VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

APLIKACE PRO OVLÁDÁNÍ PRACOVIŠTĚ ROBOTICKÉHO RAMENE

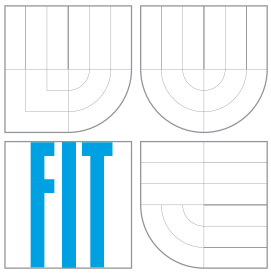
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

RADEK SEDLÁK

BRNO 2013

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

APLIKACE PRO OVLÁDÁNÍ PRACOVIŠTĚ ROBOTICKÉHO RAMENE

APPLICATION FOR CONTROLLING ROBOTIC ARM WORKCELL

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — RADEK SEDLÁK
AUTHOR

VEDOUCÍ PRÁCE — Ing. RADIM LUŽA
SUPERVISOR

BRNO 2013

Abstrakt

Tato práce se zabývá tvorbou aplikace pro vzdálené ovládání pracoviště robotického ramene na FIT VUT v Brně. Aplikace je vytvořena v jazyce C/C++ s využitím knihovny Qt. V práci jsou nejprve shrnuty obecné metodiky návrhu a testování uživatelských rozhraní. Dále je v práci popsán postup analýzy aplikace, návrhu uživatelského rozhraní, implementace aplikace a její otestování. V závěru práce jsou zhodnoceny výsledky testování a je navrženo další rozšíření aplikace.

Abstract

This thesis deals with creating the application for remote controlling robotic arm workplace at FIT BUT. The application is developed in C/C++ using the Qt library. The thesis first summarizes the general methodology for the design and testing of user interfaces. The thesis also describes the procedure for application analysis, user interface design, implementation and testing of the application. In conclusion, the test results are evaluated and proposed further expansion of the application.

Klíčová slova

uživatelské rozhraní, Qt, robotické rameno, aplikace, ovládání, metodika návrhu, metodika testování, metodika hodnocení

Keywords

user interface, Qt, robotic arm, application, controlling, design methodology, testing methodology, evaluation methodology

Citace

Radek Sedlák: Aplikace pro ovládání pracoviště robotického ramene, bakalářská práce, Brno, FIT VUT v Brně, 2013

Aplikace pro ovládání pracoviště robotického ramene

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Radima Luži.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Radek Sedlák
15. května 2013

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu Ing. Radimu Lužovi za odborné vedení při psaní této práce a za zpřístupnění pracoviště robotického ramene. Dále bych chtěl poděkovat rodičům a své přítelkyni za podporu při studiu.

© Radek Sedlák, 2013.
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

Kapitola 1

Úvod

Tvorba kvalitního uživatelského rozhraní patří bezesporu k důležitým aspektům při vývoji aplikací. Uživatelské rozhraní je jediný prostředek pro člověka, jak může s aplikací komunikovat. Pokud není toto rozhraní navrženo správně a řádně otestováno, uživatelé nemusí s aplikací pracovat efektivně. Při návrhu je třeba analyzovat, pro jakou cílovou skupinu je toto rozhraní tvořeno a jaké se od rozhraní očekávají výsledky. Při tvorbě uživatelskcýh rozhraní je také třeba určit, na jakých platformách bude aplikace spouštěna. Je zřejmé, že tvorba rozhraní pro např. dotykový mobilní telefon bude odlišná od rozhraní pro desktopovou stanici.

Cílem této práce je prostudovat a shrnout metodiky návrhu uživatelských rozhraní, dále se zaměřit na způsob hodnocení a jeho metrik, navrhnout a, v neposlední řadě, implementovat aplikaci, která bude sloužit ke vzdálenému ovládání pracoviště robotického ramene na FIT VUT v Brně.

Výsledná aplikace byla otestována a také bylo provedeno vyhodnocení intuitivnosti navrženého rozhraní a komfortu uživatele vzhledem ke zpoždění komunikace na síti. Po vyhodnocení všech testů bylo navrhnuto rozšíření této aplikace.

V práci jsou nejprve popsány teoretické informace k návrhu uživatelských rozhraní, způsobu jeho hodnocení a testování. Následuje analýza a návrh tvorby samotné aplikace. V této kapitole jsou shrnuty požadavky na výslednou aplikaci, dále je zde popsán návrh grafického uživatelského rozhraní (front-end část) a způsob jakým bude aplikace fungovat (back-end část). Po analýze a návrhu je popsána samotná implementace celé aplikace, problémy, které při řešení vznikly, a jejich řešení. Předposlední kapitola je zaměřena především na testování a vyhodnocení jeho zpětné vazby. V této kapitole jsou popsány metody, kterými byla aplikace testována a jejich přínos do výsledné podoby aplikace. V závěru je shrnut výsledek celé práce, provedeno vyhodnocení testování a navrženo další rozšíření aplikace.

Pro vypracování práce bylo nutné nastudovat knihovnu Qt a informace týkající se metodiky návrhu a způsobu hodnocení uživatelských rozhraní.

Kapitola 2

Teoretický úvod do uživatelských rozhraní

Tato kapitola si klade za cíl seznámit čtenáře této práce s pojmy souvisejícími s oblastí tvorby uživatelských rozhraní. Pojmy uvedené níže pomohou čtenáři lépe vstoupit do problematiky návrhu, tvorby a testování uživatelských rozhraní.

Metodika – Dle [8] je metodika teoreticko-praktické schéma určující postup provádění odborné činnosti, vychází z vědeckého poznání a empirie, přesně vymezuje jednotlivé postupy pro výkon dané činnosti.

Human-Computer Interaction (HCI) je disciplína zabývající se návrhem, hodnocením a implementací interaktivních výpočetních systémů pro lidské využití a studiem hlavních jevů, které je obklopují.[1]

Uživatelské rozhraní (User interface), také nazývané „UI“ nebo jednoduše rozhraní, je prostředek, kterým osoba ovládá softwarovou aplikaci nebo hardwarové zařízení. Dobré uživatelské rozhraní poskytuje „uživatelsky-přívětivé“ zkušenosti, umožňuje uživateli komunikovat se softwarem nebo hardwarem přirozenou a intuitivní cestou.[5]

Použitelnost je dle [17] atribut kvality, který posuzuje, jak snadno se uživatelské rozhraní používá.

Uživatelské zkušenosti (User experience) (UX) — ohledně UX se vyskytuje mnoho podobných definic. Např. dle Nielsena[18] UX zahrnuje všechny aspekty interakce koncového uživatele se společností, jejími službami a jejími produkty. Alben[9] definuje UX jako souhrn všech aspektů, jakým způsobem lidé používájí interaktivní produkt: jak cítí produkt ve svých rukách, jak moc rozumí tomu, jak produkt pracuje, jaké mají pocity, když produkt používají, do jaké míry slouží produkt svému účelu a jak dobře zapadá do oblasti, ve které je využíván.

Kapitola 3

Metodika návrhu uživatelských rozhraní

Metodika návrhu uživatelských rozhraní nezasahuje pouze do oblasti tvorby počítačových aplikací. Uživatelská rozhraní se totiž nenachází pouze u počítačových programů, ale u mnoha různých věcí (objektů), jako je např. automobil, pračka, mobilní telefon, bankomat atp. Tato kapitola se však bude zabývat metodikou návrhu uživatelských rozhraní v oblasti vývoje počítačových aplikací.

Návrh uživatelského rozhraní hraje v oblasti tvorby aplikace velice důležitou roli. Pokud totiž bude u aplikace brilantně zpracována pouze funkčnost a uživatelské rozhraní nebude dostatečně kvalitní, může se stát, že uživatelům, kteří budou chtít tuto aplikaci využít k usnadnění své práce, se tato práce pouze zhorší. Při vývoji aplikace je tedy nutné se návrhem uživatelského rozhraní zabývat. Následující podkapitoly popisují kritéria a aspekty správného a moderního přístupu k návrhu uživatelského rozhraní.

Je třeba zmínit, že metodika návrhu uživatelských rozhraní úzce souvisí s metodami a způsoby hodnocení a testování. V této práci je snaha metody rozdělit na část návrhu a část testování a hodnocení. V reálném životním cyklu vytvářené aplikace je ovšem v podstatě nemožné tyto metody rozdělit, protože při návrhu jsou již metody hodnocení a testování používány.

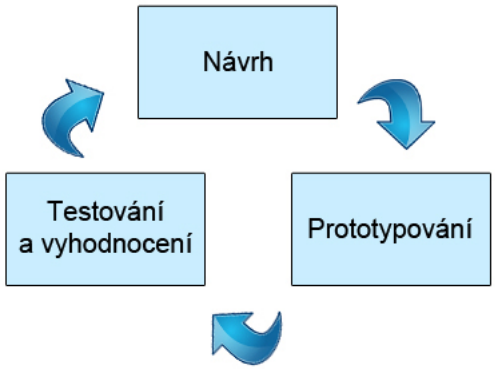
3.1 Iterativní návrh

Uživatelské rozhraní by mělo být navrhováno iterativně, protože je prakticky nemožné navrhnout uživatelské rozhraní, které by bylo navrženo najednou od začátku až do konce bezchybně. Dokonce ani experti v oblasti použitelnosti nedokáží na jeden pokus navrhnout perfektní uživatelské rozhraní, proto by měl být návrh rozdělen do několika iterací.[19]

Iterativní přístup zahrnuje postupné upřesňování a zdokonalování návrhu využíváním testování a hodnocení jednotlivých fází vývoje. Obvykle je po zkompletování návrhu nějaké části tato část otestována a jsou vyhodnoceny nalezené chyby. Tyto chyby jsou následně v dalším kroku iterace opraveny a krok je znovu otestován, zda se neobjeví chyby další. Iterativní návrh tedy funguje na principu zdokonalování původního návrhu na základě zkušeností, získaných z předchozí iterace, nikoliv na slepém zkoušení různých možností metodou pokus – omyl. Pokud je třeba volit mezi alternativami různých prvků, nebo rozhraní, nejedná se o iterativní přístup, ale o jinou metodiku. Toto porovnávání dvou různých návrhů je většinou využíváno pouze k měření a testování, nikoliv ke zdokonalování. [13]

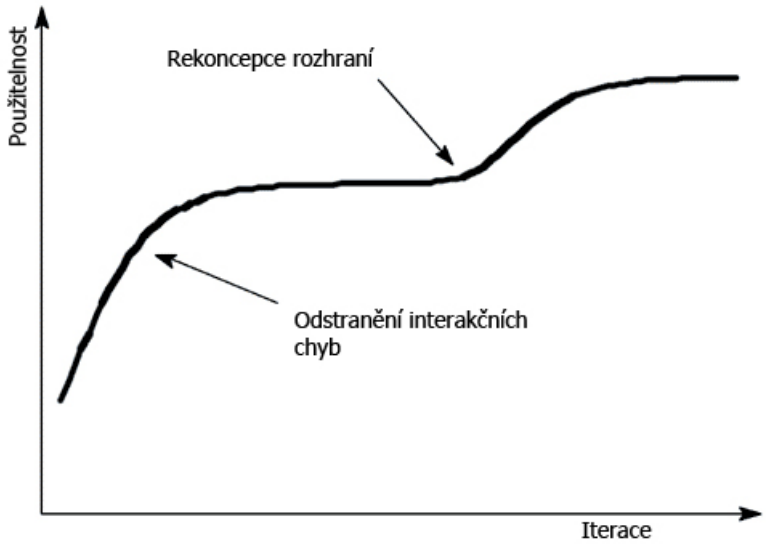
Proces iterativního návrhu může být aplikovaný od začátku vývoje uživatelského rozhraní až do jeho konce. Z důvodu jednoduchosti a ceny prováděných úprav je nejvhodnější využít jej již v ranných fázích vývoje.

Prvním krokem v iterativní metodě je vytvoření návrhu. Následuje vytvoření prototypu návrhu. Jakmile je prototyp vytvořen, měl by být řádně otestován, či ohodnocen. Nalezené chyby by následně v další iteraci měly být opraveny.



Obrázek 3.1: Proces iterativního návrhu

Na obrázku 3.1 je zobrazen vztah mezi použitelností a počtem iterací. Ideálně by měla každá iterace zvyšovat míru použitelnosti navrhovaného rozhraní. Tato míra použitelnosti by měla být lepší než v iteraci předchozí. Ne vždy je to však pravda. Pokud se totiž v návrhu provedou nějaké změny, může se ukázat, že další iterace způsobí zhoršení použitelnosti. Z tohoto důvodu nemusí být křivka na obrázku 3.1 v reálných případech tak hladká.[20]



Obrázek 3.2: Kvalita rozhraní jako funkce počtu iterací: Míra použitelnosti roste s každou další iterací, dokud nedosáhne potencionální hranice [2]

[2]Převzato a přeloženo z: http://www.nngroup.com/articles/iterative-design/

3.2 User centred design (Uživatelsky zaměřený návrh)

User centred design je metoda, která přibírá do vývoje reálné uživatele a zaměřuje se na ně v každé fázi návrhu. Hlavním cílem této metody je snaha optimalizovat výsledný produkt tak, aby byl co nejvíce přizpůsoben požadavkům uživatelů. Je více než nežádoucí, aby se uživatel musel učit novým návykům a měnit svoje zaběhnuté metody. Uživatelské požadavky jsou považovány za důležité po celou dobu vývoje a proto jsou do něj ihned začleňovány. Požadavky jsou následně více upřesňovány a zdokonalovány využitím testovacích a hodnotících metod. Metoda dodržuje normu ISO 9241-210, 2010 – Human-centred design for interactive systems, která popisuje šest principů zajišťujících, že je návrh uživatelsky zaměřený:

- Návrh je založen na jasné znalosti uživatelů, úkolů a prostředí.
- Uživatelé jsou zapojeni do návrhu a vývoje.
- Návrh je řízený a vylepšovaný uživatelsky zaměřeným testováním.
- Proces tvorby je iterativní.
- Konstrukční tým disponuje multi-disciplinárními znalostmi a perspektivami.

