Slovenská technická univerzita v Bratislave
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ
Študijný program: SOFTVÉROVÉ INŽINIERSTVO

icPoint
Kandidát na najlepší multimediálny produkt roku 2007

Mailový alias: netrollers@gmail.sk

Členovia tímu:
Bc. Michal Dobiš
Bc. Vladimír Hlaváček
Bc. Hoang Xuan Linh
Bc. Michal Jajcaj
Bc. Dušan Lamoš

Máj, 2006

Zadanie

Kandidát na najlepší multimediálny produkt roku 2007

Počet tímov: 1

Vedúci tímu: Mgr. Alena Kovárová

Každý rok prebieha medzinárodná súťaž The EUROPRIX Top Talent Award (TTA). Je to súťaž multimediálnych produktov, ktoré vytvorili mladí ľudia, sú niečím nové, neobyčajné. Ide skrátka o to, vytvoriť niečo, čo tu ešte nebolo a dokáže to uchvátiť.

Cieľom projektu je navrhnúť a vytvoriť produkt, ktorý by bol dostatočne dobrý na to, aby sa uchádzal o nomináciu alebo dokonca o výhru v súťaži TTA07. Ešte pred návrhom je nevyhnutné spraviť si obraz o tom, aké produkty sa tejto súťaže zúčastňovali po minulé roky, vybrať kategóriu, v ktorej máme šancu uplatniť sa, či už preto, že v tíme je človek, ktorý ovláda nové technológie alebo preto, že daná kategória bola slabo zastúpená.

Fantázii sa medze nekladú, programovať môžete v čomkoľvek, podstatné je, aby to

- uchvátilo - bolo niečím nové,
- dobre fungovalo,
- dobre vyzeralo (preto v tíme musí byť aj človek, ktorý má cit pre dizajn),
- malo aj plne funkčnú ANGLICKÚ verziu!

Do módy v poslednej dobe prichádzajú programy, ktoré aj telesne postihnutým umožňujú pracovať s počítačom, či sa na ňom niečo učiť. Skúste svoje nápady orientovať týmto smerom, nie je to však nevyhnutné.

Predslov

Každý z nás sa už určite prechádzal pod nočnou oblohou. Stačilo sa len pozrieť hore nad seba a videli sme Mesiac, planéty Slnečnej sústavy, hviezdy, pás Mliečnej dráhy, iné galaxie, kopy galaxií a súhvezdia. V určitých obdobiach padajú meteority, meteoritové roje a kométy prelietavajú okolo Zeme. Okrem prírodných objektov je na nočnej oblohe vidieť aj pomaly sa pohybujúce družice, vesmírne stanice ako ISS a dosť často aj lietadlá. Celkovo je možné voľným okom vidieť viac ako 2000 nebeských telies pri dobrých atmosférických podmienkach.

Pri pozorovaní nočnej oblohy vyvstáva množstvo otázok. Tieto otázky môžu byť filozofického charakteru, napríklad existuje niekde tam v diaľke inteligentný život? Kým k nám mimozemšťania neprídu alebo nepošlú signál, na podobnú otázku zrejme odpoveď nenájdeme. Človek sa môže pýtať aj praktickejšie. Pre navigáciu je dôležitá otázka, kde je Polárka? Jej hľadanie sa učí už na základných školách a patrí k základným spôsobom určovania svetových strán. Čo však ostatné hviezdy? Ako ich nájsť, alebo ako nájsť súhvezdie, ktoré predstavuje naše astrologické znamenie? A zrejme už pračlovek si pri pozorovaní nočnej oblohy položil otázku „Čo sú vlastne tie biele bodky, ktoré vidím?.“

Cieľom nášho projektu je vytvoriť aplikáciu, ktorá pomôže jednoducho zodpovedať otázky toho druhého typu. Chceme, aby používateľ nemusel hľadať hviezdy na hviezdnej mape alebo v počítači. Našou snahou je vytvoriť aplikáciu, v ktorej by určovanie hviezdy pracovalo pomocou ukazovadla. Vám bude stačiť len ukázať na nočnú oblohu cez sklo a odpovede dostanete.

Projekt vznikol ako kandidát na najlepší multimediálny produkt roku 2007 v súťaži EUROPRIX TTA v rámci predmetu Tvorba softvérového systému v tíme. Vedúcim projektu je Mgr. Alenka Kovárová a členmi tímu sú nasledovní študenti inžinierskeho štúdia v odbore Softvérové inžinierstvo: Michal Dobiš, Vladimít Hlaváček, Michal Jajcaj, Linh Hoang Xuan, Dušan Lamoš.

Predkladaná dokumentácia je rozdelená do dvoch častí. Prvá obsahuje dokumentáciu k projektu samotnému a je členená podľa jednotlivých fáz životného cyklu projektu. Druhá časť obsahuje dokumentáciu k riadeniu projektu, ponuku tímu, plán projektu, úlohy členov tímu a ďalšie smernice. Je k nej priložené CD s inštalačnými súbormi vytvoreného systému, obrazom webovej stránky a používateľskou dokumentáciou.

icPoint
Kandidát na najlepší multimediálny produkt roku 2007

Projektová dokumentácia

Obsah

0. Úvod

Predkladaný dokument obsahuje projektovú dokumentáciu k softvérovému systému vytváranému v rámci predmetu Tvorba softvérového systému v tíme. Jeho cieľom je vysvetliť riešenie zadania s názvom Kandidát na najlepší multimediálny produkt roku, ktorého výstup bude kandidátom do celoerópskej súťaže venovanej novým prístupom v oblasti multimédií *Europrix Top Talent Award* [1]. Navrhovaný systém predstavuje multimediálne riešenie pre amatérskych astronómov a bežných užívateľov, umožňujúce spoznávať hviezdnu oblohu intuitívnym spôsobom prostredníctvom ukazovania na hviezdy. Názov systému je icPoint.

Dokument je rozdelený do niekoľkých kapitol, ktoré zodpovedajú jednotlivým krokom riešenia projektu.

Prvá kapitola je venovaná opisu riešeného problému, obsahuje informácie o súťaži Europrix TTA, ďalej rôzne námety, ktoré neskôr neboli realizované a stručnú charakteristiku systému icPoint. Ďalšia kapitola rozoberá analýzu problémovej oblasti, rozoberá existujúce riešenia v danej oblasti, a astronomické súradnicové systémy.

V tretej kapitole sú rozoberané technológie, potrebné pre realizáciu projektu. Špecifikácii je venovaná štvrtá kapitola, v ktorej je podrobnejšie popísané správanie sa systému, ako aj ďalšie požiadavky naň. Piata kapitola dokumentu obsahuje hrubý návrh riešenia, pričom popisuje predpokladanú štruktúru nami navrhovaného systému, konkrétne algoritmy a postupy využívané v aplikácii. Kapitolu uzatvára návrh akceptačných testov.

Šiesta kapitola sa zaoberá prototypom vybraných častí systému, popisuje ich funkcionalitu a venuje sa dôvodom výberu práve týchto častí systému pre prototypovanie. Siedma kapitola poskytuje podrobný návrh systému. Preposledná kapitola uvádza stručne záznam o testovaní produktu. Dokument končí zhodnotením prínosov našej práci v tíme.

K dokumentu je priložená technická dokumentácia prototypu.

0.1. Skratky

Skratka	Plný význam
DVD	Digital Versatile Disc, optické médium na ukladanie informácií
IR	Infrared, infračervené svetlo
FIR	Fast Infrared, štandard pre vysokorýchlostné infračervené prenosy
fps	frames per second, počet snímok za sekundu

GPL	General Public Licence, druh licencie pre šírenie a používanie softvéru
GUI	Graphical User Interface, grafické používateľské rozhranie aplikácie
ID	Identifikátor
IP	Internet Protocol, protokol, využívaný pre prenos dát na sieti Internet
IrCOMM	Infrared Communications Protocol, protokol pre bezdrôtovú komunikáciu pomocou infračerveného svetla
IrDA	Infrared Data Association, názov organizácie ako aj označenie skupiny protokolov pre komunikáciu pomocou infračerveného svetla
IrLAN	Infrared Local Area Network, protokol pre vytváranie počítačových sietí typu LAN na báze IR prenosov
IrLAP	Infrared Wireless Link Access Protocol, jeden z protokolov linkovej vrstvy komunikačného štandardu IrDA
IrLMP	IrDA Link Management Protocol, ďalší z protokolov linkovej vrstvy komunikačného štandardu IrDA
IrMC	Infrared for Mobile Communications, protokol pre IR komunikáciu mobilných zariadení
IrOBEX	Infrared Object Exchange, protokol používaný mobilnými zariadeniami pre výmenu objektov cez IR rozhranie
IrTran-P	Infrared Transfer Picture, protokol určený na prenos digitálnej fotografie cez IR rozhranie
ISS	International Space Station, medzinárodná orbitálna stanica
kbps	kilobits per second, kilobity za sekundu
LAN	Local Area Network, lokálna počítačová sieť
LED	Light-emitting diode, polovodičová elektronická súčiastka

	vyžarujúca úzkospektrálne svetlo
LLC	Logical Link Control, vrstva komunikačného protokolu, zabezpečujúca logické spojenie
MAC	Media Access Control, vrstva komunikačného protokolu, zabezpečujúca prístup k fyzickému prenosovému médiu
Mb	Megabit, jednotka dátového objemu
MIME	Multipurpose Internet Mail Extensions, rozšírenie dokumentov, obsahujúce popis obsahu dokumentu (metainformácia)
MS	Microsoft
PC	Personal Computer, osobný počítač
PDA	Personal Digital Assistant, mobilné zariadenie poskytujúce funkcie na prenos a spracovanie informácií
PHY	Physical Signalling Layer, najnižšia vrstva protokolu IrDA
SDL	Simple DirectMedia Layer, multiplatformová knižnica pre prácu s multimédiami a grafikou
SIR	Serial Infrared, štandard pre sériovú IR komunikáciu
TTA	Top Talent Award, medzinárodná súťaž multimediálnych produktov
USB	Universal Serial Bus, univerzálna sériová zbernica
VFIR	Very Fast Infrared, štandard pre vysokorýchlostné infračervené prenosy
WWW	World Wide Web, celosvetová počítačová sieť (Internet)

1. Opis riešeného problému

V tejto časti dokumentu opisujeme kontext systému. Najzávažnejším a najpriamejším spojením je súťaž, do ktorej sa má vytváraný systém zapojiť. Prvá časť dokumentu preto predstavuje Europrix a vyzdvihuje najdôležitejšie vlastnosti projektu úspešného v konkurencií tejto súťaže. Druhá časť sa venuje námetom, ktorých realizáciu sme zvažovali v priebehu prvých týždňov semestra. Kapitolu uzatvára stručný opis zvoleného zadania systému, ktorý vyvíjame s cieľom úspešne reprezentovať fakultu v medzinárodnom zápolení.

1.1. Europrix TTA

The EUROPRIX Top Talent Award [1] je európska súťaž pre študentov a mladých profesionálov, ktorí pracujú na inovatívnych projektoch v oblasti e-obsahu a návrhu. Pri vývoji sa môžu používať ľubovoľné multimédiá a platformy. Úlohou tímu, ktorý má záujem uspieť v tejto súťaži je vytvoriť a prezentovať skutočne originálny a pútavý produkt.

Súťaž Europrix poskytuje dostatočnú voľnosť pri výbere konkrétneho zamerania projektu. Realizuje sa v ôsmich kategóriách:

- Broadband/online,
- Offline/Interactive DVD,
- Mobile Contents,
- Games,
- Interactive Computer Graphics,
- Content Tools & Interface Design,
- Interactive Installations & Interactive TV,
- Digital Video & Animations.

Vo všetkých kategóriách sa kladie dôraz najmä na široké využitie najrôznejších multimédií, ktoré so sebou prinášajú rôzne inovatívne technológie. Rozhodujúcim kritériom úspešnosti na súťaži je preto výber témy projektu.

Výsledky minulých ročníkov súťaže ukazujú, že sa vyzdvihujú najmä nezvyčajné produkty, často zachádzajúce až na pokraj reálnej použiteľnosti. Je zreteľná snaha o motiváciu skúšať nové, doposiaľ neuplatnené nápady a technológie. Vzhľadom na multimediálnu

povahu projektov sa automaticky predpokladá kvalitne spracované grafické používateľské rozhranie. Súťažiaci vo všetkých kategóriách si ďalej musia uvedomiť, že v súčasnosti bežne používané rozhranie človek – počítač je príliš oklieštené. Medzi úspešnými produktmi boli značne zastúpené systémy s menej bežným vstupom, respektíve výstupom k používateľovi.

Zaujímavou adaptáciou výstupu bol napríklad projekt „Let them sing for you“, ktorý umožňoval skladať nové piesne iba zadaním textu. Jednotlivé slová vyberal z existujúcich piesní známych interpretov a spájal ich do jedného celku. Evidentne zaujalo netradičné poňatie DVD s názvom „Murphy's loch“, v ktorom mohol používateľ prechodom medzi kapitolami meniť dej filmu.

Originálnym vstupom bývajú často hudobné nástroje, medzi ktoré sa podarilo v projekte „The ToneLadders“ zaradiť aj rebrík. Ďalším úspešným prispôsobením vnímania počítača bolo ovládanie hry pomocou polohy postavy človeka v produkte „Kick as KUNG FU“, alebo vytváranie zvukových a obrazových efektov na základe tempa pohybu v projekte „Interactive Pojection System“. Nakoniec bežnému vnímaniu počítačov sa úplne vymyká projekt „Outerspace“, ktorý predstavoval umelou inteligenciou riadené robotické rameno správajúce sa ako domáce zvieratko.

Pomerne často boli zastúpené projekty používajúce nezvyčajný hardvér alebo bežný hardvér použije v jeho neobyčajnej aplikácií. Pre maximalizáciu pravdepodobnosti úspechu v súťaži je teda užitočné, keď projekt v príjemnom používateľskom rozhraní zaujímavým spôsobom prezentuje originálnu myšlienku, pričom využíva ľahko dostupné hardvérové prostriedky netradičným spôsobom.

1.2. Námety

Počas prvých týždňov prác na projekte sme rozoberali viaceré smery, ktorými by sa mohol projekt uberať a postupne sme ohraničovali jeho finálnu podobu.

Prvotný nápad bol zameraný na podporu audiovizuálnej komunikácie medzi ľuďmi. Cieľovou skupinou pre tento nápad mali byť predovšetkým študenti a vyučujúci na každom stupni od škôlky až po doktorandov. Systém mal umožňovať rôzne možnosti vzdelávania sa na diaľku. Primárnou by bola komunikácia pomocou obrazu a zvuku, ďalej zdieľaná tabuľa, na ktorú by mohli používatelia zapisovať svoje myšlienky a tak ich zdieľať s ostatnými, podporné funkcie na prevod hovorenej reči do textu a rôzne iné rozšírenia.

Ďalším návrhom bolo vytvorenie multimediálnej domácnosti. Cieľom malo byť umožnenie používateľovi ovládať domáce spotrebiče novou, netradičnou formou pomocou špeciálneho headsetu s laserovým ukazovadlom. Ovládanie by sa uskutočňovalo dávaním

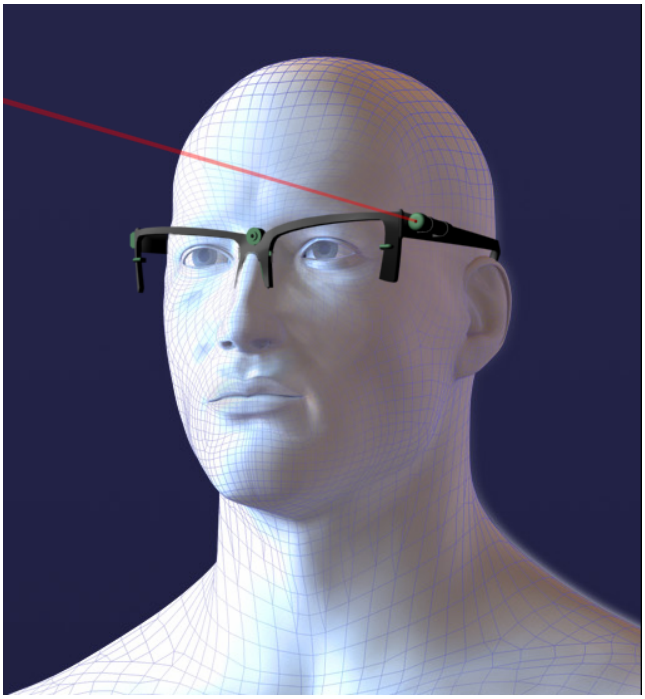
povelov žmurkaním očí, ktoré by boli snímané kamerou, alebo iným podobným zariadením schopným zachytiť žmurknutie oka. Následne by na základe preddefinovaných povelov mohol používateľ ovládať ten spotrebič, na ktorý by aktuálne mieril ukazovadlom. Samotné ovládanie spotrebičov by sa dialo vysielaním infračervených signálov zo zariadenia pripojeného k počítaču. Tento námet však postupne so sebou prinášal viaceré problémy, ktoré nezapadali do rozsahu projektu. Išlo najmä o problém detekcie žmurkania používateľa, pri ktorom by bolo potrebné použiť ďalšiu kameru, ako aj vysielanie infračervených signálov domácim spotrebičom. Infračervené zariadenia na osobných počítačoch a notebookoch nie sú kompatibilné s tými pre domáce spotrebiče. Oba tieto problémy boli viac hardvérového charakteru, preto bolo od tohto zámeru upustené.

Po zamietnutí nápadu multimediálnej domácnosti sme sa ďalej venovali iným možnostiam využitia laserového ukazovadla a kamery. Tieto dva nástroje sme považovali za zaujímavé, preto sme aj naďalej rozmýšľali nad inými možnosťami ich využitia. Vznikli rôzne nápady ako napríklad využiť laserové ukazovadlo na simuláciu hrania na virtuálnej gitare alebo kreslenie pomocou ukazovadla na virtuálne plátno, ktoré by bolo snímané kamerou a nakreslený "obraz" by bol prenášaný do počítača. Cieľom oboch týchto nápadov bolo umožniť používateľovi učiť sa hrať na gitare, respektíve kresliť.

Konečná myšlienka, ktorá nakoniec udáva cieľ projektu, má tiež náučný charakter a otvára nové možnosti pohľadu na nočnú oblohu.

1.3. Systém icPoint

icPoint je astronomický systém pre amatérskych pozorovateľov nočnej oblohy. Základnou ideou je, že používateľ otočením hlavy udá smer, ktorým sa pozerá. Jeho názov predznamenáva celú jeho funkcionalitu - čítaný s anglickou výslovnosťou znamená *„vidím bod"* a ak sa *c* chápe ako skratka z *control* získame *„kontrolujem bod"*. Bodom, ktorý používateľ vidí je typicky červená bodka z laserového ukazovadla umiestneného na jeho hlave [Obr. 1]. Otočením hlavy spôsobí pohyb bodu v priestore okolo neho.



Obr. 1 Ukazovadlo na hlave používateľa

Z pohľadu používateľa systém pozostáva len z webovej kamery pripojenej k prenosnému počítaču, ktorá slúži na nájdenie bodu z ukazovadla. Pri pohľade cez okno, sklo na konferenčnom stolíku, alebo iný priehľadný materiál, rozpoznáva *icPoint* smer pohľadu a poskytuje multimediálne informácie o nebeskom telese, na ktoré sa používateľ pozerá. Umožňuje tak dozvedieť sa pohodlným spôsobom vedecké údaje o každom, voľným okom viditeľnom nebeskom telese. Každý človek môže taktiež napomôcť pri vytváraní spoločnej kolaboratívnej encyklopédie obsahujúcej nie len odborné, ale aj iné zaujímavé údaje o Mesiaci, planétach, hviezdach a súhvezdiach. Príkladom môžu byť najrôznejšie mýty a legendy zo všetkých kútov sveta, do počúvania ktorých sa možno ponoriť pri pozeraní na nočnú oblohu posiatu hviezdami.

Začínajúcim nadšencom - astronómom poskytuje *icPoint* možnosť navigácie k zvolenému nebeskému telesu, prípadne umožňuje preskúšanie schopnosti určovania hviezd. Pri pohľade hlavou smerom k displeju počítača môže používateľ jednoducho ovládať kurzor myši, kedy jej želanú polohu určuje červený bod z ukazovadla. Maximálne pohodlie je dosahované umožnením hlasového ovládania celej aplikácie.

2. Analýza problémovej oblasti

V tejto kapitole sa nachádza analýza problémovej oblasti vyvíjaného systému. Prvá časť sa venuje existujúcim populárnym programom pre astronómov. V druhej časti kapitoly uvádzame používané astronomické súradnicové systémy, ktoré sú základom pri určovaní nebeských telies. Nakoľko systém icPoint poskytuje alternatívne ovládanie počítača využiteľné aj pre pohybovo hendikepovanými osobami, kapitolu uzatvára analýza existujúcich systémov z tejto oblasti.

2.1. Existujúce astronomické programy

Astronómia je oblasť pre vedu aj snílkov. Existuje preto množstvo veľmi precíznych a veľa pútavých programov. Pre podrobnejšiu analýzu sme zvolili dva, ktoré spájajú exaktnosť vedy a príjemnosť zaujímavého grafického podania. Oba majú navyše otvorené zdrojové súbory a možno ich neskôr použiť aj v iných projektoch.

2.1.1. Stellarium

Ide o program, ktorý simuluje pohľad na hviezdnu oblohu z ľubovoľného miesta na zemi [2]. Ide vlastne o planetárium v počítači. Zobrazuje vyše 120 tisíc hviezd z katalógu Hipparchos, planéty s mesiacmi a iné objekty na oblohe. Dokáže realisticky simulovať hviezdnu oblohu s atmosférou aj bez nej. Zvolenú oblasť oblohy je možné približovať podľa požiadaviek používateľa.

Program zobrazuje informácie o hviezdach ako meno, jasnosť, pozíciu v horizontálnej a ekvatoriálnej súradnicovej sústave. Umožňuje zobraziť súhvezdia aj s ich kresbami. Ovládanie programu je realizované myšou, prípadne klávesovými skratkami. Okrem toho je možné používať skripty, napríklad pre zobrazenie zatmenia slnka alebo mesiaca. Umožňuje aj priamo ovládať ďalekohľad pripojený k počítaču. Program je veľmi pekne graficky spracovaný, okrem štandardného obrázku zeme je možné stiahnuť ďalšie. Ide o open source uvoľnený pod licenciou GPL, existuje vo verziách pre MS Windows, Linux a prípadne ďalšie. Pre zobrazovanie grafiky používa knižnicu SDL.

2.1.2. Celestia

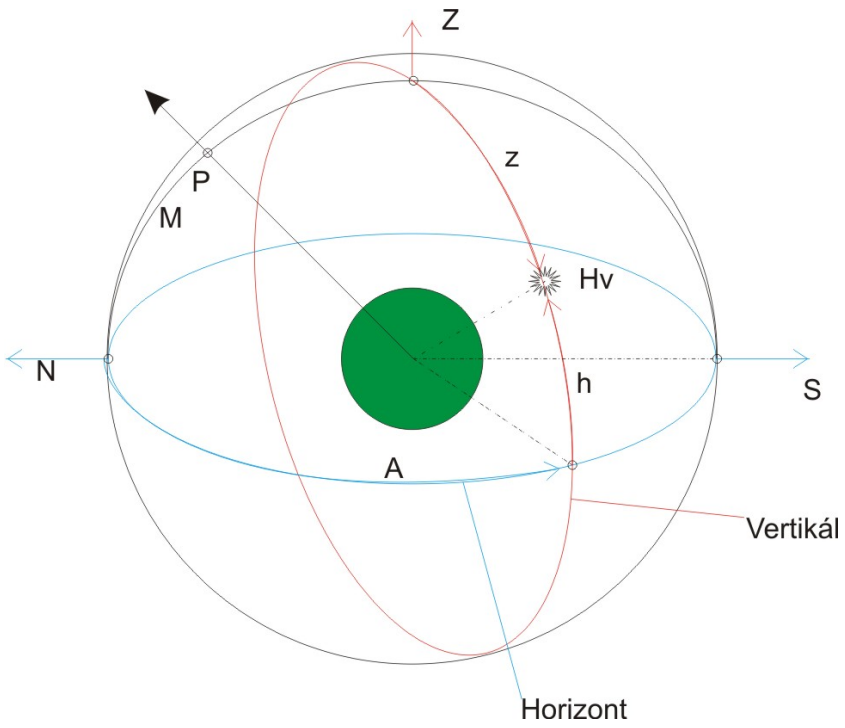
Tento program [3] nie je zameraný na simuláciu astronomických pozorovaní. Miesto toho umožňuje veľmi efektívne vizualizácie objektov v slnečnej sústave a v celej galaxií v 3D prostredí. V programe sa na ovládanie používa najmä myš. Umožňuje otáčať sa okolo vlastnej

osi a prechádzať medzi jednotlivými objektmi. Je možné zobraziť si obežné dráhy planét, spojnice hviezd pre súhvezdia a názvy jednotlivých objektov. Program tiež umožňuje kontrolu času ľubovoľným smerom.

Dôležitou súčasťou programu je katalóg objektov, ktorý umožňuje rýchlo prejsť na niektorý objekt v ňom a zobrazí ho v 3D projekcií. Okrem planét, mesiacov, hviezd a iných prírodných telies katalóg obsahuje napríklad aj družice a vesmírne stanice (ISS, Hubble). Do Celestie je možné jednoducho stiahnuť ďalšie objekty pre túto databázu, textúry pre planéty a dokonca niektoré fiktívne objekty (k dispozícií sú napríklad objekty zo seriálov a filmov). Tiež je možné používať skripty pre navigáciu. Podobne ako Stellarium ide o open source program (GPL) s verziami pre viaceré operačné systémy a s množstvom doplnkov.

2.2. Astronomické súradnicové systémy

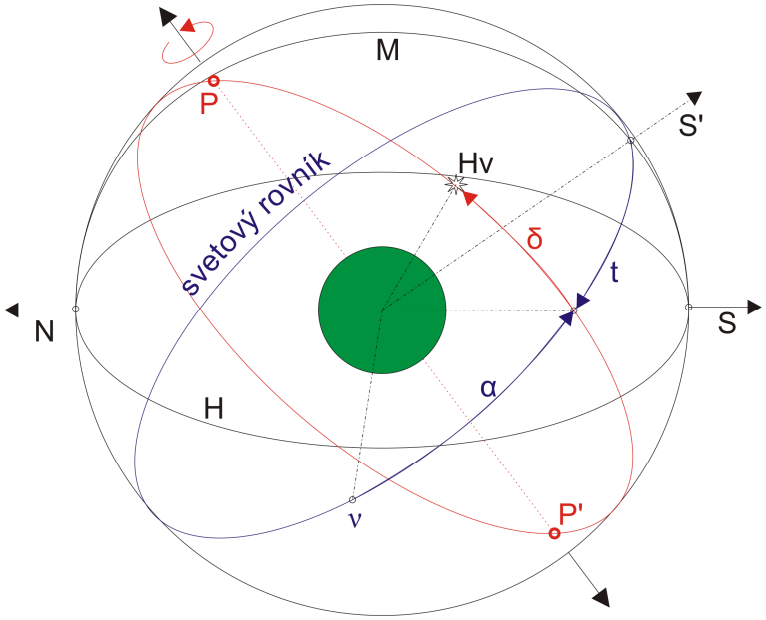
Pozícia nebeských telies sa určuje vo viacerých súradnicových sústavách [4]. Horizontálna súradnicová sústava vychádza z pozície pozorovateľa. Poloha hviezdy sa určuje na takzvanej nebeskej sfére, čo je guľová plocha, na ktorú sa hviezdy premietajú ako body. Základnou rovinou je horizontálna rovina [Obr. 2], kolmá na smer tiaže daného miesta. Pretína nebeskú sféru v najväčšej kružnici – horizonte. Druhou základnou kružnicou je meridián, najväčšia kružnica kolmá na horizont, prechádzajúca zenitom, nadirom a svetovým pólom. Najväčšie kružnice, ktoré prechádzajú zenitom a nadirom sú vertikálne kružnice. Nulovým bodom horizontálnej súradnicovej sústavy je severný bod N, od ktorého sa meria azimut smerom na juh cez východný bod až po vertikálnu kružnicu prechádzajúcu hviezdou. Tiež sa môže merať od južného bodu. Druhá súradnica je výška hviezdy, ktorá sa meria od horizontu po vertikálnej kružnici smerom k hviezde, kladne k zenitu, záporne k nadiru.



Obr. 2 Nebeská sféra

A – azimut, h – výška, z – zenitová vzdialenosť, Z – zenit, S – južný bod,
N – severný bod, P – svetový pól, M – meridián, Hv – hviezda.

Pri rovníkovej súradnicovej sústave [Obr. 3] je základnou rovinou rovina zemského rovníka, ktorá pretína nebeskú sféru v najväčšej kružnici (svetový rovník). Pólmi sú priesečníky rotačnej osi Zeme s nebeskou sférou – severný svet. pól P a južný svet. pól P'. Druhou základnou kružnicou je meridián (pevná rovníková s. s)., alebo aj najväčšia kružnica prechádzajúca pólmi a jarným bodom. Jedna súradnica je deklinácia δ, čo je uhlová vzdialenosť nebeského telesa od rovníka meraná pozdĺž najväčšej kružnice prechádzajúcej pólmi a telesom (deklinačná kružnica). Deklinácia sa vyjadruje v stupňoch kladne smerom k severnému pólu, záporne k južnému. Druhou súradnicou je uhlová vzdialenosť deklinačnej kružnice meraná pozdĺž rovníka od meridiánu (hodinový uhol t) alebo od jarného bodu (rektascenzia α). Vyjadrujú sa v stupňoch, alebo v časovej miere od 0^h do 24^h. Poloha meridiánu na nebeskej sfére závisí od času a od zemepisnej polohy. Deklinácia hviezdy sa pri dennom pohybe nemení, lebo hviezdy opisujú na oblohe kružnice rovnobežné s rovníkom. Pri určovaní hodinového uhlu je potrebné teda udať aj čas pozorovania a zemepisnú šírku stanovišťa. Rektascenzia je od polohy a od času nezávislá, počíta sa od jarného bodu v proti dennému pohybu oblohy až k deklinačnej kružnici hviezdy. Rektascenziu a hviezdny uhol spája hviezdny čas, ktorý je definovaný ako hodinový uhol jarného bodu. Táto súradnicová sústava sa používa pre určenie polohy hviezd v hviezdnych katalógoch a hviezdnych mapách.



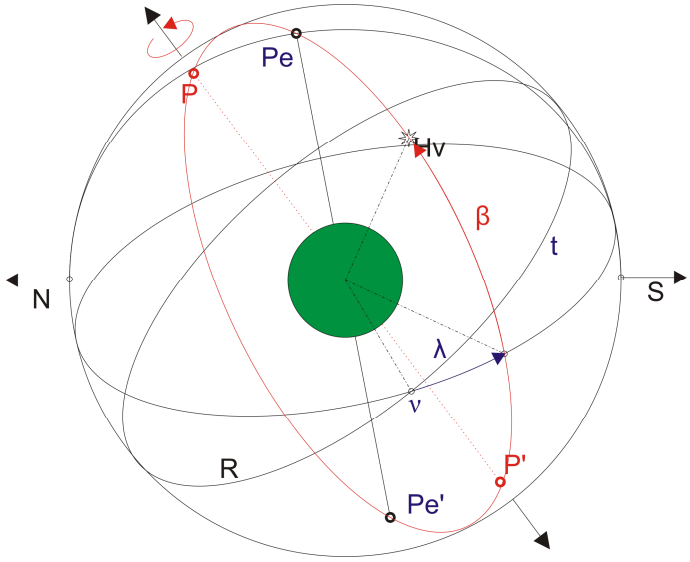
Obr. 3 Rovníková súradnicová sústava

P – severný svetový pól, P’ – južný svetový pól,

H – horizont, M – meridián, Hv – hviezda,

α – rektascenzia, δ – deklinácia, t – hodinový uhol

Pri ekliptikálnej súradnicovej sústave [Obr. 4] je základnou rovinou rovina obežnej dráhy Zeme okolo Slnka – rovina ekliptiky so severným P_e a južným P_e’ pólom. Druhou základnou kružnicou je najväčšia kružnica na sfére, prechádzajúca pólmi ekliptiky a jarným bodom. Jarný bod je jeden z priesečníkov rovníka s ekliptikou, druhý je jesenný bod. Tieto body sa nazývajú aj body rovnodennosti, ekvinokciálne body. Ekliptikálna šírka β sa meria pozdĺž šírkovej kružnice telesa od ekliptiky po teleso, kladne smerom k severnému pólu ekliptiky, záporne smerom k južnému. Vyjadruje v stupňoch od 0° po 90°. Ekliptikálna dĺžka λ sa meria od jarného bodu k šírkovej kružnici telesa po ekliptike od 0° po 360° v smere zdanlivého pohybu Slnka. Ekliptikálne súradnice spolu so sklonom ekliptiky (pre Zem ε = 23°27’) jednoznačne určujú polohu nebeského telesa vzhľadom na ekliptiku. Používa sa najmä na určenie polohy telies v slnečnej sústave.



Obr. 4 Ekliptikálna súradnicová sústava

Pe, Pe' – póly ekliptiky,

β – ekliptikálna šírka, λ – ekliptikálna dĺžka, R – svetový rovník

V dôsledku zmien vyvolávaných precesiou a nutáciou sa polohy základných rovín ekliptikálnej a rovníkovej súradnicovej sústavy menia v priestore. Preto je pri ich určovaní potrebné uviesť epochu alebo ekvinocium, na ktoré sa vzťahujú (napr. na zač. roka 2000).

2.3. Existujúce systémy pre pomoc pohybovo hendikepovaným osobám

Keďže veríme, že naša aplikácia je úplne nová, tak v tejto časti len analyzujeme niektoré, jej podobné produkty. Budeme analyzovať dva typy produktov, a to produkty ovládané očami a produkty ovládané hlasom.

2.3.1. Ovládanie počítača pohybom očí

Z produktov ovládaných očami budeme analyzovať tri exitujúce systémy, a to počítač Iriscom, zariadenie Look Device a počítač MyTobii10.

Iriscom [5] je počítač, ktorý umožňuje ovládanie očami namiesto myši na základe pohybu zrenice. Používateľovi je umožnené písať text poskytnutou virtuálnou klávesnicou. Kliknutie myši sa vykonáva pomocou žmurknutia. Na detekciu pohybu očí sa používa jediná kamera pripojená k počítaču. Výhoda toho počítača je, že umožňuje nevidiacim ovládať počítač očami. Jeho nevýhody zahrňujú vysokú cenu (cca. 6000 eur) a variabilitu výsledku v rôznych svetelných podmienkach.

Look Device [6] predstavuje okuliare, ktoré umožňujú hendikepovanej osobe prezerať web pomocou očí. Na týchto okuliaroch je umiestnených niekoľko senzorov na detekciu pohybu očí. Výhodou toho zariadenia je takisto umožnenie ovládania počítača nevidiacim očami a zároveň možnosť ovládať počítač na diaľku. Jeho nevýhodou je relatívna nepríjemnosť použitia pre používateľa a skutočnosť, že systém zatiaľ funguje len v laboratórnych podmienkach a ešte nebol aplikovaný v praxi.

MyTobiiP10 [7] predstavuje riešenie podobné systému Iriscom. Základným rozdielom je, že kliknutie myši sa realizuje zameraním zraku na vybraný objekt určitú dobu, pričom táto doba sa dá užívateľsky nastaviť. Nevýhoda toho systému spočíva takisto v jeho cene, ktorá predstavuje 17000 USD.

2.3.2. Ovládanie počítača hlasovými povelmi

Z produktov ovládaných hlasom budeme analyzovať program Vspeech [8], ktorý umožňuje používateľovi hlasom ovládať počítač. Ponúka možnosti ako spustenie vybranej aplikácie hlasovým povelom, ovládanie menu programu prostredníctvom hlasu a ovládanie systémových príkazov. Okrem týchto funkcií systém v aplikácii Internet Explorer automaticky listuje možné odkazy na stránke a používateľ si môže vybrať odkaz a presunúť sa na cieľovú stráku vyslovením hlasového príkazu. VSpeech taktiež umožňuje používateľovi vytvoriť svoj slovník a naučiť systém na svoj hlas aby ho program bol schopný rozpoznať.

