VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

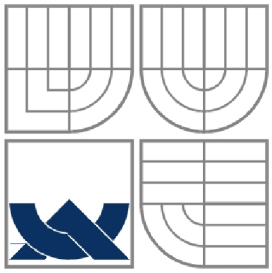
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SYSTÉM PRO SLEDOVÁNÍ AKTIVIT PROGRAMÁTORA

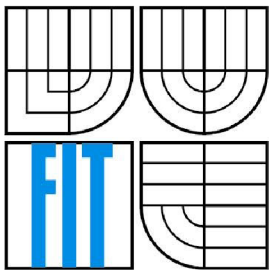
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE KAROL JARKOVSKÝ
AUTHOR

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SYSTÉM PRO SLEDOVÁNÍ AKTIVIT PROGRAMÁTORA

PROGRAMMING ACTIVITY TRACKING SYSTEM

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR

KAROL JARKOVSKÝ

VEDOUCÍ PRÁCE
SUPERVISOR

ING. LADISLAV RUTTKAY

BRNO 2009

Abstrakt

Práce se zabývá problematikou systémů a sledování aktivity programátorů po právní, technické a technologické stránce. Uvažuje postavení takových systémů v současné legislativě a zaměřuje se na určení hranic, které monitorovací systém odlišují od jeho nelegální formy. Pojednává o možnostech sledování procesů a uživatelských aktivit na systémech Microsoft Windows s použitím technik dostupných pro platformu .NET. Rozebírá možnosti pro přenos informací ze sledované stanice na zpracovávající server pomocí standardních protokolů na přenos určených. Identifikuje potenciální hrozby spojené s takovým přenosem. Návrh a implementace rozšíření popisuje s použitím modelovacího jazyka UML a návrhových vzorů.

Klíčová slova

Sledovací systém, .NET, Windows API, Microsoft Windows Service, Microsoft Web Service, UML

Abstract

The thesis is concerned with the subject of systems designed for monitoring the activity of programmers with respect to legal, technical and technological aspect. It ponders on the role and condition of such systems in the present-day legislation and focuses on the definition of the boundaries that distinguish the monitoring system from its illegal version. The paper also deals with the options concerning monitoring of processes and users' activities within Microsoft Windows systems using technologies included in .NET Framework. The thesis further analyses the possibilities of information transmission from the monitored station to the processing server by means of standard protocols designed for transmission and identifies potential threats associated. The UML modelling language is used to describe design and architecture of the implemented solution.

Keywords

Tracking system, .NET, Windows API, Microsoft Windows Service, Microsoft Web Service, UML

Citace

Jarkovský Karol: Systém pro sledování aktivit programátora. Brno, 2009, bakalářská práce, FIT VUT v Brně.

Systém pro sledování aktivit programátora

Prohlášení

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

……
Karol Jarkovský
20. 5. 2009

Poděkování

Děkuji za možnost vypracovat toto zajímavé zadání pod vysoce odborným vedením Ing. Ladislava Ruttkaye. Zároveň děkuji kolegům z Kentico software, kteří participovali jako odborní poradci ve věcech spojených s celkovým návrhem architektury finálního řešení.

Obsah

Obsah 1
1 Úvod 3
2 Sledovanie aktivity a legislatíva 4
2.1 Sledovať znamená kontrolovať 4
2.2 Koho a prečo sledovať? 4
2.3 Pohľad zamestnanca 5
2.4 Legislatíva 5
2.4.1 Povinnosti zamestnancov 5
2.4.2 Práva zamestnancov 6
2.5 Prirodzený stav 6
3 Techniky sledovania aktivity 7
3.1 Základné rozdelenie 7
3.2 Off-line sledovacie systémy 7
3.3 Poznámky k technológii 8
4 Procesy a komunikácia 9
4.1 Rozhranie Win32 API 9
4.1.1 História a súčasnosť 9
4.1.2 Windows API funkcie volané z .NET 10
4.2 Platform Invoke (P/Invoke) 10
4.3 Komunikácia správami 11
4.3.1 Formát správ 12
4.3.2 Správy v OS Windows 12
4.4 Zachytávanie správ v .NET 12
4.4.1 Techniky Hooks a GlobalHooks 13
4.5 Správa ako zdroj informácií 14
5 Prenos informácií 15
5.1 Webové služby ASP .NET 15
5.1.1 Protokol SOAP 15
5.2 Bezpečnosť webových služieb 17
5.2.1 Druhy potenciálnych hrozieb a webové služby 17
6 Návrh, dizajn a implementácia 19
6.1 Klientska časť 19
6.1.1 Aktivita užívateľa 19
6.1.2 Implementačné detaily 19