Norma dále popisuje čtyři základní kroky životního cyklu interaktivních aplikací:

- Specifikování kontextu použití
- Specifikování organizačních záležitostí
- Vytvoření návrhu řešení
- Vyhodnocení návrhu vůči požadavkům

Obrázek 3.3: ISO 9241-210: Human-centred design for interactive systems [3]

[28] [6] [4]

[3] Převzato a přeloženo z: http://uxlabs.co.uk/services/

3.3 Card sorting

Card sorting je levná, spolehlivá, rychlá uživatelsky zaměřená metoda, pomocí které lze zjistit, jak si potencionální uživatelé představují rozložení a seskupení prvků v uživatelském rozhraní. Jádro metody spočívá v tom, že uživatelé dostanou řadu karet. Na každé kartě jsou informace o její funkcionalitě v aplikaci. Uživatelé potom tyto karty mají setřídit, uspořádat, nebo seskládat tak, jak to podle nich dává nejlepší smysl. Tuto metodu je vhodné využívat buďto před samostatným návrhem uživatelského rozhraní, kdy uživatelé podle svého subjektivního pohledu dají vývojářům informace o tom, kam které prvky rozmístit a jak aplikaci strukturovat. Druhá možnost je využít tuto metodu jako metodu testovací, či hodnotící – po době, kdy je produkt nasazen do provozu lze zjistit, zda uživatelům dosavadní uspořádání a strukturalizace prvků v rozhraní vyhovuje, či nikoliv. Poté je možné případné nedostatky přehodnotit a uživatelské rozhraní přepracovat. [27]

3.4 Mock-up

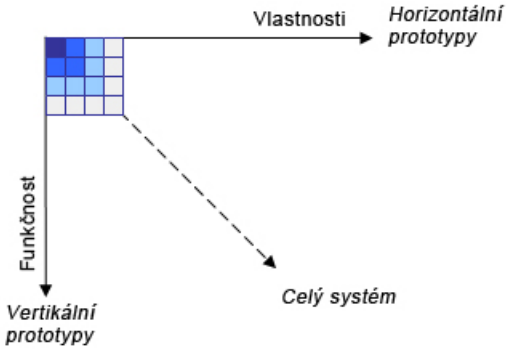
Mock-upy návrháři používájí v prvních fázích návrhu. Jelikož to jsou prvotní návrhy, je nutné, aby náklady na jejich tvorbu byly co nejmenší. Proto se k tvorbě mock-upů využívá papírových kartiček, tabulí, papírových náčrtů apod. Mock-upy slouží ke zjištění zpětné vazby uživatelů v oblasti použitelnosti, návrhu a funkčnosti navrhované aplikace. Výhodou mock-upů je, že při diskuzi a hodnocení návrhu mohou uživatelé návrh sami editovat, stačí jim k tomu pero, tužka, nůžky, či jiné běžně využívané kancelářské pomůcky. Mock-upy ale nemusí být využívány pouze pro zjištění informací od reálných uživatelů. Je vhodné je použít také mezi návrháři k lepšímu pochopení a znázornění návrhů. Na následujícím obrázku je ukázka mock-upu pro tvorbu aplikace kalendáře pro děti.[3]

Obrázek 3.4: Mockup – aplikace kalendář pro děti[4]

[4]Převzato z: [3]

3.5 Prototyping (Prototypování)

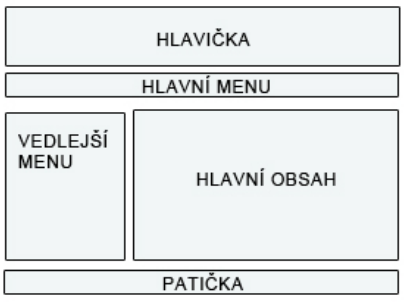
Prototypování je velice podobné mock-upům. Narozdíl od mock-upů však prototypování probíhá o něco později, využívá totiž dražších technik vyobrazení návrhu. Prototypy jsou částečně hotová řešení aplikace, na kterých je možno demonstrovat již implementované vlastnosti aplikace, či její funkcionalitu. Prototypy mohou být buďto horizontální, nebo vertikální. Horizontální prototypy zobrazují široký rozsah vlastností, které jsou ale jen velmi málo implementované. Vertikální prototypy mají jen velice málo vlastností, ale na druhé straně jsou tyto vlastnosti téměř plně implementovány. Prototypy se nezaměřují na grafické detaily, ale na obsahovou stránku a funkcionalitu, což je při návrhu uživatelských rozhraní stěžejní prvek.



Obrázek 3.5: Vertikální a horizontální prototypy[5]

3.6 Wireframe („Drátěný model“)

Wireframe, neboli drátěný model se používá k modelování kostry uživatelského rozhraní – rozložení prvků v okně, dialogu, na obrazovce, či webové stránce. V oblasti webových prezentací se je nyní wireframe velice často používaný a našel své pojmenování jako „skica webu“. Wireframe slouží k demonstraci rozmístění prvků bez detailnějšího grafického zpracování. Velikou výhodou je doba a cena, za kterou je možné takový model vytvořit. [7]



Obrázek 3.6: Příklad wireframe-u webové stránky

[5]Převzato a přeloženo z: [26]

Kapitola 4

Způsob hodnocení a testování použitelnosti uživatelských rozhraní

Hodnocení a testování použitelnosti zastřešuje metody, které se snaží odhalit problémy v použitelnosti navrhovaného uživatelského rozhraní.

Dle J.Nielsena[22] existují 4 směry, jak se hodnocení uživatelského rozhraní dá provést. První z nich je automatické ohodnocení (měření použitelnosti na základě běhu specifikace daného rozhraní nějkým programem), dále empirické (použitelnost ohodnocena na základě testování realnými uživateli), formální (využití přesných modelů a vzorců ke spočítání použitelnosti) a neformální (probíhá podle určitých pravidel a dále na základě obecných dovedností a zkušeností jednotlivých hodnotitelů). V současné době je z hlediska techniky automatické měření prakticky nemožné, kvůli vysokým nárokům na výpočetní výkon. Formální metody jsou velice těžko aplikovatelné. Empirické metody jsou nyní hlavním způsobem hodnocení uživatelských rozhraní, využívají testování samotnými uživateli. Často je ale obtížné, nebo drahé zajistit dostatečný počet uživatelů k otestování všech aspektů všech verzí vyvíjeného návrhu. Tento nedostatek vedl k zavedení neformální kontroly uživatelských rozhraní, která již není tak náročná. Tato metoda je také často vyžadována z důvodu omezeného rozpočtu projektu, nebo omezeného časového prostoru pro tvorbu dané aplikace. Přestože neformální metody nejsou tak náročné, jako metody empirické, jsou vysoce efektivní. Několik studií prokázalo, že neformální metody kontroly použitelnosti (inspekční metody) jsou schopny najít mnoho problémů použitelnosti, které jsou při uživatelském (empirickém) testování přehlíženy. Ale je také nutno dodat, že uživatelské testování je schopno odhalit nedostatky, se kterými si inspekční metody neporadí. Proto je vhodné při testování a hodnocení uživatelských rozhraní využívat kombinace uživatelského testování a inspekčních metod.

Kapitola 4.1 se zabývá inspekčními (neformálními) metodami. Metody heuristic evaluation, cognitive walkthrough, feature inspection,standards inspection obvykle využívají k hodnocení pouze jednoho hodnotitele. Naproti tomu metody pluralistic walkthrough a consistency inspections využívají k hodnocení skupiny hodnotitelů.

Kapitola 4.2 se zabývá metodami empirickými. Jak již bylo výše zmíněno, tak tyto metody hodnotí uživatelské rozhraní na základě zapojení reálných uživatelů do samostatného hodnocení.

4.1 Inspekční metody

Pojem inspekční metody zastřešuje sadu metod, které se snaží vyhodnotit uživatelské rozhraní a najít problémy v oblasti použitelnosti. Hodnotitelé u těchto metod jsou experti z oblasti použitelnosti uživatelských rozhraní, kteří na základě znalostí a zkušeností hodnotí a hledají chyby dle postupů u jednotlivých metod.

4.1.1 Heuristic evaluation (Heuristické hodnocení)

Heuristické hodnocení je metoda, při které hodnotitelé zkoumají uživatelské rozhraní a posuzují, zda toto rozhraní dodržuje doporučené principy použitelnosti (heuristiky). Hlavním cílem této metody je objevení nedostatků návrhu testovaného rozhraní. *Heuristické hodnocení je jedním z nejvíce neformálních metod využívaných ke kontrolování použitelnosti v oblasti HCI* 2 [24].

Výhodou tohoto hodnocení je, že existuje velké množství heuristik, které se navzájem nevylučují. Známé problémy jsou také často rozděleny do strukturovaných kategorií seřazených podle důležitosti dopadu na efektivnost práce uživatele s rozhraním. Heuristické hodnocení je prospěšné především v raných fázích návrhu, proto je často prováděno pro získání zpětné vazby k případům užití, které jsou pro danou aplikaci stanoveny. Tato hodnocení mohou výrazně snížit počet zjištěných problémů při pozdějších testech s reálnými uživateli aplikace. Narozdíl od metod uživatelského testování, které je velice nákladné a náročné, stačí pro heuristickou metodu jeden expert, což výrazně snižuje náklady a čas strávený testováním. Většina hodnocení může být provedena během několika dní (samozřejmě záleží na složitosti hodnocení, rozsahu testování atd.). V souvislosti s tím, že může být heuristická metoda prováděna jen jedním expertem, je často tato metoda odsuzována proto, že jsou výsledky hodnocení ovlivněny odbornou znalostí hodnotícího experta. Při využívání této metody musí být samozřejmostí, že výsledek může být subjektivně ovlivněn.

Napříč všemi výhodami a nevýhodami jsou tyto metody často používány k hodnocení, kdy je k tvorbě uživatelského rozhraní vymezen velice krátký časový prostor a množství dostupných financí omezuje využití ostatních testovacích metod.

Tato metoda byla vyvinuta konzultantem použitelnosti uživatelskcýh rozhraní Jakobem Nielsenem na základě jeho několikaletých zkušeností v oblasti školení a konzultací.

V následující podkapitole jsou uvedeny heuristiky, které jsou v dnešní době nejvíce využívány. Tyto heuristiky vyvinul sám Jakob Nielsen společně s Rolfem Molichem v roce 1990. Konečná sada heuristik však byla vydána Nielsenem v roce 1994 a je používána dodnes.

Nielsenovy heuristiky[14]

> **Visibility of system status:**
> The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.
>
> **Match between system and the real world:**
> The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
>
> **User control and freedom:**
> Users often choose system functions by mistake and will need a clearly marked

"emergency exit"to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.

Consistency and standards:
Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Error prevention:
Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

Recognition rather than recall:
Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Flexibility and efficiency of use:
Accelerators—unseen by the novice user—may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

Aesthetic and minimalist design:
Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Help users recognize, diagnose, and recover from errors:
Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

Help and documentation:
Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Volný překlad Nielsenových heuristik:

Viditelnost stavu systému:
Systém by měl vždyt udržovat uživatele informované o tom, co se právě děje, a to v odpovídající zpětné vazbě v přiměřené časové lhůtě.

Shoda mezi systémem a reálným světem:
Systém by měl „mluvit“ jazykem uživatele, se slovy, frázemi a pojmy známými pro uživatele, spíše než termíny systémově orientovanými. Následuje konvence reálného světa, informuje v přirozeném a logickém pořadí.

Uživatelská kontrola a svoboda:
Uživatelé často chybně zvolí funkci a potřebují jasně označený „nouzový východ“ k vrácení se do původního stavu, aniž by museli používat složitější postup. Podpora akcí „zpět“ a „vpřed“

Konzistence a standardy:
Uživatelé by se neměli divit, že rozdílná slova, situace, nebo akce znamenají stejné věci. Následování platformních konvencí.

Prevence chyb:
Ještě lepší, než dobře zpracované chybové zprávy, je opatrný design, který předchází problémům vyskytujícím se v první řadě. Buď eliminujte stavy náchylné k chybám, nebo je zajistěte předložením potvrzujících dialogů uživatelům.

Rozpoznání namísto zapamatování:
Minimalizujte zatížení uživatelské paměti vytvořením viditelných objektů, akcí a možností. Uživatel by si neměl pamatovat informace z jedné části dialogu do druhé. Návod k použití by měl byt viditelný nebo snadno přístupný kdykoliv je to vhodné.

Flexibilita a efektivnost užití:
Akcelerátory, neviditelné pro nové uživatele, mohou často urychlit interakci pro zkušené uživatele. Tím pádem může systém uspokojit jak nezkušené, tak zkušené uživatele. Je třeba dovolit uživatelům přizpůsobit si podle sebe často prováděné akce.

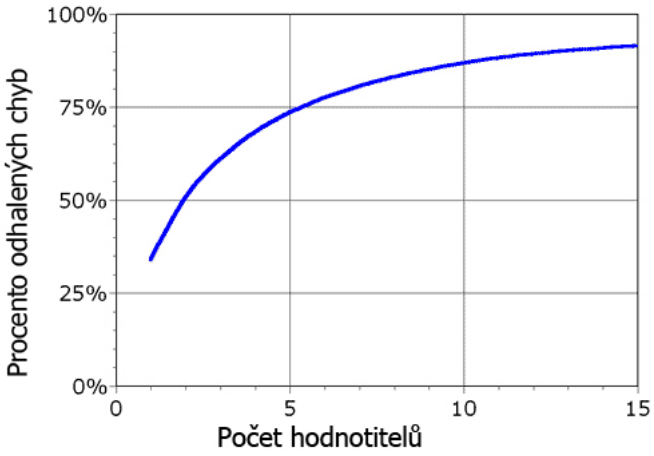
Estetický a minimalistický design:
Dialogy by neměly obsahovat informace, které jsou irelevantní, nebo potřebné jen vzácně. Každá informace navíc konkuruje relevantní informaci a snižuje její relativní význam.