Výhoda tohto programu je celkom zrejmá. Umožňuje používateľovi vhodným spôsobom ovládať počítač hlasom. Ponúka veľa funkcií pre využitie. Jeho hlavnou výhodou je však skutočnosť, že je zadarmo.

Nevýhoda spočíva v tom, že rozpoznávanie hlasu nie je celkom presné. Navyše je program pomerne pomalý.

3. Analýza potrebných technológií

V tejto kapitole sa venujeme zhodnoteniu dostupných technológií, algoritmov a prostriedkov, ktoré hrajú kľúčovú úlohu v navrhovanom systéme. Poskytujeme prehľad algoritmov hľadania útvarov v obraze, charakteristiku dostupných astronomických katalógov, analýzu technológie ovládania hlasom a v štvrtej časti kapitoly uvádzame schopnosti súčasných webových kamier. V piatej časti študujeme prínos kolaboratívneho vytvárania databáz. Kapitola končí skúmaním možností realizácie infračerveného diaľkového ovládača na domáce spotrebiče. Táto technológia bola kľúčovou pre jeden z našich námetov v začiatkoch semestra a mohla by byť neskorším rozšírením existujúceho systému.

3.1. Algoritmy rozpoznávania obrazu

Hľadanie útvarov v obraze patrí k problémom, ktoré sú ľahko riešiteľné človekom. Realizácia takéhoto hľadania presným algoritmom spustiteľným na počítači je však netriviálna úloha. Jediným vstupom popisovaného algoritmu je dvojrozmerné pole bodov udávajúcich farbu na danej pozícií.

Základným prístupom k riešeniu problematiky hľadania útvarov v obraze je identifikácia súvislých oblastí približne rovnakej farby. Pri prechode jednotlivými pixlami sa zvyšujú hodnoty na tých pozíciách, ktoré susedia s odlišnou farbou a znižujú tým, ktorých okolie je s nimi zhodné. Uvedeným postupom sa vytvorí mapa hrán v obraze a segmentáciou sa získava zoznam adeptov na hľadané objekty. Nedostatkom tejto metódy je, že hľadané útvary sú často zmesou rôznych farieb.

Vylepšenie predstavujú algoritmy schopné nájsť útvary s podobnými znakmi. Klasifikácia útvarov sa realizuje počas dlhotrvajúceho trénovania poskytovaním pozitívnych a negatívnych príkladov hľadaného obrazca [9]. Prezretie poskytnutého obrázku a identifikácia útvarov podobných pozitívnym príkladom býva už pomerne rýchle a môže byť používané aj v systémoch reálneho času. Využitie je takmer neobmedzené – určovanie, čo počítač vidí umožňuje dokonca rozpoznávanie gest rúk, alebo hľadanie tvárí v obraze. Napriek sile takýchto algoritmov, nie sú príliš vhodné na hľadanie bodu dopadu svetelného lúča s laserového ukazovadla. Príčinou je, že takýto útvar nadobúda tvary od príjemnej kružnice až po úplne roztiahnutú elipsu zaznamenanú kamerou skôr ako obdĺžnik. Farba a tvar tohto bodu taktiež značne závisí od pozadia, na ktoré ukazovadlo svieti.

Druhým princípom hľadania útvarov je porovnávanie dvoch obrazov. Ak máme možnosť získať obrázok pozadia, na ktorom nie je žiadny hľadaný útvar, môžeme využiť

princípy používané pri detekcii pohybu. Pred spustením samotného algoritmu sa oba obrázky sa prevedú do čiernobieleho formátu. Následne možno jednoduchým porovnávaním aktuálneho obrazu z kamery s pozadím vytvoriť dvojrozmerné binárne pole, kde na pozícií (x,y) je 1 ak zmena pixla na danej pozícií aktuálneho obrazu oproti pixlu z obrázka pozadia je väčšia ako daný prah (minimálna zmena). Nakoľko v súčasných kamerách sa stáva, že farba niektorého pixla sa zmení aj bez relevantnej zmeny v snímanom priestore, značným problémom je redukcia šumu. S cieľom zamedziť takémuto objavovaniu falošných pozitívnych nálezov zmien sa bežne ako relevantná hodnota berie až priemer susedných bodov v obraze (typicky osmice pixlov). Použitím tejto metódy sa často odstránia aj malé zmeny, medzi ktoré bohužial vo väčšine prípadov patrí aj bod z ukazovadla a preto je toto vylepšenie metódy pre navrhovaný systém nevhodné. Jej funkcionalita sa však dá dosiahnuť aj tak, že v nájdenom objekte vypočítame jeho veľkosť a ak je menšia ako stanovený minimálny rozmer detekovaného objektu, bude sa ignorovať.

Vo výstupe popísaného algoritmu je ďalej potrebné identifikovať okraje súvislých oblastí. Pre tento účel je použiteľná napríklad knižnica OpenCV [10] primárne zameraná na hľadanie útvarov v statickom obraze (úvodné odseky tejto kapitoly). Okrem najrôznejších implementácií algoritmov rozpoznávania obrazu poskytuje aj metódy schopné nájsť popis vonkajšieho okraja oblasti v dvojrozmernom binárnom poli, ktorý je výstupom prvej časti algoritmu založenom na porovnávaní aktuálneho obrazu a pozadia. Zo získaného poľa pozícií definujúcich vonkajšie okraje zmenených oblastí možno výpočtom priemeru získať stredy oblastí, ktoré predstavujú hľadané body z ukazovadla.

Ďalšou úlohou, ktorú treba v systéme icPoint riešiť je nájdenie monitora. V tomto prípade je možné dočasne zmeniť celý obraz na monitore, čo vyvolá zmenu oproti pozadiu a možno použiť identickú metódu ako je popísaná vyššie. Nakoľko treba identifikovať štvoruholník ako priemet obdĺžnika monitora do obrazu kamery, je nutné rozpoznať štyri vrcholy zmenenej oblasti. Na získanom poli pozícií definujúcich vonkajší okraj zmenenej oblasti, teda na okraj obrazu monitora treba aplikovať vzdialenostnú funkciu – počítať vzdialenosť stredu oblasti od jednotlivých bodov okraja. Hľadané body sa nachádzajú v 4 lokálnych maximách tejto funkcie.

3.2. Astronomické katalógy

Katalóg hviezd je usporiadaný zoznam hviezd, ktorý obsahuje informácie o hviezdach. Zvyčajne obsahujú strednú polohu hviezd, obyčajnú a zdanlivú hviezdnu veľkosť, radiálnu rýchlosť, vlastný pohyb, spektrálnu triedu, paralaxu a iné. Katalógy obsahujúce polohu hviezd

na sfére sa nazývajú pozičné katalógy. Zvyčajne sa nové katalógy zostavujú už s existujúcich alebo priamym pozorovaním. Pri porovnávaní hodnôt z katalógov je potrebné zohľadniť precesiu.

Súčasné hviezdne katalógy sa podľa presnosti delia na fundamentálne, pásmové a prehľadové. Vo fundamentálnych katalógoch sú čo najpresnejšie absolútne polohy a vlastné pohyby pomerne malého počtu hviezd, získané z mnohých pozorovaní. Tieto katalógy slúžia ako základ pre určenie polohy hviezd v ročenkách a iných katalógoch. Pásmové katalógy obsahujú presné relatívne pozície hviezd nadviazané na polohy hviezd fundamentálnych katalógov. Prehľadové katalógy obsahujú veľký počet hviezd do určitej hviezdnej veľkosti s menšou presnosťou než v predchádzajúcich katalógoch. Sú vhodné na identifikáciu hviezd a na štatistické práce. Verejne prístupné katalógy je možné nájsť napr. v [11].

Voľným okom je viditeľných viac ako 2000 hviezd. Planét viditeľných voľným okom je päť – Merkúr, Venuša, Mars, Jupiter a Saturn.

3.3. Ovládanie hlasom

Pri ovládaní počítaču, konkrétne softvéru, hlasom, je potrebné najskôr zachytiť hlasový povel, ktorý je následne analyzovaný a na základe toho je vykonaná určitá akcia. Na zachytenie povelu je využitý mikrofón pripojený k počítaču. Jednou z metód analýzy zachyteného zvukového signálu je analýza výskytu špecifických zložiek frekvencií[12][13]. Takúto analýzu je možné vykonať po aplikovaní Fourierovej transformácie na pôvodný signál. Cieľom projektu však nie je vytvárať vlastné riešenie rozpoznávania hlasu, ale jeho praktické použitie. Z tohto dôvodu sme sa ďalej sústredili na analýzu existujúcich riešení.

Na ovládanie aplikácii hlasom existuje viacero komerčných možností, tie však nie sú v našom prípade vhodné. Jedným dôvodom je, že sú to samostatné aplikácie, ktoré nie je možné použiť v rámci iného projektu, ako aj to, že sú to aplikácie komerčné a tým pádom nevhodné pre náš projekt.

Vhodným sa zdá byť použitie prostriedku Microsoft Agent[14][15], ktorého použitie je licencované podľa licencie EULA. Microsoft Agent obsahuje nasledujúce funkcie, ktoré môžu byť z nášho pohľadu zaujímavé:

- Text-to-speech engine – umožňuje prevádzať písaný text do hlasovej podoby. Má viacero možných nastavení hlasu a rýchlosti čítania. Je však obmedzený iba na anglický jazyk. Táto možnosť by sa dala využiť pri podávaní jednoduchých informácií používateľovi.

- Speech recognition engines – umožňuje rozoznať vstupné slová na text. Obmedzenie je tiež na anglický jazyk, to by však nemalo byť prekážkou alebo problémom, pretože v našom využití počítame iba s pár hlasovými príkazmi, ktoré bude používateľ používať.

Microsoft Agent je možné využiť ako súčasť iných aplikácií ako ich komponent. Zároveň umožňuje vytvárať zložitejšie štruktúry príkazov, kde sa príkaz neskladá iba z jedného hlasového povelu, ale z ich kombinácie. Na definovanie takýchto štruktúr sa používajú špeciálne gramatiky.

3.4. Súčasné webové kamery

Podľa [17] je webová kamera definovaná ako kamera, ktorej obraz môže byť v reálnom čase sprístupňované pomocou WWW, aplikácii ako Yahoo, ICQ, atď. alebo video konferenčných aplikácií.

Webová kamera je charakterizovaná parametrami ako rozlíšenie, obnovovacia (alebo tiež snímková) frekvencia, uhol pohľadu, atď.

Web kamery môžeme rozdeliť na dva typy. Sú to jednak IP kamery, ktorých obraz je prístupný cez WWW a pripojujú sa k počítaču cez sieťový konektor (zvyčajne *Ethernet*). Ďalej poznáme štandardné webové kamery (ich obraz nemôže byť prístupný cez WWW), ktoré sa k počítaču pripájajú cez rozhranie USB. Nazývame ich aj anglickým termínom W*ebcam*. IP kamery sú zvyčajne oveľa nákladnejšie, ako kamery typu webcam.

V tabuľke [Tab. 1] sú uvedené niektoré bežne dostupné modely webových kamier.

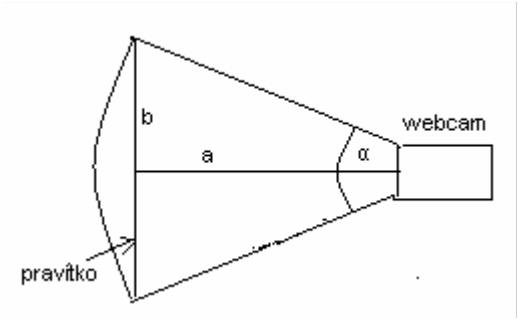
Tab. 1 Príklady dostupných webových kamier

Názov	**Typ**	**Rozlíšenie**	**Snímková frekvencia**	**Cena**
GENIUS Express II GE111	Webcam	640 x 480	30 fps	350 Sk
GENIUS Messenger	Webcam	640 x 480	30 fps	477 Sk
MSI StarCam 370i	Webcam	640 x 480	30 fps	688 Sk
MICRONET IP kamera SP5530	IP kamera	640 x 480	30fps	7690 Sk
MICRONET IP kamera SP5510	IP kamera	640 x 480	30fps	4823 Sk

Poznámka: Rozlíšenie je uvedené v pixloch, snímková frekvencia v snímkach za sekundu

Keďže medzi technickými údajmi webových kamier zvyčajne nebýva uvedený uhol pohľadu kamery, musíme ho určiť meraním. Spôsob merania je pomerne jednoduchý. Pred objektív kamery umiestnime horizontálne (resp. vertikálne) pravítko tak, aby rovina pravítka

bola kolmá na os pohľadu kamery a aby sa konce pravítka presne dotýkali hraníc obrazu, snímaného kamerou. Následne odmeriame vzdialenosť od objektívu kamery k pravítku. Spôsob merania je znázornený na obrázku [Obr. 5].



Obr. 5 Spôsob merania uhla pohľadu kamery

Teda

$tg\left(\frac{\alpha}{2}\right) = \frac{b}{2 \times a}$, pričom b je dĺžka pravítka, a je vzdialenosť od kamery k pravítku a α je uhol pohľadu.

Tak $\frac{\alpha}{2} = arctg\left(\frac{b}{2 \times a}\right)$ => $\alpha = 2 \times arctg\left(\frac{b}{2 \times a}\right)$

Pre kameru *ZSMC USB PC* je horizontálny uhol pohľadu ≈ 40° (a = 11cm, b = 8cm) a vertikálny uhol je ≈ 30° (a = 15cm, b = 8cm).

3.5. Kolaboratívne encyklopédie

Pod pojmom kolaboratívne encyklopédie zvyčajne rozumieme elektronickú formu encyklopédie, ktorá je voľne prístupná pre určitý okruh užívateľov, pričom jej obsah je tvorený spoločne všetkými užívateľmi. Základ takejto encyklopédie tvorí tzv. systém *wiki* (z havajského základu *wiki wiki*, čo znamená „rýchly“ alebo „veľmi rýchly“), čo je vlastne termín pre stránky na webe, ktoré umožňujú všetkým užívateľom pridávať a editovať svoj obsah. Prvým systémom wiki bol web organizácie Portland Pattern Repository [16], ktorá bola zároveň prvou „úschovňou“ programových návrhových vzorov. V súčasnosti jednoznačne najpopulárnejším príkladom kolaboratívnej encyklopédie je Wikipedia [17], ktorú pravdepodobne nie je nutné v tomto texte bližšie popisovať.

Hlavnou výhodou kolaboratívnej encyklopédie oproti tradičným encyklopédiám je nezávislosť na niekoľkých konkrétnych autoroch, ktorých možnosti sú pochopiteľne obmedzené. Nemenej dôležitá je možnosť neustálej a rýchlej aktualizácie obsahu. Keďže do verejne prístupnej encyklopédie môže prispievať neporovnateľne väčší počet autorov súčasne,

ako u klasickej encyklopédie, môže pomerne v krátkom čase vzniknúť rozsiahly a podrobný zdroj údajov. Nevýhodou je potreba kontroly relevantnosti, správnosti, prípadne nezávadnosti pridávaného obsahu, aby sa zamedzilo šíreniu nepravdivých informácií, prípadne spoločensky neakceptovateľných materiálov.

So zohľadnením hore spomenutých prínosov kolaboratívnych encyklopédií sme sa rozhodli zvoliť podobný prístup aj pri tvorbe multimediálneho obsahu pre systém icPoint, či už sa bude jednať o podrobné vizualizácie vesmírnych telies, doplňujúce informácie k nim, alebo hovorené slovo. Užívatelia tak majú možnosť vytvárať rozsiahlu multimediálnu databázu, ktorej rozsah by prekračoval (nielen) časové možnosti autorov systému icPoint.

3.6. Infračervené diaľkové ovládanie

IrDA je štandardom pre infračervenú komunikáciu, za ktorou stojí Infrared Data Association [18] [19]. Úlohou tejto asociácie je koordinovať a publikovať protokoly na zaistenie kompatibility medzi produktmi rôznych výrobcov. Hneď na úvod je potrebné oznámiť, že IrDA nie je rovnaký systém a nemá nič spoločné s infračervenými systémami používanými v diaľkových ovládačoch televízorov, kamier, audio spotrebičov. Tieto dva systémy sú kompletne nekompatibilné.

IrDA používa infračervené svetlo vlnovej dĺžky medzi 850nm a 900nm. Táto vlnová dĺžka je rovnaká pre oba systémy, tak pre komunikáciu medzi dvoma zariadeniami ako aj pre diaľkové ovládače domácich spotrebičov. Tu sa však končia všetky spoločné vlastnosti oboch systémov.

Všetky IrDA zariadenia komunikujú vysielaním modulovaných lúčov infračerveného svetla relatívne vysokými dátovými rýchlosťami. Protokol IrDA je poloduplexným protokolom, takže zariadenia buď prijímajú, alebo vysielajú signály.

Štandardy IrDA sa dajú hrubo rozdeliť na dve skupiny, IrDA Data a IrDA Control. IrDA Data protokoly sa zaoberajú interakciou s inými zariadeniami za účelom výmeny dát. IrDA Control protokol slúži hlavne ako rozhranie so zariadeniami, ktoré signály iba vysielajú, čiže ide o jednosmernú komunikáciu. Tieto dva protokoly nie sú schopné spolupracovať. Ich vlastnosti sú zhrnuté v tabuľke.

Tab. 2 Štandardy IrDa

	IrDA Data	IrDA Control
Rýchlosť prenosu	SIR – Asynchrónny prenos, 9600 – 115200 bps FIR – Synchrónny prenos, do 4 Mb/s VFIR – Synchrónny prenos, do 16 Mb/s	75 kbps / počet zariadení
Vlastnosti	Obojsmerná komunikácie rôznymi rýchlosťami, vrátane detekcie a opravy chýb a automatickej detekcie zariadení v blízkosti	Umožňuje hosťujúcemu zariadeniu komunikovať s viacerými zariadeniami (1:n). Má rýchle reakčné časy a nízke oneskorenie.
Použitie	Osobné počítače, notebooky, mobilné telefóny, PDA…	Klávesnice, myši, ovládanie domácich zariadení

IrDA Control používa trojprotokolový zásobník (protokoly PHY, MAC, LLC). Vysielané dáta sú modulované nosnou frekvenciou 1.5 MHz a následne vysielané pomocou infračervenej LED diódy, čím vzniká celková prenosová rýchlosť okolo 75 kbps. Modulačná schéma IrDA Control bola navrhnutá tak, aby čo najmenej interferovala s existujúcimi zariadeniami pracujúcimi s modulačnou frekvenciou 40KHz, ako sú diaľkové ovládania.

IrDA Data používa zložitejší protokolový zásobník, lebo potrebuje zaistiť rôzne aplikácie ako bod – bod komunikáciu, prenos súborov a prístup do LAN. Tieto protokoly sú: PHY, IrLAP, IrLMP, TinyTP, IrCOMM, IrOBEX, IrTran-P, IrMC, IrLAN.

Z uvedených informácii vyplýva, že použitie integrovaných infračervených portov, alebo USB infračervených zariadení nie je vhodné na vysielanie signálov zariadeniam za účelom ich diaľkového ovládania. Pri hľadaní zdrojov na túto analýzu, ako aj pri samotnom skúšaní použiť integrované infračervené zariadenie na vysielanie signálov som narazil na pár internetových stránok, ktoré síce takéto použitie označili za možné, ale väčšinou išlo o veľmi špecifické zariadenia [20][21].

Riešením, ako ovládať domáce spotrebiče pomocou infračervených signálov vysielaných z počítača môže byť vytvorenie vlastného prijímača/vysielača[22]. Existuje viacero variant, ktoré sa dajú zostrojiť. Negatívom tohto riešenia je, že vyžaduje sériový port, ktorý nie je prítomný na takmer všetkých nových notebookoch a celkovo predstavuje výrazné skomplikovanie situácie, či už spojené s vývojom takéhoto zariadenia ako aj s tým, že každý používateľ by si musel dané zariadenie tiež sám vyrábať.

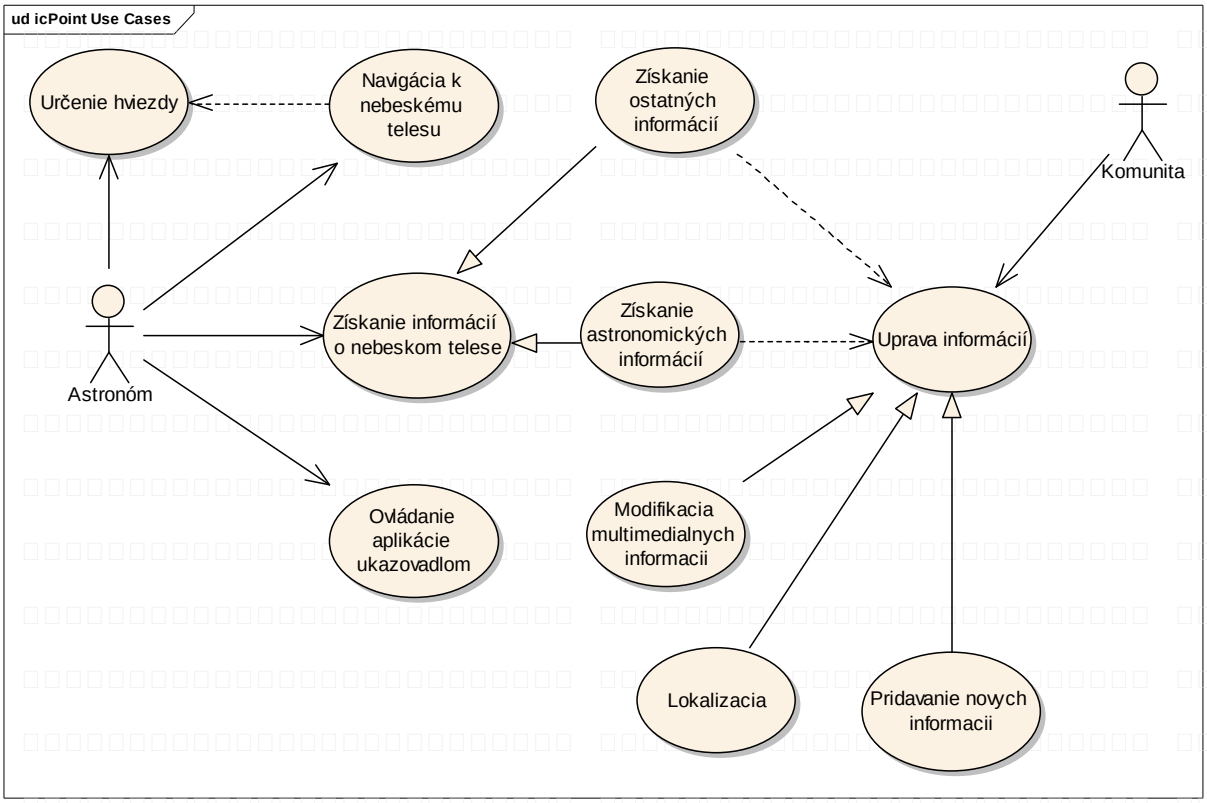
4. Špecifikácia

V tejto kapitole sa venujeme požiadavkám na našu aplikáciu. Kapitola je rozdelená na dve časti. V prvej časti sú funkcionálne požiadavky, zachytené v diagrame prípadov použitia. Prípady použitia sú popísané a niektoré významnejšie sú doplnené aj diagramami aktivít. V druhej častí sú doplnené nefunkcionálne požiadavky na systém.

4.1. Funkcionálne požiadavky

Pri použití aplikácie môžeme identifikovať dvoch hráčov. Astronóma, ktorý je priamym používateľom aplikácie a komunitu. Astronóm využíva aplikáciu pre určenie nebeského telesa, na ktoré sa pozerá, a na navigáciu ku zvolenému telesu. Pri tejto činnosti využíva svetelné ukazovadlo, pomocou ktorého môže taktiež ovládať aplikáciu.

Priamo z aplikácie môže astronóm získavať informácie o nebeských telesách. Môže ísť o vedecké informácie z katalógov, alebo o multimediálne informácie (videá, zvuk, obraz), ktoré bude spravovať komunita používateľov. Komunita môže tieto informácie dopĺňať a upravovať a tiež vytvárať preklady do iných jazykov. Diagram prípadov použitia môžeme vidieť na obrázku [Obr. 6].



Obr. 6 Diagram prípadov použitia aplikácie

4.1.1. Hráči

Astronóm

Opis: Používateľ aplikácie, ktorý od nej očakáva informácie o hviezdach a ovláda ju pomocou ukazovadla.

Prípady použitia:

- Určenie hviezdy.
- Navigácia k nebeskému telesu.
- Získanie informácií o nebeskom telese
- Ovládanie aplikácie

Komunita

Opis: Môže ísť o používateľa aplikácie alebo o niekoho iného, kto pomôže používateľovi upraviť informácie. Môže pridávať nové informácie alebo modifikovať existujúce. Môže aj prekladať existujúce informácie do nových jazykov.

Prípady použitia:

- Úprava informácií

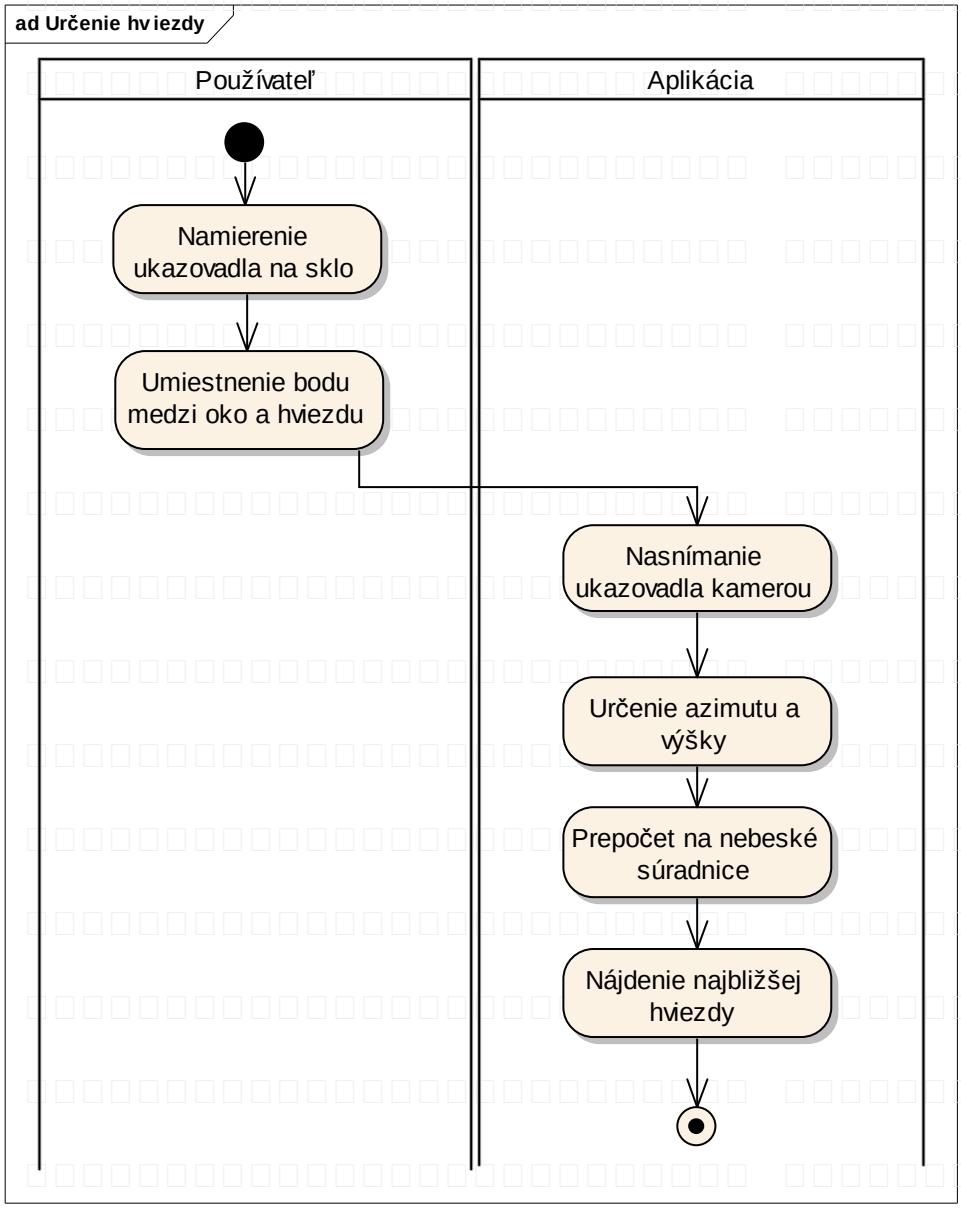
4.1.2. Prípady použitia

Určenie hviezdy

Roly: Astronóm
Vstup: Pozícia červeného bodu na skle a poloha používateľa.
Výstup: Meno hviezdy
Priorita: vysoká

Opis: Používateľ sa pozerá na hviezdu cez sklo a červený bod ukazovadla, ktorý sa na skle ukazuje. Po potvrdení (napríklad hlasom) aplikácia spracuje obraz z kamery a zistí

pozíciu červeného bodu na skle. Na základe toho určí, ktorým smerom a na ktorú hviezdu sa používateľ pozerá. Diagram aktivít pre prípad použitia môžeme nájsť na obrázku [Obr. 7].



Obr. 7 Diagram aktivít pre určenie hviezdy

Navigácia k nebeskému telesu

Roly: Astronóm

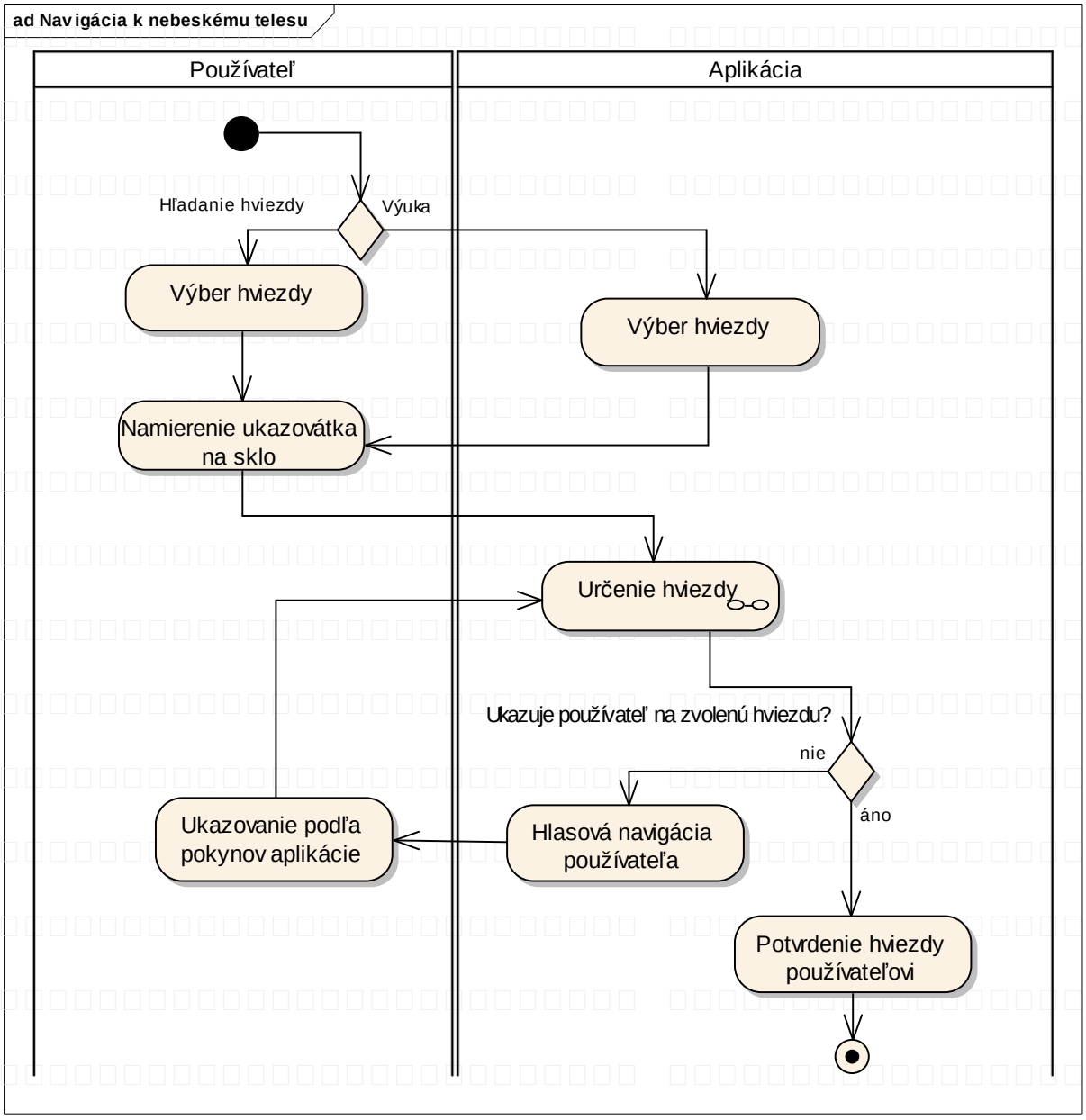
Vstup: Pozícia červeného bodu na skle, poloha používateľa, meno hviezdy

Výstup: hlasová navigácia používateľa k určenej hviezde

Priorita: stredná

Opis: Aplikácia umožní astronómovi navigáciu k zvolenej hviezde. Pomocou snímania smeru, na ktorý sa astronóm pozerá, ho bude zvukovo navádzať tak, aby sa pozeral

na zvolenú hviezdu. Softvér tiež môže po astronómovi žiadať, aby ukázal na náhodne vybranú hviezdu. Po hlasovom potvrdení aplikácia zistí, či astronóm určil správnu hviezdu. Takto môže aplikácia naučiť astronóma orientovať sa na nočnej oblohe. Diagram aktivít pre tento prípad použitia je zobrazený na obrázku [Obr. 8].



Obr. 8 Diagram aktivít pre prípad použitia navigácia k nebeskému telesu

Získanie informácií o telese

Roly: Astronóm

Vstup: meno nebeského telesa

Výstup: informácie o nebeskom telese, môžu byť dva prípady:

- Astronomické informácie, to sú informácie o nebeskom telese ako jeho súradnice, veľkosť, vzdialenosť k zemi, atď.
- Ostatné informácie, najmä nevedeckého charakteru, napr. mýty a legendy o určenom nebeskom telese

Priorita: vysoká

Opis: Keď si používateľ vyberie nebeské teleso, tak podľa jeho výberu (t.j. typ informácií) mu aplikácia bude poskytovať informácie o danom nebeskom telese. Prezentácia informácií používateľovi bude pozostávať z grafiky (fotografie, obrázky, 3D simulácie), prehrávania vytvorených audio súborov a z čítaného textu.

Ovládanie aplikácie

Roly: Astronóm

Vstup: Pozícia červeného bodu na monitore.

Výstup: pozícia myši a akcie v prípade potvrdenia používateľom

Priorita: nízka

Opis: Keď sa používateľ pozerá na monitor, aplikácia pomocou web kamery zistí pozíciu červeného bodu (laserového ukazovadla) v rámci obrazovky. Potom aplikácia posunie kurzor na tú pozícii a používateľ môže tak ovládať aplikáciu. Potvrdzovanie (tlačidla myši) môžu byť realizované hlasovo alebo gestami.

Úprava informácií

Roly: Komunita

Priorita: stredná

Tento prípad použitia je rozdelený na tri prípady. Všetky budú prístupné pomocou web rozhrania z internetu alebo z klientskej aplikácie. Overenie platnosti informácií bude zabezpečené pomocou hlasovania používateľov, prípadne potvrdením kvalifikovanou osobou. Tieto upravené informácie môžu byť s aplikáciou priamo distribuované, alebo po vyžiadaní používateľa môžu byť do aplikácie doplnené.

1. Pridávanie nových informácií

Vstup: meno nebeského telesa, doplnkové informácie

Výstup: nie sú

Opis: Komunita doplní do databázy cez webové rozhranie nové informácie o nebeskom telese, napríklad lokálne mýty a legendy.