6.1.3 Správa lokálnych dát 20
6.1.4 Životný cyklus klientskej aplikácie 21
6.1.5 Diagram tried 22
6.2 Serverová časť 23
6.2.1 Implementačné detaily 23
6.2.2 Dáta na strane serveru 27
6.2.3 Webová služba 28
6.2.4 Zabezpečenie webovej služby 28
7 Záver 29
Literatúra 30
Zoznam príloh 31
Príloha 1: Inštalačný manuál 32
Príloha 2: Práca s webovým rozhraním 35

1 Úvod

Cieľom tejto technickej správy je ponúknuť čitateľovi obraz o problematike systémov na sledovanie aktivít užívateľov, presnejšie podskupiny užívateľov – programátorov. V dobe kedy svetom otriasa ekonomická kríza a spoločnosť sa snaží šetriť finančné zdroje je jednou z možností ako tento cieľ dosiahnuť zefektívnenie využívania prostriedkov, s ktorými spoločnosť disponuje. Pri prenesení tejto skutočnosti do firemného prostredia ide využiteľnosť zdrojov ruka v ruke s produktivitou zamestnancov. Efektívnejšie využitie pracovnej doby sa rovná väčšej produktivite čo ma v konečnom dôsledku pozitívny dopad na riadenie a správu firemných prostriedkov.

K získaniu prehľadu o produktivite zamestnancov je potrebné disponovať informáciami o aktivitách, ktoré počas pracovnej doby vyvíjajú. Táto potreba je motiváciou pre bakalársku prácu, ktorej súčasťou a výsledným produktom je implementácia sledovacieho systému s architektúrou klient – server.

Po úvodnom pohľade na postoj práva, pracovného a trestného, k sledovacím systémom sa text práce venuje možnostiam sledovania aktivity užívateľov a technikám umožňujúcim monitorovanie vykonávať. Ďalšia časť práce je venovaná komunikácii procesov v systéme a technológiám, pomocou ktorých výmena informácií medzi procesmi prebieha. Obsahuje popis systémových služieb, ktoré sú na implementáciu sledovacieho systému ideálne. Záver teoretickej časti patrí prenosu správ medzi klientom a serverom pomocou webových služieb. V časti popisujúcej implementáciu finálneho riešenia je možné nájsť informácie o architektúre, návrhu a dizajne aplikácie a detailoch, ktoré ju sprevádzali.

2 Sledovanie aktivity a legislatíva

Sledovanie aktivity užívateľov operačných systémov sa stáva čoraz častejším prostriedkom na kontrolu zamestnancov zo strany manežmentu. Záznamy takéhoto systému obvykle slúžia ako dôkazový materiál v zložitých interných šetreniach zahájených voči jednotlivcom podozrivým z porušovania pracovnej kázne a disciplíny.

Veľmi citlivou témou súčasnosti je ochrana súkromia a osobných údajov. Aplikácie sledovacieho typu sú často považované za niečo čo prekračuje hranicu legálnosti. V tejto časti sú podané najčastejšie dôvody pre zavedenie takýchto systémov do firemných štruktúr a jeho dopady. Ďalej kapitola popisuje polohu sledovania a následného záznamu aktivity užívateľa v rámci právneho systému krajín Európskej únie.

2.1 Sledovať znamená kontrolovať

Na začiatok je potrebné povedať, že práca pojednáva o sledovacom systéme, ktorého účelom nie je zbierať osobné informácie o užívateľovi zneužiteľné treťou stranou ako heslá, čísla bankových účtov, čísla kreditných kariet a pod. Určitým tabu by mali ostať aj záznamy o stlačených klávesoch. Systém slúži výhradne na kontrolu morálky jednotlivca tvoriaceho širšiu skupinu ľudí, od ktorých sa očakáva podávanie stabilného pracovného výkonu.