Pomoc uživatelům poznat, diagnostikovat a zotavit se z chyb:
Chybové zprávy by měly být vyjadřovány v normálním jazyce(ne v kódech), měly by správně indikovat problém a konstruktivně navrhovat řešení.

Nápověda a dokumentace:
Přestože je lepší, když je systém použitelný bez dokumentace, nápověda a dokumentace by měla být vždy poskytnuta. Veškeré tyto informace by měly být snadno dohledatelné, zaměřené na úkol, který uživatel provádí a měly by sepisovat kroky, které by měly být provedeny. Tyto informace neměly by být příliš obsáhlé.

Optimální počet účastníků

Dle Jakoba Nielsena se optimální počet odborníků, kteří jsou zapojeni do heuristického hodnocení, pohybuje od tří do pěti osob. Počet osob by však měl být stanoven na základě analýzy nákladů, které chceme do této testovací metody vložit. Dle grafu na obrázku 4.1.1 je zřejmé, že se zvyšujícím se počtem hodnotitelů, roste procento odhalených chyb. Ovšem rostou také náklady na testování, protože je nutno zajistit další a další odborníky.



Obrázek 4.1: Křivka zobrazuje procento odhalených chyb závislých na počtu hodnotitelů

Nielsen a Landauer v roce 1993 prezentovali vztah, kterým lze jistit počet odhalených chyb.

$$\text{ProblemsFound}(i) = N(1 - (1 - l)^i) \tag{4.1}$$

ProblemsFound(i) představuje počet různých odhalených chyb nalezených i nezávislými hodnotiteli. N představuje celkový počet problému použitelnosti v rozhraní a l představuje procento odhalených chyb jedním hodnotitelem. [15] [23]

4.1.2 Cognitive walktrough (Kognitivní průchod)

Metoda Cognitive walkthrough se zaměřuje na to, jak snadné je pro nové uživatele splnit úkoly v systému. Rozhraní je obvykle znázorněno v podobě papírového náčrtu, nebo funkčního prototypu, ale také to může být i úplně hotová aplikace. Metoda je úkolově orientovaná, zatímco heuristické vyhodnocení hodnotí návrh jako celek a tak zajišťuje odchycení problémů, které tato a další podobné metody nedokáží odhalit. Metoda se opírá o myšlenku, že se uživatelé se systémem raději učí plněním různých úkolů, než čtením manuálů a dokumentací.

Metoda začíná analýzou plněného úkolu, ze které lze specifikovat posloupnost kroků, nebo akcí, které uživatel bude potřebovat k úspěšnému splnění daného úkolu, a dále specifikováním reakcí systému na jednotlivé kroky, či akce. V další fázi vývojáři postupně jako skupina prochází jednotlivé kroky a u každého kroku si pokládají řadu otázek. Při odpovídání na otázky jsou odhalovány případné chyby, které jsou sepisovány. Po průchodu všech kroků je sestaven seznam potencionálních chyb a problémů. V poslední fázi již jen zbývá provést opravy v programu.

Účinnost metod jako je Cognitive walkthrough se těžko měří. Většinou měření účinnosti této metody zahrnuje porovnání počtu odhalených problémů použitelnosti touto metodou a počtu odhalených problémů metodami odlišnými.

V následující podkapitole jsou popsány časté otázky kladené při průchodu jednotlivými kroky.

Průchod jednotlivými kroky

Při průchodu jsou kladeny tyto čtyři otázky:

- Pokusí se uživatelé dosáhnout efektu, který tomuto kroku náleží? – Ví uživatel, že ke splnění jeho úkolu musí použít právě tento krok?
- Je uživatel upozorněn, že je dostupná správná akce? – Vidí uživatel, že je akce dostupná a chápe její význam správně?
- Spojí si uživatel správnou akci s efektem, kterého chce dosáhnout? – Uživatelé často používají strategii splnění úkolu podle popisků. Tato strategie vede ke zvolení akce, pokud se její popisek shoduje s popisem plněného úkolu.
- Pokud byla provedena správná akce, pozná uživatel, že se přiblížil ke splnění zadaného úkolu? – Slouží ke zjištění zpětné vazby systému po tom, co uživatel provede akci.

[28]

4.1.3 Formal usability inspection (Formální kontrola použitelnosti)

Tato metoda spočívá v kombinaci heuristického ohodnocení a zjednodušené formy metody Cognitive walkthrough. Dle názvu metody by se mohlo zdát, že se k hodnocení využívá nějakého formalismu, není tomu ale tak. Formální kontrola použitelnosti probíhá v posloupnosti 6-ti kroků s přesně definovanými rolemi jednotlivých prvků obou komibnovaných metod:

1. Sestavení týmu: Do týmu je třeba zařadit osoby, které se podílejí na tvoření dobře použitelných návrhů. Obvykle to jsou lidé z oblasti návrhu, záruky kvality, dokumentace a technické podpory. Každý z těchto zvolených lidí přinese do hodnocení různý pohled na vytvořený návrh, což zvyšuje pravděpodobnost objevení chyb.

2. Přiřazení rolí členům týmu: Každá osoba v týmu má mimo kontrolování návrhu přiřazenu roli, kterou hraje v průběhu hodnocení. Role jsou následující:

   (a) Moderátor: Jak již název napovídá, moderátor zahajuje celé setkání členů týmu. Získává potřebné materiály, organizuje průběh setkání a koordinuje zaznamenávání nalezených chyb.

   (b) Vlastník: Osoba, která vytvořila návrh rozhraní a která způsobila případné chyby. Po objevení chyb tyto chyby opravuje.

   (c) Zapisovatel: Zaznamenává nalezené chyby v průběhu celého setkání.

   (d) Inspektor: Kdokoliv jiný. Prochází kontrolovaný návrh a oznamuje ostatním, jaké chyby nalezl. Jelikož se kontroly účastní i ostatní role, tak jsou také považovány za inspektory.

3. Zprostředkování dokumentů: Tento krok zahrnuje zajištění a zprostředkování popisu produktu, jeho návrhu, uživatelských skupin, uživatelských úkolů, použitých heuristik atp.

4. Prozkoumání návrhu: V této fázi inspektoři prochází testovaný návrh a na připravený formulář zaznamenávají všechny nalezené nedostatky. Pokud je předem stanovena forma, jakou budou inspektoři zapisovat nalezené problémy, výrazně se urychlí proces při následné diskusi nad nalezenými problémy. Před začátkme testování si všichni inspektoři musí uvědomit, jaké heuristiky měly být použity a v průběhu testování je musí mít na paměti. Procházení obyvkle probíhá metodou Kognitivního průchodu — procházení daného úkolu krok za krokem dle daného scénáře. Při zaznamenávání chyby je důležité poznamenat, při jakém úkolu chyba vznikla a na jakém místě. V případě formální kontroly programu se místo vzniku chyby udává číslem řádku, ale v uživatelském rozhraní se ve většině případů číslo řádku nevyskytuje, proto se místo nálezu chyby udává například pomocí obrázku.

5. Držení se formálního setkání : V průběhu schůzky moderátor spolu s celou skupinou prochází postupně jednotlivé úkoly. Inspektoři při průchodu jednotlivými kroky diskutují, které chyby objevili. Obyvkle jsou v průběhu této diskuse zjištěny nové chyby.

6. Stanovení priorit a opravení chyb: Všechny chyby, které byly v průběhu testování objeveny, jsou předány osobě, která je odpovědná za jejich nápravu. Toto předání obvykle zařizuje moderátor, který také, v případě potřeby, zorganizuje setkání nezbytná pro vyřešení problémů.

[23]

4.1.4 Pluralistic walkthrough (Pluralitní průchod)

Metoda Pluralistic walkthrough je metoda obdobná metodě Cognitive walkthrough. Je to metoda založená na větší skupině spolupracujících uživatelů. Tým se skládá ze tří skupin uživatelů – reprezentativních uživatelů, vývojářů a odborníků na použitelnost rozhraní. Uživatelé by měli reprezentovat cílovou skupinu a jsou považování za primární účastníky hodnocení použitelnosti. Vývojáři odpovídají na otázky ohledně designu a navrhují řešení problémů, se kterými se uživatelé v průběhu testování setkají. Odborníci slouží jako moderátoři a poskytují zpětnou vazbu na design a také doporučují jeho vylepšení. Role moderátorů plní odborníci také tím, že vedou uživatele úkoly a usnadňují spolupráci mezi vývojáři a uživateli. Tato role je dobrá k vyloučení defenzivního přístupu vývojářů ke kritice jejich produktu. [23]

4.1.5 Feature inspection(Funkční kontrola)

Metoda Feature inspection se zaměřeuje na úkoly, které by uživatelé s aplikací mohli provádět a na všechny funkce aplikace, které k plnění těchto úkolů budou využívat. Každá funkce aplikace je analyzována a vyhodnocena pro její srozumitelnost, dostupnost, užitečnost a použitelnost. Tato metoda zdůrazňuje význam funkcí, které jsou důležité pro zdokonalení použitelnosti aplikace. [23]

4.1.6 Consistency inspection(Kontrola konzistence)

Consistency inspection je kvalitativní metoda hodnocení a zdokonalování UI. Rozhraní je metodicky přezkoumáváno za účelem zkontrolování jednotnosti grafického návrhu a to jak v rámci jednotlivých oken aplikace, tak i mezi souvisejícími okny. Kontrola je prováděna v oblasti grafiky (barva, typografie, rozložení prvků, ikony), v oblasti textu (tón, styl, pravopis) a v oblasti interakce (konzistence kroků prováděných úkolů, konzistence názvů příkazů). Kontrola konzistence by se samozřejmě neměla přehánět, protože věci, které je třeba od sebe odlišit, by měly být zřetelně odlišené. [2] [23]

4.1.7 Standards inspection(Kontrola standardů)

Tato metoda hodnocení uživatelského rozhraní spočívá v kontrole, zda navržené UI odpovídá všem standardům, které toto UI má splňovat. Kontrolovány mohou být například dodržené standardy dané společnosti, ve které je UI vytvářeno, standardy, které vyžaduje zákazník, a dále různé ISO standardy, jako je např. ISO 9241, které pokrývá oblast HCI. Dále mohou být kontrolovány standardy v oblasti reakce systému, doby odevzy, dostupnosti odlišným uživatelským skupinám apod. K využití hodnocení touto metodou je zapotřebí odborníka, který je dobře s testovanými standardy obeznámen. V průběhu testování a hodnocení zmíněnou metodou obvykle odborník sepisuje, zda aplikace standardy splňuje, nebo zda se vyskytují nějaké problémy. Ve výsledku je tedy znám seznam nedodržených standardů, které je pak nutné opravit. [28]

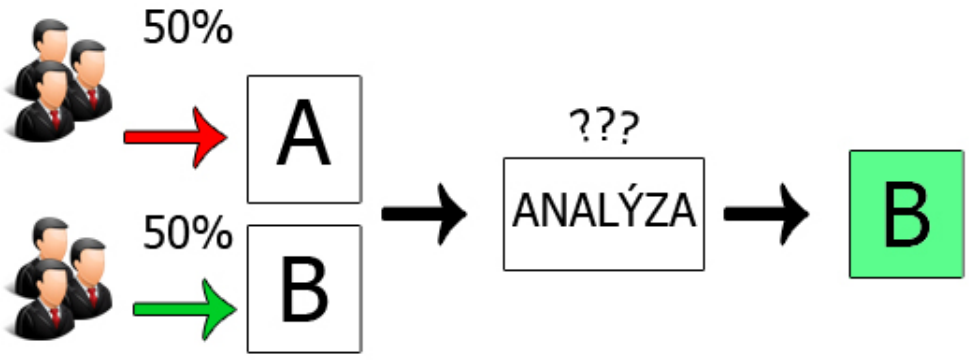
4.2 Empirické metody

Empirické metody jsou založeny na testování uživatelského rozhraní reálnými uživateli aplikace. Empirické testování přináší výhodu oproti inspekčnímu testování, protože reální uži-

vatelé uvažují jinak, jak experti. Proto je možné odhalit velké množství chyb, které by inspekční metody odhalit nedokázaly.