2. Lokalizácia

Vstup: nová lokalizácia

Výstup: nie sú

Opis: Komunita vytvorí nový preklad niektorých existujúcich informácií. Môže dopĺňať preklad multimediálnych informácií o hviezdach, ale aj lokalizovať aplikáciu samotnú alebo informácie z katalógov.

3. Modifikácia multimediálnych informácií

Vstup: meno nebeského telesa, doplnkové informácie.

Výstup: nič

Opis: Cez internetové rozhranie komunita požívateľov upraví existujúce informácie, ktoré aplikácia o nebeských telesách poskytuje.

4.2. Nefunkcionálne požiadavky

Okrem funkcionálnych požiadaviek, ktoré sme identifikovali, by naša aplikácia mala spĺňať ďalšie požiadavky pre bezpečnosť používateľa, softvérové a hardvérové požiadavky a požiadavky na rozhranie.

4.2.1. Bezpečnosť pre oči používateľov alebo ľudí v blízkosti.

Z dôvodu, že laserový lúč je nebezpečný pre oči používateľa, bude nutné zmierniť jeho efekty alebo upozorniť používateľa na túto skutočnosť. Nebudeme používať ukazovadlo s intenzívnym lúčom.

4.2.2. Softvérové požiadavky

Pre potreby ovládania aplikácie hlasom bude nutný softvér pre spracovanie hlasu, napríklad Microsoft Agent.

4.2.3. Hardvérové požiadavky

Predbežné hardvérové požiadavky na osobný počítač budú vychádzať z požiadaviek použitého implementačného prostredia a ďalších požiadaviek, ktoré vyplývajú z potrebných multimediálnych technológií. Tiež bude potrebná web kamera na získavanie obrazu.

4.2.4. Iné požiadavky

Aplikácia bude pre svoju funkčnosť a lepšie pohodlie používateľa využívať nasledovné predmety:

- Laserové ukazovadlo, s jeho pomocou môže aplikácia vedieť kam sa používateľ pozerá.
- Okuliare, na ktorých bude ukazovadlo umiestnené.
- Sklo, na ktoré bude svietiť laserové ukazovadlo pri určovaní smeru pohľadu používateľa

4.2.5. Požiadavky na rozhranie

Rozhranie aplikácie musí byť používateľsky prívetivé. V základnej verzii musí byť lokalizované do anglického jazyka.

4.3. Údaje v aplikácií

Údaje prístupné pre používateľa môžeme rozdeliť do dvoch kategórií. Prvá sú vedecké údaje o nebeských telesách, napríklad ekvatoriálne súradnice pre hviezdy, magnitúda, vzdialenosť od Zeme. Tieto údaje sú získavané na základe hviezdnych katalógov a používateľ ich nemôže upravovať. Druhá kategória sú údaje multimediálne. Vzťahujú sa tiež na nebeské telesá, ale používateľ ich môže meniť. Predstavujú nevedecké informácie o nebeských telesách.

Okrem týchto dvoch kategórií aplikácia bude uchovávať rôzne meta informácie, napríklad o jazyku alebo type informácií.

Návrh logického dátového modelu pre lokálnu databázu údajov je možné nájsť v kapitole 5.1.9.

5. Hrubý návrh riešenia

V tejto kapitole predstavujeme navrhovanú architektúru systému icPoint. Prvá časť kapitoly člení systém na moduly, ktoré stručne popisuje. Druhá časť kapitoly poskytuje návrh, ako využitím ukazovadla, skla a kamery pripojenej k počítaču určovať smer, ktorým sa pozerá používateľ systému. Kapitola končí návrhmi akceptačných testov.

5.1. Architektúra systému

Architektúra systému je vytvorená zo zreteľom na požiadavky. Systém pozostáva z dvoch častí. Hlavná časť aplikácie icPoint pracuje na osobnom počítači astronóma. Je rozdelená do niekoľkých balíkov, ktoré spolu navzájom spolupracujú.

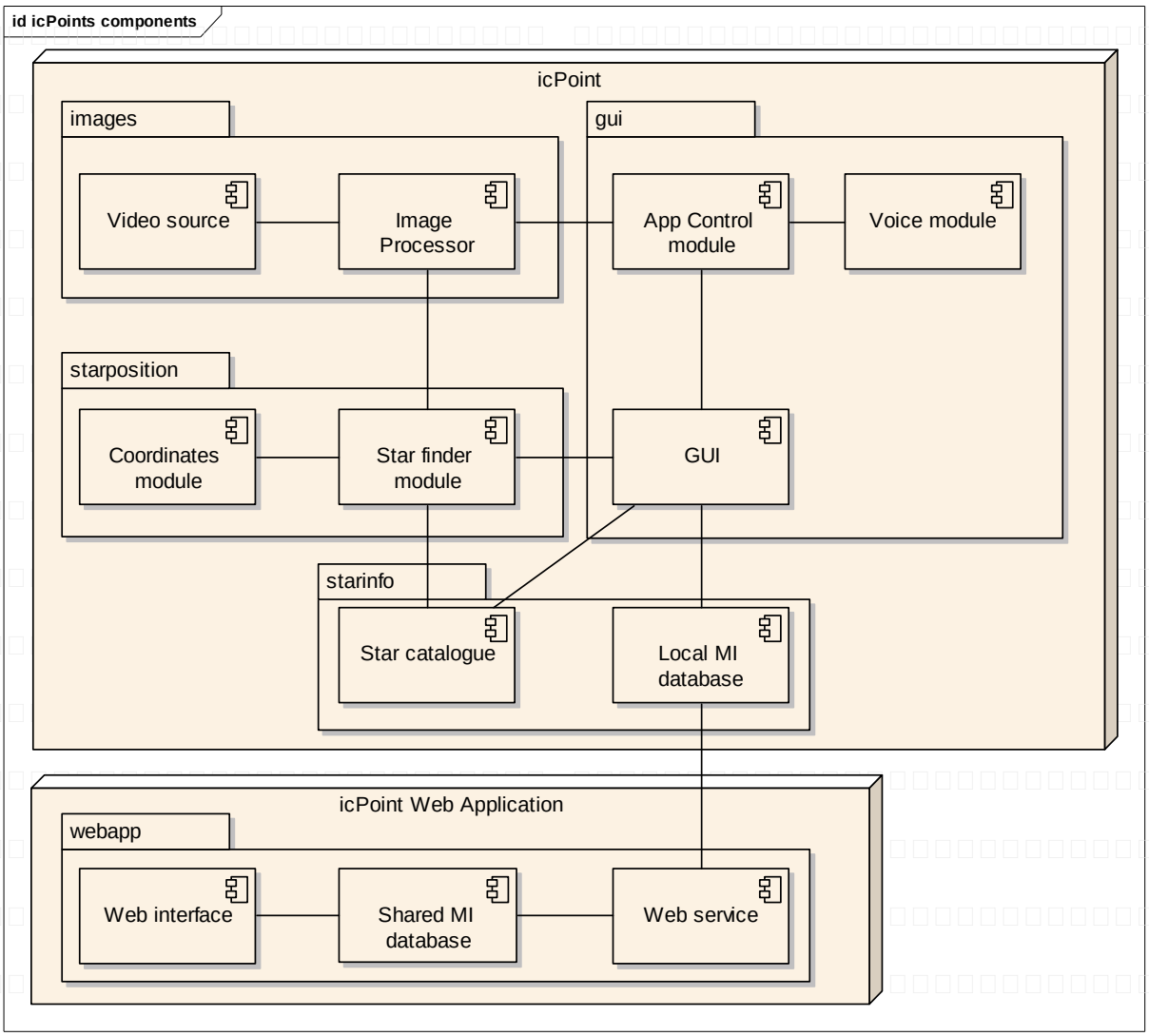
Balík images zahŕňa komponenty, ktoré zabezpečujú čítanie a spracovanie obrazu. Na načítanie obrazu z kamery slúži komponent Video source. Jeho úlohou je poskytovať obraz pre Image processor na spracovanie a získanie pozície ukazovadla. Informácie o pozícií ukazovadla v rámci skla budú využívať moduly v balíku starposition. Modul Starfinder využije modul Coordinates na prepočet súradníc do súradníc platných pre nebeské telesá. Na základe údajov z hviezdneho katalógu nájde požadované nebeské teleso.

Používateľské rozhranie a ovládanie aplikácie icPoint majú na starosti komponenty v balíku gui. Informácie z modulu Image Processor využíva aj komponent App control, pokiaľ používateľ ukazuje na obrazovku počítača. Spolupracuje s modulom Voice module pre rozpoznávanie hlasu a čítanie textových informácií. Samotné grafické rozhranie aplikácie predstavuje komponenta GUI. Zobrazuje informácie z ostatných modulov pre používateľa a na interakciu s ním môže okrem myši používať aj interakciu cez modul App control.

V balíku starinfo sa nachádzajú komponenty pre správu informácií o nebeských telesách. Komponent Star catalogue obsluhuje informácie z hviezdnych katalógov. Okrem vedeckých informácií je potrebná aj lokálna databáza multimediálnych informácií, ktorú zabezpečuje komponent Local MI database. Tento modul tiež slúži na komunikáciu so zdieľanou databázou informácií cez webovú službu. Oba komponenty v module starinfo sú prepojené na komponent gui.

Druhá časť aplikácie je prístupná cez internet na webovom serveri. Komunikáciu s aplikáciou používateľa zabezpečuje Web service. Jej úlohou je prenášať informácie medzi lokálnou a zdieľanou databázou na internete (komponent Shared MI database). Správa tejto databázy je možná pomocou webového rozhrania oddeleného od databázy.

Celý diagram komponentov vrátane vzťahov medzi nimi môžeme vidieť na obrázku [Obr. 9].



Obr. 9 Diagram komponentov aplikácie

5.1.1. Video source

Komponent video source zahŕňa všetky triedy pre prácu s kamerou. Ich úlohou je poskytovať obraz alebo video z kamery podľa požiadaviek.

5.1.2. Image processor

Image processor spracováva obraz z video kamery. Pomocou algoritmov rozpoznávania obrazu nájde na obraze prijímanom z modulu Video source ukazovadlo. Pri ukazovaní na sklo hľadá okrem bodu ukazovadla aj odraz z ukazovadla v skle. Odraz je

farebne odlíšený od bodu ukazovadla. Tieto dva body slúžia ako základ pre výpočet priamky otočenia používateľa. Prvý bod tejto priamky je bod ukazovadla na skle. Druhý sa vypočíta z odrazu ukazovadla, ktoré je upevnené na hlave používateľa. Využije sa pri tom známa vzdialenosť kamery od skla. Pri známom azimute otočenia kamery vieme určiť azimut a výšku otočenia používateľa. Tieto údaje sa predajú modulu Star finder.

Ak sa používateľ pozerá na obrazovku počítača, tento modul to deteguje a na obraze z kamery nájde pozíciu ukazovadla v rámci obrazovky. Túto pozíciu predá do modulu App Control module.

5.1.3. Star finder module

Modul Star finder obsahuje triedy pre efektívne a rýchle nájdenie najbližšieho nebeského telesa k určeným súradniciam. Na prepočet súradníc využije modul Coordinates. Musí zabezpečiť rozlišovanie jednotlivých druhov telies podľa toho ktoré je najbližšie k daným súradniciam. To znamená napríklad rozlišovanie Mesiaca, pretože ten je pomerne veľký oproti ostatným telesám na nočnej oblohe. Pre súradnice hviezd z analýzy vyplynulo, že sú určené stálymi rovníkovými súradnicami (prakticky nemenné minimálne pre obdobie cca 25 rokov). Preto pri určovaní hviezd využije prepočet z horizontálnej súradnicovej sústavy do rovníkovej a nájde najbližšie hviezdy k týmto súradniciam.

Pri hľadaní planét je nutné využívať iný postup, pretože nie sú tak pevne určené. Vypočíta skutočnú polohu planét, prevedie ju na horizontálne súradnice a skontroluje, či sa používateľ nepozerá daným smerom. Vzhľadom na to, že viditeľných planét je iba päť, nevzniká veľké oneskorenie. Mesiacmi planét sa nie je potrebné zaoberať, vzhľadom na to, že voľným okom nie sú na nočnej oblohe viditeľné. Okrem planét modul kontroluje aj pozíciu Mesiaca.

Po určení všetkých najbližších telies odovzdá modul Star finder tieto informácie grafickému rozhraniu pre ich zobrazenie používateľovi.

5.1.4. Coordinates module

Modul Coordinates implementuje astronomické algoritmy potrebné pre prepočet súradníc a vyhľadávanie nebeských telies. Obsahuje funkcie na prepočet z horizontálnej súradnicovej sústavy do rovníkovej a na prepočet z ekvatoriálnej do horizontálnej. Okrem toho obsahuje ďalšie algoritmy na výpočet juliánskeho dátumu, započítanie skreslenia atmosféry a na výpočty pozícií planét a Mesiaca. Tieto funkcie sú využívané modulom Star finder pri hľadaní najbližšej hviezdy.

5.1.5. App control module

App control slúži na ovládanie aplikácie pomocou ukazovadla. Získava informácie z Image processoru, ktoré obsahujú pozíciu ukazovadla v rámci obrazovky. Na túto pozíciu presunie myš. Okrem toho deteguje, ak používateľ vykonal gesto. Gesto je určitý obrazec, ktorý používateľ môže vytvoriť pri pohybe ukazovadlom. Gestá bude možné konfigurovať v aplikácií cez používateľské rozhranie. Ak používateľ vykoná gesto, tento modul zabezpečí vykonanie konfigurovanej funkcie. V spolupráci s modulom Voice module zabezpečuje tiež vykonanie funkcií na hlasový pokyn od používateľa.

5.1.6. Voice module

Ide o modul s dvoma funkciami. Prvá je prijímanie hlasových povelov od používateľa. Modul rozpoznáva hlasové povely a podľa toho predá riadenie modulu App control. Okrem hlasových povelov môže tiež rozpoznávať aj iné zvukové signály (napríklad pískanie). Druhá funkcia modulu je čítanie textových informácií pre používateľa.

5.1.7. GUI

Tento modul zahŕňa grafické rozhranie aplikácie icPoint. Používateľ môže v rozhraní aplikácie prezerať informácie z katalógov a z lokálnej databázy multimediálnych informácií. Informácie používateľ môže upravovať priamo z aplikácie a potom nahrať do zdieľanej databázy informácií na internete. Pre zobrazenie multimediálnych informácií musí rozhranie umožňovať prehrávanie videa, zvuku a zobrazovanie textu a obrázkov.

Pri pohľade na hviezdnu oblohu cez sklo môže rozhranie aplikácie zobrazovať simulovanú hviezdnu oblohu a na nej zvýrazňovať hviezdu, na ktorú sa aktuálne ukazuje. Ak používateľ potvrdí hviezdu, rozhranie ukáže informácie o nej. Môže hlasovo voliť, či chce vedecké alebo nevedecké multimediálne informácie.

5.1.8. Star catalogue

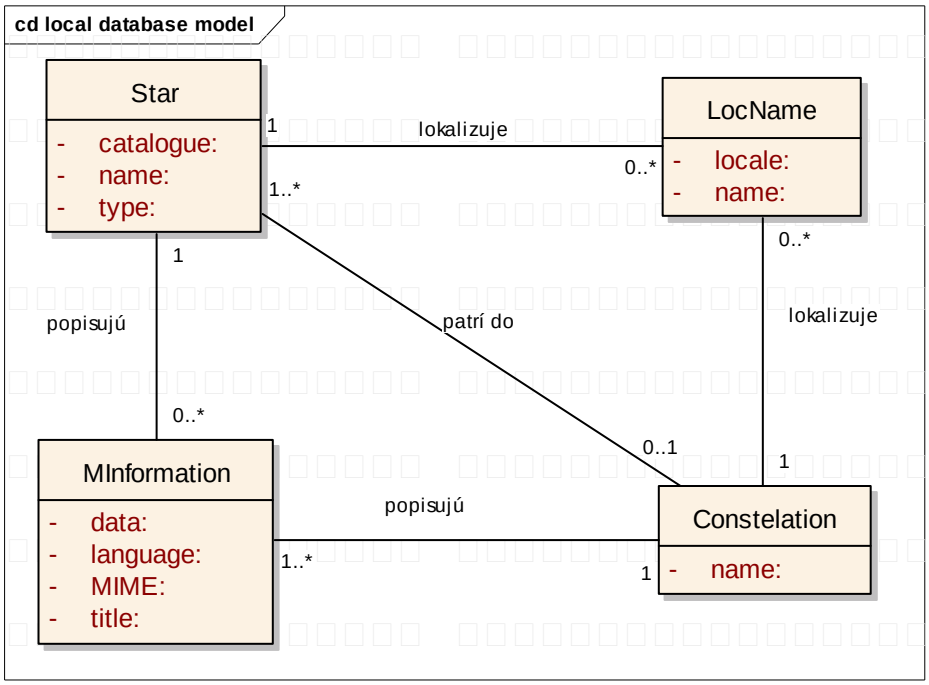
V tomto module je zahrnutá správa hviezdnych katalógov. Nemusí ísť o jediný katalóg, modul umožní aj správu viacerých a potom umožní používateľovi zvoliť si, ktorý katalóg chce používať. Pozície hviezd v katalógu sa používajú pri hľadaní hviezd modulom Star finder. Tu sa však prehľadávajú iba hviezdy viditeľné voľným okom, preto katalóg musí poskytovať zoznam týchto hviezd. Pre prehliadanie hviezd cez grafické rozhranie môže byť poskytnutý celý katalóg hviezd.

5.1.9. Local MI database

Modul vykonáva správu lokálnej databázy multimediálnych údajov. Zabezpečuje funkcie pre prístup do tejto databázy ako vkladanie, mazanie, zmenu údajov a vyhľadávanie. Informácie z databázy bude zobrazovať grafické rozhranie. Okrem toho modul predstavuje aj klienta pre webovú službu, cez ktorú sa pristupuje k zdieľanej databáze dát na internetovom serveri.

Logický dátový model pre databázu je zobrazený na obrázku [Obr. 10]. Do lokálnej databázy údajov sa ukladajú informácie o nasledovných entitách:

- Star – nebeské teleso, môže ísť o planétu alebo hviezdu. Má atribúty:
- name – identifikátor hviezdy v rámci katalógu
- catalogoue – identifikátor katalógu
- type – typ nebeského telesa, môže byť hviezda, planéta, galaxia, hmlovina a pod.
- Constelation – predstavuje súhvezdie. Do súhvezdia môže patriť 1 až n hviezd, ale nebeské teleso nemusí patriť do žiadneho súhvezdia (napríklad planéta). Súhvezdie určujú nasledovné atribúty:
- name – meno súhvezdia
- MInformation – entita uchovávajúca multimediálne informácie. Tieto informácie môžu popisovať nebeské teleso alebo súhvezdie, pričom ku každému súhvezdiu alebo nebeskému telesu vieme priradiť ľubovoľný (0..*) počet entít MInformation. Charakterizujú ich nasledovné atribúty:
- title – má popisný charakter, môže slúžiť ako titulok pre zobrazenie danej informácie
- MIME – určuje typ dát pomocou MIME typu
- data – samotné dáta, sú uchovávané priamo v databáze
- language – jazyk, v ktorom sú dáta uložené
- LocName – entita, ktorá uchováva lokalizované názvy nebeských telies a súhvezdí. Každá lokalizácia sa vzťahuje na jedno súhvezdie alebo nebeské teleso a k jednému telesu alebo hviezde môže byť priradených viacero lokalizovaných názvov. Entita obsahuje nasledovné atribúty:

- locale – identifikuje jazyk v ktorom je daný názov
- name – lokalizovaný názov objektu
- 

Obr. 10 Logický dátový model pre lokálnu databázu údajov

Pretože ide o jednoduchú relačnú databázu, bolo by pri implementácií vhodné využiť niektoré existujúce riešenie pre súborové relačné databázy (napríklad SQLite [23]).

5.1.10. Web service

Modul implementuje webovú službu na ktorej budú prístupné údaje zo zdieľanej databázy. Pomocou tohto modulu sa multimediálne informácie prenášajú z a do lokálnej aplikácie podľa potrieb používateľa. Na strane používateľa môžu byť cachované, alebo môže vždy dochádzať k obnove týchto údajov cez web service, záleží podľa implementácie lokálnej databázy.

5.1.11. Shared MI database

Modul Shared MI database predstavuje zdieľanú databázu multimediálnych údajov, ktorá bude prístupná z webového servera pre komunitu používateľov. Okrem údajov pre lokálne databázy musí uchovávať aj informácie potrebné pre autentifikáciu a autorizáciu používateľov, overenie platnosti údajov, údaje pre ankety a podobne. Tento dátový model je závislý od možností, ktoré bude webové rozhranie poskytovať.

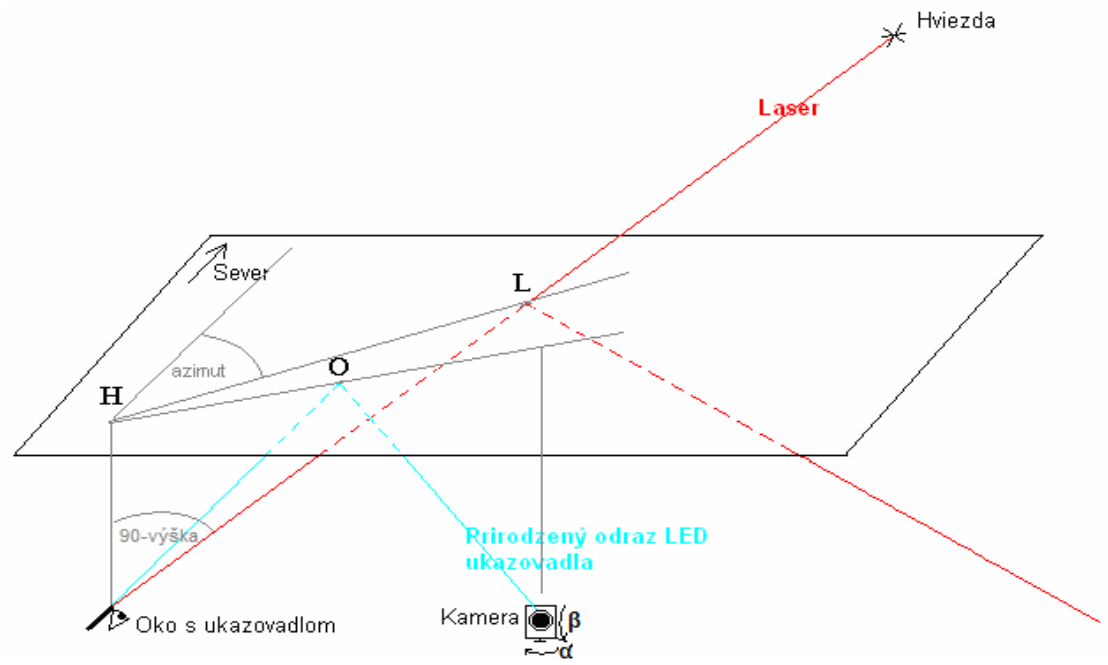
5.1.12. Web interface

Webové rozhranie internetovej aplikácie umožňuje editovanie zdieľanej databázy údajov. Bude zrejme realizované formou wiki. Umožňuje pridávanie informácií o nebeských telesách a súhvezdiach, zmenu týchto informácií, upload súborov so zvukom alebo videom na server, prípadne odstránenie existujúcich údajov. Umožňuje tiež tvorbu ankiet a hlasovanie o pravdivosti a platnosti informácií. Výsledky hlasovaní budú prístupné pre používateľa, keď si bude chcieť lokálnu databázu rozšíriť o tieto údaje. Rozhranie umožňuje aj prístup k inštalačným súborom aplikácie, dokumentácií, informáciách o projekte a rôznych novinkách.

5.2. Určovanie pozície hviezd

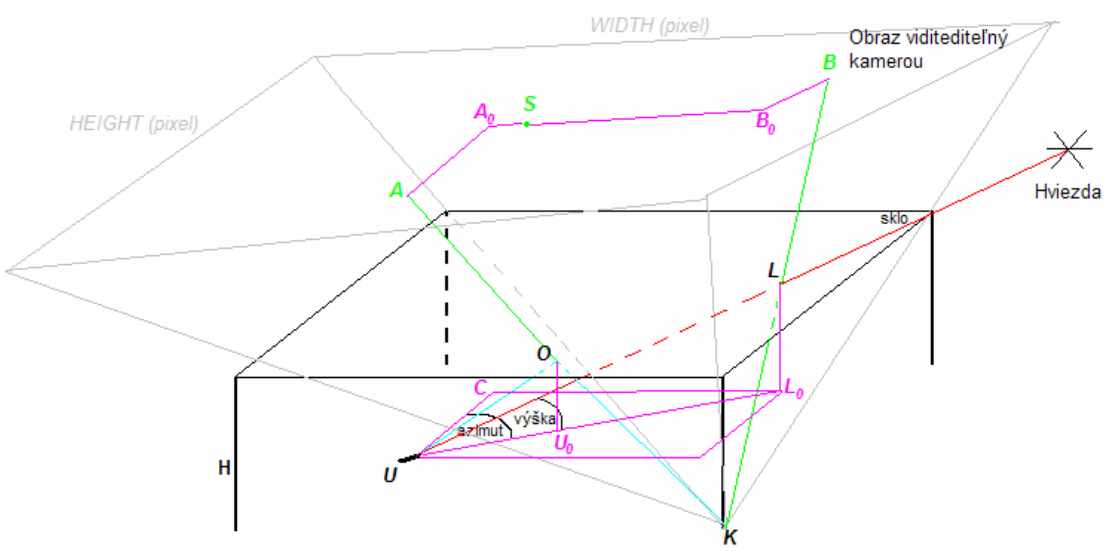
Kritickou úlohou v systéme je určenie smeru, kam sa používateľ na oblohe pozerá. Pre zistenie smeru treba získať 2 body v priestore a preložiť nimi priamku s cieľom určiť jej sklon k horizontu a jej otočenie vzhľadom na sever. Získané uhly predstavujú azimut a výšku, a sú potrebné pre určenie nebeského objektu, na ktorý používateľ ukazuje.

Elegantným riešením problému je použitie videokamery otočenej smerom k oblohe. Používateľ nachádzajúci sa v jej blízkosti má na hlave umiestnené ukazovadlo, ktoré svieti smerom k pozorovanej hviezde. Kamera sníma len priemetňu (priehľadný materiál, napríklad sklo), na ktorej rozpoznáva dva body [Obr. 11]. Prvým je bod dopadu svetelného lúča z ukazovadla na priemetňu (bod *L*). Druhým rozpoznávaným bodom je odraz zdroja tohto lúča, prípadne samostatnej LED umiestnenej na hlave používateľa (bod *O*).



Obr. 11 Zjednodušený obrázok pozície hviezdy a odrazu na priemetni

V domácnostiach sa bežne nachádzajú konferenčné stolíky so sklenným povrchom. Predpokladáme využitie najmä tohto typu priemetne a preto možno vzdialenosť oka od priemetne a vzdialenosť kamery od priemetne pokladať za rovnakú. Poloha oka sa pritom môže meniť a nie je možné brať ju ako fixnú. Aby bolo možné vidieť väčšiu plochu z malej vzdialenosti, je ďalej potrebné dať kameru mierne nabok a umožniť jej otočenie okolo osi sever-juh (uhol α) a osi východ-západ (uhol β). Popísanú situáciu zobrazuje nasledujúci obrázok [Obr. 12], kde body *A* a *B* sú pozície v obraze, na ktorých kamera vidí body *O* a *L*. Body A_0 a B_0 znázorňujú priemet bodov *A* a *B* do x-ovej osi obrázka, bod *S* je stred obrázka (v pixloch). Výška stolíka je označená písmenom *H*. Pomyselnú vzdialenosť kamery K od obrázka, teda veľkosť úsečky *KS* reprezentuje značka *h*.



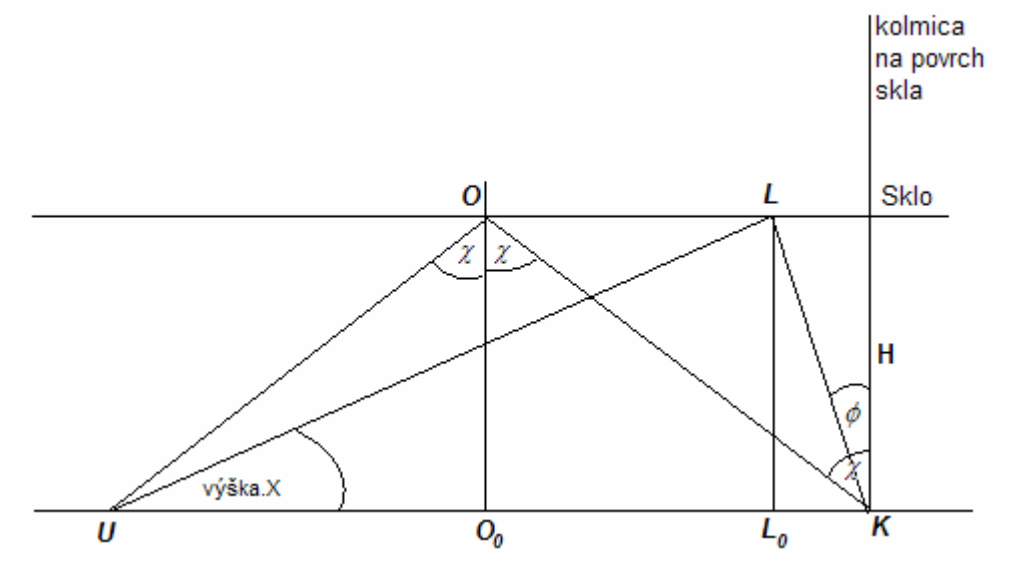
Obr. 12 Priemety bodov v obraze

Pri známom horizontálnom (u_1) a vertikálnom (u_2) uhle pohľadu kamery možno odvodiť $h = \frac{WIDTH}{2 \times tg\left(\frac{u_1}{2}\right)}$, respektíve $h = \frac{HEIGHT}{2 \times tg\left(\frac{u_2}{2}\right)}$. Priamo z obrázka možno potom písať rovnicu pre *x*-ovú zložku uhla medzi kolmicou a priamkou *KO* (rovnica 1), respektíve kolmicou a priamkou *KL* (rovnica 2).

$$\chi = uhol(A_0KS) + \alpha = arctg\left(\frac{A.x - S.x}{h}\right) + \alpha = arctg\left(\frac{(A.x - S.x) * tg\left(\frac{u_1}{2}\right)}{WIDTH}\right) + \alpha \quad (1)$$

$$\phi = uhol(B_0KS) + \alpha = arctg\left(\frac{B.x - S.x}{h}\right) + \alpha = arctg\left(\frac{(B.x - S.x) * tg\left(\frac{u_1}{2}\right)}{WIDTH}\right) + \alpha \quad (2)$$

[Obr. 13] zobrazuje prierez rovinou danou osou *x* a zvislou čiarou. Body *U* a *C* majú identickú *x*-ovú súradnicu a preto ich možno v tomto priemete považovať za identické. Z obrázka je zreteľný postup ako získať dĺžku úsečky CL_0 (rovnica 3).



Obr. 13 Priemet do roviny osi X (východ-západ) a zvislej osi

$$CL_0 = 2 \times |K.x - O_0.x| - |U.x - L_0.x| = 2 \times H \times tg\phi - H \times tg\chi \quad (3)$$

Výpočet dĺžky úsečky *CU* je analogický rovniciam 1 až 3, pričom sa počíta so súradnicami *y* miesto *x*, namiesto *WIDTH*, u_1 a α sa použije postupne *HEIGHT*, u_2 a β. Výpočet azimutu a výšky je potom udaný rovnicami (4) a (5), v ktorých sa v oboch zlomkoch vzdialenosť kamery od priemetne (*H*) sa vykráti a výsledné rovnice na ňom nie sú závislé.

$$výška = arctg\frac{H}{L_0U} = arctg\left(\frac{H}{\sqrt{|CL_0|^2 + |CU|^2}}\right) \quad (4)$$

$$azimut = arctg\frac{L_0C}{CU} \quad (5)$$

Popísané rovnice predpokladajú, že kamera je otočená tak, aby vrch obrazu získavaného z kamery bol nasmerovaný priamo k severu. Ak tento fakt neplatí postačuje pripočítať uhol otočenia kamery k výslednému azimutu.

5.3. Návrh testov

V tejto fáze tvorby projektu sa zameriame na akceptačné testovanie nami vytváraného produktu. Kapitola obsahuje možné scenáre pre použitie systému a očakávané akcie systému. Zameriava sa predovšetkým na ovládanie systému prostredníctvom ukazovadla. Ďalšia časť systému, pridávanie informácií do databázy prostredníctvom webového rozhrania nie je zahrnutá, keďže v momentálnej fáze projektu by boli testovacie scenáre príliš všeobecné.

ID: Scenár 1

Názov: Určenie hviezdy

Vstupné podmienky: nie sú

Výstupné podmienky: Používateľovi sa zobrazí obrazovka s informáciami o vybranej hviezde

Krok	Akcia	Očakávaná reakcia
1	Používateľ namieri ukazovadlo na sklo tak, aby bod ukazoval z jeho pohľadu na želaný objekt a potvrdí voľbu	Systém zobrazí obrazovku s informáciami o najbližšej hviezde

ID: Scenár 2

Názov: Navigácia k nebeskému telesu

Vstupné podmienky: nie sú

Výstupné podmienky: Používateľ sa pomocou navigácie dostane k želanej hviezde, v prípade módu, kde má užívateľ nájsť systémom určenú hviezdu je výstupnou podmienkou správne určenie, či bola ukázaná správna hviezda a výpis príslušnej správy

Krok	Akcia	Očakávaná reakcia
1	Používateľ zadá názov hviezdy, ktorú chce vyhľadať	Systém ho hlasovo naviguje k zadanej hviezde
2	Používateľ sa dostatočne priblíži ukazovadlom k zadanej hviezde	Systém používateľa upozorní a vypíše informácie o vybranej hviezde

Alternatívny scenár:

Krok	Akcia	Očakávaná reakcia
1	Používateľ vyberie možnosť dotazovania na polohu hviezdy zo strany systému	Systém vyberie hviezdu z vybranej časti hviezdnej oblohy a požiada používateľa o jej nájdenie prostredníctvom ukazovátka
2	Používateľ vyberie hviezdu a potvrdí voľbu	Systém vypíše správu, či používateľ vybral hľadanú hviezdu, alebo nie

ID: Scenár 3

Názov: Ovládanie aplikácie

Vstupné podmienky: užívateľ namieri laserový lúč na monitor

Výstupné podmienky: Kurzor myši sa presunie na vybrané miesto

Krok	Akcia	Očakávaná reakcia
1	Používateľ namieri ukazovadlo na sklo tak, aby bod ukazoval z jeho pohľadu na želaný objekt a potvrdí voľbu	Systém zobrazí obrazovku s informáciami o najbližšej hviezde
2	Používateľ potvrdí voľbu	Systém reaguje analogicky, ako pri potvrdení voľby prostredníctvom myši

6. Prototyp

Kapitola poskytuje prehľad prototypovaných častí systému. Prvá časť kapitoly uvádza dôvody, ktoré nás viedli k výberu častí, ktoré sme zaradili do prototypu. Nasledujúce odseky kapitoly uvádzajú popis funkcionality vytvorených častí systému.

6.1. Analýza rizík

Cieľom našej práce na prototype bolo najmä overenie základných schopností systému. Za najdôležitejšiu a zároveň najkritickejšiu úlohu považujeme určenie, na ktorú hviezdu sa používateľ systému pozerá. Z toho dôvodu sme sa rozhodli implementovať nasledujúce funkcie:

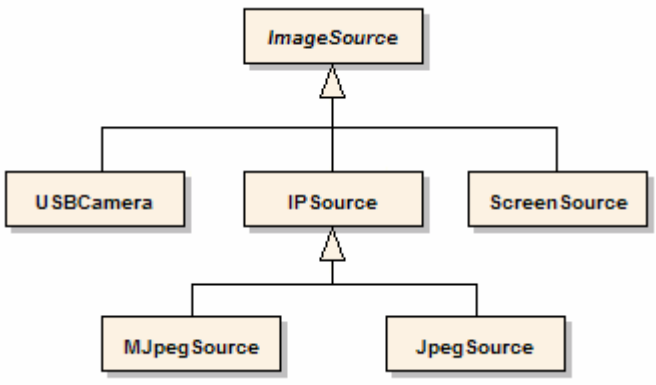
- Získanie obrazu z kamery
- Hľadanie bodu v obraze a vypočítanie azimutu a výšky
- Prepočet hvezdárskych súradníc a určenie najbližšej hviezdy

Pre umožnenie testovania v reálnych podmienkach sme moduly implementujúce uvedené funkcie integrovali do systému s jednoduchým používateľským rozhraním určeným na testovanie.