2.2 Koho a prečo sledovať?

Každá firma je závislá na výkone svojich pracovníkov. Nasadenie a pracovná morálka v tíme ľudí určuje výsledky firmy ako celku. Pri stále vyššej úrovni postupov zavádzaných do firemných procesov sa tou najnebezpečnejšou a najnepredvídateľnejšou formou vo firme stávajú zamestnanci. Tri najpodstatnejšie faktory hovoriace v prospech zavedenia sledovacieho systému do podnikových štruktúr je krádež firemných dát, zatiahnutie škodlivého kódu do siete a v neposlednom rade možnosť kontrolovať dodržiavanie interných podnikových smerníc.

Údaje získané sledovaním je možné využiť na rôznych úrovniach vo firemnej štruktúre. Nielenže pomáhajú priebežne identifikovať potenciálne škodlivé aktivity užívateľov vo firemnej sieti, ale dokážu podať pomerne presný a hlavne aktuálny obraz o vyťaženosti ľudských zdrojov. Týmto nepriamo pomáhajú relevantným oddeleniam regulovať pracovnú silu a v neposlednom rade izolovať jednotlivcov, ktorí sú pre firmu dôležití a naopak.

2.3 Pohľad zamestnanca

Na jednej strane dobrý sledovací systém dokáže ponúknuť vedeniu mnoho užitočných informácií, ale na strane druhej dokáže pri nedodržaní ľudských princípov medzi zamestnancami vyvolať nevôľu voči zamestnávateľovi, spôsobiť napätie a zhoršiť vnútrofiremné vzťahy.

Zamestnávateľ má právo kontrolovať nakladanie s firemnými prostriedkami, či už ide o výpočtovú techniku alebo ľudské zdroje. Je však dobré informovať zamestnancov o zavedenom systéme. Treba poukázať na skutočné prínosy systému a využiť ho na spravodlivé odmeňovanie zamestnancov.

Nasadenie sledovacieho systému je na území Českej republiky vhodné konzultovať s Úradom pre ochranu osobných údajov, ktorý môže eliminovať prípadné nedostatky v politike monitorovania zamestnancov.

2.4 Legislatíva

V dnešnom firemnom prostredí je veľmi motivujúcim faktorom znižovanie nákladov paradoxne pri zvyšovaní výkonov. Logickým dôsledkom takéhoto trendu je aj zavádzanie sledovacích systémov slúžiacich na zbieranie údajov potrebných k efektívnemu regulovaniu firemných zdrojov.

Čoraz častejšie sa však s takýmito tendenciami v spoločnosti objavuje otázka či sledovanie a používanie sledovacích systémov nie je náhodou nezákonný spôsob kontroly a či ich používanie neprekračuje hranice určene legislatívou.

2.4.1 Povinnosti zamestnancov

V minulosti bolo používanie sledovacích systémov veľmi rozporuplné a mnohými zástupcami z radov laickej aj odbornej verejnosti považované za obmedzovanie osobnej slobody. Situácia je dnes však iná a oporu je možné nájsť aj v súčasne platných zákonoch.

Český zákonník práce [1] v §73 odst. 1 definuje povinnosti zamestnanca ako:

(1) Zaměstnanci jsou povinni zejména

a) pracovat svědomitě a řádně podle svých sil, znalostí a schopností, plnit pokyny nadřízených vydané v souladu s právními předpisy a dodržovat zásady spolupráce s ostatními zaměstnanci,

b) plně využívat pracovní doby a výrobních prostředků k vykonávání svěřených prací, plnit kvalitně, hospodárně a včas pracovní úkoly,

c) dodržovat právní předpisy vztahující se k práci jimi vykonávané; dodržovat ostatní předpisy vztahující se k práci jimi vykonávané, pokud s nimi byli řádně seznámeni,

d) řádně hospodařit s prostředky svěřenými jim zaměstnavatelem a střežit a ochraňovat majetek zaměstnavatele před poškozením, ztrátou, zničením a zneužitím a nejednat v rozporu s oprávněnými zájmy zaměstnavatele.

Bližší pohľad na obsah písmena b) jasne hovorí, že pracovná doba má byť plne využitá na vykonávanie priradenej práce. Zamestnávateľ má teda právo kontrolovať či zamestnanec dodržuje ustanovenia prítomné v Zákonníku práce a navyše má právo kontrolovať či sa s prostriedkami firmy nakladá vhodne a efektívne. To vyplýva z obsahu písmena d).