4.2.1 A/B testing (A/B testování)

Metoda testování A/B je založena na uživatelském testování, používá se většinou u webových uživatelských rozhraní. Jedné skupině uživatelů těchto rozhraní je zobrazována jedna varianta webových stránek (A) a druhá varianta (B) je zobrazena druhé polovině. Důležitým aspektem této metody je zachování konzistence a diskrétnosti zobrazovaných verzí. Nástroje pro toto testování tedy musí zajistit, aby uživatel viděl pořád tu stejnou verzi stránek, jinak by testování nemělo význam. Většinou je k dosažení tohoto požadavku využíváno cookies[1]. Po nasazení nástroje je nutné čekat, dokud se nenasbírá dostatečné množství informací pro následné vyhodnocení. Vzorek dat musí být dostatečně obsáhlý a pestrý. Po vyhodnocení nashromážděných dat je zjištěno, o kolik procent je která varianta lepší. Nyní už jen zbývá lepší variantu nasadit pro všechny uživatele a zvažovat provedení A/B testu v nějaké další oblasti rozhraní. Na následujícím obrázku je A/B testování názorně vyobrazeno. [25] [10]



Obrázek 4.2: A/B testování s vyhodnocení B jako lepší varianty [2]

4.2.2 Thinking aloud(Myšlení nahlas)

Metoda myšlení nahlas spočívá v tom, že uživatel pracuje s daným uživatelskýcm rozhraním (prototypem, papírovým modelem, nebo dokumentací) a při tom myslí nahlas – spontáně (nebo na výzvu) vyslovuje myšlenky, fakty, očekávání, pochybnosti, které mu při práci s rozhraním přidapnou na mysl. Metoda je levná, robustní, flexibilní a jednoduchá k naučení. Metoda má kořeny v kognitivních metodách, kde se vyskytuje jistá forma diskuze. Myšlení nahlas bylo zavedeno do oblasti uživatelských rozhraní p. Lewisem a jeho kolegy v roce 1982 a od té doby je úspěšně aplikováno k hodnocení a testování uživatelských rozhraní. Výhody metody:

- **Levné** - není třeba žádné speciální vybavení, stačí poznámkový blok a tužka k zapisování poznatků z toho co si uživatel myslí

[1] Cookies - Malé množství dat, které je uloženo na straně uživatele.
[2] Ikony postav v tomto i dalších obrázcích převzaty z http://www.icons-land.com

- **Robustní** - jelikož v metodě testují běžní uživatelé, tak se jejich myšlenky neopírají o žádné metodiky, ale jednají dle svého vlastního uvážení, což může odhalit širokou řadu problémů.
- **Flexibilní** - metodu je možné využít v jakýchkoliv fázích návrhu. Protože jediným nástrojem využitým k testování jsou samotní uživatelé, je možné tuto metodu využít u různorodcých uživatelských rozhraní – webové stránky, intranet, podnikový software, mobilní aplikace aj.
- **Jednoduché k naučení** - k testování základních věcí není vůbec těžké se naučit tuto metodu používat

Obrázek 4.3: Myšlení nahlas – uživatel řeší problém a říká, co si myslí. Vývojář si píše poznámky

[12] [16]

4.2.3 Constructive interaction (Konstruktivní spolupráce)

Tato metoda je variantou metody Myšlení nahlas, rozdíl je ovšem v tom, že při testování je rozhraní testováno nikoliv jedním uživatelem, ale dvěma. Hlavní výhoda metody spočívá v tom, že je více přirozenější, než metoda s jedním uživatelem. Lidé jsou zvyklí se při společném řešení daného problému na tomto řešení domlouvat. Z tohoto je zřejmé, že při testování rozhraní ve dvojicích bude vysloveno daleko více komentářů nesoucích důležité informace, než při testování jedním uživatelem. Nevýhodou metody pak ale může být, že různí lidé mohou mít různé návyky a postupy, jak daný úkol řešit, a nemusí se podařit dosáhnout jejich vzájemné spolupráce při testování.

Metoda konstruktivní spolupráce je zvláště vhodná pro testování uživatelských rozhraní pro děti. Přimět je, aby postupovaly podle pokynů v metodě Myšlení nahlas, může být někdy složité, proto je vhodnější zvolit metodu dvojic. Využití metody Konstruktivní spolupráce je vhodné pro projekty, u kterých je snadné získání velkého množství uživatelů. Je třeba totiž využít dvakrát tolik uživatelů, než při jedno-uživatelské metodě.

Obrázek 4.4: Konstruktivní interacke – dva uživatelé spolu řeší problém a komunikují. Vývojář si píše poznámky

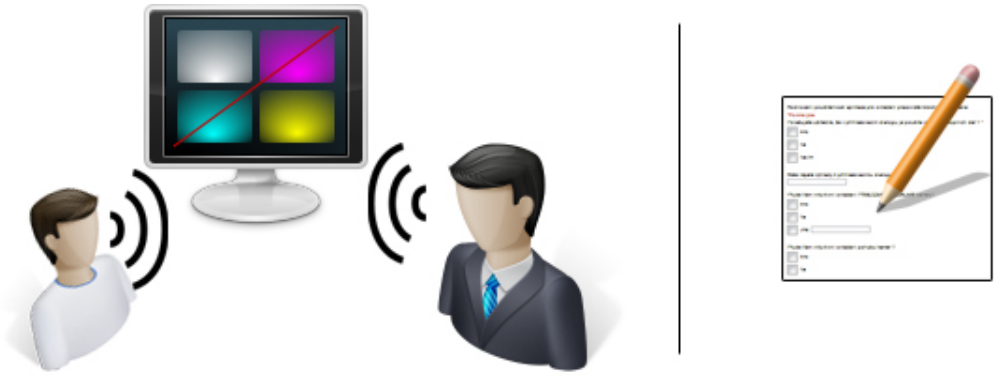
[21]

4.2.4 Dotazovací technika (Query techniques)

Dotazovací metoda poskytuje přímé odpovědi na otázky, které jsou z oblasti použitelnosti uživateli položeny. Výsledky této metody jsou vždy subjektivní, proto je potřeba shromáždit větší množství informací a ty pak následně vyhodnotit. Výhodou je, že subjektivní pocity testovaných uživatelů mohou odhalit nedostatky, které by objektivní metody odhalit nedokázaly. Dotazování probíhá formou přímých rozhovorů při testování, nebo vyplněním připraveného dotazníku.

Rozhovor poskytuje přímý a strukturovaný přístup k informacím, měl by být připraven dopředu. Rozhovor je možné v průběhu přizpůsobovat aktuální situaci.

Dotazníky jsou méně flexibilní, protože jsou vytvořeny předem. Mohou být ale rozšířeny pro větší skupinu uživatelů, protože jsou časově nenáročné. Dotazník může obsahovat otázky s jednou možnou odpovědí, více možnými odpověďmi, nebo také otázky, na které je možné odpovědět textem. Po vytvoření dotazníku, by ale měl být dotazník ozkoušen, zda je navržen správně.



Obrázek 4.5: Dotazovací technika – Vlevo je vyobrazen rozhovor, vpravo vyplňování dotazníku

[11] [21]

Kapitola 5

Analýza a návrh aplikace

Tato kapitola se zabývá analýzou a návrhem aplikace. V podkapitolách jsou popsány požadavky, které byly stanoveny při zadávání tématu této práce a jimiž se řídí další návrh celého uživatelského rozhraní. Jelikož bylo formou konzultací s vedoucím práce a formou diskusí nad vzhledem a chováním aplikace průběžně rozhraní aplikace testováno a analyzováno, byly tyto informace využity jako zpětná vazba sloužící k průběžnému zdokonalování již ve fázích vývoje.

5.1 Analýza požadavků na aplikaci

Podkapitola se zabývá analýzou a rozborem požadavků, které by aplikace v konečné fázi měla splňovat. Prvním krokem bylo seznámení se s prostředím pracoviště robotického ramene na FIT VUT v Brně. Robotické rameno je umístěno v laboratoři A223.1, která se nachází vedle kanceláře A223. Na pracovišti je nainstalována serverová stanice, která slouží pro ovládání robotického ramene. Dále jsou v laboratoři IP kamery, které jsou v místnosti rozmístěny a umožňují vzdálené sledování pracoviště.

Aplikace musí být navržena tak, aby ji uživatel bez znalosti matematické teorie v oblasti kinematiky robotů byl schopen ovládat samostatně.

Jedním ze základních požadavků na aplikaci je, že tato aplikace bude ovládat robotické rameno vzdáleně pomocí internetové sítě. Uvnitř místnosti poběží serverová část a na stanici, ze které bude rameno ovládáno bude spuštěna klientská část. Komunikace mezi klientskou a serverovou částí musí být spolehlivá. Ke komunikaci klienta se serverem bude sloužit protokol, který je zapsán ve formátu XML, tento protokol je dále popsán v kapitole 5.2.2. Aplikace bude implemtována v programovacím jazyce jazyka C/C++ s využitím knihovny Qt, která zajišťuje multiplatformní nástavbu pro tvorbu grafických uživatelských rozhraní.

Pro monitorování chování robotického ramene na pracovišti musí aplikace v reálném čase zobrazovat obrazová data IP kamer, které jsou na pracovišti nainstalovány. Jelikož mají kamery možnosti pohybu, je nutné zajistit, aby uživatel aplikace také mohl tyto kamery ovládat.

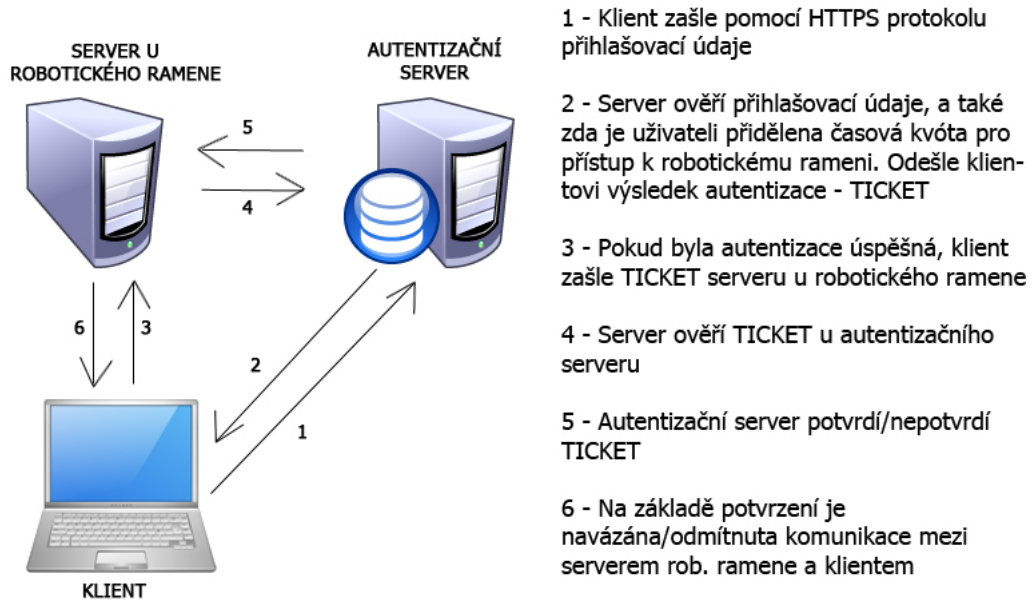
5.2 Návrh aplikace

V této podkapitole je popsán návrh celé aplikace. V první podkapitole je popsán návrh autentizace klienta vůči serveru, následuje popis protokolu, pomocí kterého bude klientská

část aplikace komunikovat se serverovou části pracoviště, dále je popsán návrh přihlašovacího okna a po té návrh hlavního okna celé aplikace. Jelikož byl stav aplikace v době jejího vytváření průběžně testován, tak se toto testování také promítlo do samotného návrhu a pomocí zpětné vazby byl návrh aplikace průběžně zdokonalován.

5.2.1 Autentizace uživatelů

Jelikož je potřeba k pracovišti robotického ramena zajistit výlučný přístup, je třeba zahrnout autentizační metody, na jejichž základě bude ověřován přístup k pracovišti ramene přes klientskou část aplikace. Na obrázku 5.2.1 je vyobrazen postup autentizace a následné komunikace klienta a serveru. Z důvodu, že bude paralelně s navrhnovanou aplikací existovat informační systém, který bude zajišťovat rezervace času pracoviště rob. ramene, bylo navrženo, že autentizace uživatelů bude probíhat vůči autentizačnímu serveru, na kterém bude spuštěn již zmiňovaný informační systém.



Obrázek 5.1: Znázornění autentizace uživatele a potvrzení komunikace serveru u robotického ramene

5.2.2 Komunikace mezi klientem a serverem na pracovišti

V této podkapitole je popsán protokol, kterým komunikuje navrhovaná klientská aplikace se serverovou částí. Data protokolu jsou ve formátu XML. Pokud je uživatel úspěšně autentizován, server zasílá zprávu pravidelně po cca 100 ms. Komunikace probíhá pomocí TCP socketů, které již mají implementovány mechanismy nezbytné pro spolehlivou komunikaci v rámci počítačové sítě. V bloku 5.1 je popsán protokol komunikace ze strany serveru ke klientovi a v bloku 5.2 je popsán protokol od klienta k serveru.

```
<message>
  <session>
    <user>uživatelské jméno</user>
    <message_id>ID zprávy</message_id>
    <timestamp> <!-- timestamp odpovídá času odeslání zprávy s˜nanosekundovou
    přesností -->
      <sec> sekundy </secs>
      <nsec> nanosekundy </nsecs>
    </timestamp>
  </session>
  <status>
    <joints>
      <joint name="J1">
        <angle>value</angle>
        <moving> YES/NO </moving> <!-- pokud se kloub pohybuje nebo ne -->
        <speed> hodnota </speed> <!-- rychlost pohybu kloubu (muze se stat,
        ze kloub provadejici nejaky pohyb ma momentalni rychlost 0) -->
        <max>maximální hodnota úhlu</max>
        <min>minimální hodnota úhlu</min>
      </joint>
      ...
    </joints>
    <gripper type="two-finger">
      <fingers_count>hodnota</fingerscount>
      <fingers>
        <finger name="name">
          <finger_pos> hodnota </finger_pos> <!-- poloha prstu v˜mm -
          u˜dvouprstého chapadla staší poloha jednoho prstu vůči druhému -->
          <finger_max>maximální hodnota</finger_max>
          <finger_min>minimální hodnota</finger_min>
        </finger>
      </fingers>
    </gripper>
    <errors>
      <error component="gipper/arm/camera/sensor/other" ID="XXX"> error kód
      </error> <!--komponenta může být chapadlo nebo rameno, kamera,nebo
      nějaký senzor, ID je identifikátor komponenty - použitelné pro kamery
      a˜senzory - zatím nepoužité(ID=0 pro chapadlo a˜rameno), jiná chyba je
      v˜současné době pouze chybou při interpretaci skriptu-->
    </errors>
  </status>
  <sensors> <!-- informace ze senzorů -->
  </sensors>
  <video> <!-- informace o˜video-streamu, od kamera atd. -->
  </video>
</message>
```

Kód 5.1: "Komunikace server ⇒ klient"

```
<message>
  <session>
    <user> uživatelské jméno </user>
    <pwd> hash hesla </pwd>
    <message_id>ID zprávy</message_id> <!-- ID se s~každou zprávou o~1
    zvyušeje -->
    <timestamp> <!-- timestamp odpovídá času odeslání zprávy s~nanosekundovou
    přesností -->
      <sec> sekundy </secs>
      <nsec> sekundy </nsecs>
    </timestamp>
  </session>
  <code lang='LUA'>
Skript pro robotické rameno v~jazyce předaném atributem 'lang' - nyí je
podporován pouze jazyk LUA
  </code>
</message>
```

Kód 5.2: "Komunikace klient ⇒ server"

Funkce pro ovládní ramene v jazyce LUA

Pro ovládání robotického ramene slouží funkce, které jsou volány pomocí komunikačního protokolu. V době tvorby této práce jsou dostupné pouze funkce v jazyce LUA. Následuje seznam funkcí s jejich popisy.