Menej kritickou funkciou systému je ovládanie hlasom. Dôležitým je ovládanie ako také, preto sme sa rozhodli v prototype vytvoriť jednoduchší variant a pokročilé ovládanie pomocou používateľom definovaných povelov sme ponechali na neskoršiu fázu vývoja systému. Pre overenie schopnosti prepojenia na databázu sme do prototypu zaradili lokálnu databázu.

6.2. Získavanie obrazu z kamery

Prvým vytvoreným modulom systému bol VideoSource. Jeho úlohou je poskytovať systému obraz z pripojenej webovej, alebo IP kamery. Zjednodušený diagram tried komponentu (pozri Obr. 14) ukazuje závislosť medzi spôsobmi pripojenia na kameru. Po výbere typu kamery stačí vo zvyšku systému stačí používať

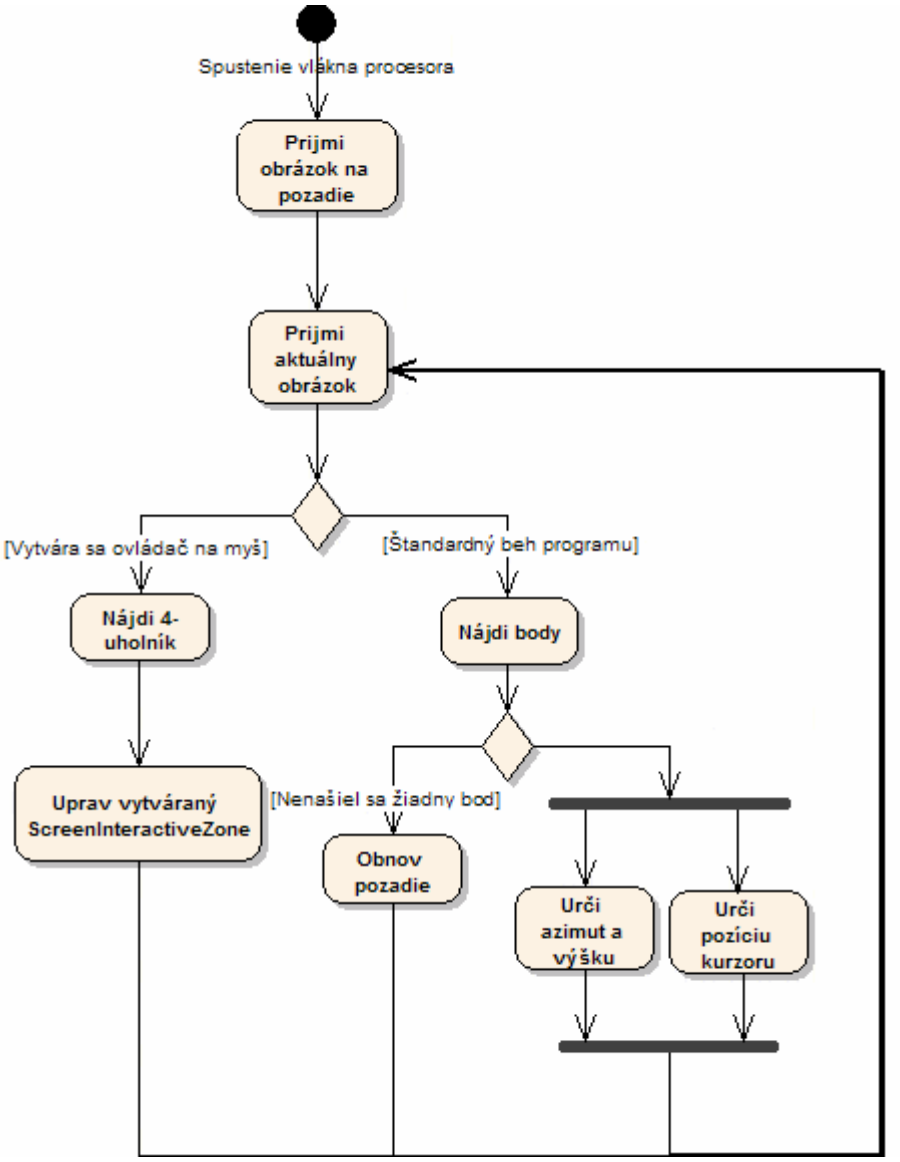


Obr. 14. Diagram tried modulu VideoSource

základnú triedu ImageSource, ktorá po získaní obrázku vyvoláva príslušnú udalosť. Testovanie komponentu s rôznymi rozlíšeniami obrazu pri 1 až 30 obrázkoch za sekundu preukázalo, že príjem obrazu z jednej kamery neprináša žiadne podstatné výkonnostné nároky na systém.

6.3. Hľadanie bodu v obraze a výpočet azimutu a výšky

Spracovanie obrazu z kamery zabezpečuje v systéme modul ImageProcessor. Procesor pracuje v cykloch (pozri Obr. 15) začínajúcich príjmom obrázka. Nasleduje vyhľadanie bodov a ich vyhodnotenie. Ďalší obrázok sa prijíma až keď je predchádzajúci spracovaný, čím sa zamedzí zdržovaniu systému.



Obr. 15. Cyklus ImageProcessora

Vyhľadávanie útvarov v obraze začína porovnaním aktuálneho obrazu s obrazom pozadia v čierno-bielom formáte. V získanom binárnom poli sa následne určia okraje zmenených oblastí tak, že sa postupuje po bodoch binárneho poľa označených ako true pridŕžajúc sa jednej strany oblasti až k začiatočnému bodu okraja (pozri Obr. 16). Ak je rozpoznaná oblasť dostatočne veľká, teda nie je prehlásená za šum, jej stred predstavuje hľadaný bod.

Obr. 16. Určovanie okraju oblasti

Začína sa na prvom nájdenom zmenenom bode (prvý zhora a zľava), potom postupuje po zmenených bodoch tak, aby po ľavej strane bol vždy nezmenený

Azimut a výška sa počítajú podľa návrhu danom kapitolou 5.2 Určovanie pozície hviezd. Implementácia je doplnená o určenie kvadrantu kam sa používateľ pozerá, ktorý sa výpočtom dĺžky úsečiek stráca.

Prototyp obsahuje aj ešte nie kompletne otestovanú metódu pre aproximáciu danej oblasti štvoruholníkom, čo je dôležité pri ovládaní kurzoru myši pre nájdenie monitora. Algoritmus začína rovnako ako pri hľadaní bodu a pokračuje hľadaním 4 lokálnych maxím vzdialenostnej funkcie. Vzdialenostná funkcia predstavuje vzdialenosť bodu okraju od stredu. Šum a aproximácia spojitého priestoru len niekoľkými pixlami spôsobí, že lokálnych maxím je viac než potrebujeme. Preto sa po rozpoznaní N maxím funkcie tieto filtrujú tak, že body [k+1] sa odstráni vtedy, ak je vo vzdialenosti L od priamky medzi bodmi [k] a [k+2]. Vzdialenosť L sa zvyšuje až kým nemáme iba hľadaný počet vrcholov. Prototyp je doplnený aj o výpočet novej pozície kurzoru myši na základe vzdialeností od hrán 4-uholníka. Prototyp však neobsahuje nahradenie oblasti monitora v obrázku pozadia aktuálnym snímkom monitora, nakoľko potrebné projektívne zobrazenie plánujeme do systému implementovať až v letnom semestri.

6.4. Určenie najbližšej hviezdy

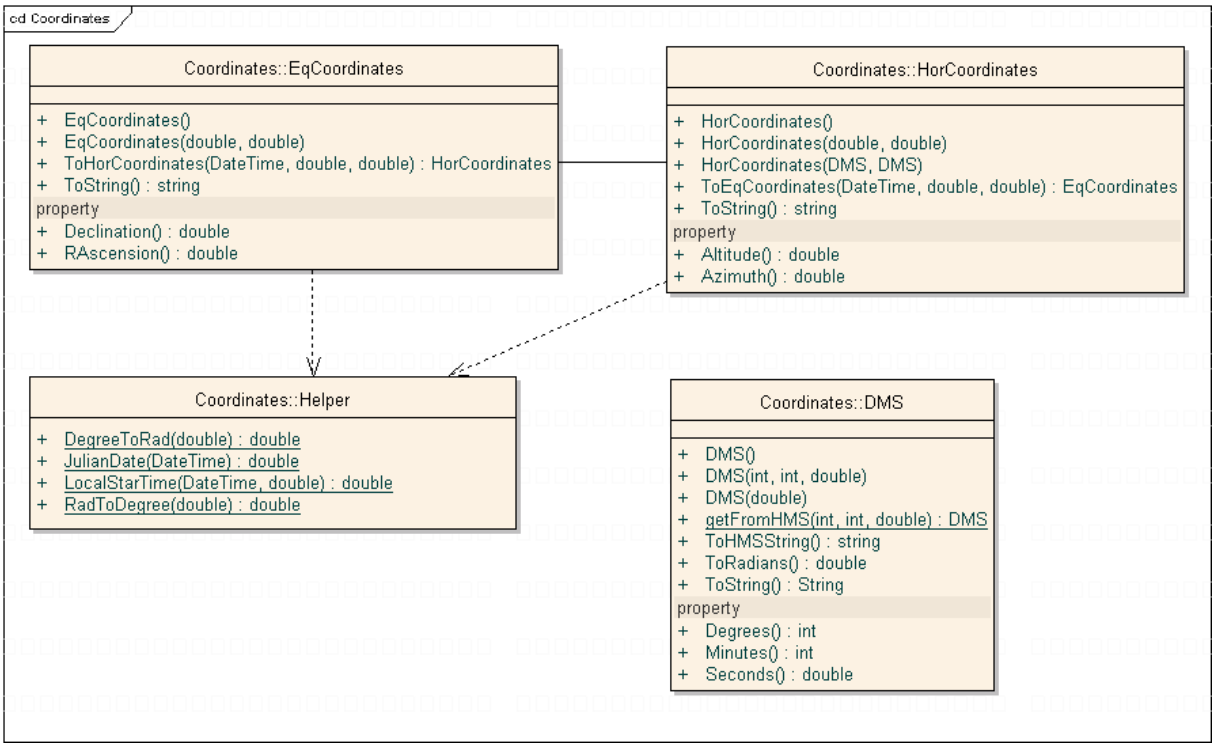
Vyhľadávanie v hvezdárskych katalógoch zabezpečujú v rámci prototypu nasledovné moduly:

- Coordinates – prevody súradníc medzi jednotlivými súradnicovými systémami a pomocné triedy pre prácu s uhlami
- Star Catalogue – katalógy nebeských telies, vyhľadávanie v katalógoch na základe ich špecifických súradnicových sústavách
- Star Finder – správa katalógov pre vyhľadávanie, prevod medzi súradnicovými sústavami pre konkrétne katalógy

6.4.1. Modul Coordinates

Modul Coordinates zabezpečuje prevod medzi rôznymi súradnicovými sústavami. Pre úroveň prototypu je implementovaný iba prevod medzi rovníkovou a horizontálnou súradnicovou sústavou.

Súradnice sú pre každú sústavu reprezentované jednou triedou. Trieda EQCoordinates predstavuje súradnice v rovníkovej súradnicovej sústave, v ktorej sa udáva poloha hviezd. Vlastnosti triedy sú deklinácia a rektascenzia v rovníkovej súradnicovej sústave. Pre horizontálnu súradnicovú sústavu sa používa trieda HorCoordinates, ktorá obsahuje vlastnosti azimut a výšku. Obidve triedy okrem konštruktorov obsahujú metódy pre prevod na opačnú súradnicovú sústavu. Tieto metódy vyžadujú ako parametre čas pozorovania a geografické súradnice príslušného miesta na Zemi, keďže horizontálna súradnicová sústava je na nich závislá. Modul obsahuje ďalšie dve pomocné triedy. Metódy triedy Helper poskytujú prevod na Juliánsky dátum a výpočet lokálneho hviezdneho času, ktoré sú potrebné pri prepočte súradníc. Trieda DMS zapuzdruje uhly, ktoré môžu byť pomocou metód a vlastností tejto triedy získavané v stupňoch, radiánoch alebo v hodinách podľa potreby. Diagram tried môžeme vidieť na Obr. 17.



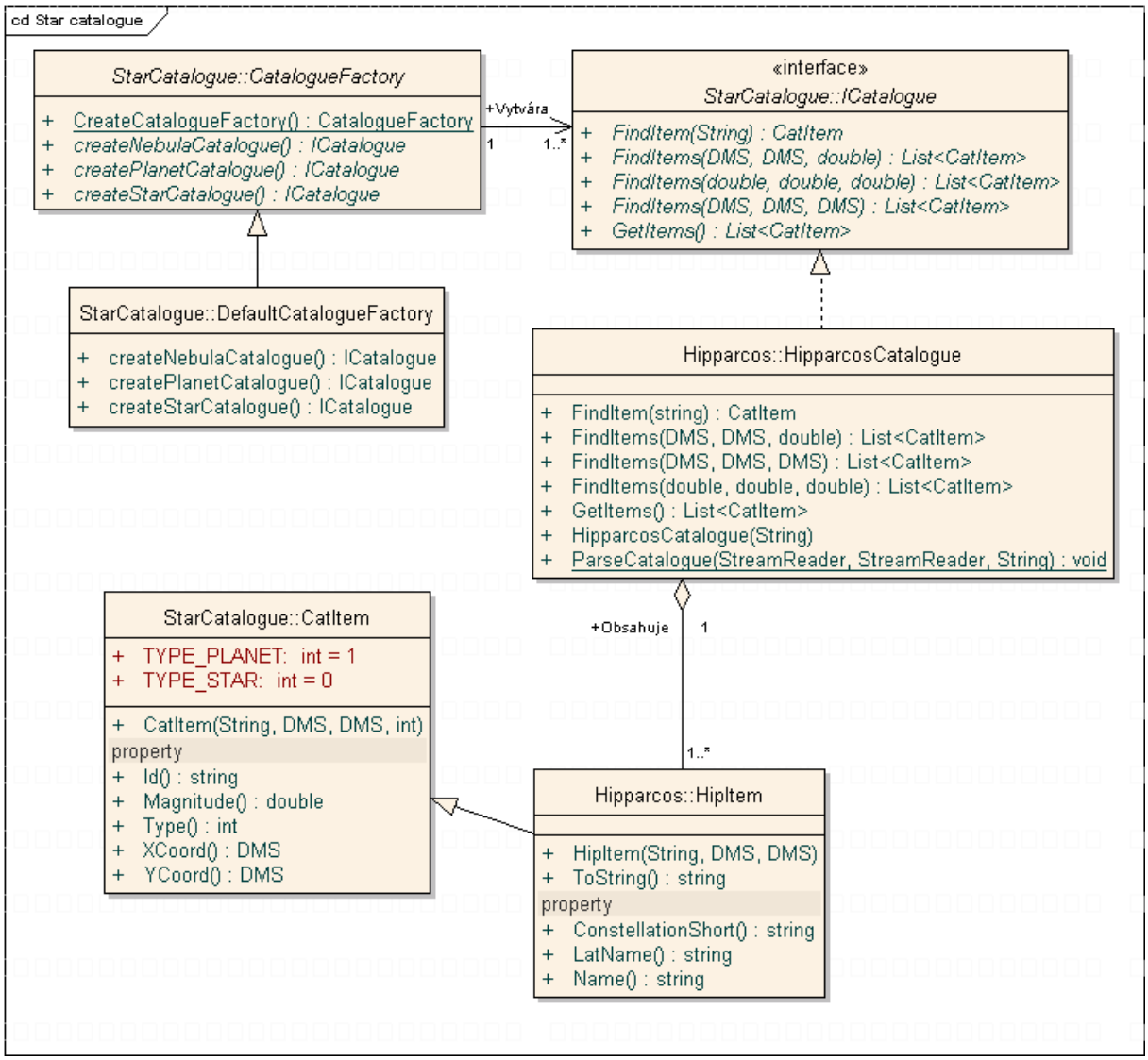
Obr. 17. Diagram tried modulu Coordinates

6.4.2. Modul Star Catalogue

Triedy modulu Star Catalogue zabezpečujú správu a načítavanie hviezdnych katalógov. Prístup ku katalógu nebeských telies definuje rozhranie ICatalogue. Deklarované metódy slúžia pre prístup k položkám katalógu cez id v alebo súradnice. Pre úroveň prototypu je implementovaný katalóg hviezd, ktorý vychádza z katalógu Hipparcos. Poloha hviezd sa v tomto katalógu udáva v rovníkovej súradnicovej sústave a preto metódy v katalógu očakávajú súradnice v tejto súradnicovej sústave. Trieda HipparcosCatalogue používa XML súbor pre reprezentáciu katalógu ako jednoduchú databázu. Preto obsahuje aj metódu ParseCatalogue, ktorá tento súbor vytvorí na základe originálneho súboru katalógu.

Trieda CatItem obsahuje vlastnosti nebeských telies a bude slúžiť na ich reprezentáciu. V tejto triede sú definované iba súradnice, magnitúda, typ a identifikácia položky podľa katalógu. Samotné hviezdy katalógu Hipparcos sú zastúpené triedou HipItem, ktorá dedí vlastnosti triedy CatItem a v súčasnej verzií dopĺňa názov hviezdy, pokiaľ je tento definovaný.

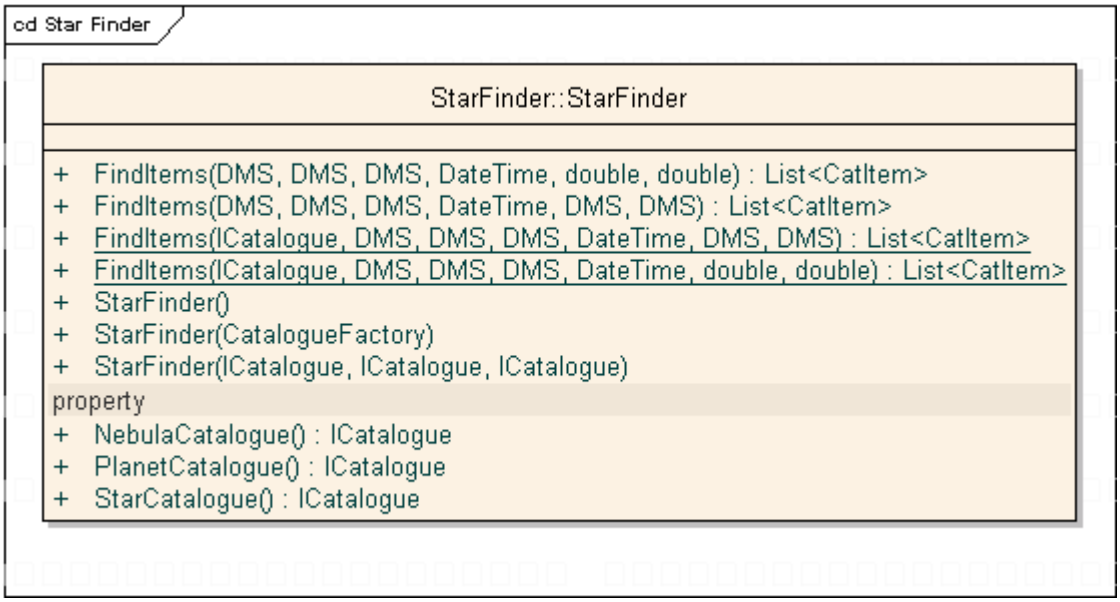
Vytváranie katalógov zabezpečujú potomkovia triedy CatalogueFactory. Na úrovni prototypu ide o jednu triedu DefaultCatalogueFactory, ktorá implementuje iba metódu createStarCatalogue, kde vytvorí inštanciu triedy HipparcosCatalogue. Triedy a vzťahy medzi nimi v module Star Catalogue sú zobrazené na Obr. 18.



Obr. 18. Diagram tried modulu Star Catalogue

6.4.3. Modul Star Finder

Úlohou modulu Star Finder je vyhľadávať v katalógoch hviezd podľa zadaných súradníc. Na úrovni prototypu obsahuje jednu triedu StarFinder, ktorá zapuzdruje vyhľadávanie v katalógoch na základe súradníc horizontálnej súradnicovej sústavy. Trieda je zobrazená na Obr. 19. Jeho úlohou je tiež zjednodušiť správu katalógov a zabezpečenie vytvorenia iba jednej inštancie tried katalógov. Metódy pre vyhľadávanie v katalógu očakávajú súradnice v horizontálnej súradnicovej sústave, uhlovú vzdialenosť, do ktorej sa vyhľadáva a čas a miesto pozorovania. V prototype sa vyhľadáva iba v katalógu hviezd, vo finálnej verzii bude vyhľadávanie rozšírené o hľadanie všetkých prírodných viditeľných telies v danej oblasti. Okrem toho trieda umožňuje aj priamy prístup ku katalógom pomocou svojich vlastností.



Obr. 19. Trieda StarFinder zabezpečujúca vyhľadávanie v katalógoch

6.5. Ovládanie hlasom

Jednou z funkcionálnych požiadaviek uvedených v špecifikácii projektu je ovládanie aplikácie hlasom. V prototype je ovládanie aplikácie hlasom riešené použitím nástroja SAPI. Ten umožňuje rozpoznávať anglicky hovorené slová a na základe ich rozpoznania vykonať určenú činnosť. Aplikácia obsahuje základnú množinu príkazov, ktoré je možné aktivovať hlasom. Tieto príkazy sú definované v externom súbore, takže je možné ich meniť.

Nevýhodou a obmedzením použitia SAPI je možnosť rozpoznania iba anglických slov. A aj napriek tomu, že nie je predpokladaná veľká množina hlasových príkazov na ovládanie, môžu nastať problémy s rôznou výslovnosťou definovaných hlasových povelov. Riešením tohto problému by mal byť vlastný algoritmus na rozpoznávanie hlasových príkazov vo finálnej verzii projektu. Jeho princíp by bol založený na porovnávaní preddefinovaných hlasových povelov so vstupmi z mikrofónu. Tým by sa vyriešil problém rôznej výslovnosti slov rôznymi používateľmi, pretože každý používateľ by do nastavení aplikácie vložil vzory svojich vlastných hlasových povelov.

Prezentácia informácií používateľovi tiež využíva možnosti SAPI, konkrétne prevod textu na hovorené slovo. Text, ktorý je transformovaný na reč je získaný z HTML kódu. Ten však obsahuje veľa metainformácií, ktoré nie je vhodné podávať používateľovi. HTML kód preto obsahuje vopred definované značky na výber iba tých častí, ktoré sú podstatné. Tieto značky nemajú vplyv na zobrazenie obsahu HTML stránky v rozhraní aplikácie. Pri

transformácii textu na reč sú najskôr získané vybrané časti a následne sú používateľovi prezentované zvukovou formou.

6.6. Lokálna multimediálna databáza

Multimediálne informácie v systéme sú viacerých druhov. Medzi základné patria tie, ktoré plánujeme implementovať v letnom semestri – 3D interaktívne pohyby v slnečnej sústave, „rozbalenie“ mapy mesiaca, a aktuálny výrez oblohy s žiariacimi nebeskými telesami doplnenými o obrázky súhvezdí. Takto definovať vieme len niekoľko objektov, pričom množstvo ľudí má podrobnejšie informácie a môže sa o nich podeliť v kolaboratívnej encyklopédii. Do prototypu sme sa rozhodli zaradiť lokálnu databázu multimediálnych údajov, ktorá bude neskôr aktualizovaná na základe zdieľanej kolaboratívnej encyklopédie. Realizovať ju v tejto fáze projektu sme sa rozhodli z dôvodu potreby databázy na počítači bez plnohodnotného databázového systému. Zvolili sme súborovú databázu.

6.6.1. Dostupné súborové databázy

Pre implementáciu lokálnej databázy sme zvolili súborovú relačnú databázu, ktorá uloží celú databázu v jednom súbore na disku. Takto aplikácia nebude závislá na externom databázovom serveri. Ako možnosti pre súborovú databázu sme zvažovali MS Access a SQLite.

MS Access

MS Access je nástroj na správu relačnej databázy od spoločnosti Microsoft, je súčasťou balíka Microsoft Office. Kombinuje relačný systém Microsoft Jet Database Engine (systém práce s databázou, na ktorom bola postavených niekoľko produktov Microsoftu) s grafickým používateľským rozhraním.

Výhody

- Jednoduchý, použiteľný a známy pre všetkých členov tímu
- Jednoducho akceptuje dáta z iných databázových systémov, ktoré majú ODBC konektivitu (Oracle, SQL Server)
- Dobrá spolupráca s .NET rámcom

Nevýhody

- Nemá spúšťacie udalosti (trigger) databázy a uložené procedúry. Nie je vhodný pre správu veľkých databáz
- Málo typov dát

SQLite

SQLite [23] je relačný databázový systém obsahujúci v relatívnej malej knižnici napísanej v jazyku C. Je vyvíjaný D. Richardom Hippom a šírený pod licenciou verenej domény. SQLite pracuje bez serveru, ktorý by sa musel štartovať a spravovať, využíva vlastné súbory na lokálnom disku.

Výhody

- Maximálna veľkosť databázy 2 teraslabiky
- Veľkosť reťazca a BLOB-u je obmedzená len voľnou pamäťou systému
- Jednoduché, ľahko pochopiteľné API
- Voľne dostupná databáza aj so zdrojovými kódmi

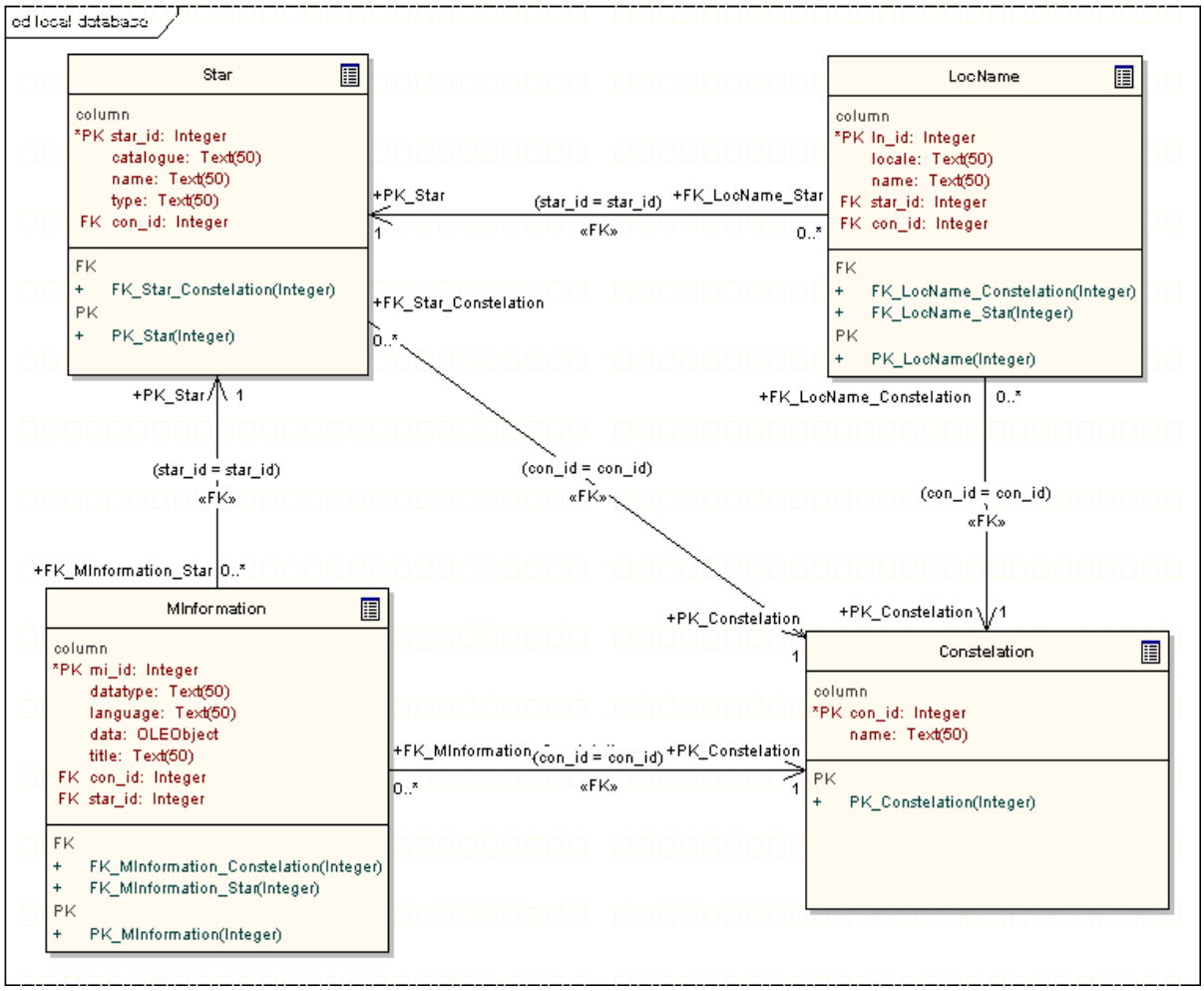
Nevýhody

- Nie všetky vlastnosti jazyka SQL sú podporované
- Povolená iba jedna aktívna transakcia
- Chýbajúce skúsenosti s databázou

Pre lokálnu databázu údajov sme vybrali databázový systém MS Access, pretože je lepšie podporovaný na platforme .NET a tím má s jeho použitím väčšie skúsenosti.

6.6.2. Lokálna databáza

Fyzický model údajov vychádza z logického modelu pre lokálnu databázu. Doplnené sú typy údajov podľa zvolenej databázy, privátne a cudzie kľúče. Fyzický model údajov je zobrazený na Obr. 20 ako diagram tabuliek.



Obr. 20. Diagram tabuliek lokálnej databázy

6.7. Zhodnotenie prototypu

Prototyp mal overiť základné predpoklady použitia systému. Podarilo sa nám implementovať a otestovať rozpoznávanie bodov v obraze. Využitím nájdených bodov určuje systém smer pohľadu používateľa. Prvý test v reálnych podmienkach odhalil značnú nepresnosť, ktorá - ako sa neskôr ukázalo – bola spôsobená chybou vo výpočtoch. Napriek tomu sme stanovili postup ako zabezpečiť vyššiu presnosť. Prvou pomôckou je umiestnenie hlavy na látku („húpaciu sieťku“) udržiavajúcu konštantnú vzdialenosť hlavy od skla. Druhý krok uľahčuje postup určovania sklonu kamery tak, že sa vždy bude mieriť na stred stola, pričom systému sa v budúcnosti budú zadávať len rozmery tohto stola. Nakoniec prototyp poukázal na fakt, že ukazovadlo by bolo lepšie umiestniť nie na kraj hlavy, ale medzi oči, čo používateľovi zjednoduší výber pozorovaného objektu. Druhý test v reálnych podmienkach sme plánovali vykonať posledný týždeň semestra, no neustále zamračená obloha nám to nepovolila. Vzorové príklady však systém už spracovával správne aj po integrácii komponentov pre prácu s hviezdami.

Podarilo sa nám implementovať pre astronomický systém nevyhnutné prepočty hvezdárskych súradníc, ktorých správnosť sme overili na hviezdach. Naštudovali sme literatúru potrebnú pre zvládnutie vyhľadávania planét a voľným okom viditeľných hviezdokôp, čo nám umožní bezproblémovú implementáciu tejto funkcionality do finálnej verzie produktu.

Do prototypu sme sa rozhodli zaradiť základné ovládanie hlasom, ktoré je v terajšej verzii určené na testovacie účely a preto sme ho so zvyškom systému ešte neintegrovali. Pre stanovené obmedzenia na prototyp sa javí ako postačujúce rovnako ako aj čítanie filtrovaného HTML pomocou syntézy reči.

Vytvorený prototyp pokladáme za dobrý základ pre úspešnú implementáciu celého systému.

7. Podrobný návrh

Kapitola poskytuje prehľad kľúčových návrhových rozhodnutí pri realizácii systému icPoint. Jej prvá časť uvádza výsledok práce na module pre analýzu obrazu. Druhá časť kapitoly sa zameriava na detailný návrh katalógu nebeských telies a algoritmov pre vyhľadávanie v nich. Tretia podkapitola uvádza podrobný návrh lokálnej databázy.

Štvrtá časť kapitoly prezentuje hlavný modul aplikácie, ktorý spája všetky komponenty systému, ktorým zároveň poskytuje potrebné grafické používateľské rozhranie.

Záver kapitoly je vyhradený pre opis podrobného návrhu zdieľanej databázy, webovej aplikácie slúžiacej pre tvorbu kolaboratívnej encyklopédie a webovej služby pre prístup k vytvoreným dátam.

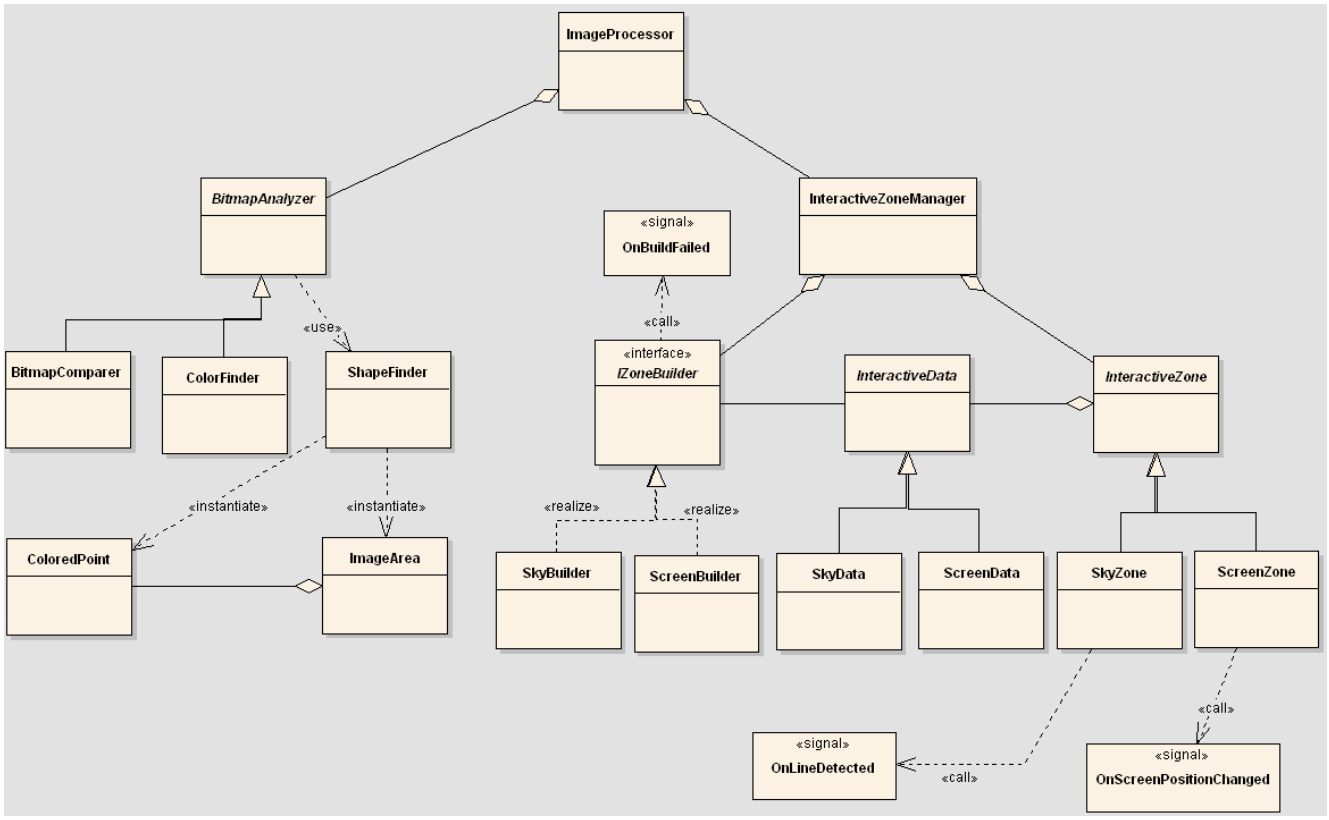
7.1. Analýza obrazu

V kapitole 6.3 sme predstavili spôsob hľadania bodov v obraze, ktorý bol použitý pri tvorbe prototypu. Jeho základnou myšlienkou je porovnávať aktuálny snímok z kamery s obrazom pozadia, ktorý sa príležitostne obnovuje. Spôsob sa už v čase testovania prototypu potvrdil ako užitočný pre vyhľadávanie bodov potrebných pre výpočet smeru pohľadu používateľa k hviezdam. Po dokončení ďalšej funkcionality sa však ukázalo, že pre potreby určovania pozície kurzoru myši na obrazovke nie je vhodný a museli sme pre tento účel vymyslieť alternatívnu procedúru.