O porušovaní ochrany osobnej slobody teda z tohto pohľadu nemôžeme hovoriť pokiaľ nie je sledovací software inštalovaný na súkromnom počítači, ktorý nie je majetkom firmy.

2.4.2 Práva zamestnancov

Ako bolo už v úvode tejto kapitoly naznačené línia medzi legálnym a nelegálnym monitorovaním je veľmi tenká a preto existuje rada legislatívnych doplnkov, ktoré upevňujú postavenie zamestnanca v tejto veci.

S ohľadom na článok 13 Listiny základných práv a slobôd [2], s ohľadom na nariadenie Európskeho súdu pre ľudské práva týkajúce sa čl. 8 Dohovoru o ochrane ľudských práv a základných slobôd [3] má zamestnanec prirodzené právo na ochranu súkromia, ktorého súčasťou je aj právo na ochranu osobných údajov a to tiež vo vzťahoch pracovno-právnych. Predpokladá sa, že zamestnávateľ bude takéto právo rešpektovať. Tieto práva však musia byť v rovnováhe s legitímnymi právami zamestnávateľa.

2.5 Prirodzený stav

Momentálny stav vo firemných prostrediach si viac či menej vyžaduje určitý druh kontroly dodržovania pracovnej doby zamestnancov. Nejde pritom len o to, ponúknuť riadiacim skupinám nástroj na umravnenie zamestnancov ako by mohol niekto poznamenať, ale vo väčšine prípadov ide o prirodzenú potrebu získať prehľad o vyťaženosti prostriedkov rôzneho charakteru.

Pri dodržaní práv zamestnancov a ich informovaní o prebiehajúcom monitorovaní je zavedenie sledovacieho systému prínosom pre obe zúčastnené strany a počiatočné investície spojené s nasadzovaním systému sú návratne vo veľmi krátkom čase. Keďže v konečnom dôsledku dochádza aplikáciou systému do firemných štruktúr k šetreniu finančných prostriedkov je možné považovať takýto systém za prospešný, umožňujúci spätné smerovanie financií k zamestnancom.

3 Techniky sledovania aktivity

Sledovanie aktivity užívateľov sa rozvíja podobným tempom ako dnešný hardvér alebo softvér. Kým na začiatku 90. rokov bola otázka sledovania aktivity užívateľov na výpočtových staniciach nerelevantná a z technického hľadiska ani nerealizovateľná, s nástupom obdobia rozmachu internetu dochádzalo postupne k zavádzaniu systémov, ktoré odhaľovali v oblasti zbierania dát o aktivitách systémov dovtedy skryté možnosti.

Spektrum produktov ponúkaných na trhu so sledovacími systémami v súčasnosti číta stovky až tisíce a kategorizácia je pre lepšiu orientáciu nezainteresovaného pozorovateľa takmer nevyhnutná. Táto kapitola si preto kladie za cieľ podať ucelený obraz o základných technikách používaných pri monitorovaní aktivít užívateľov vo vnútri systému. V závere definuje obraz riešenia implementovaného v rámci bakalárskej práce.

3.1 Základné rozdelenie

Najčastejším prvotným rozdelením systémov je členenie na základe módu, v ktorom pracujú. Hovoriť tak môžeme o on-line a off-line systémoch.

V poslednej dobe sú stále častejšie používané on-line monitorovacie systémy, ktoré získavajú informácie o užívateľovej aktivite priamo za behu. Spracovávateľom dát je veľmi často tretia osoba, ktorá sleduje prácu náhodného užívateľa prostredníctvom obrazu jeho plochy, dokáže zachytávať komunikáciu po sieti alebo sledovať akcie procesov priamo v reálnom čase.

Druhou skupinou sú systémy, ktoré získavajú údaje o aktivitách užívateľa počas celého behu systému, ale takto nadobudnuté informácie nie sú analyzované v reálnom čase. Miesto toho sú vo formáte určenom charakterom informácií ukladané na sledovanej stanici alebo sú šírené sieťou ďalej, k stanici na správu takýchto dát určenej.