Ovládání ramene

`ARMSetAngleAbs(AXIS,angle_val_deg)` – nastavení absolutní hodnoty úhlu kloubu, `AXIS` – číslo kloubu, `angle_val_deg` – hodnota
`ARMSetAngleRel(AXIS,angle_val_deg)` – nastavení relativní hodnoty úhlu kloubu, `AXIS` – číslo kloubu, `angle_val_deg` – hodnota
`ARMGetAngle(AXIS)` – zjištění natočení kloubu, `AXIS` – číslo kloubu
`ARMSetDestPosition(X,Y,Z,hand_normal_X,hand_normal_Y,hand_normal_Z)` – nastavení ruky do koncové pozice
`ARMSetSpeed(speed)` – nastavení rychlosti pohybu
`ARMGrip(X,Y,Z,force_N)` – uchopení objektu na souřadnicích X,Y,Z – vysokoúrovňová funkce
`ARMPut(X,Y,Z)` – polož předmět na danou pozici – vysokoúrovňová funkce

Ovládání chapadla

`HANDGrip(force_N = 10)` – zavření chapadla
`HANDDrop()` – otevření chapadla
`HANDSetFingerPosRel(value_mm)` – nastavení relativní pozice změny prstu – kladná hodnota prsty otevírá – síla je daná napevno 10N
`HANDSetFingerPosAbs(value_mm)` – nastavení absolutní pozice změny prstu – kladná hodnota prsty otevírá – síla je daná napevno 10N

Ovládání kamer

`CAMERASetAngle(CAMERA,angleX,angleY,angleZ)` – nastavení pohledu kamery
`CAMERASetView(CAMERA,VIEW_NUMBER)` – nastaví do pohledu `VIEW_NUMBER` obraz z kamery `CAMERA`

Ostatní

`OTHERRandom()` – funkce vrátí náhodné celé číslo

Ukázky zpráv

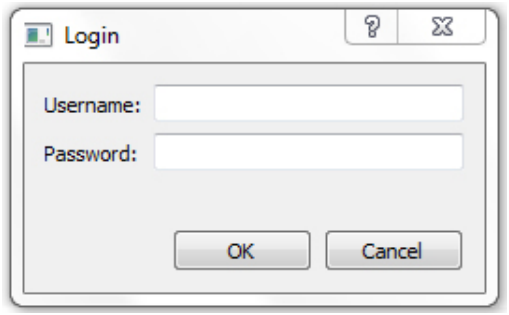
V následujícím XML zápisu je uvedena ukázka zprávy k serveru robotického ramene.

```
1  <message>
2        <session>
3              <user>user1</user>
4              <pwd>pass</pwd>
5              <message_id>1</message_id>
6              <timestamp><sec>6987012</sec><nsec>5000</nsec></timestamp>
7        </session>
8        <code lang='LUA'>
9          HANDDrop();
10         HANDGrip(10);
11         </code>
12 </message>
```

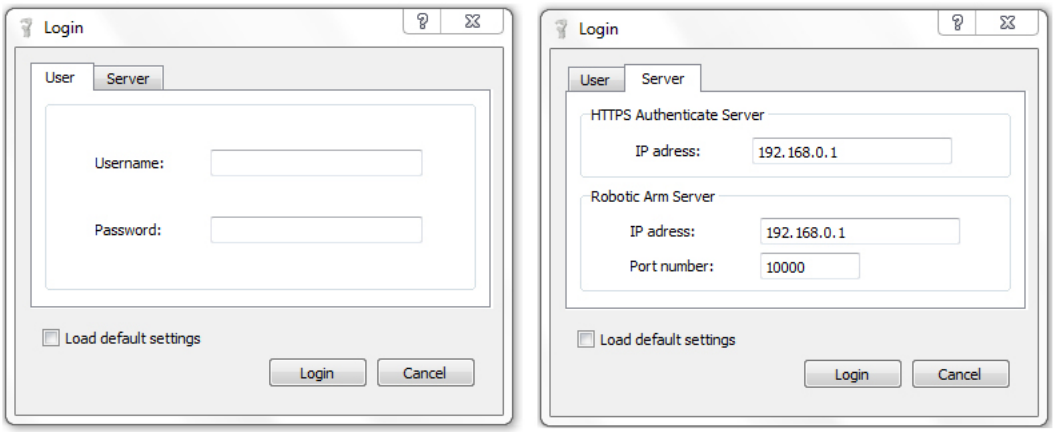
Kód 5.3: "Ukázka zprávy k serveru - otevření a zavření ruky ramene"

5.2.3 Přihlašovací dialog

Po spuštění aplikace bude zobrazeno uživateli přihlašovací dialogové okno. V tomto okně bude muset uživatel vyplnit přihlašovací údaje, adresu a port serveru robotického ramene a adresu autentizačního serveru. Adresy serverů budou v aplikaci nastaveny na výchozí hodnoty, které se budou po spuštění aplikace automaticky vyplňovat do přihlašovacího formuláře. Pro případ, že by se v budoucnu chtěl uživatel přihlašovat k jinému serveru, tak je možnost tyto adresy v konfiguraci aplikace změnit. Pole která mohou nabývat pouze specifických hodnot jako je např. IP adresa povolují zápis pouze validních hodnot. Pro přihlášení bude sloužit tlačítko, které také bude reagovat na stisk klávesy ENTER . Na obrázku 5.2.3 je znázorněn první návrh přihlašovacího okna. Na obrázku 5.2.3 je znázorněno okno po několika iteracích, kdy bylo třeba doplnit výše zmíněné údaje, které v prvním návrhu chyběly. Také přibylo zaškrtávací tlačíko pro spuštění hlavní aplikace ve výchozím nastavení velikosti a stavu okna, z důvodu, kdyby uživatel například pracoval s více monitory a pak chtěl aplikaci spustit jen na jednom monitoru.



Obrázek 5.2: První návrh přihlašovacího dialogu

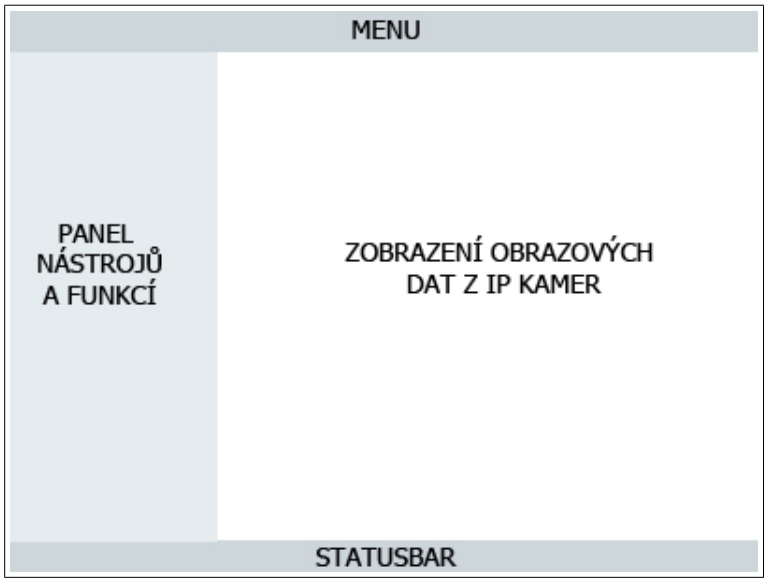


Obrázek 5.3: Přihlašovací dialog ve finální verzi

5.2.4 Hlavní aplikace

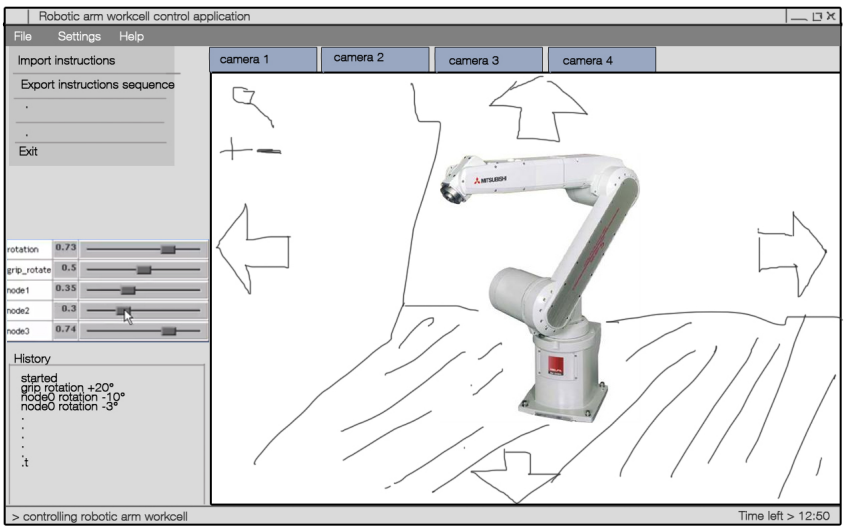
Celá aplikace bude obsahovat dynamicky se měnící prvky, u kterých to bude vhodné, aby bylo zajištěno bezproblémové zobrazení na monitorech s různým rozlišením. Největší prostor zabírá pohled kamery, který zajišťuje vzdálené zobrazení oblasti pracoviště s ramenem. Hlavní okno bude vytvořeno standardním způsobem, který v sobě zahrnuje stavový řádek, umožňuje tvorbu menu, pomocí kterého budou volány jednotlivé hlavní funkce aplikace. Celá aplikace bude využívat v co největší míře klávesových zkratek, které budou sloužit ke zefektivnění rychlosti používání rutinních akcí.

Vzhled aplikace bude dle ověřených postupů – pracovní nástroje a aktivní prvky budou zobrazeny na levé straně, stavová lišta bude ve spodní části, zobrazení kamer bude v pravé části aplikace. Navržené rozložení je vyobrazeno na následujícím obrázku.



Obrázek 5.4: Návrh rozmístění částí v hlavním okně aplikace.

Na následujícím obrázku je znázorněn první návrh hlavního okna aplikace.



Obrázek 5.5: První návrh rozhraní

5.3 Zobrazení kamer a jejich ovládání

Na výše uvedeném obrázku s prvním návrhem hlavního okna jsou také vyobrazeny ovládací prvky kamer. Prvky byly původně navrženy v obraze kamery, ale toto řešení se později při testování s uživateli projevilo jako nevhodné. V dalším návrhu, který je na obrázku 5.6, je vyobrazeno nové umístění ovládacích prvků kamer.

Obrázek 5.6: Nové umístění ovládacích prvků kamer

Při návrhu ovládání a zobrazování obrazu z kamer byl vytvořen dotazník. Pomocí tohoto dotazníku bylo zkoumáno, jakými způsoby běžní uživatelé ovládají interaktivní obrazové prvky, jako jsou obrazy z kamer, nebo např. internetové mapy – s mapami stejně jako s kamerami lze pohybovat a jejich obraz přibližovat a oddalovat. Dotazník vyplnilo 40 uživatelů, kteří pokrývají podle vyplněných informací uživatelské skupiny od začátečníků, přes pokročilé až k expertům. Dotazník a jeho výsledky jsou uvedeny v příloze B C. Z dotazníku je patrné, že drtivá většina uživatelů používá pro přibližování a oddalování obrazu kolečko myši a dále tlačítka, která označují přiblížení a oddálení. Pro pohyb obrazu uživatelé dle získaných dat využívají převážně funkci „drag and drop“ a dále také tlačítka určená k posunu.

Výsledky dotazníku vedly k zakomponování nejvíce používaných metod ovládní do vytvářené aplikace.

Kapitola 6

Implementace aplikace

V této kapitole je detailněji popsána implementace celé aplikace, postupy a myšlenky, které byly při vytváření aplikace aplikovány.

6.1 Zvolení nástrojů

Celá aplikace je implementována v jazyce C/C++. Jelikož byl požadavek vytvořit aplikaci s grafickým uživatelským rozhraním, tak byl využit volně dostupný framework Qt s otevřeným zdrojovým kódem. Tento framework je distribuovaný pod licencí LGPL v2.1. Zajišťuje prostředky pro cross-platformní programování aplikací, včetně grafických nástaveb. Výsledná aplikace tedy nebude závislá na operačním systému a prostředí.