Prvá časť tejto podkapitoly prezentuje základ vnútornej štruktúry modulu ImageProcessor. Nasledujúce dve časti rozoberajú metódy výpočtu azimutu a výšky, a určovania novej pozície kurzoru myši.

7.1.1. Podrobný návrh modulu ImageProcessor

Potreba používania dvoch rôznych metód vyhľadávania vyústila do nutnosti rozšíriť existujúcu vnútornú štruktúru komponentu ImageProcessor (pozri Obr. 21). Pri návrhu súčiastky sme sústredili na to, aby bolo možné nie len jednoducho pridávať spôsoby vyhodnocovania bodov (Interaktívne zóny), ale aj pohodlne vytvárať nové druhy analyzátorov obrazu (BitmapAnalyzer).



Obr. 21. Diagram tried komponentu ImageProcessor

Základnú fasádu komponentu vytvára trieda ImageProcessor. Navonok komponentu je okrem nej viditeľná len abstraktná trieda InteractiveData a jej podtriedy SkyData a ScreenData. Tie slúžia pre komunikáciu vlastností vyhľadávania a zabezpečujú notifikáciu o jeho výsledkoch pomocou signálov (delegáty a udalosti) OnLineDetected, OnScreenPositionChanged a oznamovanie chybového stavu využitím OnBuildFailed.

Trieda ImageProcessor obsahuje zoznam analyzátorov obrazu BitmapAnalyzer – vždy minimálne jeden BitmapComparer, ktorý dopĺňa niekoľko ColorerFinderov v závislosti od parametrov vyhľadávania. Táto trieda zabezpečuje základné operácie nad obrazom z kamery, ktorý rozdeľuje na zóny pre jednotlivé vyhľadávače tak, že každý jeden BitmapAnalyzer si udržuje zoznam oblastí obrázka, v ktorých nemá vyhľadávať. Jej podtriedy BitmapComparer a ColorFinder definujú samotný algoritmus označovania pixlov. Nad výsledným binárnym poľom pixlov, kde každá hodnota označuje zaujímavosť bodu, sa následne realizuje identifikácia ucelených oblastí pomocou triedy ShapeFinder, ktorá slúži na identifikovanie stredov oblastí a na rozpoznanie hľadaných n-uholníkov. Na základe výsledkov hľadania potom v spolupráci s triedou BitmapAnalyzer vytvorí buď zoznam nájdených bodov ColoredPoint alebo nájdených tvarov ImageArea.

Parametre potrebné pre výpočet smeru pohľadu sa do komponentu dostávajú v rámci inštancií podtried abstraktnej triedy InteractiveData. Na základe nimi poskytnutých údajov sa v triede InteractiveZoneManager vytvára konkrétny staviteľ interaktívnej zóny reprezentovaný rozhraním IZoneBuilder. Implementácie tohto rozhrania slúžia na tvorbu samotných zón SkyZone a ScreenZone (dediacich od abstraktnej triedy InteractiveZone) zabezpečujúcich samotný výpočet smeru pohľadu – buď azimutu a výšky, alebo novej pozície kurzoru myši. O správu staviteľov (IZoneBuilder) a vytvorených interaktívnych zón (InteractiveZone) sa stará trieda InteractiveZoneManager, na ktorú fasáda komponentu (trieda ImageProcessor) väčšinu požiadaviek príslušného typu po úvodnej kontrole priamo deleguje.

7.1.2. Smer pohľadu k hviezdam

Výpočet azimutu a výšky sme v letnom semestri už nemodifikovali. Je založený na porovnávaní obrazu pozadia s aktuálnym snímkom z kamery, kde sa identifikujú dva body ako odrazy z laserového ukazovadla na skle. Implementácia samotných rovníc pre určenie azimutu a výšky nad týmito bodmi bola už priamočiarou realizáciou návrhu (pozri kapitolu 5.2). Rozsiahlejšie úvahy boli potrebné len v súvislosti s určovaním správneho kvadrantu a príslušnou úpravou získaných uhlov.

Porovnávanie dvoch obrázkov ako základ pre vyhľadanie bodov sme zvolili preto, lebo v obraze z kamery môže byť množstvo rušivých elementov. Okrem klasického šumu treba počítať s tým, že na skle sa objavia odrazy z pouličného osvetlenia, poprípade bude vidieť svetlo z Mesiaca. Vyhľadávanie odrazov svetla z ukazovadla čisto na základe ich farby by preto neviedlo k správnym výsledkom.

7.1.3. Pozícia kurzoru myši

Ovládanie kurzoru myši vyžaduje prípravnú fázu založenú na nájdení štvoruholníka monitoru v obraze z kamery a na zistení jeho otočenia. Jej postup je nasledovný:

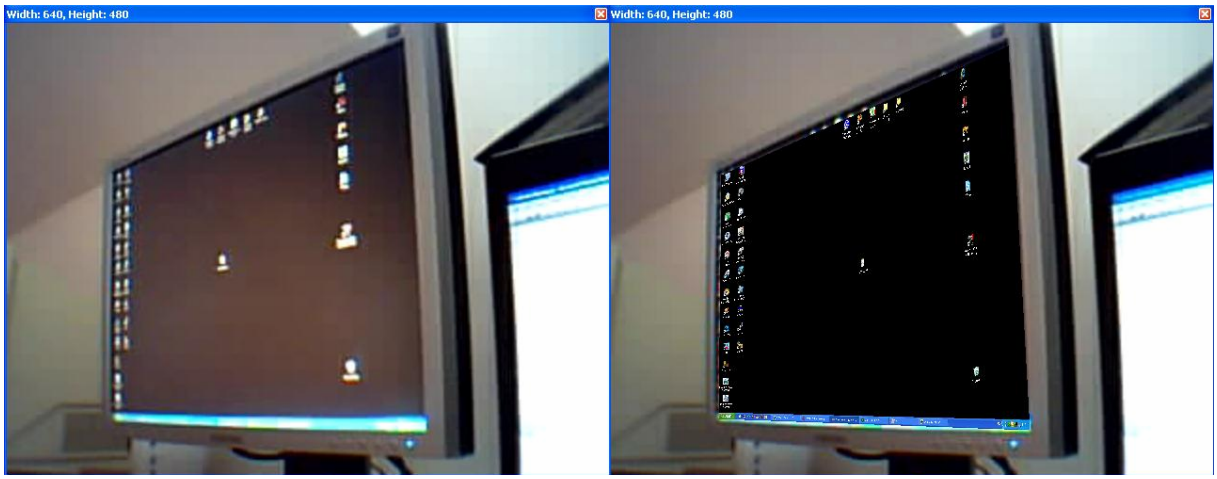
1) Prijmi obraz z kamery a obnov podľa neho pozadie v Compareri
2) Zmeň farbu celého displeja na bielu
3) Prijmi niekoľko (vždy minimálne jeden, no dobré je viac pre prípad kamery s automatickým prispôsobovaním svetelnosti) obrázkov a obnov podľa nich pozadie v BitmapComparer-i.
4) Rozdeľ monitor na 4 časti a jednotlivé štvrtiny (rohy obrazovky) vyplň postupne červenou, zelenou, modrou a čiernou farbu (pozri Obr. 22).

Obr. 22. Farby monitora pre určenie jeho otočenia v obraze z kamery

5) Prijmi snímok z kamery a hľadaj v ňom štvoruholník. Základ postupu nájdenia 4-uholníka je opísaný v časti 6.3:
   a) Nájdi okraje zmenenej oblasti
   b) Urči lokálne maximá vzdialenostnej funkcie (funkcia predstavujúca vzdialenosť bodu okraju od stredu). Vzhľadom na šum a aproximáciu spojitého priestoru v reálnom svete len niekoľkými bodmi v digitálnom svete, je lokálnych maxím pomerne veľa a treba ich ďalej filtrovať.
   c) Filtruj maximá tak, že sleduješ kedy začínajú klesať a stúpať. Ponechaj len zmeny. Tento krok slúži najmä pre urýchlenie nasledujúceho kroku tým, že zmenší počet vrcholov.
   d) Postupuj po všetkých zostávajúcich bodoch a o každom over, či neleží na priamke tvorenej jeho susednými bodmi. Toleranciu pre to, čo znamená ležať na priamke (vzdialenosť od priamky) postupne zvyšuj, až kým neostanú len 4 vrcholy.
6) Zisti, ktoré z otočení nájdenej oblasti sa líši svojou farebnosťou najmenej od požadovaných farieb daných 4-tým bodom algoritmu) a vytvor podľa neho príslušný objekt triedy ImageShape.

Výstupom popísaného algoritmu je inštancia triedy ScreenZone, ktorej sa dodá tvar obrazovky v snímku z kamey (ImageShape). Pôvodne sme mali v úmysle realizovať ďalší krok – hľadanie ukazovadla - pomocou porovnávania pozadia a aktuálneho obrazu z kamery. To by povoľovalo mať na monitore aj rozsiahle oblasti podobnej farby ako produkuje laserové ukazovadlo. Implementovali sme aj príslušné projektívne zobrazenie, ktoré vždy modifikovalo príslušnú časť pozadia obrazu z kamery aktuálnym „screenshot-om“ obrazu produkovaného grafickou kartou na daný monitor. Obr. 23 ilustruje problémy, spôsobené skreslením kamery. Ďalšie zdokonaľovanie zobrazovania by malo odstrániť drobné

nedostatky v podobe niektorých zaoblení v obraze, no neboli sme schopní odstrániť tak výrazné zmeny farieb v závislosti od svetelnosti a vzdialenosti jednotlivých častí obrazovky.



Obr. 23. Skreslenie kamery a projektívne zobrazenie

Riešením problému bolo použitie vyhľadávania oblastí zafarbených podobne ako produkuje laserové ukazovadlo. Pre každú interaktívnu zónu ovládajúcu kurzor myši (ScreenZone) sa preto vytvára nový ColorFinder. Ten v každom cykle ImageProcessora (prijatie obrázku, pozri kapitolu 6.3) v danej oblasti vyhľadáva body blízke požadovanej farbe ukazovadla. Výsledný efekt je uspokojivý a ovládanie kurzoru myši laserovým ukazovadlom sa stáva bezproblémovou činnosťou.

7.2. Katalóg nebeských telies

V prototype bolo implementované vyhľadávanie hviezd na základe katalógu Hipparcos. Z dôvodu vyhľadávania bolo potrebné doplniť aj algoritmy pre prevod medzi horizontálnou a rovníkovou súradnicovou sústavou. Tieto algoritmy boli implementované správne a vďaka návrhu tried katalógu je s nimi možné jednoducho pracovať a pridávať nové implementácie. Preto budú doplnené iba niektoré atribúty pre existujúce triedy podľa ďalších potrebných algoritmov. Zároveň bude potrebné dopracovať implementácie katalógov pre vyhľadávanie planét a ďalších nebeských telies

7.2.1. Modul Coordinates

Modul Coordinates zabezpečuje prevod medzi rôznymi súradnicovými sústavami pre ďalšie moduly. Jeho časť bola implementovaná v rámci prototypu, bude doplnená o triedy umožňujúcu prevody na ekliptikálnu súradnicovú sústavu. Návrh tohto modulu vychádza z objektového návrhu, pričom každý druh súradnicovej sústavy bude zastúpený triedou.

Inštancie tejto triedy budú predstavovať súradnice v danej súradnicovej sústave. Vzhľadom na používané sústavy bude modul obsahovať nasledovné triedy:

- HorCoordinates – trieda pre reprezentáciu súradníc v horizontálnej súradnicovej sústave, jej atribútmi budú azimuth (azimut) a altitude (výška)
- EqCoordinates – trieda pre reprezentáciu súradníc v rovníkovej súradnicovej sústave, jej atribútmi budú rAscension (rektascenzia) a declination (deklinácia)
- EclCoordinates – trieda pre reprezentáciu súradníc v ekliptikálnej súradnicovej sústave, jej atribútmi budú dĺžka (longitude) a šírka (latitude)

Triedy pre reprezentáciu súradníc budú obsahovať metódy pre vytvorenie inštancie reprezentujúcej inú súradnicovú sústavu podľa potreby prevodov v iných moduloch. Okrem týchto tried budú v module ďalšie pomocné triedy pre výpočty:

- Helper – pomocná trieda pre niektoré algoritmy, ktoré sú potrebné pre prevody medzi všetkými súradnicovými sústavami
- DMS – trieda pre zjednotenie reprezentácie uhlov. Aplikácia bude počítať uhly v radiánoch, ale niektoré sa musia zobraziť v stupňoch, minútach, sekundách alebo v hodinách, prípadne radiánoch. Prepočty pre požadované jednotky bude vykonávať táto trieda.

Na Obr. 24 sa nachádza diagram navrhovaných tried vrátane ich atribútov a metód.

Obr. 24. Diagram tried pre modul Coordinates

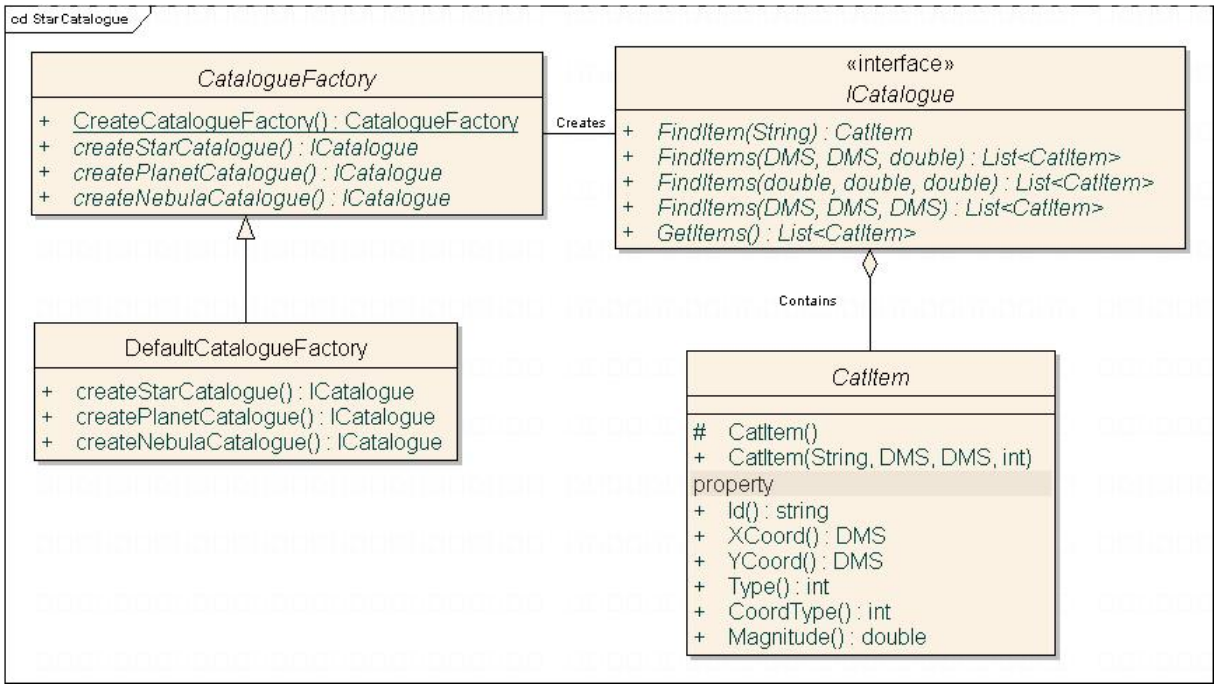
7.2.2. Modul Star Catalogue

Úlohou modulu StarCatalogue je zabezpečiť správu a načítavanie katalógov nebeských telies. Je potrebné oddeliť katalógy pre jednotlivé typy nebeských telies a umožniť pridávanie ďalších katalógov. Preto je základom modulu rozhranie ICatalogue, ktoré definuje metódy pre jednotlivé katalógy. Metódy rozhrania umožnia získať zoznam všetkých položiek katalógu a vyhľadávať v nich podľa id položky a podľa zadanej pozície. Toto vyhľadávanie bude očakávať parametre v rovníkovej súradnicovej sústave, pretože ide o základnú sústavu pre katalógy hviezd, hmlovín a iných objektov (okrem planét, pre ktoré je základnou sústavou ekvatoriálna).

Jedna položka v katalógu bude reprezentovaná pomocou potomka abstraktnej triedy CatItem. Definuje základné atribúty každej položky – id, x-súradnicu, y-súradnicu (pre danú sústavu), typ (hviezda, planéta ...), typ použitej súradnicovej sústavy (doplnenie oproti prototypu pre algoritmus vyhľadávania planét) a magnitúda telesa. Určenie typu nebeského telesa pri všeobecnom vyhľadávaní bude možné pomocou typu alebo priamo pomocou identifikácie typu potomka tejto triedy.

Vytváranie katalógov zabezpečujú potomkovia triedy CatalogueFactory. Definuje metódy pre vytvorenie katalógu planét, hviezd a v prípade potreby aj ďalších objektov. Základným potomkom je trieda DefaultCatalogueFactory, ktorá ako katalóg hviezd použije katalóg Hipparcos a pre planéty použije katalóg založený na algoritmoch z knihy Astronomické algoritmy pro kalkulátory[24]. Podľa možností budú doplnené ďalšie katalógy, najmä katalóg hmlovín a galaxií.

Na Obr. 25 sa nachádza diagram základných tried a rozhraní.

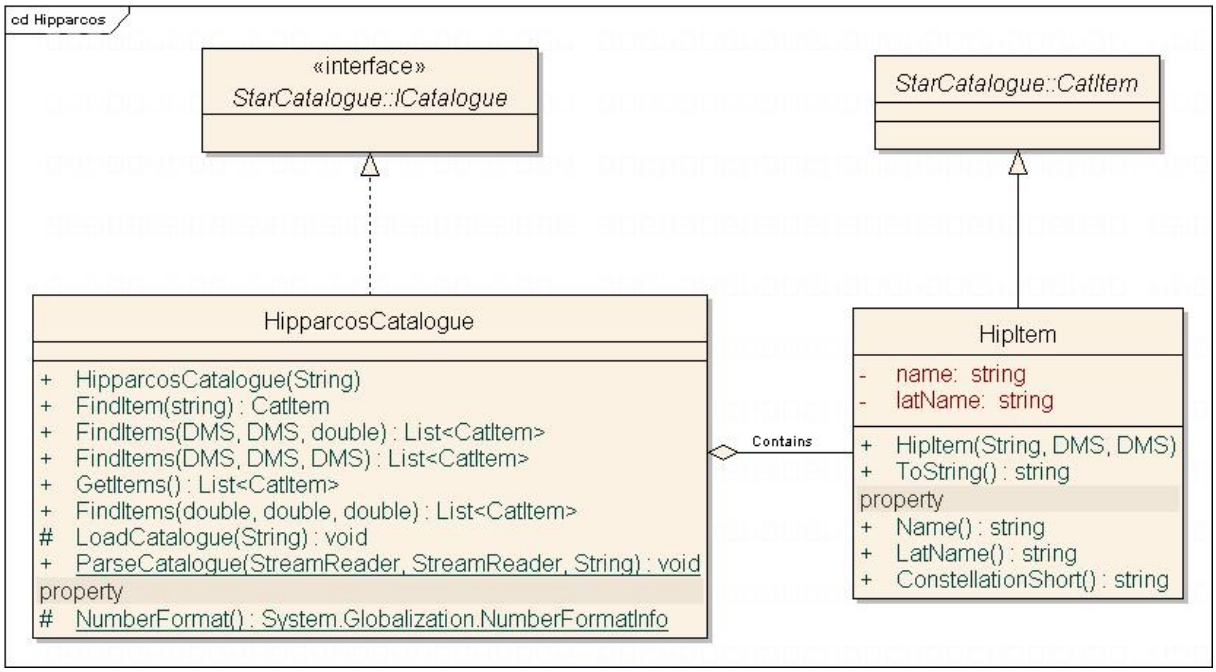


Obr. 25 Diagram tried pre rozhrania a základné triedy modulu StarCatalogue

Katalóg hviezd

Pre katalóg hviezd bol na základe analýzy problematiky vybraný katalóg Hipparcos. Tento katalóg je poskytovaný napríklad na adrese [25] ako textový súbor vrátane popisu jeho formátu. Potrebné údaje z tohto súboru bude obsahovať implementácia katalógu hviezd v externom XML súbore.

Triedy katalógu hviezd (pozri Obr. 1) boli implementované už v rámci prototypu a nebudú zmenené. Katalóg hviezd je reprezentovaný triedou HipparcosCatalogue, ktorá implementuje rozhranie ICatalogue. Okrem metód tohto rozhrania obsahuje metódy pre vytvorenie XML súboru zo súboru katalógu a načítanie XML súboru pri vytváraní inštancie triedy. Položky v katalógu sú reprezentované triedou HipItem, potomkom triedy CatItem. Dopĺňa atribúty pre uchovávanie latinského mena hviezdy a skratky súhvezdia.



Obr. 26 Triedy katalógu hviezd

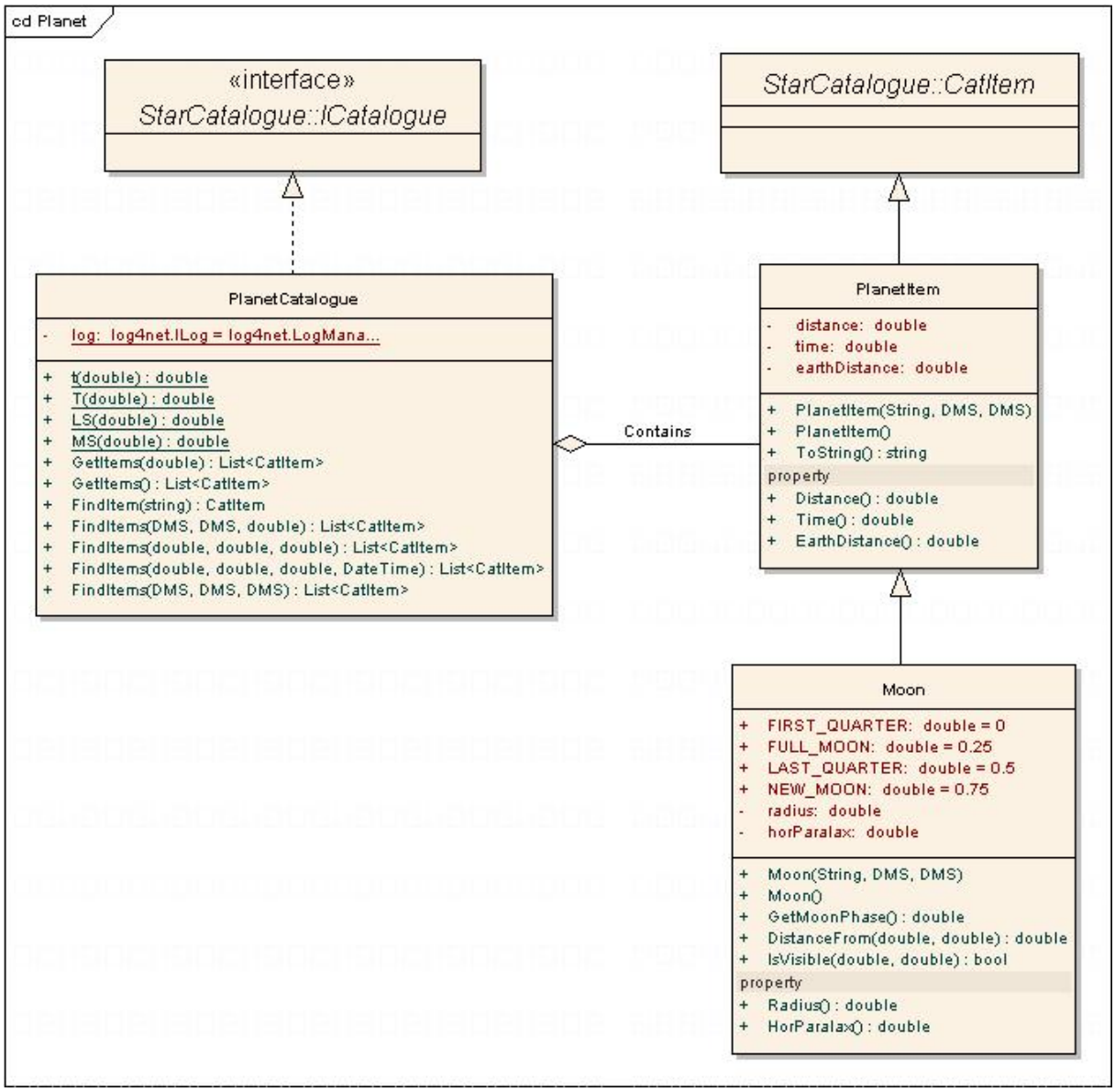
Katalóg planét

Planéty nemajú pevnú pozíciu v žiadnej súradnicovej sústave (ako hviezdy v rovníkovej), preto bude nutné ich pozície vypočítavať priamo pre zadaný čas. Algoritmy budú vychádzať z knihy Astronomické algoritmy pro kalkulátory [24]. Vypočítanie pozície planét na základe algoritmov budú mať za úlohu metódy triedy PlanetCatalogue, ktorá implementuje rozhranie ICatalogue. Okrem implementácie rozhrania obsahuje aj metódu pre priamy výpočet pozície v danom čase, metódy rozhrania ICatalogue ju budú volať s aktuálnym časom.

Položky katalógu (planéty) budú reprezentované triedou PlanetItem. Ide o potomka triedy CatItem a obsahuje ďalšie atribúty pre vzdialenosť od Slnka (distance), Zeme (earthDistance) a čas v ktorom bola pozícia vypočítaná. Sú potrebné najmä z dôvodu prepočtov medzi rôznymi súradnicovými sústavami. Mesiac je reprezentovaný zvlášť potomkom triedy PlanetItem, dopĺňa metódy pre zistenie fázy Mesiaca, uhlového polomeru

a zisťovanie, či prekrýva niektorú hviezdu na oblohe (táto metóda bude zrejme potrebná pre samotné zobrazovanie Mesiaca v aplikácií).

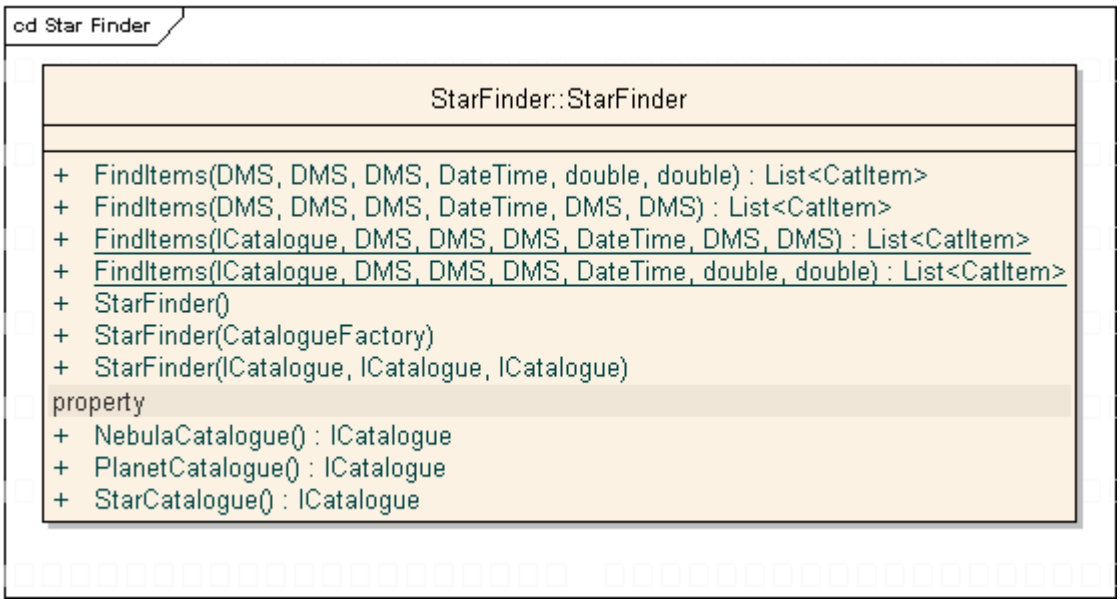
Na Obr. 27 sa nachádzaju zobrazené triedy katalógu planét.



Obr. 27 Triedy katalógu planét

7.2.3. Modul Star Finder

Úlohou modulu Star Finder je vyhľadávať v katalógoch hviezd podľa zadaných súradníc. Tento modul bol v rámci prototypu navrhnutý dostatočne a nebude zmenený. Obsahuje triedu StarFinder, ktorá zabezpečuje jednotný prístup ku katalógom a vyhľadávanie vo viacerých katalógoch zároveň. Trieda je zobrazená na Obr. 28.



Obr. 28. Trieda StarFinder zabezpečujúca prístup ku katalógom

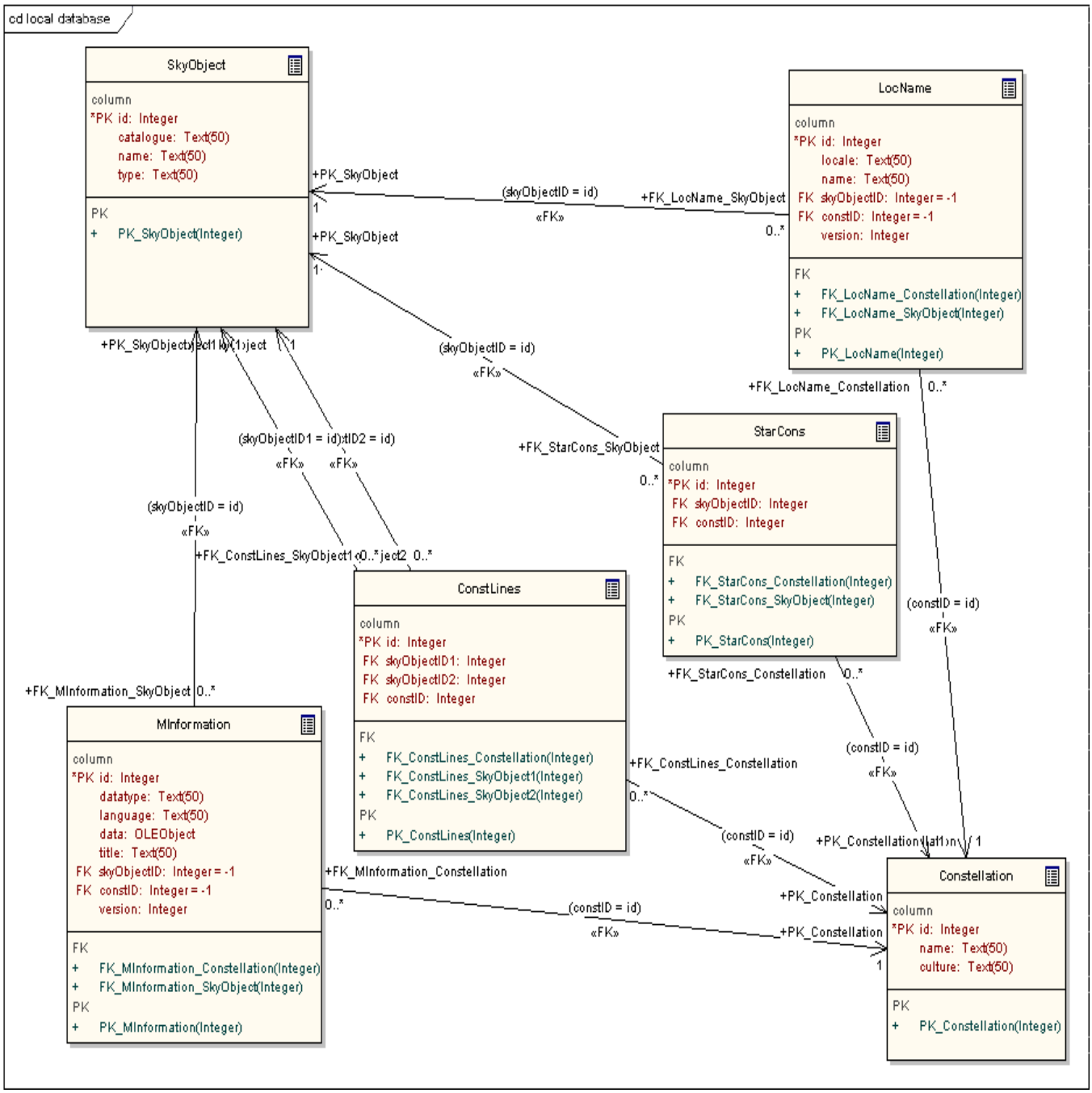
7.3. Lokálna databáza

Modul pre lokálnu databázu zabezpečuje uchovávanie multimediálnych informácií a lokalizovaných názvov pre nebeské telesá. Môžu byť do lokálnej databázy dopĺňané zo zdieľanej databázy pomocou webových služieb a zobrazujú sa na požiadavku používateľa.

7.3.1. Tabuľky v databáze

Návrh lokálnej databázy vychádza z fyzického modelu ale boli pridané dve tabuľky StarCons a ConstLines. Tabuľka StarCons reprezentuje vzťah n ku n medzi tabuľkou SkyObject a tabuľkou Constellation. Jedno súhvezdie môže obsahovať viaceré hviezdy a naopak jedna hviezda môže byť vo viacerých súhvezdiach, pretože súhvezdia môžu byť z rôznych kultúr. Tabuľka ConstLines uchováva čiary v súhvezdí. Aby relácie v databáze boli konzistentné tak sme pridali do tabuliek SkyObject, Constellation, MInformation záznam s identitou, ktorá má hodnotu -1.

Diagram tabuliek je zobrazený na Obr. 29.



Obr. 29 Diagram tabuliek lokálnej databázy

7.4. Integrácia systému a používateľské rozhranie

Úlohou prototypu nebolo vytvoriť ucelený systém. Našim cieľom bolo overiť si funkčnosť a použiteľnosť technológií vhodných pre vývoj systému, preveriť našu schopnosť realizovať rizikové časti systému a otestovať ich presnosť. Integráciu systému a vytvorenie primeraného používateľského systému sme implementovali až v letnom semestri.

Kapitola 5.1 prezentovala architektúru systému, v ktorej základom integrácie hlavnej aplikácie bol balík „gui“ s jeho komponentami GUI, App Control module a Voice module

slúžiacim na rozpoznávanie a syntézu hlasu. Prvá časť tejto podkapitoly prezentuje architektúru práve tmeliacej časti systému, ktorú dopĺňa aj prepojením na grafické používateľské rozhranie. Jej druhá časť sa zameriava na opis realizácie zobrazovania hviezdnej oblohy. Podkapitola končí opisom realizácie hlasového ovládania.