Keďže sa táto práca zaoberá sledovacím systémom zo skupiny off-line bude aj ďalší text pojednávať o tomto type monitorovacích nástrojov. Oproti klasickým off-line systémom rozširuje funkcionalitu o možnosť zasielať zozbierané údaje na server, k centralizovanému spracovaniu, hneď potom čo je detekované aktívne pripojenie.

3.2 Off-line sledovacie systémy

Systémy tejto skupiny sa môžu deliť do podskupín na základe ich prevedenia na softvérové a hardvérové. Ako už názvy napovedajú líšia sa v spôsobe existencie v rámci sledovaného systému.

Hardvérové zariadenia fungujú zväčša na princípe odchytávania impulzov produkovaných nejakým periférnym zariadením. Keylogger, ako sa odborne takéto sledovacie zariadenie nazýva,

tvorí medzistupeň od periférneho zariadenia k stanici, na ktorej beží sledované prostredie. Do internej pamäte zariadenia sa zaznamenávajú vyvolané akcie a uchovávajú sa na dodatočnú analýzu. Zjavným nedostatkom u takýchto zariadení je limitovaná kapacita pamäte určenej na záznam udalostí a potreba mechanicky dolovať uložené dáta.

Omnoho rozšírenejšiu podskupinu tvoria softvérové keyloggery. Ide o druh aplikácie, ktorá zaznamenáva udalosti prebiehajúce v systéme a údaje ukladá priamo na disk alebo odosiela po sieti ďalej v závislosti na tom, ako bola pôvodne navrhnutá. Odpadá teda problém s pamäťou a doručovanie dát je veľmi jednoducho realizovateľné automaticky bez nutnosti manuálne do akcie zasahovať.

3.3 Poznámky k technológii

Za ideálny stav by sme s ohľadom na možnosti sledovania užívateľových aktivít mohli označiť taký stav, kedy by jednotlivé akcie odohrávajúce sa na ľubovoľnej stanici boli monitorovateľné z centrálneho bodu v sieti. A to navyše bez potreby spúšťať na monitorovanej stanici akúkoľvek klientsku časť sledovacieho systému.

Dôležitým faktom, ktorý je potrebné uviesť, a ktorý určuje spoločnú črtu väčšiny off-line sledovacích softvérových systémov je, že kvôli limitáciám dnes používaných sieťových protokolov a adresárových služieb prístup popísaný vyššie nie je možné realizovať. Prehľad o spustených procesoch, čase strávenom prácou s jednotlivými aplikáciami a pod. je možné získať výhradne pomocou prostriedkov bežiacich na cieľovom systéme. Z toho vyplýva, že každý takýto systém pozostáva z klientskej a serverovej časti.

Sledovací systém vo svojej klientskej časti umiestnenej v monitorovanom systéme potrebuje získavať údaje o všetkých aktivitách užívateľa. V ideálnom prípade tak, aby pri samotnej práci neprekážal, a zároveň doba jeho behu bola zhodná s behom systému a to bez ohľadu na to, či došlo počas tejto doby k odhláseniu zo systému. Ako najvhodnejšia forma sa v tomto prípade javí aplikácia bežiaca na pozadí, odchytávajúca všetku komunikáciu medzi časťami systému.

Ako už bolo naznačené v predchádzajúcom odstavci informácie o akciách sú získavané na základe odchytávania správ, ktoré si systém a jednotlivé aplikácie medzi sebou v rámci systémovej komunikácie odosielajú.

Pri uvažovaní o spracovaní získaných údajov je podstatný tiež fakt, že nikdy nie je možné spoliehať sa na dostupnú vzájomnú konektivitu monitorovanej stanice a spracovávajúceho bodu v sieti. Preto je potrebné uvažovať dvojstupňovú techniku pri ukladaní a preposielaní získaných dát. Pod pojmom dvojstupňová technika je myslený spôsob spracovania, ktorý primárne ukladá informácie lokálne a po overení spojenia predáva údaje do spracovateľskej (rodičovskej) stanice. Až po úspešnom odoslaní údajov sú lokálne dáta zmazané.