6.2 Přihlašovací okno

Po spuštění aplikace je uživateli zobrazeno přihlašovací okno, které zajišťuje přihlášení do sytému. Přihlašovací okno je založeno na třídě `QDialog`, která poskytuje všechny potřebné vlastnosti pro toto okno. V tomto přihlašovacím okně je pomocí třídy `QTabWidget` uděláno rozdělení nastavení na část pro informace o uživateli a část pro konfiguraci autentizačního serveru a serveru na pracovišti. V sekci pro nastavení konfiguračních údajů (IP adres a portu) serverů jsou všechny uživatelské vstupní prvky opatřeny kontrolou platných údajů. V průběhu návrhu bylo do přihlašovacího okna přidáno zaškrtávací tlačítko pro zajištění načtení okna aplikace ve výchozí poloze z důvodu, kdyby uživatel náhodou umístil okno aplikace tak, že by při příštím spuštění bylo nedostupné.

6.3 Hlavní okno

Po úspěšném přihlášení je přihlašovací okno skryto a uživatel se dostane do hlavního okna celé aplikace. Hlavní okno je vytvořeno na základě třídy `QMainWindow`, která zajišťuje standardní prvky pro hlavní okno aplikace jako je menu a statusbar.

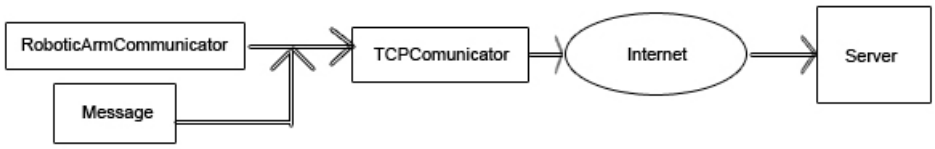
Menu slouží pro zajištění přístupnosti k základním funkcím aplikace – vypnutí aplikace, změna a nastavení IP adres a portu serverů a ke zobrazení informací o aplikaci. Dále je v menu obsažena funkce importování souboru, která zajišťuje otevření zadaného souboru a odeslání uvedeného programu na server.

Ve statusbaru jsou zobrazovány informace, které slouží k informování uživatele. Na pravé straně statusbaru je plánováno zobrazení času zbývající do vypršení relace. Pro vytvoření statusbaru byla využita již hotová třída `QStatusBar`.

Hlavní okno je rozděleno dle návrhu vyobrazeného na obrázku 5.2.4.

6.4 Komunikace klient-server

Pro komunikaci s pracovištěm robotického ramene byla vytvořena třída, které zajišťuje metody potřebné pro komunikaci klienta se serverem. Využívá třídy popisující komunikační protokol. Tyto třídy poskytují rozhraní pro získání správného zápisu komunikačního protokolu v jazyce `XML`. Třída zajišťuje vygenerování správného zápisu protokolu a předání dat další třídě, která již data posílá přes TCP k serveru, který ovládá robotické rameno. Na následujícím obrázku je vyobrazeno propojení tříd, které zajišťují komunikaci se serverem



Obrázek 6.1: Vyobrazení tříd, které slouží k zajištění komunikace se serverem robotického ramene

6.5 Kamery a jejich ovládání

Jelikož tvořená aplikace slouží ke vzdálenému ovládání pracoviště robotického ramene, tak je velice důležité poskytnou co nejlepší prostředí pro pozorování tohoto pracoviště. V době vzniku této práce se na pracovišti nachází 4 IP kamery, které jsou umístěny tak, aby pokryly co největší snímaný prostor.

V hlavním okně aplikace se blok s kamerami nachází v pravé části aplikace a zabírá největší část aplikace. Kamery jsou zastřešeny pomocí karty se záložkami, kdy každá karta poskytuje přístup k ovládání a obrazu vždy jedné kamery. Pro snadnější manipulaci se záložkami reagují tyto záložky na klávesové zkratky: `Ctrl+Tab` slouží k přepnutí na další záložku, `Ctrl+Shift+Tab` slouží k přepnutí na předchozí záložku, `Ctrl+ číslo záložky` slouží k přepnutí na konkrétní záložku. Mechanismus záložek je vytvořen tak, aby při zneaktivnění záložky se tato záložka přepla do pasivního režimu a nesnažila se získávat obraz z kamer. Tímto je dosaženo, že aplikace nebude zbytečně využívat procesorový čas, přidělenou paměť a přenosové pásmo.

Jak již bylo zmíněno, tak každá záložka obsahuje ovládání a zobrazování obrazu z jedné kamery. Pro ovládání kamer slouží tlačítka umístěná nalevo od obrazu, dále po najetí na obraz kamery je možné jej posunout pomocí metody drag and drop – tento způsob pohybu je omezený z důvodu složitých výpočtů, které by vedly k přesným informacím a souřadnicím posunu kamery. Pro přibližování obrazu z kamer slouží posuvník v levé části pod tlačítky ovládajícími pohyb. Druhou alternativou k přibližování je metoda ovládání pomocí kolečka myši. Získávání obrazu a ovládání kamer je realizováno pomocí HTTP rozhraní

kamer, které je dostupné již od výrobce. Obraz kamery se vždy přizpůsobuje velikosti okna aplikace tak, aby byl vždy vidět celý záběr kamery. Jelikož je zobrazování plynulého obrazu pouhým překreslováním obrázků velice náročné, do budoucna by bylo vhodné přepracovat zobrazování videa z kamery na metodu s RTSP streamingem.

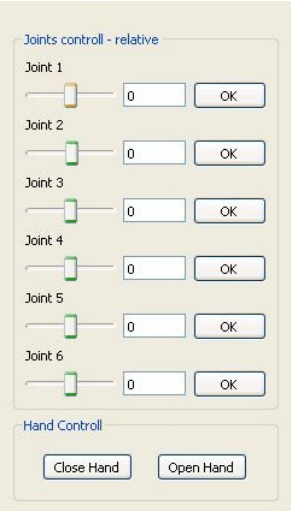
6.6 Prvky pro ovládání robotického ramene

Jelikož je celé pracoviště robotického ramene ve fázích vývoje, nejsou všechny funkce pro ovládání robotického ramene implementovány. Při tvorbě aplikace proto byly zahrnuty ovládácí prvky funkcí ramene, které jsou již implementované. Jedná se o funkce pro otevření a uzavření chapadla a dále funkce pro ovládání jednotlivých kloubů ramene. Aplikace tedy neumožňuje využívání žádných vysokoúrovňových funkcí jako je např. uchopení objektu na souřadnicích `X,Y,Z`.

Pro ovládání pohybu kloubů slouží posuvník propojený s textovým vstupem. Vstupní pole pro ruční zadání hodnoty je opatřeno validátorem, který zajišťuje zadání korektní hodnoty. Po zadání hodnoty natočení kloubu je stisknutím tlačítka OK odeslán příkaz k otočení daného kloubu. Ovládácí prvky kloubů jsou nastaveny tak, aby generovaly funkce pro relativní natočení daného kloubu.

K ovládní chapadla robotického ramene slouží dvě tlačítka. První tlačítko chapadlo otevře a druhé jej zavře.

Následující obrázek vyobrazuje zpracování prvků pro ovládání robotického ramene.



Obrázek 6.2: Ovládací prvky robotického ramene

Kapitola 7

Testování aplikace

V následujícím textu je popsán postup jakým byla celá aplikace testována. Kapitola úzce souvisí s kapitolou 5.2, protože testování a hodnocení bylo prováděno již od počátku tvorby, tak jak je doporučováno v obecných metodikách návrhu a tvorby uživatelských rozhraní.

7.1 Zvolení metod testování

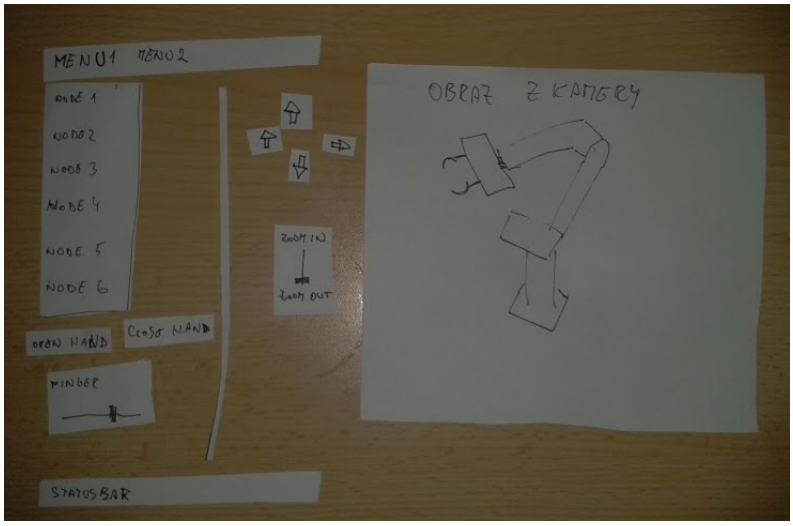
Jelikož je testování velice důležitým prvkem při tvorbě aplikací s uživatelským rozhraním, bylo nutné se rozhodnout, jaké metody budou pro testování vytvořené aplikace pro ovládání pracoviště robotického ramene vhodné.

Z inspekčních metod přidapadla v úvahu metoda **cognitive walkthrough**, která byla využita jako forma konzultace s vedoucím této práce – tato metoda přinesla úpravy změn přihlašovacího dialogu, zavedla možnost změny IP adres a postupně řídila celý průběh vývoje aplikace.

Pro empirické testování bylo domluveno 10 uživatelů, kteří se do tohoto testování v různých fázích zapojovali. Při testování byly využity metody **Myšlení nahlas**, **Konstruktivní spolupráce** a také **dotazovací metody**: **Rozhovor** a **Dotazník**. V průběhu vývoje aplikace byly v největší míře využívány metody Myšlení nahlas a Rozhovor, z nichž bylo odhaleno hned několik chyb v návrhu a tyto metody také přinesly další různá doporučení vedoucí ke zdokonalení aplikace. Pro zdokonalení ovládání obrazu získaného z IP kamery byl vytvořen dotazník, který vplnilo 40 uživatelů.

7.2 Využití metody card sorting

Po stanovení a analýze požadavků na výslednou aplikaci byla na vzorku 3 uživatlů aplikována technika Card sorting. Tato technika je popsána v kapitole 3.3. Pomocí papírových bloků, které představovaly jednotlivé bloky v aplikaci dle návrhu, byli uživatelé dotazování na poskládání těchto bloků tak, jak považují za nejvodnější rozmístění. Uživatelé se ve všech případech shodli, že prostor pro zobrazování obrazů kamer, snímajících pracoviště, by měl být zobrazen vpravo. Vlevo by měly být uspořádány ovládací prvky pro kontrolu robotického ramene. Menu celé aplikace bylo standardně umístěno v horní části celé aplikace tak, jak již poskytuje třída QWindow frameworku Qt. Stavový řádek byl také standardně umístěn do spodní části aplikace.



Obrázek 7.1: Seskládané části aplikace při užití metody card sorting

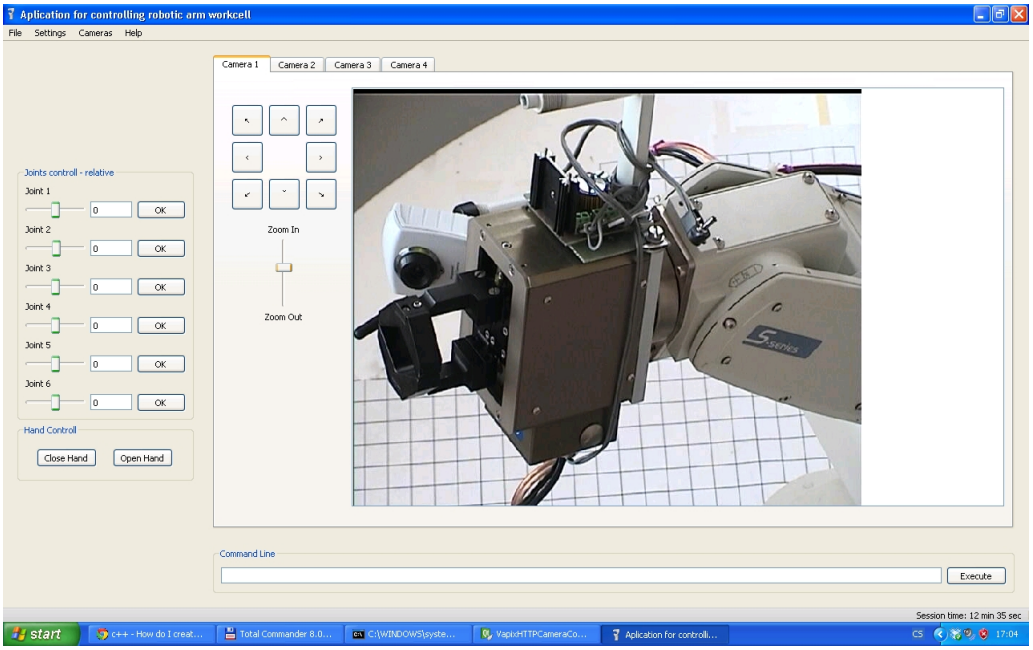
7.3 Ovládací prvky kamer

V prvním návrhu umístění prvků pro ovládání kamer bylo navrženo umístění těchto prvků přímo do obrazu s kamerou – šipky pro pohyb kamer by byly umístěny po okrajích obrazu viz. obrázek 5.2.4. Pomocí testování a následného rozhovoru s testery bylo zjištěno, že ovládání by bylo vhodné pro dotykové zařízení, jako je např. tablet. Pro desktopovou aplikaci ovšem toto řešení nebylo vhodné, protože ovládací prvky byly umístěny daleko od sebe, což pro zařízení, které je ovládáno pomocí myši bylo velice omezující. Po vyhodnocení zpětné vazby byl vytvořen nový návrh ovládacích prvků. V tomto návrhu jsou zahrnuty také nové prvky pro ovládání nejen do přímých směrů, ale jejich kombinací – např. pohyb nahoru doprava. Tyto tlačítka pro ovládání pohybu kamer byly umístěny vlevo od obrazu kamer a seskupeny do jednoho panelu. Nutnost pohybu myší pro ovládání natáčení kamer se několiknásobně zmenšil. Pod panel pro ovládání byl umístěn posuvník, který slouží k přibližování obrazu kamer.