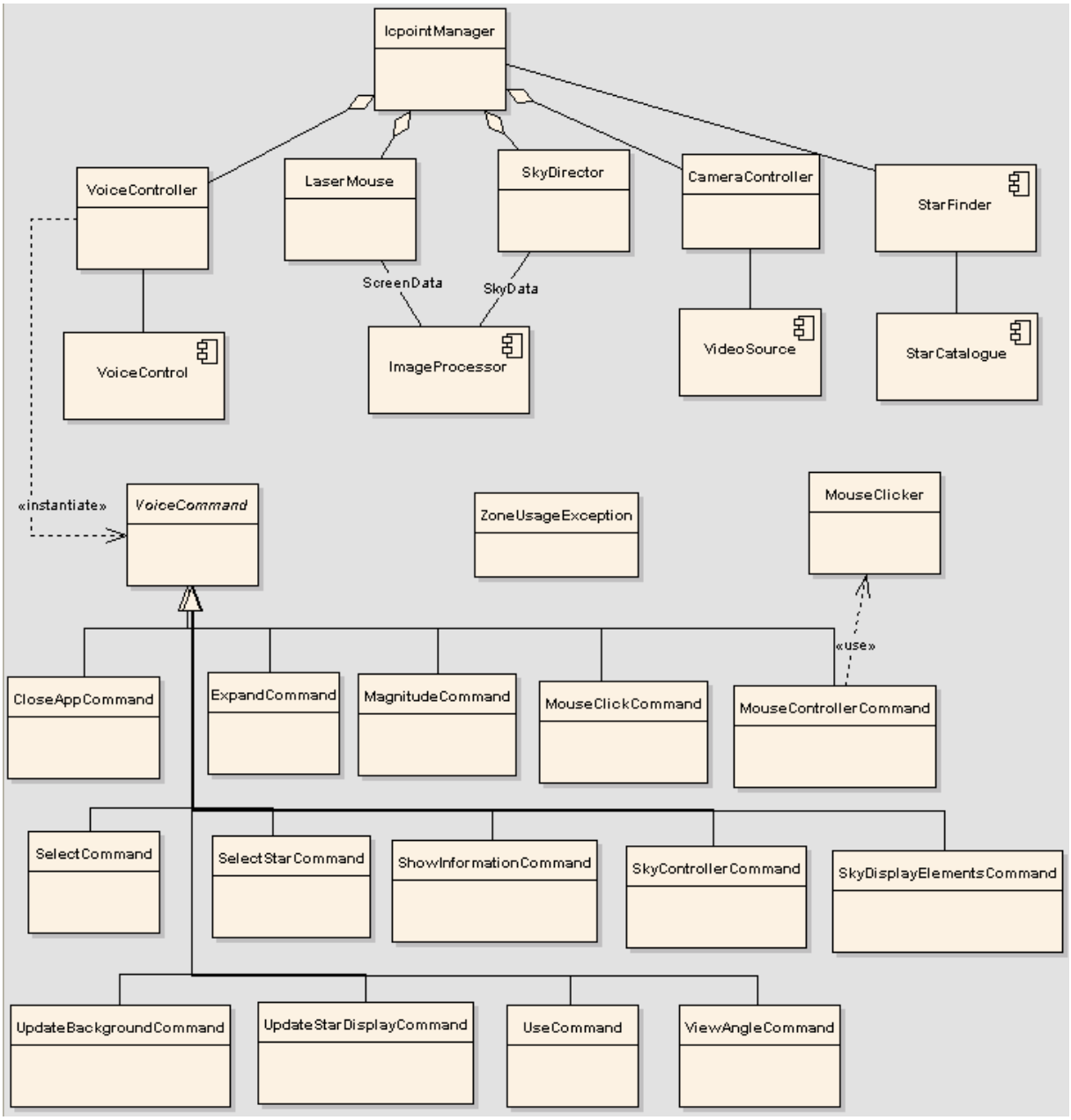
7.4.1. Architektúra spojenia komponentov a GUI

Tmeliacim elementom systému je trieda IcpointManager, ktorá spravuje inštancie kľúčových komponentov systému (pozri Obr. 30) prostredníctvom pomocných tried, tvoriacich fasádu jednotlivým komponentom podľa ich funkcie. Výnimkou je modul StarFinder tvorený jedinou triedou, ktorá z pohľadu zvyšku systému sama tvorí vhodnú fasádu pre funkcionalitu týkajúcu sa katalógov nebeských telies a vyhľadávania v nich. Trieda IcpointManager má vždy jedinú inštanciu (návrhový vzor Singleton).

CameraController definuje rozhranie pre správu kamery, ku ktorej pristupuje prostredníctvom základnej triedy všetkých zdrojov obrázkov - ImageSource komponentu VideoSource. Inštanciu konkrétnej triedy vytvára použitím triedy VideoSourceFactory z balíka VideoSource, ktorá vytvára objekty len na základe mena triedy a parametrov konštruktora uchovávané v textovej podobe v nastaveniach aplikácie. Popísaným spôsobom je možné jednoducho nahradiť alebo doplniť komponent VideoSource inou implementáciou s ďalšími triedami umožňujúcimi používanie nových typov kamier bez nutnosti zasahovania do hlavnej časti aplikácie a bez potreby jej rekompilácie. VideoSourceFactory ďalej umožňuje pomocou reflexie získať zoznam implementovaných ovládačov (koncových tried z balíka VideoSource), čím zjednodušuje výber triedy pri nastavovaní ovládača zdroja obrázkov.

Obdobná potreba sa vynorila aj v prípade rozhrania na rozpoznávanie hlasu. VoiceController vytvára inštanciu rozpoznávača hlasu z balíka Voice Control, ktorú konfiguruje pomocou XML súboru s gramatikou. V XML súbore sú definované názvy gramatických pravidiel a zoznamy ich variant spolu s textom, ktorý má Voice control rozpoznávať. Tvárnosť a modifikovateľnosť sme sa rozhodli dosiahnuť použitím návrhového vzoru Príkaz (Command). Abstraktná trieda VoiceCommand definuje rozhranie pre všetky hlasové príkazy, ktorých realizácia je zapúzdrená v jej jednotlivých podtriedach. Výber konkrétnej triedy sa realizuje priamo na základe názvu pravidla rozpoznaného balíkom Voice control a rozoznaná alternatíva pravidla slúži ako parameter príkazu. Množstvo vytvorených príkazov ilustruje Obr. 30, čím zároveň jasne vizualizuje ich širokú a jednoduchú rozšíriteľnosť.

Poslednými triedami pripojenými k IcpointManager-u sú LaserMouse a SkyDirector. Tie slúžia na zjednodušenie rozhrania komponentu ImageProcessor, ktorého vonkajšie rozhranie definované rovnomennou triedou je pre používanie v zvyšku systému zbytočne príliš všeobecné. Uvedené dve triedy upravujú toto rozhranie pre potreby používania práve jedného ovládača kurzoru myši a práve jedného rozpoznávania smeru pohľadu k hviezdam.

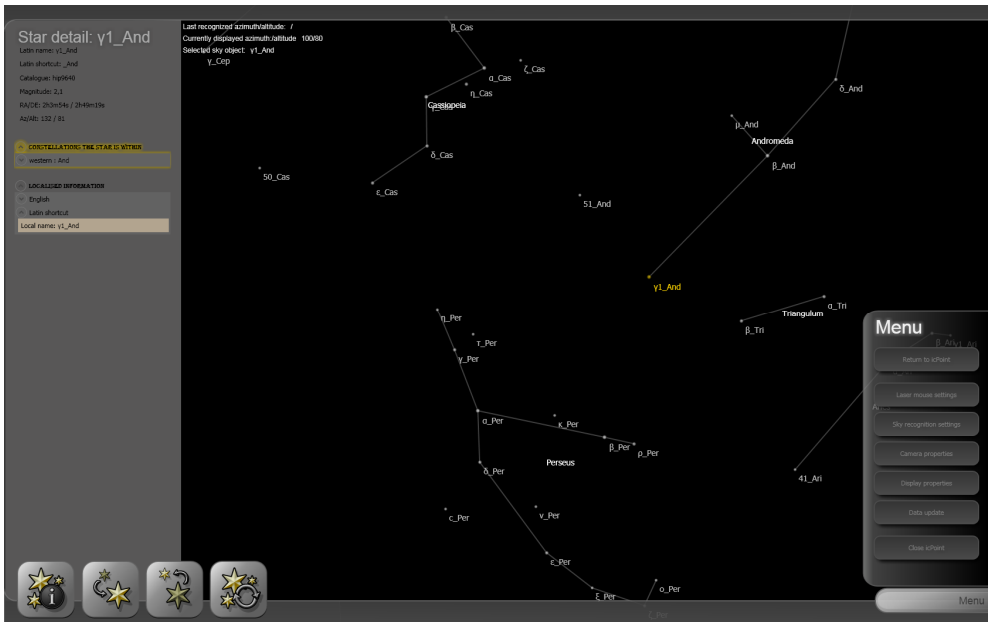


Obr. 30. Prepojenie častí aplikácie a príkazy hlasového ovládania

Nastavovanie parametrov a menu

Väčšina komponentov vyžaduje poskytnutie parametrov, ktoré môže nastavovať používateľ. Dlhodobo sú všetky uchovávané vo vlastnostiach (settings) hlavnej časti systému

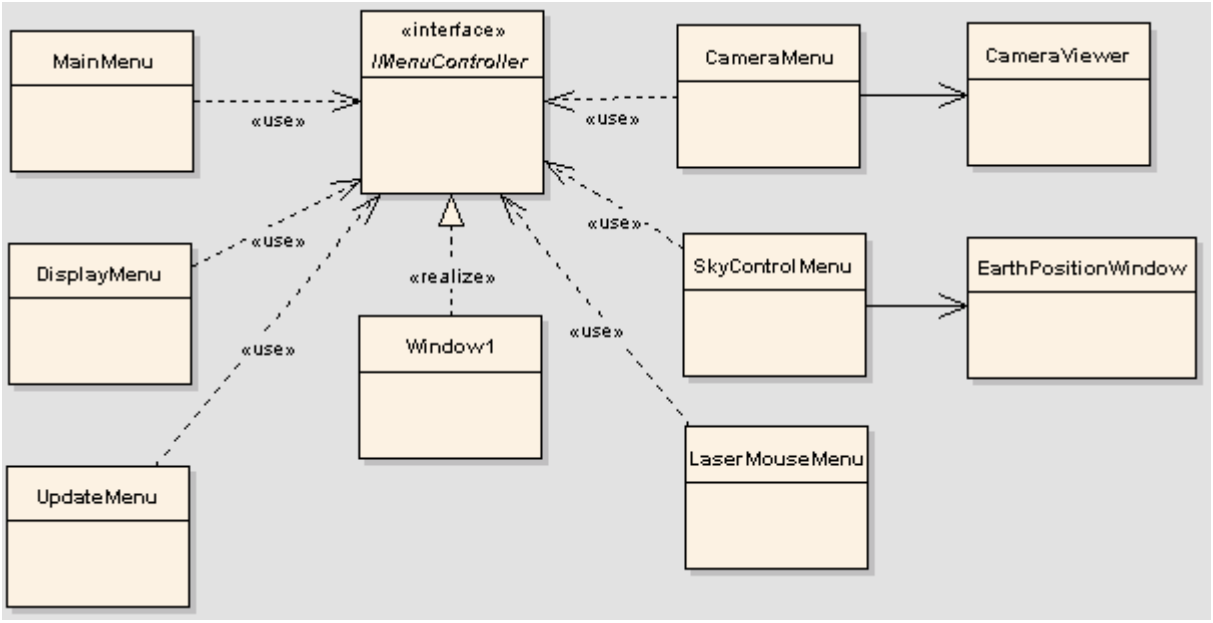
icPoint, no ich manuálne nastavovanie v čisto textovej podobe by bolo prinajmenšom nepraktické. Kandidát na najlepší multimediálny produkt roku 2007 musí mať používateľské rozhranie pekné a prívetivé (pozri Obr. 31). Preto sme ho realizovali pomocou technológie Windows Presentation Foundation (WPF [27], Avalon).



Obr. 31. Hlavné okno aplikácie so zobrazeným menu a oknom StarDetailDisplayer

Samostatné okno pre zobrazovanie snímok z kamery je realizované triedou CameraViewer. Oddelene je aj okno pre výber polohy na zemeguli EarthPositionWindow poskytujúce príjemnú grafickú alternatívu k ručnému zadaniu presnej zemepisnej dĺžky a šírky. Využíva sa pritom mapa sveta a vyhľadávanie miest poskytované GoogleMaps API [26]. Obe okná sú vyvolateľné z menu.

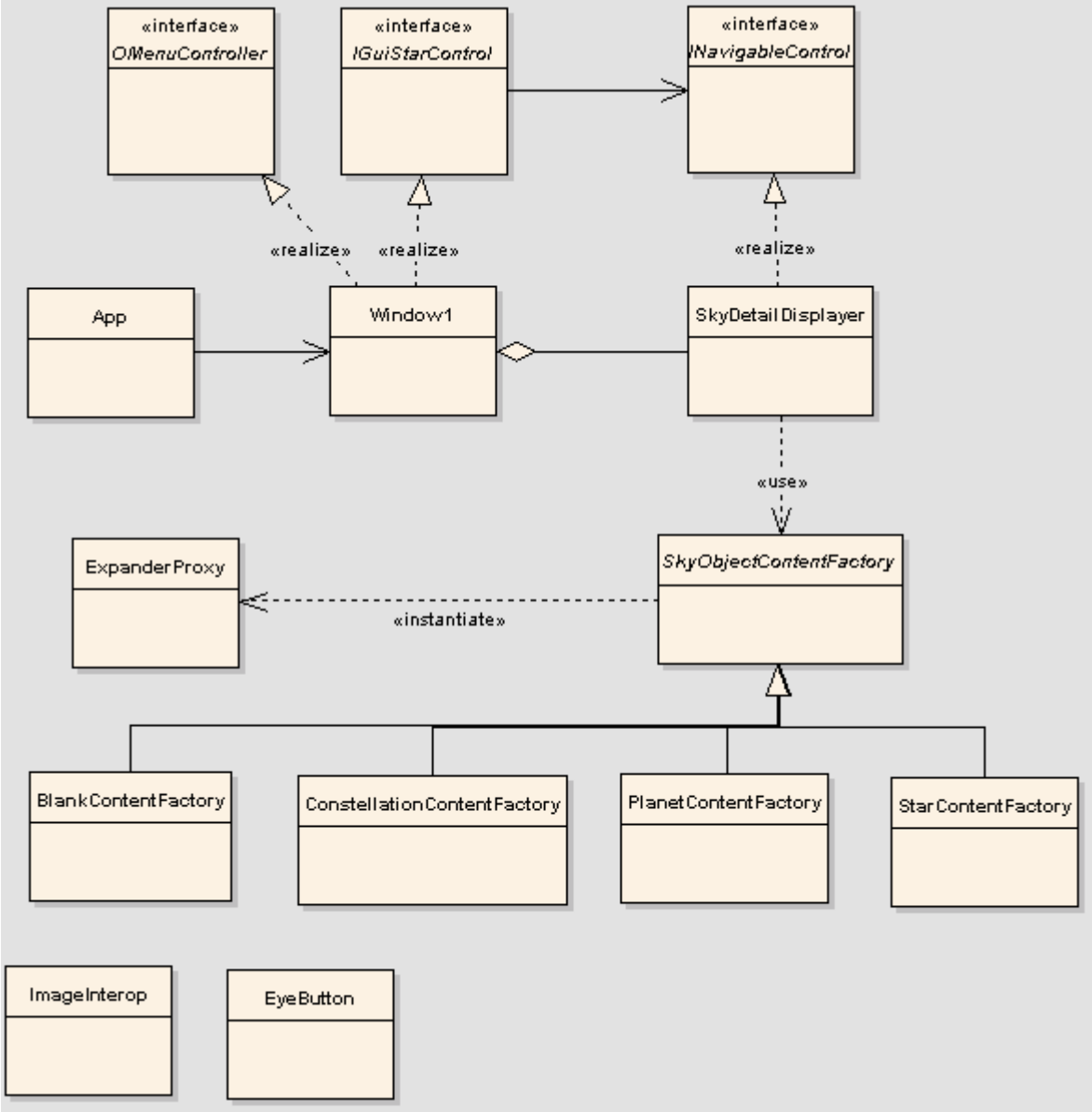
WPF obsahuje komponent Frame, ktorý môže mať ako svoj obsah ľubovoľnú HTML stránku alebo XAML Page. Menu sme sa rozhodli navrhnúť tak, že každé jedno submenu je samostatná stránka (Page), ktorá pre zmeny v menu používa metódy rozhrania IMenuController (pozri Obr. 32). Konkrétnu implementáciu tohto rozhrania dostáva ako parameter konštruktora, ktorou je typicky objekt triedy hlavného okna aplikácie – Window1. Použitie rozhrania IMenuController zjednodušuje prípadnú neskoršiu modifikáciu a jednoznačne definuje metódy potrebné pre riadenie menu.



Obr. 32. Návrh menu

Prezentácia multimediálnych informácií

Aplikácia sa spúšťa ako inštancia triedy App, ktorá vytvára hlavné okno grafického používateľského rozhrania reprezentované triedou Window1 (pozri Obr. 33). Všetky texty v ňom, ako aj v ostatných viditeľných častiach systému sú ukladané ako zdroje (resources), ktoré môžu byť ľahko (nezávisle na samotnom zdrojovom kóde) upravované a lokalizované ich prekladom do iného jazyka.



Obr. 33. Diagram tried pre prezentáciu multimediálnych informácií

Window1 slúži ako mediator (sprostredkovateľ) pre rôzne prvky používateľského rozhrania. Okrem toho, že implementuje už spomínané rozhranie pre ovládanie menu (IMenuController), realizuje aj rozhranie definované ako IGuiStarControl. Toto určuje základné operácie, ktoré musí zabezpečovať GUI aplikácie vo vzťahu k výberu hviezd a zobrazovanému výrezu oblohy. Samotné rozhranie označuje, že jeho implementácia musí obsahovať odkaz na navigovateľné okno s prezentovateľným obsahom (rozohranie INavigableControl), ktoré je v prípade hlavného okna systému icPoint realizované triedou SkyDetailDisplayer.

SkyDetailDisplayer je vlastný element grafického používateľského rozhrania (UserControl), ktorý je možné zobraziť pomocou volania pre zobrazenie hviezdy, planéty, súhvezdia alebo prázdneho obsahu. Pre všetky druhy obsahu definuje základné rozdelenie na nadpis a kontainery pre zobrazenie základných informácií, prepojení na iné objekty a multimediálnych dát. Kontainery sú realizované ako schovateľné panely (expander), ktorých obsah sa napĺňa použitím továrne SkyObjectContentFactory ako základnej triedy aplikovaného návrhového vzoru AbstractFactory. Konkrétna realizácia továrne (BlankContentFactory, ConstellationContentFactory, PlanetContentFactory alebo StarContentFactory) sa vyberá podľa použitej metódy pre zobrazenie okna s multimediálnymi informáciami SkyDetailDisplayer, ktorý vďaka tomu zaisťuje len spoločnú funkcionalitu a definuje rozloženie elementov.

Predpokladáme, že množstvo zobrazovaných informácií o nebeských telesách bude vysoké. Používateľ si o zvolenom telese väčšinou želá zobraziť len niektoré z nich a bolo by preto nevhodné a zbytočne zdržujúce, keby sa vždy vytvárali všetky elementy GUI potrebné pre ich prezentáciu. Rozhodli sme sa implementovať aj prezentáciu samotných multimediálnych informácií ako schovateľné zoznamy, pričom viditeľný je na začiatku len názov. Ostatné sa (ako definuje vzor Proxy – zástupca) vytvárajú až v momente, keď sú potrebné triedou ExpanderProxy. K nej pristupuje zvyšok systému od momentu jej vytvorenia len ako k jej rodičovskej triede - schovateľnému zoznamu Expander.

7.4.2. Zobrazovanie hviezd

Keďže grafické rozhranie aplikácie je kompletne implementované s použitím Windows Presentation Foundation, aj pre zobrazenie hviezd bolo použité toto rozhranie. Vstupnými údajmi pre zobrazenie hviezd je poloha hviezdy v horizontálnych súradniciach (azimuth a výška) a smer pohľadu uvedený takisto v horizontálnych súradniciach. Azimut a výška predstavujú uhly, merané z polohy pozorovateľa, jedná sa teda vlastne o sférické súradnice. Tento fakt vedie pomerne priamočiaro k zobrazovaniu hviezd ako bodov umiestnených na povrchu gule, v ktorej strede sa nachádza používateľ (resp. kamera). Polomer gule je možné zvoliť prakticky ľubovoľný, keďže vzdialenosť hviezd možno z nášho pohľadu považovať za nekonečnú. Od polomera gule však závisí miera sférického skreslenia pri zobrazovaní hviezd, vytvárajúceho dojem určitej perspektívy.

Pri samotnom zobrazovaní hviezd (principiálne ide o mapovanie zo sférických súradníc hviezdy do pravouhlých dvojrozmerných súradníc obrazovky) možno postupovať rôznymi spôsobmi. Najzložitejší spôsob by pravdepodobne viedol cez analytickú geometriu

a hľadanie súradníc priesečníka priamky, spájajúcej stred gule s hviezdou, a roviny priemetne. Nájdenie súradníc tohto priesečníka nepredstavuje síce problém, my však potrebujeme získať súradnice tohto bodu v rámci roviny priemetne, čiže vlastne vzdialenosti tohto bodu od priamok, korešpondujúcich s okrajmi zobrazovanej plochy. Pri tomto musíme zohľadňovať natočenie tejto plochy v priestore atď. Aj keď je táto varianta realizovateľná, vzhľadom na zdĺhavosť odvodenia a obmedzený čas pre riešenie bolo nutné zvážiť ine alternatívy.

Ďalšou alternatívou bolo využitie knižnice tretej strany, umožňujúcej mapovanie dvojrozmerných ovládacích prvkov na trojrozmerné objekty. Hviezdy by v takomto prípade boli zobrazené na dvojrozmernú plochu, ktorou by bola následne obalená guľa a celok by sa zobrazil s použitím zabudovaných funkcií WPF pre trojrozmerné zobrazovanie. Táto varianta však takisto prinášala veľké množstvo úskalí, najmä potrebu ošetrenia prípadov, keď sa objekt vykresľoval na okraji dvojrozmernej plochy a došlo by tak k jeho „useknutiu“ atď. Táto alternatíva teda tiež nebola vyhovujúca.

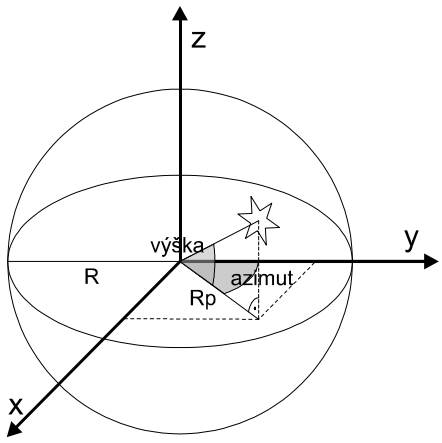
Treťou alternatívou bol transformačný proces, rozdelený na niekoľko fáz. V prvej fáze sa sférické súradnice hviezdy s použitím goniometrických funkcií prevedú na pravouhlé priestorové súradnice. Súradnice sa vypočitajú nasledovným spôsobom:

$$X = \sin(az) * R_p$$

$$Y = \cos(az) * R_p$$

$$Z = \sin(alt) * R$$

Kde *az* je azimuth a *alt* je výška v horizontálnych súradniciach, R je polomer gule, na ktorej povrch sú umiestňované hviezdy. R_p je dĺžka priemetu úsečky spájajúcej stred gule s hviezdou do vodorovnej roviny.

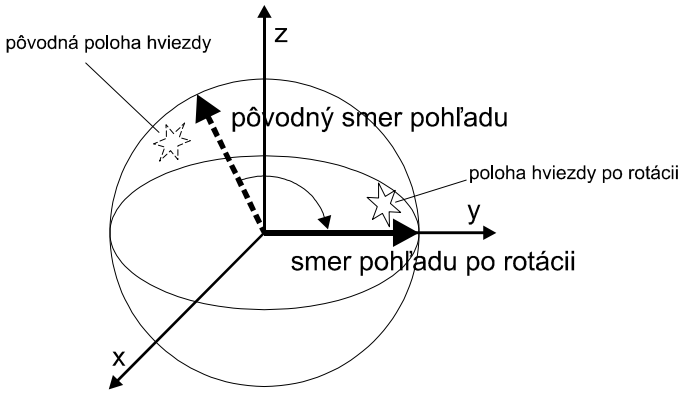


Obr. 34Prevod horizontálnych súradníc na pravouhlé

V druhej fáze sa priestorové súradnice hviezdy transformujú pomocou rotácie v priestore tak, aby priamka, určujúca zadaný smer pohľadu po aplikovaní rotačnej transformácie prechádzala niektorou z osí pravouhlej súradnicovej sústavy. Samotné vzorce, použité pre priestorovú transformáciu vychádzajú z transformačných matíc, bežne využívaných v priestorovej grafike. Matica pre rotáciu okolo osi X vyzerá nasledovne:

$$\mathcal{R}_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix} = \exp\left(\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -\theta_x \\ 0 & \theta_x & 0 \end{bmatrix} \right)$$

Kde θ_x je uhol rotácie okolo osi X.

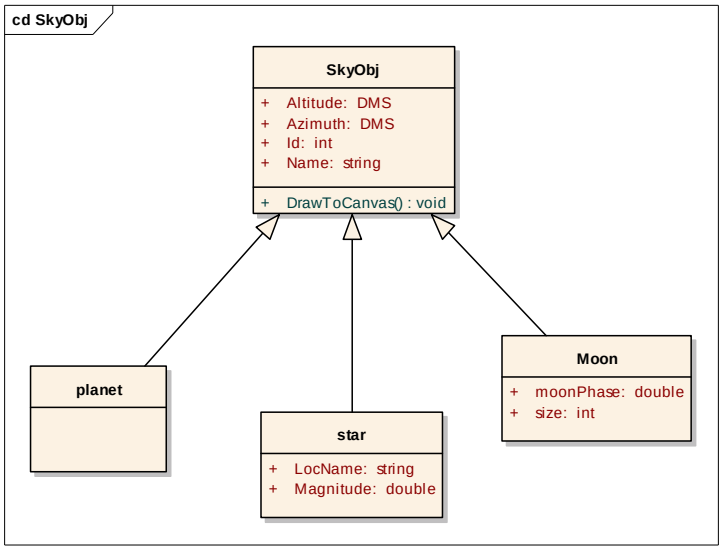


Obr. 35 Rotačná transformácia súradníc

Po vykonaní rotačnej transformácie súradníc bodu môžeme v treťej fáze jednoduchým spôsobom získať ortogonálny priemet súradníc bodu do roviny. Stačí, keď ako súradnice bodu v rovine uvažujeme priestorové súradnice v zvyšných dvoch osiach, kolmých na smer pohľadu (ktorý je teraz rovnobežný s treťou osou). Takto získané súradnice možno následne priamo použiť pre zobrazenie hviezdy na obrazovke.

Zobrazovanie je zapuzdrené do triedy SkyDisplay. Vstupom pre zobrazovanie je zoznam zobrazovaných nebeských objektov so súradnicami, geografická poloha pozorovateľa, dátum a čas, smer pohľadu, zorný uhol a niekoľko ďalších parametrov, spojených so samotným zobrazovaním.

Zobrazované objekty sú reprezentované prostredníctvom tried. Hlavnou triedou je trieda *SkyObj*, reprezentujúca ľubovoľný nebeský objekt. Od nej sú odvodené triedy pre hviezdy, planéty a mesiac. Súhvezdia sú reprezentované triedou *constellation*, obsahujúcou súradnice hviezd a údaje o spojniciach hviezd.



Obr. 36 Diagram tried nebeských objektov

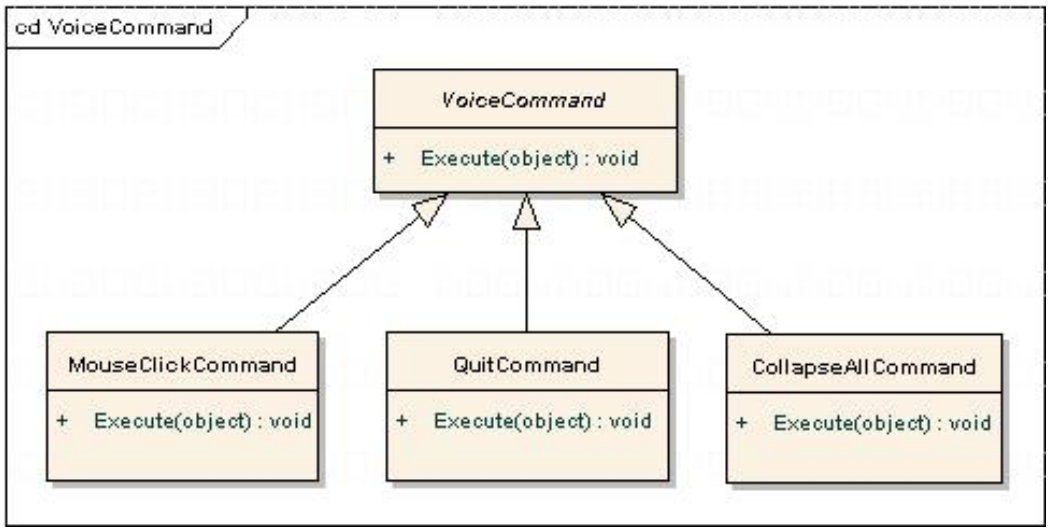
7.4.3. Rozpoznávanie hlasu

V počiatočných fázach projektu bolo plánované vytvoriť vlastný subsystém na rozpoznávanie hlasových povelov. Ten mal byť založený na zaznamenávaní hlasových povelov a ich následnej analýzy pomocou Fourierovej transformácie na zistenie frekvenčných zložiek zachytených zvukových vĺn. Získané vzorky by boli porovnávané s vopred zaznamenanými na vyhodnotenie ich zhody.

Pri implementácii uvedeného systému však bolo zistené, že zaznamenávanie a spracovanie zvuku s parametrami, potrebnými pre presné rozoznanie povelov, je náročné na systémové prostriedky. Tie sú však kriticky dôležité pri spracovaní obrazu, preto bolo od vytvárania vlastného systému upustené a na tento účel sme sa rozhodli použiť Microsoft Speech API (ďalej SAPI).

Definície hlasových príkazov, ktoré program akceptuje, sa budú nachádzať v externom súbore formátu XML. Tento súbor predstavuje gramatiku pre hlasové povely. Takáto gramatika umožňuje definovať nie len samotné výrazy, slúžiace ako povely pre funkcie aplikácie, ale aj zložené príkazy, skladajúce sa z viacerých povelov. Výhodou použitia zložených príkazov je, že zriedkavejšie dochádza k nesprávnemu interpretovaniu zachyteného povelu (použitie príkazu „move left“ namiesto „left“ zníži pravdepodobnosť nesprávneho rozoznania príkazu na polovicu).

Spracovanie hlasových povelov bude založené na abstraktnej triede VoiceCommand, od ktorej budú odvodené triedy reprezentujúce jednotlivé príkazy. Navrhnutá hierarchia tried pre hlasové povely sa nachádza na Obr. 37.



Obr. 37 Hierarchia tried pre hlasové povely.

7.5. Serverová časť aplikácie

Do serverovej časti aplikácie sú zaradené tri moduly. Ich hlavnou úlohou je umožniť pridávanie nových údajov pre aplikáciu a uchovávať ich v zdieľanej databáze. Rozhranie pre dopĺňanie informácií predstavuje modul Web interface a zdieľaná databáza je realizovaná modulom Shared MI Database. Okrem toho poskytuje serverová časť aplikácie možnosti doplnenia údajov do lokálnej databázy pomocou webovej služby.

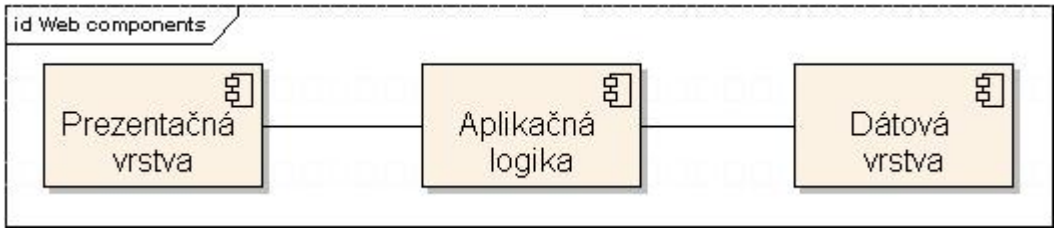
7.5.1. Modul Web interface a Shared MI Database

Tieto moduly sú súčasťou webovej aplikácie, ktorá má umožňovať používateľom pridávanie údajov do databázy. Táto časť v prototype nebola vôbec implementovaná.

Architektúra webovej aplikácie

Aplikácia bude implementovaná pomocou technológií .NET – C#, ASP.NET a s použitím MS SQL Server Express ako databázový server. Preto architektúra zohľadňuje štandardnú architektúru aplikácií na základe týchto technológií. Je zvolená architektúra pomocou vrstiev s troma vrstvami, zobrazenými na obrázku Obr. 38. Prezentačná vrstva zabezpečuje zobrazenie údajov používateľovi a získavanie požiadaviek a odpovedí od používateľa. Budú implementované priamo pomocou ASP.NET stránok a formulárov. Aplikačná logika zabezpečí spracovanie údajov, pôjde predovšetkým o udalosti, ktoré reagujú na používateľove akcie vo web stránke. Úlohou dátovej vrstvy je uchovávať údaje

a poskytovať ich podľa požiadaviek z aplikačnej vrstvy. Bude implementovaná pomocou objektovo relačného mappera a ako databázový server bude použitý MS SQL Server Express.



Obr. 38. Architektúra dynamickej webovej aplikácie

Fyzický model údajov

Fyzický model údajov vychádza z logického modelu lokálnej databázy, ktorý bol načrtnutý už v hrubom návrhu, a je doplnený vzhľadom na špecifikáciu webovej aplikácie. Možno ho rozdeliť na dve časti: časť pre uchovávanie samotných údajov pre aplikáciu a časť pre správu používateľov a histórie zmien. Model pre uchovávanie údajov pre aplikáciu zobrazuje obrázok Obr. 39.

Entita SkyObject slúži pre uchovávanie údajov o nebeských telesách. Okrem id obsahuje nasledovné atribúty:

- catalogue – identifikácia katalógu, v ktorom sa daný objekt nachádza
- name – id v katalógu
- type – typ objektu, napríklad hviezda, planéta a pod.

Pre uchovávanie základných údajov o súhvezdiach slúži entita Constellation. Okrem identifkátora id obsahuje atribúty:

- first_id – pomocný atribút potrebný pre uchovávanie histórie zmien.
- name – meno súhvezdia, preferuje sa jeho skratka
- culture – kultúra v ktorej sa dané súhvezdie vyskytuje

Nebeské teleso (hviezda) môže patriť do niekoľkých súhvezdí, ak berieme rôzne kultúry a zároveň jedno súhvezdie obsahuje viacero hviezd. Na reprezentáciu tohto n ku m vzťahu slúži entita StarCons. Obsahuje cudzie kľúče pre identifikáciu hviezdy a súhvezdia do ktorého daná hviezda podľa vzťahu patrí.

Okrem tohto vzťahu môže hviezda byť súčasťou čiary v obrazci súhvezdia. Obrazce sú potrebné pre zobrazovanie súhvezdí v aplikácií. Jedna hviezda môže byť súčasťou viacerých čiar a v rámci jedného obrazca súhvezdia existuje viacero čiar. Zároveň každá čiara obsahuje dve hviezdy. Na uchovávanie týchto údajov slúži entita ConstLines, ktorá obsahuje

podľa týchto podmienok dva cudzie kľúče pre nebeské objekty (hviezdy) a jeden cudzí kľúč identifikujúci súhvezdie. Pre entity ConstLines a StarCons sa neuchováva história zmien.