4 Procesy a komunikácia

Proces počas svojho behu v systéme vykonáva akcie, ktoré mnohokrát vyžadujú aby systém umožnil procesu prístup k systémovým prostriedkom. Môže taktiež nastať situácia, kedy proces podobnú službu požaduje od iného procesu bežiaceho na rovnakej úrovni. V oboch prípadoch je potrebné mať k dispozícii mechanizmus, ktorý umožní procesu komunikovať so systémom alebo iným procesom štandardizovanou cestou. V OS typu Windows poskytuje takéto možnosti skupina funkcií označovaných súhrne ako Win32 API [4].

Táto kapitola pojednáva o aplikačnom rozhraní Win32 API, s dôrazom kladeným na jeho dôležitosť a postavenie v rodine systémov Windows. Popisuje prostriedky, ktoré API poskytuje a vysvetľuje úlohu tohto rozhrania pre riešenie bakalárskej práce.

4.1 Rozhranie Win32 API

Úlohou tejto podkapitoly je popísať vlastnosti a funkcie Win32 rozhrania, často nazývaného Windows API, ktoré sú relevantné k téme bakalárskej práce.

Po krátkom úvode do histórie je v tejto časti práce rozoberaný najmä spôsob integrácie a volania nespravovaného kódu Windows API, tzv. „unmanaged code“ [5], z metód, ktoré sú súčasťou spravovaného .NET kódu.

4.1.1 História a súčasnosť

Win32 API reprezentuje sadu aplikačných rozhraní prítomných v operačných systémoch spoločnosti Microsoft. V súčasnosti sa názov Win32 pomaly vytráca a začína sa miesto neho udomácňovať pomenovanie Windows API. Pôvodne Win32 API totiž nevyjadruje presne skutočnosť, že toto rozhranie je podporované od prvých 16-bitových verzií až po najnovšie 64-bitové systémy Windows.

Windows API je súčasťou OS Windows od úplných začiatkov, tzn. od verzie Windows 1.0 (r. v. 1983) aj keď v tom čase ešte pod názvom Win16 API. Funkcionalita, ktorú ponúkalo Windows API sa rozširovala s každou novou verziou a stále sa rozširuje o desiatky až stovky funkcií uľahčujúcich vývojárom prácu. Tieto funkcie dávajú vývojárom na jednej strane veľkú mieru voľnosti a kontroly nad aplikáciami, na strane druhej ale vyžadujú zvýšenú pozornosť a zodpovednosť zo strany programátora pri manipulácii so systémovými prostriedkami.

V minulosti bolo kladené zo strany Microsoftu veľké úsilie na spätnú kompatibilitu jeho operačných systémov. S príchodom .NET platformy sa však vo vnútri spoločnosti začali presadzovať názory, že je potrebná zámena pôvodného modelu aplikačného rozhrania pre OS Windows za model, ktorý by pracoval so spravovaným kódom. Výsledkom je iniciatíva o vytvorenie nového Windows

API rozhrania, ktoré by natívne podporovalo spravovaný kód a používalo pre svoj beh .NET platformu.

Je hodné diskusie, nakoľko je zavedenie takéhoto spravovaného kódu do nízkoúrovňových knižníc výhodné. Je však isté, že zavedením tohto modelu dôjde k uľahčeniu práce s rozhraním operačného systému najmä z pohľadu .NET vývojára.

Jednoduchosť prístupu k funkcionalite ponúkanej Windows API však v súčasnosti nie je zďaleka samozrejmosťou. Komplikácie sú spojené najmä s importovaním (sprístupnením) nespravovaného kódu Windows API do tried, ktorých kód je spravovaný jazykom MSIL pre .NET platformu. Ide o jeden z najzložitejších problémov, ktorý sa pri vývoji aplikácií využívajúcich Windows API musí brať v úvahu. Nasleduje preto text, ktorý sa venuje práve tomuto problému a poskytuje jedno z možných riešení.

4.1.2 Windows API funkcie volané z .NET

Ako bolo nepriamo naznačené vyššie, nespravovaný preložený kód písaný v jazyku C alebo C++ je binárnemu súboru, tzv. assembly pre platformu .NET veľmi odlišný. Zatiaľ čo bežný kód nespravovaného programu je prekladaný do strojového kódu a interpretovaný priamo systémom, assembly pre .NET je prekladané do jazyka MSIL a spúšťané CLR, čo je prostredie pre beh aplikácii implementovaných pre .NET platformu. Tento základný rozdiel definuje problém kooperácie nespravovaného a spravovaného kódu.

