, „Svetelektro,“ 15 3 2011. [Online]. Available: <https://svetelektro.com/clanky/nasiel-sa-balon-universum-394.html>. [Cit. 10 2016].
- [5] Zawin, „Svetelektro,“ 14 5 2011. [Online]. Available: <http://svetelektro.com/clanky/balon-universum-2-zhodnotenie-399.html>. [Cit. 10 2016].
- [6] „What is Arduino,“ [Online]. Available: <https://www.arduino.cc/en/Guide/Introduction>. [Cit. 3 10 2016].
- [7] E. s. r. o., „Digitálne teplotné čidlo DS18B20 po zberniči 1-Wire,“ [Online]. Available: <http://www.elektrobazeny.sk/titulka/arduino/zakladne-info-18b20/>. [Cit. 3 10 2016].
- [8] Farnell, „Arduino Uno,“ [Online]. Available: <http://www.farnell.com/datasheets/1682209.pdf>. [Cit. 3 10 2016].
- [9] Farnell, „Arduino Mega2560,“ [Online]. Available: http://www.farnell.com/datasheets/810077.pdf?_ga=1.6948594.1701222862.1475489180. [Cit. 3 10 2016].
- [10] SparkFun, „SparkFun GPS Logger Shield,“ [Online]. Available: <https://www.sparkfun.com/products/13750>. [Cit. 3 10 2016].
- [11] ADH-tech, „GP3906-TLP DataSheet,“ [Online]. Available: https://cdn.sparkfun.com/assets/learn_tutorials/4/6/8/GP3906-TLP_DataSheet_Rev_A01.pdf. [Cit. 3 10 2016].
- [12] E. Tutorials, „Temperature Sensors,“ [Online]. Available: http://www.electronics-tutorials.ws/io/io_3.html. [Cit. 1 11 2016].

- [13] A. C. P. Thermistors, „4 Most Common Types of Temperature Sensors,“ 12 2015. [Online]. Available: <http://www.ametherm.com/blog/temperature-sensor-types>. [Cit. 1 11 2016].
- [14] T. E. components, „Proffuse PT100-1020,“ [Online]. Available: <http://www.tme.eu/sk/details/pt100-1020/teplotne-senzory-odporove/proffuse/>. [Cit. 1 11 2016].
- [15] D. semiconductor, „Ds18B20 Programmable Resolution 1-Wire Digital Thermometer,“ [Online]. Available: <http://www.gme.sk/img/cache/doc/530/067/ds18b20-datasheet-1.pdf>. [Cit. 1 11 2016].
- [16] GMElectronic, „DS18B20,“ [Online]. Available: <http://www.gme.sk/ds18b20-p530-067>. [Cit. 1 11 2016].
- [17] D. semiconductors, „KTY81-1 series Silicon temperature sensors,“ 8 2000. [Online]. Available: http://pdf.datasheetcatalog.com/datasheet/philips/KTY81-1SERIES_3.pdf. [Cit. 1 11 2016].
- [18] MUO, „How and Why to Add a Real Time Clock to Arduino,“ 2 2014. [Online]. Available: <http://www.makeuseof.com/tag/how-and-why-to-add-a-real-time-clock-to-arduino/>. [Cit. 29 10 2016].
- [19] Arduino, „DS1302 Real Time Clock,“ [Online]. Available: <http://playground.arduino.cc/Main/DS1302>. [Cit. 30 10 2016].
- [20] Arduino, „Timer1,“ [Online]. Available: <http://playground.arduino.cc/Code/Timer1>. [Cit. 31 10 2016].
- [21] M. T. Karol Rástočný, „Softvérové nástroje pre vývoj softvér u v tíme,“ 2016.
- [22] M. Petrinec a P. Májová, „Orienteering,“ 12 1 2016. [Online]. Available: <http://www.orienteering.sk/page/co-je-to-orientacny-beh>. [Cit. 10 2016].
- [23] „RLX,“ [Online]. Available: <http://rlx.sk/sk/433mhz-868mhz-915mhz-24ghz/3544-433mhz-rf-transmitting-module-er-wrf43301r-transmitter-receiver.html>. [Cit. 11 2016].