Entita LocName bude obsahovať lokalizované mená pre nebeské teleso alebo súhvezdie. K nebeskému telesu alebo súhvezdiu sa môže vzťahovať viacero týchto lokalizovaných názvov, každý pre iný jazyk. Preto entita obsahuje cudzie kľúče, ktoré môžu identifikovať lokalizované nebeské teleso alebo súhvezdie. Okrem toho obsahuje nasledovné atribúty:

- firstID – slúži pre identifikáciu prvého vloženého riadku, potrebné pre uchovávanie histórie
- locale – identifikácia lokalizácie mena
- name – samotné lokalizované meno

Samotné multimediálne informácie budú uchovávané v tabuľke MInformation. Môžu sa vzťahovať k súhvezdiu alebo k nebeskému objektu, a ku každému môže byť priradených viacero multimediálnych informácií. Tabuľka preto obsahuje stĺpce skyObjectID a constID, ktoré zachytávajú tieto n ku 1 vzťahy. Ďalej obsahuje nasledovné stĺpce:

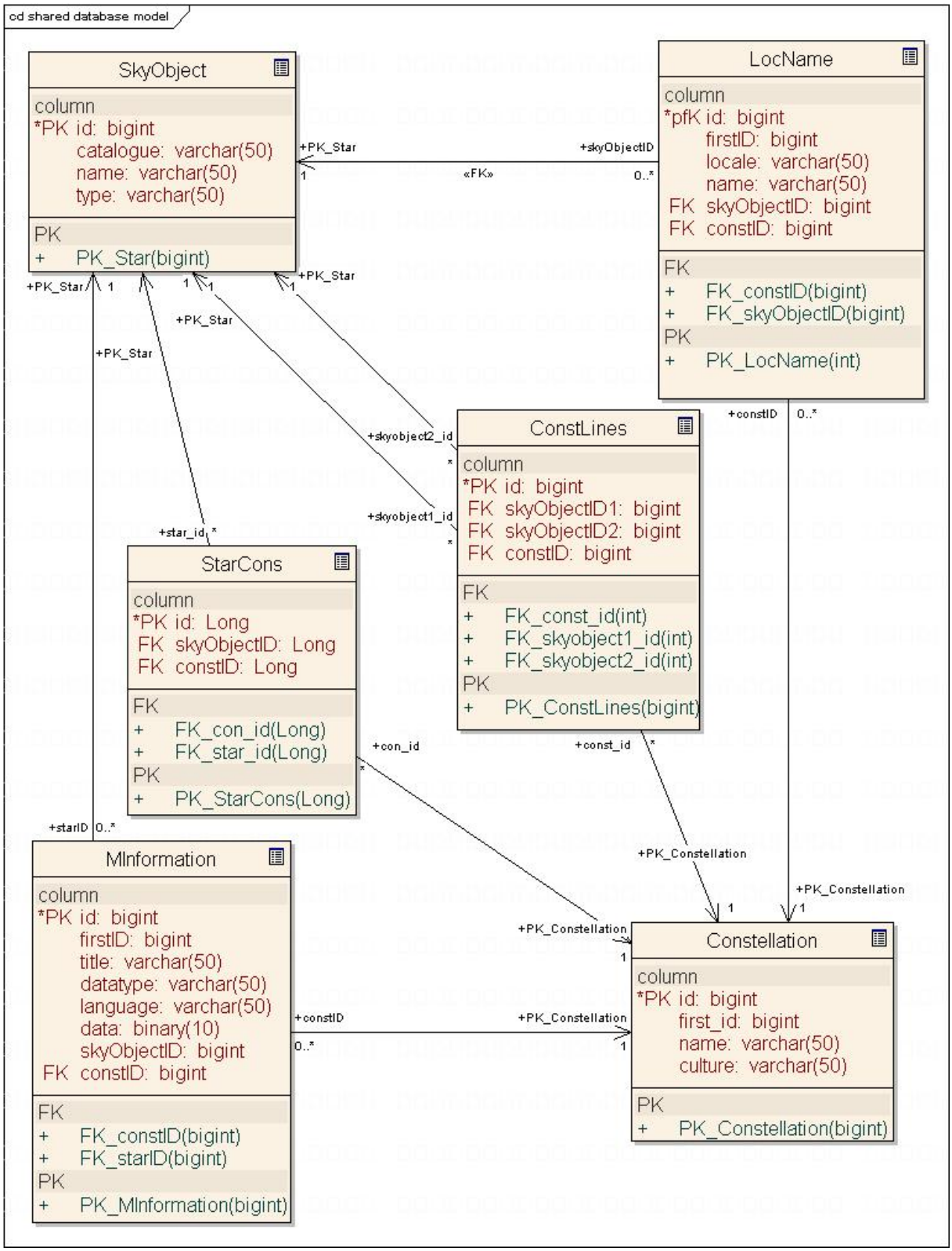
- firstID – slúži pre identifikáciu prvého vloženého riadku, potrebné pre uchovávanie histórie
- title – titulok multimediálnej informácie, môže byť aj krátky popis
- datatype – dátový typ entity, budú použité MIME typy
- data – samotné multimediálne informácie

Okrem uchovávania údajov pre aplikáciu je potrebné uchovávať aj údaje o používateľoch a zmenách v jednotlivých entitách, aby sme mali prehľad medzi zmenami, ku ktorým v údajoch došlo. Entity pre ich uchovávanie je možné vidieť na obrázku Obr. 40.

Uchovávanie údajov o používateľoch zabezpečuje trieda WebUser. Obsahuje nasledovné atribúty:

- username – prihlasovacie meno používateľa
- password – heslo používateľa
- email – emailová adresa používateľa

Okrem toho je potrebné uchovávať rolu používateľa v systéme, aby bolo možné definovať rozdielne oprávnenia pre používateľov. Rola je reprezentovaná ako samostatná entita s názvom roly ako atribútom (rolename). Jednu rolu v systéme môže mať viacero používateľov, preto obsahuje entita WebUser aj cudzí kľúč roleID na zachytenie tohto vzťahu.



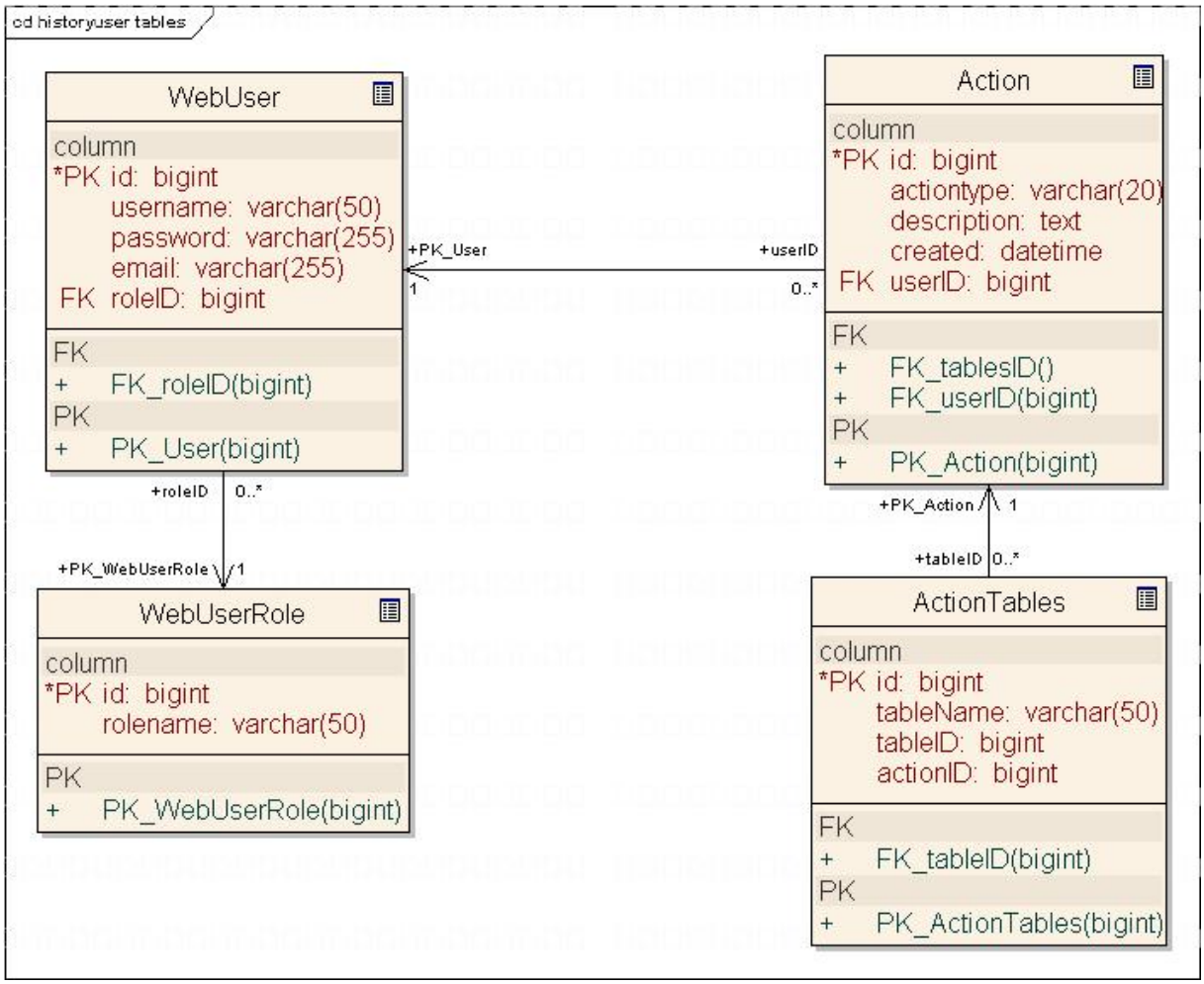
Obr. 39. Fyzický model údajov – údaje pre aplikáciu

Pre uchovávanie histórie zmien slúžia tabuľky Action a ActionTables. V tabuľke Action sa uchováva popis akcie, ktorá bola v systéme vykonaná. Obsahuje atribúty:

- actiontype – typ akcie, môže byť napríklad insert, update, delete

- description – popis akcie, bude obsahovať opis zmien v systéme
- created – čas vykonania akcie
- userID – identifikácia používateľa, ktorý akciu vykonal

Jedna akcia môže ovplyvniť viacero tabuliek alebo riadkov v tabuľke, preto je informácia o zmenených tabuľkách uchovávaná zvlášť entitou ActionTables. Entita obsahuje identifikátor akcie, ktorá bola vykonaná (actionID), názov zmenenej tabuľky (tableName) a id v tabuľke, ktorého sa zmena týka (tableID).



Obr. 40. Tabuľky pre uchovávanie histórie a používateľov

Dátová vrstva

Pre implementáciu dátovej vrstvy bude použitý objektovo relačný mapper Gentle.NET vo verzií 1.2.9, pretože ďalšie zverejnené verzie sú označené ako beta verzie. Ide o pomerne jednoduchý mapper, ale pre naše účely bude postačujúci. Dátová vrstva bude obsahovať triedy zapuzdrujúce entity v tabuľke a zároveň k nim umožňujúce prístup a úpravy.

Aplikačná a prezentačná vrstva

Tieto vrstvy budú implementované pomocou ASP.NET stránok a reakcií na udalosti jednotlivých komponent stránky. Vzhľadom na použitý objektovo relačný mapper bude ako dátový zdroj pre stránku používaný ObjectDataSource.

Jedným z najdôležitejších algoritmov je vytváranie samotnej histórie pri editácií údajových entít, aby bolo umožnené zobrazenie tejto histórie. Pre vytvorenie histórie obsahujú editovateľné entity atribút firstID. Pri vložení nového riadku (súhvezdia, mena, multimediálnej informácie) bude firstID zhodné s id vloženého riadku. Ak bude požiadavka na úpravu alebo mazanie, nebude v skutočnosti upravovaný alebo zmazaný riadok. Vloží sa do tabuľky nový riadok, ktorý bude obsahovať nové údaje, na ktoré mal byť pôvodný riadok upravený alebo všetky stĺpce budú prázdne v prípade mazania (okrem id a firstID). Nový riadok bude mať firstID rovné firstID riadku, ktorý sa upravoval. Atribút firstID riadku bude vlastne vždy ukazovať na prvý vložený riadok s danou informáciou. Pomocou toho bude potom jednoduché nájsť všetky zmeny, keďže nám stačí vyhľadávať položky s rovnakým firstID. Okrem toho bude do tabuliek Action a ActionTable vložený nový záznam, ktorý bude popisovať danú zmenu a pomocou tableName a tableID identifikovať aj samotný údajový záznam, ktorý bol ako zmena do databázy pridaný. Atribút tableID bude odkazovať vždy na id záznamu, nie na firstID, pretože tak by nebolo možné v tabuľke vyhľadať zmenený záznam.

Tento spôsob práce s údajovými entitami musia zohľadňovať aj metódy pre vyhľadávanie údajov, keď je potrebné aby štandardne zobrazovali poslednú aktuálnu zmenu. Bude využité zväčšovanie id riadku pri pridávaní nového záznamu (použitý bude štandardný spôsob generovania id pomocou automatickej inkrementácie). Vyberú sa teda tie údaje, ktoré majú najvyššie id s daným firstID.

7.5.2. Webová služba

Webová služba bude slúžiť na aktualizáciu informácií v lokálnej databáze zo zdieľanej databázy, umiestnenej v serverovej časti aplikácie. Prístup k službe bude zabezpečený protokolom SOAP, pracujúcim nad protokolom HTTP. Samotné metódy webovej služby budú poskytovať údaje pre metódy v lokálnej časti aplikácie, ktoré budú lokálne údaje aktualizovať. Z dôvodu prenosu všetkých informácií cez internet je dôležité, aby webové metódy poskytovali iba tie informácie, ktoré sú potrebné pre aktualizáciu údajov v lokálnej databáze.

Webové metódy by mali poskytovať nasledujúce funkcie:

- Získanie atribútov vybraného objektu (konštelácie, hviezdy)
- Získanie lokalizovaných informácií pre zvolený objekt a lokalizáciu

- Získanie najnovších multimediálnych informácií na základe poskytnutých verzií existujúcich v lokálnej databáze
- Overiť, či existujú novšie multimediálne informácie o zvolenom objekte

8. Testovanie

Kapitola obsahuje opis testovania aplikácie a webového rozhrania. Je rozdelená na dve časti. Prvá časť opisuje spôsoby testovania počas vývoja aplikácie, niektoré použité nástroje a testovacie údaje. Druhá časť kapitoly je venovaná záverečnému testovaniu aplikácie, ktoré bolo vykonávané externými testermi aj členmi tímu. Je založené na akceptačných testoch vytvorených počas špecifikácie, ktoré boli doplnené o niekoľko akceptačných testov pre webovú aplikáciu.

8.1. Testovanie počas vývoja aplikácie

Všetky funkcie, metódy a triedy boli čiastočne otestované počas vývoja aplikácie účastníkmi tímu. Okrem testovaním vývojárom daného komponentu boli funkcie modulu otestované aj členmi tímu, ktorí tieto funkcie využívali. Testovanie niektorých komponentov prebiehalo pomocou nástroja NUnit (Image processor, Video source). Ostatné boli testované priamo volaním daných funkcií z rozhrania. Výsledky poskytované aplikáciou boli porovnávané s astronomickým programom Stellarium aj s priamym pozorovaním nočnej oblohy. Týmto spôsobom boli overené predovšetkým astronomické algoritmy a údaje z katalógu. Overenie určovania smeru pohľadu prebiehalo v súčinnosti s testovaním astronomických komponentov. Pri testovaní web aplikácie sme sa zameriavali predovšetkým na overovanie možných zadávaných údajov a správnosť poskytovaných výsledkov.

Výsledkom testovania počas vývoja bolo odstránenie chýb, zmena niektorých algoritmov a optimalizácia rozhrania pre ovládanie pomocou ukazovadla.

8.2. Používateľské testovanie

Používateľské testovanie prebiehalo od vytvorenia prototypu. Takto testovaná bola predovšetkým hlavná aplikácia. Záverečné testovanie prebiehalo za účasti viacerých členov tímu a externých testerov dňa 7.5.2007.

8.2.1. Akceptačné testy

ID: Scenár 1

Názov: Určenie hviezdy

Vstupné podmienky: nie sú

Výstupné podmienky: Používateľovi sa zobrazí obrazovka s informáciami o vybranej hviezde

Krok	Akcia	Očakávaná reakcia
1	Používateľ namieri ukazovadlo na sklo tak, aby bod ukazoval z jeho pohľadu na želaný objekt a potvrdí voľbu	Systém zobrazí obrazovku s informáciami o najbližšej hviezde

Výsledky testu: Testovanie bolo úspešné, bolo vykonané s viacerými hviezdami súhvezdia Veľká medvedica a Pastier.

ID: Scenár 2

Názov: Navigácia k nebeskému telesu

Vstupné podmienky: nie sú

Výstupné podmienky: Používateľ sa pomocou navigácie dostane k želanej hviezde, v prípade módu, kde má užívateľ nájsť systémom určenú hviezdu je výstupnou podmienkou správne určenie, či bola ukázaná správna hviezda a výpis príslušnej správy

Krok	Akcia	Očakávaná reakcia
1	Používateľ zadá názov hviezdy, ktorú chce vyhľadať	Systém ho hlasovo naviguje k zadanej hviezde
2	Používateľ sa dostatočne priblíži ukazovadlom k zadanej hviezde	Systém používateľa upozorní a vypíše informácie o vybranej hviezde

Alternatívny scenár:

Krok	Akcia	Očakávaná reakcia
1	Používateľ vyberie možnosť dotazovania na polohu hviezdy zo strany systému	Systém vyberie hviezdu z vybranej časti hviezdnej oblohy a požiada používateľa o jej nájdenie prostredníctvom ukazovátka
2	Používateľ vyberie hviezdu a potvrdí voľbu	Systém vypíše správu, či používateľ vybral hľadanú hviezdu, alebo nie

Výsledky testu: Testovanie nebolo úspešné, táto časť aplikácie nebola dokončená.

ID: Scenár 3

Názov: Ovládanie aplikácie

Vstupné podmienky: užívateľ namieri laserový lúč na monitor

Výstupné podmienky: Kurzor myši sa presunie na vybrané miesto

Krok	Akcia	Očakávaná reakcia
1	Používateľ namieri ukazovadlo na sklo tak, aby bod ukazoval z jeho pohľadu na želaný objekt a potvrdí voľbu	Systém zobrazí obrazovku s informáciami o najbližšej hviezde
2	Používateľ potvrdí voľbu	Systém reaguje analogicky, ako pri potvrdení voľby prostredníctvom myši

Výsledky testu: Test prebehol úspešne. Testovanie bolo vykonávané počas konferencie IIT.SRC 2007 viacerými jej návštevníkmi.

8.2.2. Testovanie webovej aplikácie

Pre testovanie webovej aplikácie neboli vytvorené akceptačné testy počas návrhu, preto sú teraz doplnené. Testovanie bolo vykonávané členmi tímu aj externými testermi.

ID: Scenár 1

Názov: Pridanie lokalizovaného mena nebeskému telesu

Vstupné podmienky: Nebeské teleso nemá lokalizované meno v danom jazyku

Výstupné podmienky: Nebeskému telesu je priradené nové lokalizované meno

Krok	Akcia	Očakávaná reakcia
1	Používateľ zadá informácie pre vyhľadanie nebeského telesa	Systém zobrazí tabuľku obsahujúcu vyhľadávané nebeské teleso
2	Používateľ vyberie voľbu Names pre teleso	Zobrazenie formulára pre doplnenie lokalizovaného mena
3	Používateľ do formulára zadá meno a jeho jazyk a potvrdí formulár	Do systému je doplnené nové lokalizované meno pre daný objekt

Výsledky testu: Test prebehol úspešne podľa scenára. Bol vykonaný pre viacero (asi 10) lokalizovaných mien spolu s ich neskoršími úpravami

ID: Scenár 2

Názov: Pridanie multimediálnej informácie súhvezdiu

Vstupné podmienky: Existujúce súhvezdie v databáze

Výstupné podmienky: Súhvezdiu je priradená nová multimediálna informácia

Krok	Akcia	Očakávaná reakcia
1	Používateľ zadá informácie pre vyhľadanie súhvezdia	Systém zobrazí tabuľku obsahujúcu vyhľadávané súhvezdie
2	Používateľ vyberie voľbu List multimedia pre teleso	Zobrazenie formulára pre doplnenie multimediálnej informácie
3	Používateľ do formulára údaje	Do systému je doplnená nová multimediálna informácia pre dané súhvezdie

Výsledky testu: Test prebehol úspešne podľa scenára. Bol vykonaný pre viacero (asi 20) súhvezdí, pridávané boli ich obrázky

9. Zhodnotenie

V našom projekte sme sa snažili vytvoriť zaujímavý produkt, ktorý bude mať šancu uspieť aj na medzinárodnej súťaži Europrix. Preto sme pomerne veľký čas pred samotným začiatkom projektu venovali výberu vhodnej témy. Z viacerých nápadov sme zvolili návrh na vytvorenie softvéru pre amatérskych astronómov, ktorý umožňuje pomocou laserového ukazovadla vyberať hviezdy a poskytuje o nich vedecké aj nevedecké informácie. Veríme, že nielen nápad ale aj jeho spracovanie a systémom ponúkané možnosti v súťaži uspejú.

Vzhľadom na nedostatok času a iné povinnosti sme nestihli implementovať niektoré funkcie, ktoré by funkcie systému ďalej rozširovali. Predovšetkým bude do termínu súťaže potrebné vytvoriť anglickú verziu používateľskej príručky a opisu systému. Z pôvodnej množiny funkcií systému nie je zatiaľ implementované navádzanie používateľa k nebeskému telesu. Okrem toho sme nestihli doplniť do web stránky zabezpečenie pomocou registrácie a prihlasovania, ktoré tiež bude možné doplniť v ďalšom pokračovaní projektu.

Práca na našom projekte nám poskytla predovšetkým praktické skúsenosti s tímovou spoluprácou. Naučili sme sa odhadovať vlastné schopnosti aj schopnosti ostatných členov tímu, podľa čoho sme plánovali a rozdeľovali úlohy. Tiež sme sa snažili využiť individuálnych záujmov členov tímu na jednotlivých úlohách, čo nám vďaka lepšej motivácií umožnilo ich rýchlejšie splnenie. Okrem získania nových technických znalostí sme získali aj praktické znalosti s používaním nástrojov pre spoluprácu v tíme, najmä pre správu verzií a nástroj pre manažment projektu.

Náš tím bol zložený z ľudí, ktorí sa navzájom poznali a dokázali efektívne spolupracovať a riešiť problémy. Projekt pre nás predstavoval určitú výzvu nielen svojou experimentálnou povahou, ale aj množstvom skúseností a vedomostí, ktoré boli pre jeho realizovanie potrebné. Snažili sme sa nájsť aj vhodné riešenia problémov, ktoré nastali pri jeho realizácií. Výsledkom projektu je kvalitný produkt, na ktorom by sme radi ďalej pracovali a ešte tak zlepšili šance v súťaži.

Zoznam použitej literatúry

[1] The EUROPRIX Top Talent Award
http://www.toptalent.europrix.org/

[2] Stellarium homepage
http://www.stellarium.org/

[3] Celestia homepage
http://www.shatters.net/celestia

[4] Hajduk, A., et. al.: Encyklopédia astronómie. Obzor, Bratislava. 1987

[5] IRISCOM - Computer control by eye-tracking
http://www.iriscom.org

[6] Look device - University of Ulster news release
http://news.ulster.ac.uk/releases/2002/631.html

[7] Tobii Technology
http://www.tobii.se

[8] Vspeech website
http://vspeech.sourceforge.net/index.htm

[9] Florian, A: How to build a cascade of boosted classifiers on Haar-like features. In: OpenCV's rapid Object Detection, 2003.

[10] Intel Corporation: Open Source Computer Vision Library. 2006.
http://www.intel.com/technology/computing/opencv/index.htm

[11] Public Astronomical Catalogues and Lists
ftp://cdsarc.u-strasbg.fr/pub/cats/

[12] noReco – homemade speech recognition
http://www.mperfect.net/noReco/

[13] Voice Activation & Annunciation using Visual C# 2003
http://www.emant.com/676008.page

[14] C# Tutorial: Introduction to Microsoft Agent
http://www.deitel.com/articles/csharp_tutorials/20051111/MicrosoftAgent_Page1.html

[15] Microsoft Agent home page
http://www.microsoft.com/products/msagent/default.asp

[16] Portland Pattern Repository
http://c2.com/ppr/

[17] Wikipedia – the free encyclopedia
http://www.wikipedia.org

[18] IrDA: Infrared Data Association
http://www.cs.utexas.edu/~ypraveen/surveys/wlan_security/node13.html

[19] IrDA interfacing to PCs
http://www.eix.co.uk/Articles/IrDA/Welcome.htm

[20] Papyrus Computer Technologies Ltd.
http://www.papyrus.co.il/FAQ/infrared_(irda).htm#Q.%20What%20are%20the%20two%20basics%20of%20the%20IrDA%20standards

[21] UIR – Universal Infrared Receiver
http://fly.srk.fer.hr/~mozgic/UIR/

[22] Building an Infrared Transmitter for Your PC
http://www.hardwaresecrets.com/article.php?id=86

[23] SQLite
http://www.sqlite.org/

[24] Pokorný, Z.: Astronomické algoritmy pro kalkulátory, Praha: Hvězdárna a planetárium hl. m. Prahy, 1988, 88 s.

[25] FTP Facilities at CDS, 2001, ftp://cdsarc.u-strasbg.fr/pub/cats/I/239/, pristupované 12.12.2006.

[26] Google Maps API . Google, 2007
http://www.google.com/apis/maps/

[27] Windows Presentation Foundation. Microsoft corporation, 2007. http://wpf.netfx3.com/

[28] Gentle.NET http://www.mertner.com/confluence/display/Gentle/Home;jsessionid=450FAE34DA075E2C3077A4EA0907C29F

[29] MyGeneration, http://www.mygenerationsoftware.com/portal/default.aspx

Príloha A. Technická dokumentácia

Príloha obsahuje technické informácie o vybraných častiach v rámci implementácie prototypu.

1. Hľadanie útvarov v obraze

Hľadanie útvarov v obraze je implementované v komponente ImageProcessor. Prvým krokom je získanie binárneho poľa označujúceho body (pixely) obrazu, ktoré sú z pohľadu vyhľadávania relevantné. Abstraktná trieda BitmapAnalyzer pre tento účel volá svoju metódu MarkBitmap, ktorá je realizovaná v triedach BitmapComparer a ShapeFinder.

MarkBitmap v triede BitmapComparer slúži na porovnanie bodov aktuálneho snímku s pozadím, pričom určí, ktoré sa zmenili a ktoré nie. Uzamyká bity obrázkov a volá metódu CompareImages, ktorej kľúčová časť je nasledovná:

```
byte* curr = (byte*)currData.Scan0.ToPointer();
byte* back = (byte*)backData.Scan0.ToPointer();

for (int y = 0; y < height; y++)
{
    // for each pixel
    for (int x = 0; x < width; x++, curr += 3, back += 3)
    {
        // grayscale value using BT709
        float a = 0.2125f * curr[/*RGB.R*/2] + 0.7154f *
              curr[/*RGB.G*/1] + 0.0721f * curr[/*RGB.B*/0];
        float b = 0.2125f * back[/*RGB.R*/2] + 0.7154f *
              back[/*RGB.G*/1] + 0.0721f * back[/*RGB.B*/0];

        // difference & tresholding
        if (Math.Abs(a - b) > treshold)
        {
            pixelsChanged++;
            markers[x, y] = true;
        }
        else
            markers[x, y] = false;
    }
    curr += currOffset;
    back += backOffset;
}
```

Vyhľadávanie na základe podobnosti k hľadanej farbe, ktoré sa využíva pre ovládanie kurzoru myši, vychádza z výsledkov implementácie metódy MarkBitmap v triede ColorFinder. Jej podstatu vyjadrujú tieto riadky:

```
byte* p = (byte*)data.Scan0.ToPointer();
for (int y = 0; y < height; y++)
{
    // for each pixel
    for (int x = 0; x < width; x++, p += 3)
    {
        // grayscale value using BT709
```

```
            int dist = Math.Abs(p[/*RGB.R*/2] - clrSearch.R) +
Math.Abs(p[/*RGB.G*/1] - clrSearch.G) + Math.Abs(p[/*RGB.B*/0] -
clrSearch.B);
            if (dist < treshold)
            {
                markers[x, y] = true;
                pixels++;
            }
            else
            {
                markers[x, y] = false;
            }
        }
        p += offset;
    }
```

Odlíšenie jednotlivých metód vyhľadávania je realizované rozdelením obrázka kamery na zóny, z ktorých každú prehľadáva jediná inštancia niektorej z implementácií abstraktnej triedy BitmapAnalyzer.

Pre potreby systému icPoint nebolo možné použiť štandardný spôsob redukcie šumu, kedy sa neporovnávajú priamo pixle ale až priemer susedných (typicky) osmíc pixlov. Testovanie takejto metódy ukázalo, že často by sa odfiltroval aj hľadaný bod z ukazovadla. Namiesto toho sa nájdu obrysy zmenených oblastí v binárnom poli changes[,], ktoré označuje zmenené body obrazu. Hľadanie obrysov je realizované metódou GetContour triedy ShapeFinder:

```
    List<Point> pts = new List<Point>();
    int x=startFrom.X;
    int y = startFrom.Y;
    if (bitmap[x, y] == false)
        return pts;

    // obmedzenia
    int xMax = bitmap.GetLength(0);
    int yMax = bitmap.GetLength(1);
    int rotation = 0;//0 dolava, 1 dole, 2 doprava, 3 hore

    // budem sa drzat laveho okraja, v kazdom kroku sa musim pohnut aspon o
         jeden bod
    do
    {
        // mame bod okraja
        pts.Add(new Point(x, y));

        // najprv sa skusim pohnut v smere o jedno viac dolava (teda v
               smere o -90 stupnov oproti aktualnemu)
        rotation = (rotation + 3) % 4;
        int dirx;
        int diry;
        for (int i = 0; i < 4; i++)
        {
            GetDirection(rotation, out dirx, out diry);
            int nX = x + dirx;
            int nY = y + diry;
            bool accepted = false;
```

```
            for (int o = 0; o < 2; o++, nX+=dirx, nY+=diry)
            {
                if (nX >= 0 && nY >= 0 && nX < xMax && nY < yMax &&
                        bitmap[nX, nY])
                {
                    x = nX;
                    y = nY;
                    accepted = true;
                    break;
                }
            }
            if (accepted)
                break;

            rotation = (rotation + 1) % 4;
        }
    }
    while (x != startFrom.X || y != startFrom.Y);

    return pts;
```

Na základe obrysu je známy obvod oblasti. Ak je príliš malý, považuje sa daný bod za šum. Inak sa stred tejto oblasti označí za nájdený bod. Ak sa nehľadá bod ale n-uholník, tak sa vypočítajú vzdialenosti od stredu - vzdialenostná funkcia – a nájdu sa jej lokálne maximá. Uvedené je realizované v metóde DistanceFunction:

```
    if (vertexCount <= 1)
        return null;

    List<Point> maxims = new List<Point>();
    List<float> distances = new List<float>();
    int j = 0;
    bool growing = false;

    // najdeme stred
    PointF center = FindCentralPoint(contour);
    float lastDist = 0;

    // hladame vsetky lokalne maxima vzdialenostnej funkcie
    for (int i = 0; i < contour.Count; i++)
    {
        float dist = (float) Math.Sqrt(Math.Pow(contour[i].X - center.X, 2)
            + Math.Pow(contour[i].Y - center.Y, 2));
        if (dist > lastDist)
        {
            if(growing)
            {
                maxims[j] = contour[i];
                distances[j] = dist;
            }
            else
            {
                maxims.Add(contour[i]);
                distances.Add(dist);
                growing = true;
            }
        }
        else if (growing)
        {
            j++;
```

```
            growing = false;
        }

        lastDist = dist;
    }

    List<Point> vertices = new List<Point>();
    // mam prvy vrchol (bod TopLeft) - je nim bod 0
    vertices.Add(maxims[0]);
    int k = 1;
    // teraz hladam kedy sa prvy krat zacne zmensovat distFunction
    while(k < distances.Count)
    {
        float dist = distances[k-1];
        // najprv zbehne zmensovanie
        while (k<distances.Count && distances[k]<=dist)
        {
            dist = distances[k];
            k++;
        }
        // potom to stupa
        while (k < distances.Count && distances[k] > dist)
        {
            dist = distances[k];
            k++;
        }
        // mame i-te lokalne maximum
        vertices.Add(maxims[k-1]);
    }

    // nakoniec treba aj tak este skusit, ci nelezia na 1 priamke
    int min = 0;
    k = 0;
    while (vertices.Count > vertexCount)
    {
        // prelozim priamku bodmi vertices[k], vertices[k+2]
        // bod vertices[k+1] odstranim vtedy, ak jeho vzdialenost od
              priamky je mensia ako min
        if(ImageArea.PointLineDistance(vertices[k],vertices[(k + 2) %
            vertices.Count],vertices[(k + 1) % vertices.Count])<=min)
        {
            vertices.RemoveAt((k + 1) % vertices.Count);
        }
        else
        {
            k++;
        }
        if (k >= vertices.Count)
        {
            k = 0;
            min++;
        }
    }

    // vrati vrcholy
    return vertices;
```

2. Určenie smeru pohľadu

Implementácia smeru pohľadu používateľa ku hviezdam bola priamočiarou realizáciou návrhu zo začiatku semestra. Bolo však nutné uvedomiť si, ktorým smerom jednotlivé hodnoty azimutu a výšky stúpajú s ohľadom na to, že používateľ bude pod snímaným priesvitným materiálom ležať. Výsledkom doplnenia určovania smeru na všetky 4 kvadranty je koncový výpočet smeru pohľadu v metóde UsePoints triedy SkyZone:

```
if(pts.Count==0)
    return;

if (pts.Count == 1)
{
    data.callback(0, Math.PI/2);
}
else
{
    // azimut a vyska
    double azimut;
    double vyska;

    // vyber bodov reprezentujucich odraz hlavy a laseroveho luca
    ColoredPoint reflex, laser;
    GetPoints(pts, out reflex, out laser);
    log.Debug("odraz hlavy:" + reflex.Position + ", laser na skle:" +
laser.Position);

    // "pixlova vzdialenost" od priemetne
    double h1 = imageWidth / data.TwoTanHorizontalAngle;
    // tangens uhla medzi kolmicou a priamkou KameraOdrazhlavy
    double xiX = Math.Tan(Math.Atan((reflex.Position.X - imageWidth / 2.0)
/ h1) + data.YRotation);
    // tangens uhla medzi kolmicou a priamkou KameraLaser
    double psiX = Math.Tan(Math.Atan((laser.Position.X - imageWidth / 2.0)
/ h1) + data.YRotation);

    // "pixlova vzdialenost" od priemetne
    double h2 = imageHeight / data.TwoTanVerticalAngle;
    // uhol medzi kolmicou a priamkou KameraOdrazhlavy
    double xiY = Math.Tan(Math.Atan((reflex.Position.Y - imageHeight / 2.0)
/ h2) + data.XRotation);
    // uhol medzi kolmicou a priamkou KameraLaser
    double psiY = Math.Tan(Math.Atan((laser.Position.Y - imageHeight / 2.0)
/ h2) + data.XRotation);


    if (double.IsNaN(xiX) || double.IsNaN(psiX) || double.IsNaN(xiX) ||
double.IsNaN(psiX))
    {
        azimut = 0;
        vyska = 0;
        log.Warn("Calculating azimut failed, could not evaluate Tan() while
it is NaN");
    }
    else
    {
        // dlzka usecky CL0 bez nasobku realnej vzdialenosti od priemetne
```

```
        double CL = Math.Abs(2 * xiX - psiX);
        // dlzka usecky CL0 bez nasobku realnej vzdialenosti od priemetne
        double CU = Math.Abs(2 * xiY - psiY);

        // vypocet azimutu
        if (CU == 0)
        {
            if (laser.Position.X < reflex.Position.X)
                azimut = Math.PI / 2.0;         // ak je laser viac dolava
ako hlava, pozeram sa na vychod
            else
                azimut = 3 * Math.PI / 2.0;     // ak je laser viac doprava
ako hlava, tak sa pozeram na zapad
        }
        else
        {
            double azAng = Math.Atan(CL / CU);
            if (laser.Position.Y < reflex.Position.Y)       // ak je laser
nad hlavou, tak sa pozeram na severo-XXX
            {
                if (laser.Position.X < reflex.Position.X)   // ak je laser
viac dolava ako hlava, pozeram sa na vychod
                    azimut = azAng;                 // severo-vychodny
kvadrant
                else                                        // ak je laser
viac doprava ako hlava, tak sa pozeram na zapad
                    azimut = 2 * Math.PI - azAng;   // severo-zapadny
kvadrant
            }
            else                                            // ak je laser
nizsie ako hlava, tak sa pozeram na juho-XXX
            {
                if (laser.Position.X < reflex.Position.X)   // ak je laser
viac dolava ako hlava, pozeram sa na vychod
                    azimut = Math.PI - azAng;       // juho-vychodny
kvadrant
                else                                        // ak je laser
viac doprava ako hlava, tak sa pozeram na zapad
                    azimut = Math.PI + azAng;       // juho-vychodny
kvadrant
            }
        }

        // uprava o otocenie smerom na vychod
        azimut += data.ZRotation;
        // vypocet vysky
        vyska = Math.Atan(1 / (Math.Sqrt(Math.Pow(CL, 2) + Math.Pow(CU,
2))));

        // mame vypocitane co sme potrebovali - dame zaznam do logu
        log.Debug("azimut: " + azimut * 180 / Math.PI + ", vyska: " + vyska
* 180 / Math.PI);

        // notifikacia o novom smere pohladu
        data.callback(azimut, vyska);
    }
}
```

Výpočet novej pozície kurzoru myši na monitore je realizované v metóde UsePoints triedy ScreenZone, ktorá vyberie z bodov nájdených ColorFinderom ten najvhodnejší a zavolá preň metódu GetRelativeCoordination triedy ImageShape.

```
double a = PointLineDistance(TopLeft, TopRight, P);
double b = PointLineDistance(BottomLeft, BottomRight, P);
double c = PointLineDistance(TopLeft, BottomLeft, P);
double d = PointLineDistance(TopRight, BottomRight, P);
if((a+b == 0) || (c+d==0))
    return new PointF(0, 0);
else
    return new PointF((float)(c / (c + d)), (float)(a / (a + b)));
```

Trieda ImageShape reprezentuje tvar obrazca v snímku z kamery, v našom prípade je to tvar monitora. Napríklad jej vlastnosť TopLeft neznamená jeho najvrchnejší a najľavejší okraj, ale súradnice toho vrcholu, ktorý predstavuje ľavý horný roh pôvodnej obrazovky (tj po otočení). Metóda PointLineDistance počíta vzdialenosť bodu od priamky.

```
if (A.X == B.X)
{
    return Math.Abs( P.X - A.X);
}
else
{
    double alpha = (B.Y - A.Y) / Convert.ToDouble(B.X - A.X);
    double Tx = (alpha * (P.Y - A.Y + alpha * A.X) + P.X) / (1 + alpha *
alpha);
    double Ty = alpha * Tx + A.Y - alpha * A.X;
    return Math.Sqrt((Math.Pow(P.X - Tx, 2) + Math.Pow(P.Y - Ty, 2)));
}
```

3. Doplnenie prevodov medzi súradnicovými sústavami

Do prevodov medzi súradnicovými sústavami bola pri implementácií výslednej aplikácie doplnená trieda EclCoordinates zabezpečujúca reprezentáciu a prevody z ekliptikálnej súradnicovej sústavy. Trieda pracuje so zemskými ekliptikálnymi súradnicami, nie so súradnicami so stredom v Slnku, pretože ostatné súradnice sú tiež vzťahované na Zem.