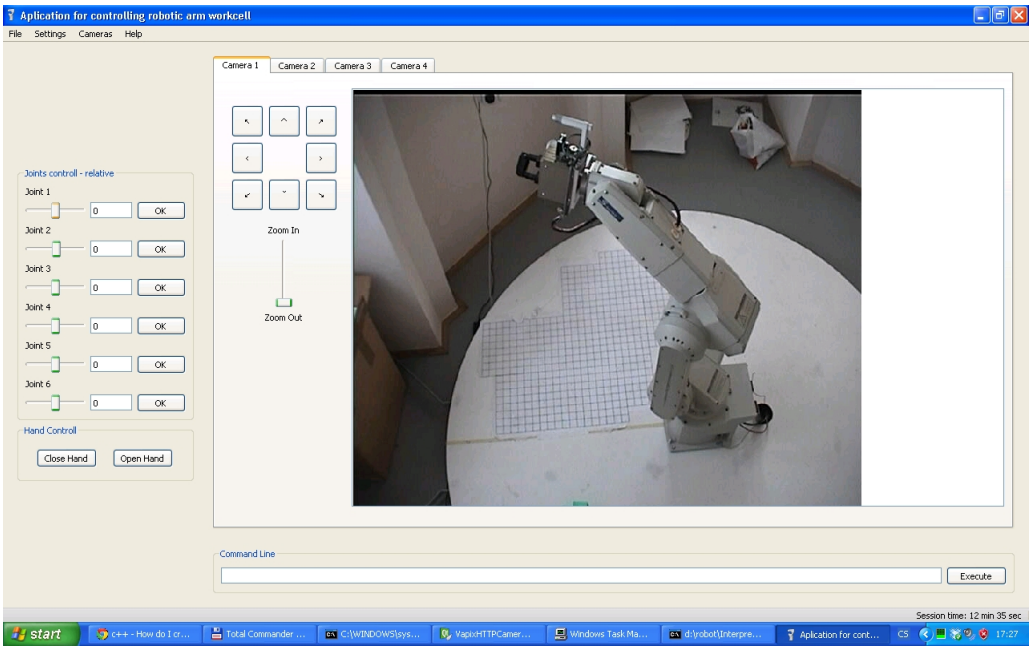
Pro získání informací ke zdokonalení ovládání záznamů z kamer byl vytvořen dotazník, na který odpovědělo 40 osob. Výsledkem dotazníku bylo zjištění, že uživatelé pro přibližování obrazových dat využívají kolečko myši a pro pohyb obrazu většinou metodu „drag and drop“.

7.4 Testování pohybů kloubů a pohybů kamer

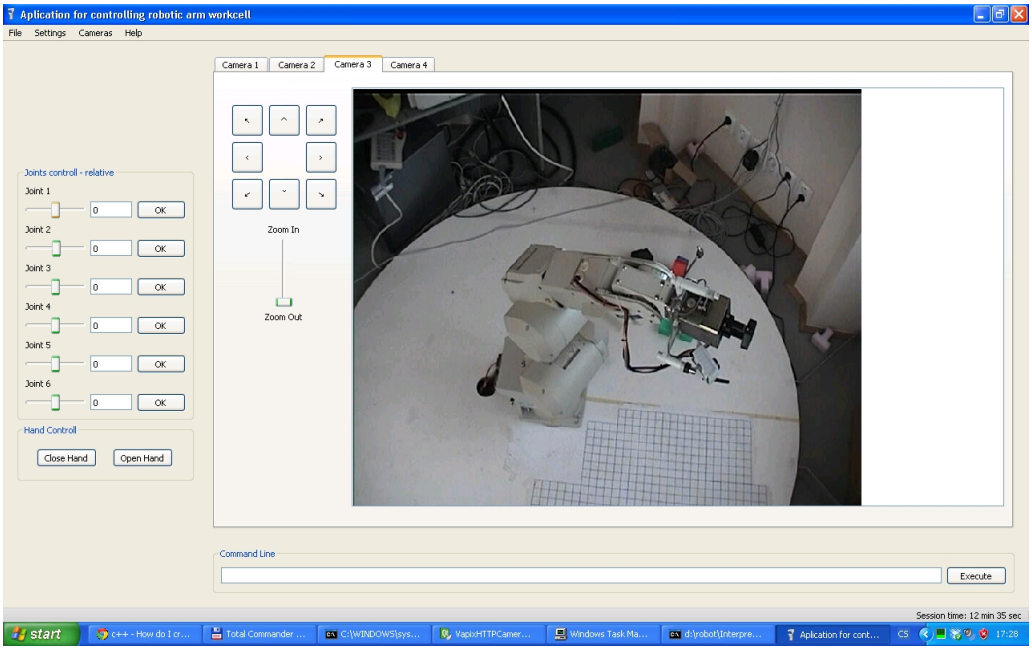
Po dokončení implementace prvků pro ovládání kloubů byly tyto prvky otestovány nejprve kontrolou generovaného `XML` protokolu, který měl zajistit příslušnou akci. Po kontrole `XML` protokolu bylo provedeno testování v reálné situaci, kdy bylo spuštěno robotické rameno a pomocí prvků aplikace bylo ovládáno. Pro zobrazování pohybů ramene byly použity a tím i otestovány prvky, které slouží k ovládání kamer. Na následujících obrázcích jsou snímky z testování ovládání ramene přes aplikaci.



Obrázek 7.2: První snímek testování



Obrázek 7.3: Druhý snímek testování



Obrázek 7.4: Třetí snímek testování

7.5 Testování hotové aplikace

Po dokončení implementace, bylo na vzorku deseti uživatelů provedeno testování zaměřené na zorientování uživatele v aplikaci, ohodnocení ovládacích prvků a zhodnocení vlivu zpoždění komunikace na komfort uživatele. Pro hodnocení byl vytvořen dotazník, který je dostupný v příloze D a jeho výsledky v příloze E. Dle výsledků se dá říct, že je uživatelské rozhraní intuitivní, byly použity vhodné ovládací prvky a vlastnosti. Uživatelé se v aplikaci orientovali bez větších problémů. Při tomto testování byl hodnocen vliv zpoždění komunikace na síti, které bylo v aplikaci simulováno. Testovaní uživatelé považují aplikaci za použitelnou, pokud časová prodleva není větší jak tři až čtyři sekundy.

Kapitola 8

Závěr

V bakalářské práci byla shrnuta a popsána teorie z oblasti návhru, testování a hodnocení uživatelských rozhraní. Dále je v práci popsán návrh, implementace a testování aplikace, která vznikla při řešení této práce. Aplikace slouží ke vzdálenému ovládání pracoviště robotického ramene na Fakultě informačních technologií VUT v Brně. Tato aplikace byla vytvořena v programovacím jazyce C/C++ s využitím knihovny Qt, která poskytuje multiplatformní nástavbu pro tvorbu standardních uživatelských rozhraní.

Při tvorbě celé aplikace byly využívány metody popsané v teoretické části práce. Testování a hodnocení probíhalo ve všech fázích návrhu.Využívání těchto metod vedlo k odhalování chyb v návrhu, doplňování chybějících prvků, či reorganizování pozic prvků v aplikaci. Pro získání informací k návrhu ovládacích prvků IP kamer byl vytvořen dotazník, pomocí kterého byly zjištěny nejpoužívanější techniky a ty byly následně do aplikace zahrnuty.

Po dokončení byla celá aplikace otestována na vzorku 10 uživatelů, kteří vyplňovali dotazník týkající se hodnocení použitelnosti uživatelské rozhraní. Výsledky dotazníku ukazují, že navržené uživatelské rozhraní je intuitivní a jsou v něm použity jednoduché, ale účelné ovládací prvky a funkce. Při testování bylo také provedeno hodnocení použitelnosti aplikace vzhledem ke zpoždění komunikace na síti. Z výsledků lze říci, že uživatelé považují aplikaci za použitelnou, pokud je doba zpoždění menší jak tři až čtyři sekundy.

Jelikož je nyní vytvořená aplikace zaměřena pouze na vzdálené ovládání pracoviště, ke kterému je umožněn přístup pouze v zaregistroavané časové kvótě, bylo by vhodné integrovat do této aplikace simulátor celého pracoviště tak, aby v případě, že uživatel nemá přístup k pracovišti, mohl pracovat se simulátorem, který by mu poskytoval stejné prostředky jako samotné pracoviště.

Při vypracovávání bakalářské práce bylo třeba naučit se programovat aplikace s grafickým uživatelským rozhraním s využitím knihovny Qt. Dále proniknout do problematiky návrhu a hodnocení uživatelských rozhraní. V neposlední řadě bylo velice zajímavé seznámit se s pracovištěm robotického ramene.

Literatura

[1] Definition and Overview of Human-Computer Interaction. [Online], [cit. 2013-05-11].
URL http://old.sigchi.org/cdg/cdg2.html#2_1

[2] Glossary - consistency inspection. [Online], [cit. 2013-05-11].
URL http://www.usabilityfirst.com/glossary/consistency-inspection/

[3] Mock-ups. [Online], [rev. 2010-02-16], [cit. 2013-05-12].
URL http://www.interaction-design.org/encyclopedia/mock-ups.html

[4] User-Centered Design. [Online], [cit. 2013-05-11].
URL https://www-01.ibm.com/software/ucd/ucd.html

[5] User Interface. [Online],[rev. 2009-03-31], [cit. 2013-05-11].
URL http://www.techterms.com/definition/user_interface

[6] What is User-Centered Design? [Online], [cit. 2013-05-12].
URL http://www.usabilityprofessionals.org/usability_resources/about_usability/what_is_ucd.html

[7] Wireframe. [Online], [cit. 2013-05-13].
URL http://www.usabilitybok.org/node/26

[8] *Všeobecná encyklopedie ve čtyřech svazcích. 3.díl.* Praha: Nakladatelský dům OP Diderot, 1997, ISBN 80-85841-35-5, 740 s.

[9] ALBEN, L.: Quality of experience: defining the criteria for effective interaction design. *interactions*, ročník 3, č. 3, Květen 1996: s. 11–15, ISSN 1072-5520, doi:10.1145/235008.235010.
URL http://doi.acm.org/10.1145/235008.235010

[10] ANDREWS, K.: Human-Computer Interaction. [Online], [rev. 2011-04-05], [cit. 2013-05-12].
URL http://www.scribd.com/doc/101724661/hci

[11] DIX, A.; FINLAY, J.; ABOWD, G. D.; aj.: *Human-computer interaction*. Pearson Education, 2004, ISBN 0130461091, 880 s.

[12] JORGENSEN, A. H.: Thinking-aloud in user interface design: a method promoting cognitive ergonomics. *Ergonomics*, ročník 33, č. 4, 1990: s. 501–507, [Online], [cit. 2013-05-08].
URL http://www.tandfonline.com/doi/abs/10.1080/00140139008927157

[13] NIELSEN, J.: [Online],[rev. 1993-11-01], [cit. 2013-05-13].
URL http://www.nngroup.com/articles/iterative-design/

[14] NIELSEN, J.: 10 Usability Heuristics for User Interface Design. [Online], [cit. 2013-05-12].
URL http://www.nngroup.com/articles/ten-usability-heuristics/

[15] NIELSEN, J.: How to Conduct a Heuristic Evaluation. [Online], [cit. 2013-05-12].
URL
http://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/

[16] NIELSEN, J.: Thinking Aloud: The #1 Usability Tool. [Online], [rev. 2012-01-16], [cit. 2013-05-13].
URL
http://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/

[17] NIELSEN, J.: Usability 101: Introduction to Usability. [Online], [cit. 2013-05-11].
URL http:
//www.nngroup.com/articles/usability-101-introduction-to-usability/

[18] NIELSEN, J.: User Experience (UX) — Our Definition. [Online], [cit. 2013-05-11].
URL http://www.nngroup.com/about-user-experience-definition/

[19] Nielsen, J.: The Usability Engineering Life Cycle. *Computer*, ročník 25, č. 3, Březen 1992: s. 12–22, ISSN 0018-9162, doi:10.1109/2.121503.
URL http://dx.doi.org/10.1109/2.121503

[20] Nielsen, J.: Iterative User-Interface Design. *IEEE Computer*, ročník 26, č. 11, 1993: s. 32–41.

[21] NIELSEN, J.: *Usability Engineering*. USA: Academic Press, 1993, ISBN 1-12-518406-9, 340 s.

[22] NIELSEN, J.: Usability inspection methods. In *Conference Companion on Human Factors in Computing Systems*, CHI '94, New York, NY, USA: ACM, 1994, ISBN 0-89791-651-4, s. 413–414, doi:10.1145/259963.260531.
URL http://doi.acm.org/10.1145/259963.260531

[23] NIELSEN, J.; MACK, R. L. (editoři): *Usability inspection methods*. New York: John Wiley and Sons, 1994, ISBN 0-471-01877-5.