Implementovaný je prevod do rovníkovej súradnicovej sústavy, používa sa nasledujúci kód na prevod:

```
            // calculate declination
            double dec = Math.Asin(Math.Cos(latitude) * Math.Sin(longitude)
* Math.Sin(eps) +
                Math.Sin(latitude) * Math.Cos(eps));
            double N = Math.Cos(latitude) * Math.Sin(longitude) *
Math.Cos(eps) -
                Math.Sin(latitude) * Math.Sin(eps);

            double D = Math.Cos(latitude) * Math.Cos(longitude);
            // calculate right ascension using N and D
```

```
            double ra = Math.Atan(N / D);
            if (D < 0)
            {
                ra = Math.PI + ra;
            }
```

4. Katalóg a vyhľadávanie v katalógu

Pri implementácií finálnej aplikácie bol doplnený katalóg umožňujúci vyhľadávanie planét na základe určeného času a pozície. Triedy katalógu dodržiavajú návrh a nedošlo k žiadnym zmenám. Výpočty pozície planét a Mesiaca pochádzajú z knihy Astronomické algoritmy pro kalkulátory [24]. Algoritmy sú súčasťou metódy public List<CatItem> GetItems(double time) a pridružených metód. Ide o pomerne zložité výpočty, najprv sa vypočítajú pozície planét v ekliptikálnej súradnicovej sústave so Slnkom v strede. Pokračuje sa konverziou týchto súradníc do ekliptikálnej súradnicovej sústavy so Zemou v strede:

```
                N = p.Distance * Math.Cos(p.YCoord.ToRadians()) *
Math.Sin(p.XCoord.ToRadians() - LeS);
                D = p.Distance * Math.Cos(p.YCoord.ToRadians()) *
Math.Cos(p.XCoord.ToRadians() - LeS) + 1;
                Le = Math.Atan(N / D);
                if (D < 0)
                {
                    Le = Math.PI + Le;
                }
                Le = Le + LeS;
                int i = (int)(Le / (2 * Math.PI));
                Le = Le - i * 2 * Math.PI;

                a = p.Distance * Math.Sin(p.YCoord.ToRadians());
                    d = Math.Sqrt(N*N + D*D + a*a);

                Wi = Math.Asin((p.Distance * Math.Sin(p.YCoord.ToRadians())
/ d));
```

Kde Wi a Le sú nové súradnice, d je vzdialenosť planéty od Zeme. Nasleduje výpočet polohy Mesiaca, ktorý je priamo počítaný v ekliptikálnej súradnicovej sústave so Zemou v strede. Metóda vráti zoznam všetkých planét (inštancie PlanetItem a MoonItem) s vypočítanými pozíciami.

Vyhľadávanie podľa súradníc a konverziu súradníc do rovníkovej súradnicovej sústavy zabezpečuje metóda public List<CatItem> FindItems(double xcoord, double ycoord, double distance, DateTime time).

```
                pi = (PlanetItem)ci;
                ecl.Longitude = pi.XCoord.ToRadians();
                ecl.Latitude = pi.YCoord.ToRadians();
                eq = ecl.ToEqCoordinates(t);
                ra2 = eq.RAscension;
                dec2 = eq.Declination;

                dist = Math.Acos(d1sin * Math.Sin(dec2) + d1cos *
Math.Cos(dec2) * Math.Cos(xcoord - ra2));
```

```
                //moon is compared separately in second part of condition,
due it's visible radius is greater.
                if ((dist < distance) || (pi.Id.Equals("M") && (dist -
((Moon)pi).Radius < distance)))
                {
                    pi.XCoord = new DMS(ra2);
                    pi.YCoord = new DMS(dec2);
                    pi.CoordType = CatItem.COORD_EQUATORIAL;
                    returnList.Add(pi);
                }
```

Na základe tejto metódy vyhľadávajú aj všetky ostatné metódy definované v rozhraní ICatalogue.

Trieda StarFinder v module StarFinder bola doplnená o tento katalóg a súčasne vyhľadáva hviezdy aj planéty v danej oblasti.

5. Ovládač myši

Na základe rozpoznávania obrazu a výpočtov v ImageProcessori sa určuje nová pozícia kurzoru myši. Samotný pohyb kurzorom myši je možné realizovať veľmi jednoducho použitím volaní, ktoré sú štandardne v .Net 2.0 nasledovne:

```
System.Windows.Forms.Cursor.Position = pt;
```

Rámec .Net však neposkytuje rutinu pre vyvolanie kliknutia myšou. Uvedenú funkcionalitu je možné realizovať volaním Win Api funkcie mouse_event v knižnici user32.dll. Realizácia základnej funkcionality – kliknutia myšou, je realizované v triede MouseClicker:

```
using System;
using System.Runtime.InteropServices;

/// <summary>
/// obladac tlacidiel mysi
/// </summary>
/// <author>Michal Dobis</author>
public class MouseClicker
{
    [DllImport("user32.dll")]
    private static extern void mouse_event(UInt32 dwFlags, UInt32 dx,
        UInt32 dy, UInt32 dwData, IntPtr dwExtraInfo);

    private const UInt32 MouseEventLeftDown = 0x0002;
    private const UInt32 MouseEventLeftUp = 0x0004;

    /// <summary>
    /// posiela kliknutie lavym kurzorom mysi
    /// </summary>
    public static void SendMouseClick()
    {
        mouse_event(MouseEventLeftDown, 0, 0, 0, new System.IntPtr());
        mouse_event(MouseEventLeftUp, 0, 0, 0, new System.IntPtr());
    }
```

```
}
```

6. Získavanie zoznamu implementovaných tried

Pre účely maximálnej rozšíriteľnosti aplikácie sa pri vytváraní niektorých tried nepoužíva ich konštruktor priamo, ale konkrétny typ vytváranej inštancie sa určuje dynamicky len z jej názvu daného nastaveniami aplikácie. Uvádzame príklad inštancovania triedy pre ovládanie kamery:

```
public static ImageSource CreateCamera(string type, object[]
    parameters)
{
    Type camType = Type.GetType("Netrollers.icPoint.VideoSource." +
          type);
    if (camType != null)
    {
        Type[] tps = new Type[parameters.Length];
        for (int i = 0; i < parameters.Length; i++)
        {
            tps[i] = parameters[i].GetType();
        }
        ConstructorInfo constuct = camType.GetConstructor(tps);
        if (constuct != null)
        {
            return constuct.Invoke(parameters) as ImageSource;
        }
    }
    return null;
}
```

Príjemné používateľské rozhranie potom vyžaduje poskytnutie zoznamu implementovaných tried (ovládačov). Nadobudnutie takéhoto zoznamu je možné realizovať nasledovne:

```
public static string[] GetImplementedCameraList()
{
    Type camType =
        Type.GetType("Netrollers.icPoint.VideoSource.ImageSource");
    Type[] tps = camType.Module.GetTypes();
    List<string> res = new List<string>();
    foreach (Type t in tps)
    {
        if (t.IsClass && !t.IsAbstract)
        {
            Type bt = t.BaseType;
            while (bt != null && bt != typeof(object))
            {
                if (bt == camType)
                {
                    res.Add(t.Name);
                    break;
                }
                else
                {
                    bt = bt.BaseType;
                }
```

```
                }
            }
        }
        return res.ToArray();
    }
```

7. Rozpoznávanie hlasu a vytvorenie príkazu

Rozpoznávanie hlasu vyžaduje ako svoj vstup definíciu gramatiky, ktorú má rozpoznávať. Gramatika sa zapisuje v formáte XML. Uvádzame príklad zápisu niekoľkých pravidiel:

```
<GRAMMAR LANGID="409">
      <RULE ID="1"  NAME="MouseClickCommand" TOPLEVEL="ACTIVE">
            <P>mouse down</P>
      </RULE>

      <RULE ID="2"  NAME="CloseAppCommand" TOPLEVEL="ACTIVE">
            <L PROPNAME="closelist">
                  <P>+close application</P>
                  <P>+quit application</P>
            </L>
      </RULE>

      <RULE ID="3"  NAME="UpdateBackgroundCommand" TOPLEVEL="ACTIVE">
            <P>update background</P>
      </RULE>

      <RULE ID="4"  NAME="UpdateStarDisplayCommand" TOPLEVEL="ACTIVE">
            <L PROPNAME="updatestars">
                  <P>update display</P>
                  <P>update sky</P>
                  <P>use current sight</P>
                  <P>show stars</P>
            </L>
      </RULE>

      <RULE ID="5"  NAME="SelectStarCommand" TOPLEVEL="ACTIVE">
            <L PROPNAME="direction">
                  <P VALSTR="NEXT">next</P>
                  <P VALSTR="NEXT">further</P>
                  <P VALSTR="PREV">previous</P>
                  <P VALSTR="PREV">closer</P>
            </L>
            <P>star</P>
      </RULE>

      ...

</GRAMMAR>
```

Rozpoznaný príkaz sa použitím udalostí jazyka C# dostáva do hlavného modulu aplikácie, kde ho spracováva nasledovný kód:

```
    VoiceCommand command = VoiceCommand.CreateCommand(e.command, e.text);
    if (command != null)
    {
```

```
        if(OnVoiceCommand!=null)
            OnVoiceCommand(e.text);
        command.Execute(e.arg);
    }
```

Statická metóda CreateCommand tiedy VoiceCommand vyvára konkrétny príkaz nasledovne:

```
    Type commandType =
           Type.GetType("Netrollers.icPoint.WpfGui.Manager.VoiceCommands." +
           type);

    if (commandType != null)
    {
        Type[] tps = new Type[] {};
        ConstructorInfo constuct = commandType.GetConstructor(tps);
        if (constuct != null)
        {
            VoiceCommand com = constuct.Invoke(new object[]{}) as
                  VoiceCommand;
            if (com != null)
            {
                log.Debug("Command created: " + commandType.Name);
                com.text = text;
                return com;
            }
        }
    }
    return null;
```

8. Výber polohy na Zemeguli pomocou mapy

Rýchla a pohodlná realizácia určovania pozície na zemeguli je použitie hotového riešenia. Príčinou je najmä obtiažnosť získania dostatočne podrobných máp a databázy geografických polôh svetových miest. Rozhodli sme sa použiť voľne dostupné Google Maps Api, ktorého jediným obmedzením je, že vyžaduje pripojenie na internet. Z licenčných podmienok ďalej vyplýva, že samotná realizácia mapy musí byť verejne dostupná.

Pre lokálnu aplikáciu bolo potrebné len to, aby stránka hostujúca mapu vedela spracovať zemepisnú šírku a dĺžku zadanú ako parameter v URL a aby aktuálne zvolenú polohu na mape prezentovala v textovej podobe. Mali sme k dispozícii voľne dostupný server s PHP, tak výsledný kód vyzerá nasledovne:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:v="urn:schemas-microsoft-com:vml">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=utf-8"/>
    <title>icPoint - choose your location</title>
    <style type="text/css">
    v\:* {
      behavior:url(#default#VML);
    }
    </style>
    <script src="http://maps.google.com/maps?file=api&amp;v=2&amp;key=XXX"
      type="text/javascript"></script>
    <script type="text/javascript">
```

```
  //<![CDATA[
  var map = null;
  var geocoder = null;
  var origPoint = new GLatLng(
     <?
     if($latitude && $longitude):
       echo "$latitude,$longitude";
     else:
       echo "48.153375053814926, 17.07348346710205";
     endif;
      ?>);
  var marker = new GMarker(origPoint, {draggable: true});

  function load() {
    if (GBrowserIsCompatible()) {
      map = new GMap2(document.getElementById("map"));
     geocoder = new GClientGeocoder();
      map.setCenter(origPoint, 13);
      updatePointInfo(origPoint);
     map.addControl(new GLargeMapControl());
      map.addControl(new GMapTypeControl());
     map.addOverlay(marker);

     GEvent.addListener(map, "click", function(overlay, point) {
        if (overlay) {
          return;
        }
       else
        {
         marker.setPoint(point);
         updatePointInfo(point);
        }
      });

     GEvent.addListener(marker, "dragstart", function() {
     });

     GEvent.addListener(marker, "dragend", function() {
       updatePointInfo(marker.getPoint());
     });
    }
  }

  function showAddress(address) {
    if (geocoder) {
      geocoder.getLatLng(
        address,
        function(point) {
          if (!point) {
            alert(address + " not found");
          } else {
            map.setCenter(point, 13);
            marker.setPoint(point);
           updatePointInfo(point);
          }
        }
      );
    }
  }

  function updatePointInfo(point) {
     document.getElementById("location_message_longitude").innerHTML= oint.lng().toString();
      document.getElementById("location_message_latitude").innerHTML=point.lat().toString();
  }
  //]]>
  </script>
</head>
<body onload="load()" onunload="GUnload()">
  <form action="#" onsubmit="showAddress(this.address.value); return false">
    <p>
      <input type="text" size="60" name="address" value="Bratislava, SVK" />
      <input type="submit" value="Go!" />
    </p>
    <div id="map" style="width: 640px; height: 480px"></div>
    <div id="location_message_longitude"></div><div id="location_message_latitude"></div>
  </form>
</body>
```

```
</html>
```

9. Grafické používateľské rozhranie

Používateľské rozhranie sme realizovali pomocou technológie Windows Presentation Foundation. Funkcionalitu sme ponechali zapísanú v jazyku C#, no nemenné časti samotnej grafiky sme vytvárali v jazyku XAML. Nasledujúci príklad ukazuje zápis základnej časti GUI komponentu StarDetailDisplayer:

```
<UserControl
      xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
      xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
      x:Class="Netrollers.icPoint.WpfGui.SkyDetail.SkyDetailDisplayer"
      Width="300" Height="600" x:Name="UserControl" Opacity="0.7"
BorderThickness="0,2,2,2" Background="#FF7E7C7C">

      <UserControl.BorderBrush>
            <LinearGradientBrush EndPoint="1,0.5" StartPoint="0,0.5"
SpreadMethod="Reflect">
                  <GradientStop Color="#FF545454" Offset="0.837"/>
                  <GradientStop Color="#FF7E7E7E" Offset="1"/>
                  <GradientStop Color="#FF3C3C3C" Offset="0.519"/>
            </LinearGradientBrush>
      </UserControl.BorderBrush>

      <StackPanel>
            <Grid Name="topDecoration">
                  <TextBlock Name="WindowTitle" TextAlignment="Left"
Width="Auto" Height="Auto" Text="Fuuu" Foreground="#DBFFFFFF"
FontFamily="Arial" FontSize="20">
                    <TextBlock.RenderTransform>
                      <TransformGroup>
                        <MatrixTransform
Matrix="1.33333,0,0,1.33333,22,12"/>
                      </TransformGroup>
                    </TextBlock.RenderTransform>
                    <TextBlock.BitmapEffect>
                      <OuterGlowBitmapEffect GlowSize="4.53542" Opacity="1"
GlowColor="#FF727272" Noise="0"/>
                    </TextBlock.BitmapEffect>
                </TextBlock>

            </Grid>

            <ScrollViewer HorizontalScrollBarVisibility="Auto"
VerticalScrollBarVisibility="Auto" Margin="15,15,15,100"
x:Name="skyPresentationScroller">
                  <StackPanel ScrollViewer.CanContentScroll="True"
Name="skyPresentationContent" Height="Auto"><!--Margin="15,15,15,100"-->
                        <StackPanel Name="commonData" Margin="5,0,5,15"/>

                        <Expander Name="starConstData" Margin="0,10"
Height="Auto" IsExpanded="True">
                              <Expander.Header>
                                    <TextBlock Name="starConstTitle"
FontFamily="Algerian"/>
                              </Expander.Header>
```

```
                              <StackPanel Name="starConstDataContainer"
Height="Auto" Background="#FF8C8C8C"/>
                        </Expander>

                        <Expander Name="localInfData" Margin="0,10,0,0"
Height="Auto" IsExpanded="True">
                              <Expander.Header>
                                    <TextBlock FontFamily="Algerian"
Text="{DynamicResource StarDetail.LocalisedInformation}"/>
                              </Expander.Header>
                              <StackPanel Name="localInformationContainer"
Background="#FF8C8C8C"/>
                        </Expander>
                  </StackPanel>
      </ScrollViewer>
      </StackPanel>
</UserControl>
```

10. Zobrazenie hviezd

Pre transformáciu horizontálnych súradníc hviezdy na súradnice obrazovky sa pri zobrazovaní využíva algoritmus, popísany v podkapitole 7.4.2. Praktická implementácia algoritmu v rámci triedy *PlanarProjection* vyzerá nasledovne:

```
//calculates 3D world coordinates from azimuth, altitude and sphere
distance
protected Point3D Get3DCoords(double azimuth, double altitude, double
distance)
{
      Point3D newPoint = new Point3D();

      newPoint.Z = Math.Sin(altitude) * distance;

      double projectedDistance = distance * Math.Cos(altitude);
      newPoint.X = Math.Sin(azimuth) * projectedDistance;
      newPoint.Y = Math.Cos(azimuth) * projectedDistance;

      return newPoint;
}


//rotate a point by the axes of the world coordinate system,
//the origin of the coordinate system is the center of rotation
protected Point3D Rotate3D(Point3D myPoint, double angleX, double angleY,
double angleZ)
{
      Point3D oldPoint = new Point3D();
      Point3D newPoint = new Point3D();

      oldPoint.X = myPoint.X;
      oldPoint.Y = myPoint.Y;
      oldPoint.Z = myPoint.Z;

      if (angleX != 0)
      {
      //rotation about the X axis
```

```
        newPoint.Y = oldPoint.Y * Math.Cos(angleX) - oldPoint.Z *
        Math.Sin(angleX);
        newPoint.Z = oldPoint.Y * Math.Sin(angleX) + oldPoint.Z *
        Math.Cos(angleX);

        oldPoint.Y = newPoint.Y;
        oldPoint.Z = newPoint.Z;
        }

        if (angleY != 0)
        {
        //rotation about the Y axis
        newPoint.X = oldPoint.X * Math.Cos(angleY) + oldPoint.Z *
        Math.Sin(angleY);
        newPoint.Z = - oldPoint.X * Math.Sin(angleY) + oldPoint.Z *
        Math.Cos(angleZ);

        oldPoint.X = newPoint.X;
        oldPoint.Z = newPoint.Z;
        }

        if (angleZ != 0)
        {
        //rotation about the Z axis
        newPoint.X = oldPoint.X * Math.Cos(angleZ) - oldPoint.Y *
        Math.Sin(angleZ);
        newPoint.Y = oldPoint.X * Math.Sin(angleZ) + oldPoint.Y *
        Math.Cos(angleZ);

        oldPoint.X = newPoint.X;
        oldPoint.Y = newPoint.Y;
        }

        return oldPoint;
}

//Projection from 3D to 2D
protected Point Projection(Point3D myPoint, double zoomFactor)
{
        double X = (myPoint.X) * zoomFactor;
        double Y = - (myPoint.Z) * zoomFactor;

        Point newPoint = new Point(Convert.ToInt32(X),Convert.ToInt32(Y));
        return newPoint;
}
```

Uvedené funkcie slúžia pre vykonanie jednotlivých etáp transformácie. Kompletná transformácia potom vyzerá nasledovne:

```
//the whole projection process
public Point ProjectPoint(double azimuth, double altitude)
{
        Point3D myPoint = Get3DCoords(azimuth, altitude,
        this.SphereDistance);
        myPoint = Rotate3D(myPoint, 0, 0, this.viewAzimuth);
        myPoint = Rotate3D(myPoint, -this.viewAltitude, 0, 0);

        Point Point2D = Projection(myPoint, zoomFactor);
```

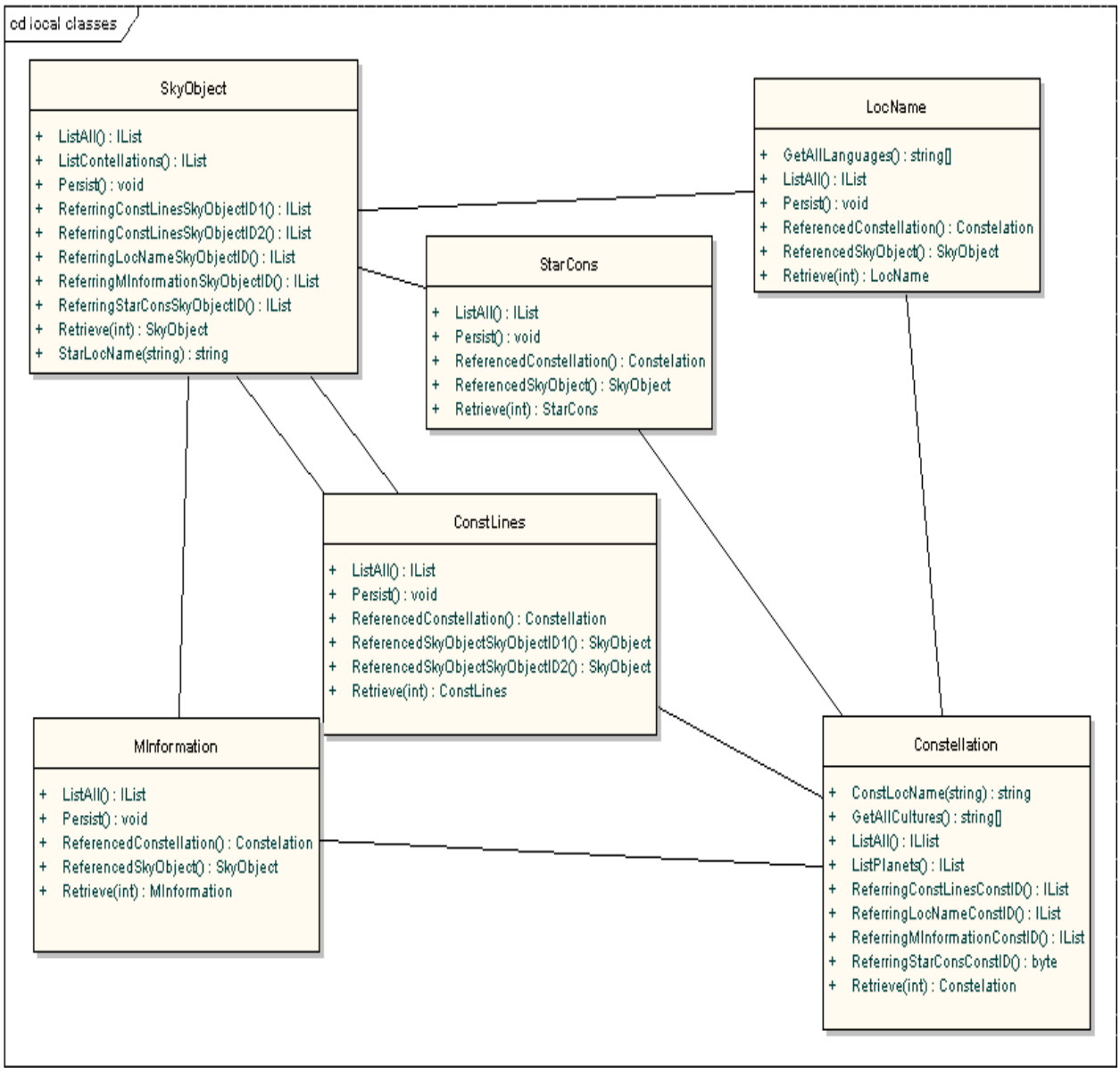
```
        //add the drawing offset
        Point2D.X = Point2D.X + this.offsetX;
        Point2D.Y = Point2D.Y + this.offsetY;

        return Point2D;
}
```

11. Lokálna databáza údajov

11.1.1. API lokálnej databázy

Na implementáciu rozhraní pre lokálnu databázu sa najprv používala technológia ADO.NET. Rozhodli sme sa ho nahradiť objektovo relačným mapperom Gentle.NET, ktorý je súčasne použitý aj vo webovej aplikácií. Na generovanie kódu tried pre Gentle.NET sme používali nástroj MyGeneration. Generovaný kód vychádza z tabuliek a relácií v databáze. Každá trieda prezentuje jednu tabuľku v databáze (atribúty triedy a tabuľky sú rovnaké). Základné funkcie na spracovanie databázy (napr. čítanie, vloženie, obnova a zmazanie) sú globálne pre všetky triedy. Naviac MyGeneration generuje ešte potrebné funkcie napr. zobrazenie všetkých záznamov danej tabuľky alebo záznamy, ktoré sa vzťahujú podľa relácie k danému záznamu, atď.

Diagram tried s metódami je zobrazený na Obr. 41. Atribúty každej triedy sú rovnaké ako v príslušnej tabuľke a význam vzťahov je podobný ako v diagrame tabuliek.



Obr. 41 Diagram tried s metódami

Ak chceme vykonať operáciu s údajmi (vložiť, obnoviť, atď.) môžeme zavolať príslušné statické metódy v triede Broker (napr. Broker.insert(), Broker.update(), atď.).

V každej triede vždy existujú tri rovnaké funkcie, to sú ListAll(), Retrieve(int id) a Persist(). Funkcia ListAll() získa všetky záznamy v tabuľke. Funkcia Retrieve(int) získa záznam podľa jeho identifikátora. Funkcia Persist() vloží do tabuľky nové údaje (Insert) alebo aktualizuje existujúce (Update). Okrem týchto metód ešte v každej triede môžu existovať metódy pre získanie entít podľa cudzieho kľúča. Ak tabuľka A má cudzí kľúč od tabuľky B, tak v triede A bude mať metódu ReferencedB(), ktorá vráti záznam tabuľky B (tento záznam sa vzťahuje k danému záznamu typu A). V triede B bude mať metódu ReferingA(), ktorá vráti jeden zoznam inštancií typu A (tieto inštancie majú cudzie kľúče v danom zázname typu).

Pre potreby používateľského rozhrania sme implementovali ďalšie metódy v triedach. Trieda Constellation má metódy ListPlanets(), ConstLocName() a GetAllCultures (). Metóda ListPlanets() vráti všetky hviezdy, ktoré sú v danom súhvezdí. Metóda GetAllCultures() vráti všetky kultúry v databáze. Metóda ConstLocName() vráti lokálne meno súhvezdia podľa jazyka. Trieda LocName má metódu GetAllLanguages(), ktorá vráti všetky používané jazyky. Trieda SkyObject má metódy ListConstellations() a StarLocName(). Metóda ListConstellations() vráti všetky súhvezdia, v ktorých je daná hviezda. Metóda StarLocName() vráti lokálne meno hviezdy podľa jazyka.

Príklad ošetrenia či LocName ukazuje na Constellation alebo nie.

```
public Constellation ReferencedConstellation()
            {
            if (this.constID == -1)
                return null;
            return Constellation.Retrieve(ConstID);
            }
        }
```

Pri zobrazení všetkých hviezd pre súhvezdie sa používa nasledovný dotaz:

```
String sql = "select so.id, so.catalogue, so.name, so.type from SkyObject
as so, Constellation as c, StarCons as s where so.id = s.skyObjectID and
s.constID = c.id and c.id =" + this.Id;
```

Nasledovný dotaz slúži na zobrazenie všetkých súhvezdí, do ktorých patrí daná hviezda:

```
String sql = "select c.id, c.culture, c.name from Constellation as c,
SkyObject as so, StarCons as s where so.id = s.skyObjectID and s.constID =
c.id and so.id = " + this.Id + " and c.culture = '" + culture +"'";
```

Zobrazenie všetkých jazykov:

```
String sql = "select distinct locale from LocName";
```

12. Webová aplikácia

Webová aplikácia je implementovaná pomocou jazyka C# a aplikačného rámca ASP.NET. Pre prístup k údajom sa používa objektovo relačný mapper Gentle.NET [28]. Umožňuje mapovať entity v databáze na triedy a pracovať s nimi objektovým spôsobom vrátane reprezentácie relácií. Triedy pre dátovú vrstvu boli vygenerované pomocou nástroja MyGeneration [29] a doplnené pre potreby aplikácie. Mapper používa atribúty pre určenie tabuľky a stĺpcov na ktorý sa má trieda a jej položky mapovať:

```
[TableName("WebUser", CacheStrategy.Temporary)]
public partial class WebUser : Persistent
{
      #region Members
      private bool isChanged;
```

```
        [TableColumn("id", NotNull=true),
        PrimaryKey(AutoGenerated=true)]
        private long id;
        [TableColumn("username", NullValue="")]
        private string username;
    ...
      }
```

Na základe týchto mapovaní a tried umožňuje mapper relatívne jednoduché vytváranie databázových dotazov pomocou objektov. Okrem prístupu pomocou týchto objektov umožňuje aj priamy prístup pomocou SQL príkazov a spätné mapovanie výsledkov dotazu na objekty.

Mapper je používaný aj pre samotné vkladanie a úpravu položiek v databáze. Ide o implementáciu navrhovaného algoritmu, ktorý vytvára v databáze históriu zmien. Je implementovaný pre súhvezdia, multimediálne informácie a lokalizované mená:

```
        LocName ln = new LocName();
        ln.FirstID = firstID;
        ln.Locale = locale;
        ln.Name = name;
        ln.SkyObjectID = skyObjectID;
        ln.ConstID = constID;

        Transaction transaction = null;
        try
        {
            transaction = new Transaction();
            transaction.Persist(ln);
            Action.InsertAction(1, Action.ACTION_UPDATE,
                 "Name='" + name + "', Locale='" + locale + "'",
                 DateTime.Now, Action.LOCNAME_TABLE, ln.Id,
transaction);

            transaction.Commit();
        }
        catch (Exception e)
        {
            if (transaction != null)
            {
                // rollback transaction and release database connection
                transaction.Rollback();
            }
            // rethrow the exception
            throw e;
         }
```

Metóda Action.InsertAction vloží do informáciu o akcií do tabuliek Action a ActionTables:

```
        Action a = new Action(type, description, created, userID);
        ActionTables at;


        transaction.Persist(a);
        for (int i = 0; i < changes.Count; i++)
        {
            at = (ActionTables)changes[i];
```

```
            at.ActionID = a.Id;
            transaction.Persist(at);
        }
```

Všetky zmeny sa vykonávajú v rámci jednej transakcie, takže ak dôjde výnimke, je táto transakcia vrátená späť a je tak zabezpečená konzistencia údajov.

Pri vyhľadávaní v týchto entitách sa používajú obmeny nasledovného SQL príkazu:, ktorý zabezpečí vyhľadanie najnovšieho záznamu.

```
String sql = "select LocName.id, LocName.firstID, LocName.locale, " +
             " LocName.name, LocName.skyObjectID, LocName.constID " +
               "from LocName " +
               "where LocName.id in ( " +
              "select max(id) from LocName where Locale !=
              'latShortcut' " +
                     "and " + objectid + " group by firstID)";
```

Jednotlivé triedy reprezentujúce údajové entity neobsahujú vždy metódy pre všetky operácie (vyhľadávanie, vkladanie, úprava a mazanie). Trieda SkyObject umožňuje iba vyhľadávanie, pretože v lokálnej aplikácií sú tieto údaje pevne dané možnosťami katalógov a nie je vhodné aby používatelia menili ich obsah. Trieda LocName obsahuje okrem získavania na základe id nebeského objektu a súhvezdia aj metódy pre vkladanie a úpravu. Neumožňuje vymazávanie, vzhľadom na to, že je možné chybnú položku zmeniť na správnu. Podobnú sadu metód obsahuje aj trieda MInformation z rovnakých dôvodov. Trieda Constellaiton obsahuje metódy pre vyhľadávanie na základe parametrov, pridanie nového súhvezdia (používateľ môže pridať svoje lokálne súhvezdia, ktoré v aplikácií nie sú) a úpravu súhvezdia. Vymazávanie neobsahuje, pretože je umožnená zmena všetkých položiek. Triedy StarCons a ConstLines obsahujú metódy pre vkladanie a vymazávanie záznamu. Ide o pomerne jednoduché entity, pre ktoré nie je zmena potrebná, a ak áno, je ju možné vykonať pomocou odstránenia a vloženia nového záznamu. Pre tieto entity sa nesleduje ani história.

Prezentačná vrstva je realizovaná pomocou ASP.NET stránok a formulárov. Používa sa podpora pre témy a lokalizáciu, ktoré umožňujú neskoršie zmeny používaného jazyka stránky a grafického dizajnu. Základnou stránkou definujúcou dizajn a menu je stránka Master.master a ostatné stránky ju používajú a dopĺňajú obsah. Pre prístup k údajom sa používa komponent ObjectDataSource a zobrazovanie zabezpečuje komponent GridView. Pridávanie údajov je realizované zvláštnymi formulármi a reakciou na udalosť pri potvrdení formuláru. Pri zachytení udalosti sú kontrolované políčka formulára a pomocou volania metódy pre vloženie údajov vykonaná požadovaná operácia.

Webová služba, ktorá je súčasťou serverovej časti aplikácie, slúži na aktualizáciu informácií v lokálnej databáze zo zdieľanej databázy. Deklarácie webových metód, ktorými táto služba disponuje sa nachádzajú v abstraktnej triede `abstract class icPointWebService`, ktorá je odvodená od triedy `System.Web.Services.WebService`. Prenos údajov medzi serverom (webovou službou) a klientom je riadený protokolom SOAP, ktorý pracuje nad protokolom HTTP.

Jednotlivé metódy webovej služby sú navrhnuté tak, aby poskytovali čo najelementárnejšie informácie, ktoré klient požaduje a nevyžadujú tak veľké objemy prenesených údajov.

Pri práci s údajmi v databáze, ktorá je v prípade implementovanej webovej služby obmedzená iba na výber údajov, sú využívané metódy objektovo-relačného mapera. Ten je používaný aj na manipuláciu s informáciami z web stránky, pretože pracujú nad rovnakou databázou.

Príloha B: Používateľská príručka