[24] Nielsen, J.; Molich, R.: Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, New York, NY, USA: ACM, 1990, ISBN 0-201-50932-6, s. 249–256, doi:10.1145/97243.97281.
URL http://doi.acm.org/10.1145/97243.97281

[25] SNÍŽEK, M.: A/B testování – kompletní průvodce. 12. 5. 2011, [Online], [rev. 2011-12-05], [cit. 2013-05-12].
URL http://www.optimics.cz/c/ab-testovani-kompletni-pruvodce

[26] Soegaard, M.: Prototyping. 2010, [Online], [rev. 2010-03-22], [cit. 2013-05-12].
URL http://www.interaction-design.org/encyclopedia/prototyping.html

[27] SPENCER, D.; WARFEL, T.: Card sorting: a definitive guide. [Online], [cit. 2013-05-12].
URL http://boxesandarrows.com/card-sorting-a-definitive-guide

[28] STONE, D.; aj.: *User interface design and evaluation*. The Morgan Kaufmann series in interactive technologies, Morgan Kaufmann, 2005, ISBN 0-12-088436-4, 669 s.

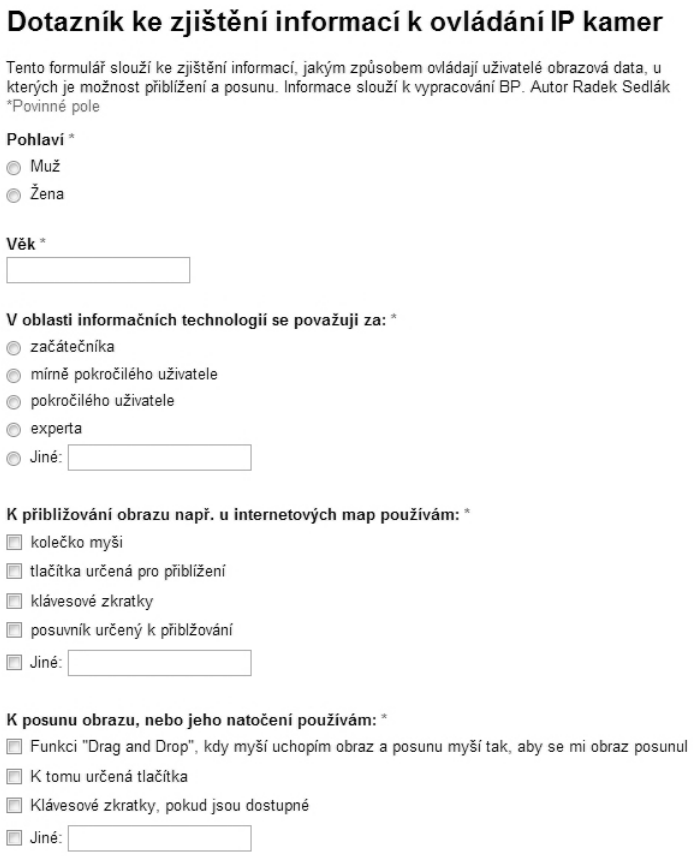
Příloha A

Obsah CD

- \doc\ – dokumentaci k aplikaci
- \dotazniky\ – dotazníky a jejich výsledky
- \src\ – zdrojové kódy aplikace
- \thesis\ – zdrojové soubory technické zprávy
- `aplikace_pro_ovladani_pracoviste_robotickeho_ramene.pdf` – technická zpráva ve formátu PDF
- Makefile – soubor pro přeložení aplikace
- README – soubor s informacemi o aplikaci

Příloha B

Dotazník k ovládání IP kamer



Dotazník ke zjištění informací k ovládání IP kamer

Tento formulář slouží ke zjištění informací, jakým způsobem ovládají uživatelé obrazová data, u kterých je možnost přiblížení a posunu. Informace slouží k vypracování BP. Autor Radek Sedlák
*Povinné pole

Pohlaví *
- Muž
- Žena

Věk *

V oblasti informačních technologií se považuji za: *
- začátečníka
- mírně pokročilého uživatele
- pokročilého uživatele
- experta
- Jiné:

K přibližování obrazu např. u internetových map používám: *
- kolečko myši
- tlačítka určená pro přiblížení
- klávesové zkratky
- posuvník určený k přibližování
- Jiné:

K posunu obrazu, nebo jeho natočení používám: *
- Funkci "Drag and Drop", kdy myší uchopím obraz a posunu myší tak, aby se mi obraz posunul
- K tomu určená tlačítka
- Klávesové zkratky, pokud jsou dostupné
- Jiné:

Příloha C

Dotazník k ovládání IP kamer – výsledky

Na následujícím grafu jsou zobrazeny získané informace z dotazníku. Zaznamenaná data jsou dostupná na přiloženém CD.

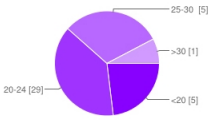
Počet odpovědí: 40

Souhrn

Pohlaví

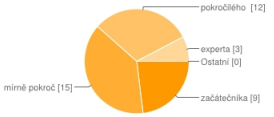
Muž	26	65 %
Žena	14	35 %

Věk



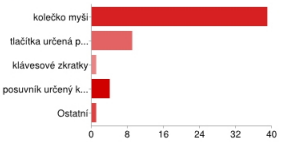
<20	5	12,5 %
20-24	29	72,5 %
25-30	5	12,5 %
>30	1	2,5 %

V oblasti informačních technologií se považuji za:



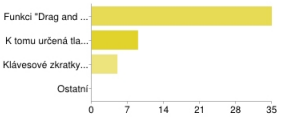
začátečníka	9	23 %
mírně pokročilého uživatele	15	38 %
pokročilého uživatele	12	31 %
experta	3	8 %
Ostatní	0	0 %

K přibližování obrazu např. u internetových map používám:



kolečko myši	39	97,5 %
tlačítka určená pro přiblížení	9	22,5 %
klávesové zkratky	1	2,5 %
posuvník určený k přibližování	4	10 %
Ostatní	1	2,5 % - dvojklik myší

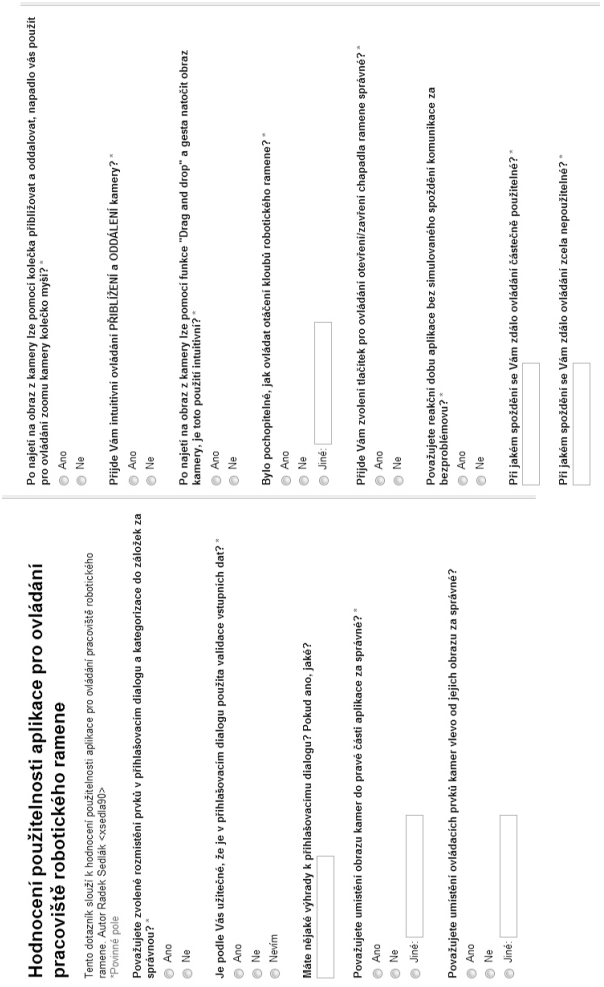
K posunu obrazu, nebo jeho natočení používám:



Funkci "Drag and Drop", kdy myší uchopím obraz a posunu myší tak, aby se mi obraz posunul	35	87,5 %
K tomu určená tlačítka	9	22,5 %
Klávesové zkratky, pokud jsou dostupné	5	12,5 %
Ostatní	0	0 %

Příloha D

Dotazník k ohodnocení výsledné aplikace



Hodnocení použitelnosti aplikace pro ovládání pracoviště robotického ramene

Tento dotazník slouží k hodnocení použitelnosti aplikace pro ovládání pracoviště robotického ramene. Autor Radek Sedlák <xsedla90>

*Povinné pole

Považujete zvolené rozmístění prvků v přihlašovacím dialogu a kategorizace do záložek za správnou? *
- Ano
- Ne

Je podle Vás užitečné, že je v přihlašovacím dialogu použita validace vstupních dat? *
- Ano
- Ne
- Nevím

Máte nějaké výhrady k přihlašovacímu dialogu? Pokud ano, jaké?

Považujete umístění obrazu kamer do pravé části aplikace za správné? *
- Ano
- Ne
- Jiné:

Považujete umístění ovládacích prvků kamer vlevo od jejich obrazu za správné?
- Ano
- Ne
- Jiné:

Po najetí na obraz z kamery lze pomocí kolečka přibližovat a oddalovat, napadlo vás použít pro ovládání zoomu kamery kolečko myši? *
- Ano
- Ne

Přijde Vám intuitivní ovládání PŘIBLÍŽENÍ a ODDÁLENÍ kamery? *
- Ano
- Ne

Po najetí na obraz z kamery lze pomocí funkce "Drag and drop" a gesta natočit obraz kamery, je toto použití intuitivní? *
- Ano
- Ne

Bylo pochopitelné, jak ovládat otáčení kloubů robotického ramene? *
- Ano
- Ne
- Jiné:

Přijde Vám zvolení tlačítek pro ovládání otevření/zavření chapadla ramene správné? *
- Ano
- Ne

Považujete reakční dobu aplikace bez simulovaného spoždění komunikace za bezproblémovou? *
- Ano
- Ne

Při jakém spoždění se Vám zdálo ovládání částečně použitelné? *

Při jakém spoždění se Vám zdálo ovládání zcela nepoužitelné? *

Příloha E

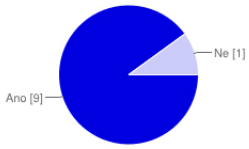
Dotazník k ohodnocení výsledné aplikace – výsledky

Na následujích grafech jsou zobrazeny získané informace z dotazníku. Zaznamenaná data jsou dostupná na přiloženém CD.

Počet odpovědí: 10

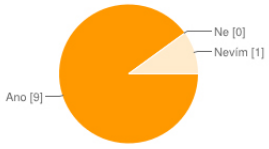
Souhrn

Považujete zvolené rozmístění prvků v přihlašovacím dialogu a kategorizace do záložek za správnou?



Ano	9	90 %
Ne	1	10 %

Je podle Vás užitečné, že je v přihlašovacím dialogu použita validace vstupních dat?



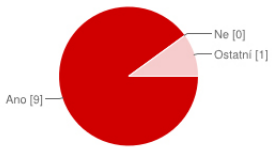
Ano	9	90 %
Ne	0	0 %
Nevím	1	10 %

Máte nějaké výhrady k přihlašovacímu dialogu? Pokud ano, jaké?

nastavení připojení a IP adres mohlo být ve stejném okně s přihlašovacím formulářem

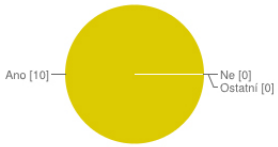
Chybí možnost zapamatování už.jména

Považujete umístění obrazu kamer do pravé části aplikace za správné?



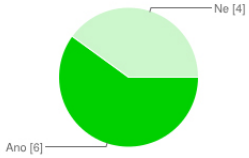
Ano	9	90 %
Ne	0	0 %
Ostatní	1	10 %

Považujete umístění ovládacích prvků kamer vlevo od jejich obrazu za správné?



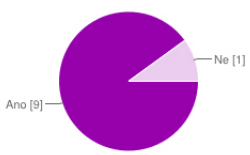
Ano	10	100 %
Ne	0	0 %
Ostatní	0	0 %

Po najetí na obraz z kamery lze pomocí kolečka přibližovat a oddalovat, napadlo vás použít pro ovládání zoomu kamery kolečko myši?



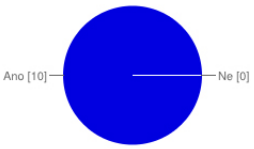
Ano	6	60 %
Ne	4	40 %

Přijde Vám intuitivní ovládání PŘIBLÍŽENÍ a ODDÁLENÍ kamery?



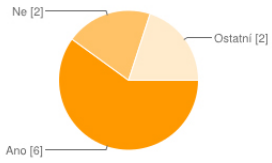
Ano	9	90 %
Ne	1	10 %

Po najetí na obraz z kamery lze pomocí funkce "Drag and drop" a gesta natočit obraz kamery, je toto použití intuitivní?



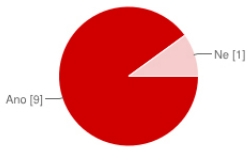
Ano	10	100 %
Ne	0	0 %

Bylo pochopitelné, jak ovládat otáčení kloubů robotického ramene?



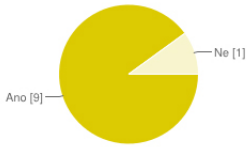
Ano	6	60 %
Ne	2	20 %
Ostatní	2	20 %

Přijde Vám zvolení tlačítek pro ovládání otevření/zavření chapadla ramene správné?



Ano	9	90 %
Ne	1	10 %

Považujete reakční dobu aplikace bez simulovaného spoždění komunikace za bezproblémovu?



Ano	9	90 %
Ne	1	10 %

Při jakém spoždění se Vám zdálo ovládání částečně použitelné?

3s 2-4s 4s 3-4s 3s 4s 1s 3-4s 2s 2-4s

Při jakém spoždění se Vám zdálo ovládání zcela nepoužitelné?

5s 5 a více sekund >8s >5s 10s 6s 3s víc jak 5s 5s 5-6s