Platforma .NET poskytuje niekoľko techník, ktoré umožňujú volať nespravovaný kód zo spravovaného a naopak. Jednotlivé možnosti sa líšia najmä podľa toho o aký typ objektu nespravovaného kódu ide – môže to byť statické DLL [6], čiže dynamicky linkovaná knižnica, alebo objekt zapúzdrený do COM rozhrania.

Pre potreby práce je potrebné získať v .NET triedach možnosť pristupovať k funkciám, ktoré sú súčasťou Windows API. Tieto funkcie sú distribuované vo forme DLL knižníc. Ideálnou technikou pre exponovanie funkcií statického DLL je technika nazývaná Platform Invoke (P/Invoke) [7].

4.2 Platform Invoke (P/Invoke)

P/Invoke dovoľuje volať funkciu napísanú v ľubovoľnom nespravovanom jazyku priamo v .NET kóde pokiaľ je táto funkcia znovu deklarovaná v spravovanom kóde. Služba takejto spolupráce je poskytovaná priamo CLR a je súčasťou vrstvy zodpovednej za komunikáciu medzi jazykmi rôznej úrovne.

Použitie techniky P/Invoke vyžaduje aby deklarácia funkcie, teda jej hlavička, bola umiestnená okrem pôvodného hlavičkového súboru aj priamo v .NET triede, z ktorej bude volaná. Import exportovateľnej funkcie písanej v nespravovanom kóde sa deje pomocou `DllImport` atribútu.

Napríklad pre sprístupnenie funkcie `ShowWindow()` z DLL knižnice `user32.dll` rozhrania Windows API by kód importu vyzeral nasledovne:

```
// Import funkcie ShowWindow() z kniznice user32.dll
[DllImport("user32.dll")]
static extern IntPtr ShowWindow(IntPtr hWnd,int nShow);
```

Deklaráciou externej funkcie bolo povedané, že implementácia konkrétnej funkcie je prítomná v uvedenej DLL knižnici. Kód spravovaný .NET prostredím tak získal prístup k funkcii a môže ju volať a používať tak, ako by išlo o metódu spravovaného kódu. Podobným spôsobom je možné zviditeľniť akúkoľvek externú funkciu pokiaľ je jej export povolený pri implementácii DLL.

Niektoré funkcie z Windows API ako parametre, vstupné alebo výstupné, očakávajú typ, ktorý je reprezentovaný na .NET platforme odlišne ako vo Windows API. V takýchto prípadoch je potrebné explicitne určiť odpovedajúci objekt .NET k typu z Windows API. Explicitnému predefinovaniu sa hovorí marshaling. Nasledujúci príklad ukazuje ako je možné predefinovať parameter funkcie z pôvodného `LPTSTR` na jeho ekvivalent v .NET – parameter typu `string`.

```
// Marshaling LPTSTR na string
[DllImport("msvcrt.dll")]
public static extern int puts([MarshalAs(UnmanagedType.LPStr)]string
m);
```

Aplikáciou vyššie uvedených poznatkov je možné sprístupniť a používať akékoľvek funkcie Windows API. To je jedna z podmienok úspešnej implementácie riešenia tejto práce.

4.3 Komunikácia správami

Aplikácie bežiace v systémoch typu Windows sú, na rozdiel od v minulosti používaných systémov typu MS-DOS, riadené udalosťami. Systém vyvoláva udalosti, generuje určité stavy alebo poskytuje vstupy na ktoré aplikácie reagujú. Znamená to, že program takpovediac čaká, pokiaľ mu nie je predané riadenie v podobe vstupov, na ktoré dokáže reagovať. Správy však nemusia byť generované iba systémom. Samotné aplikácie môžu generovaním správ informovať napríklad o svojom stave alebo reagovať na akciu vyvolanú iným prvkom v systéme.

Udalosti a k nim odpovedajúce správy sú generované ako následok akcie užívateľa. Vznikajú napríklad pri pohybe myšou, stlačení klávesy klávesnice a pod. Vyvolané môžu byť tiež systémom ako reakcia na činnosť určitej aplikácie. Nemenej často však ide o správy generované aplikáciou reagujúcou na akcie inej aplikácie.

4.3.1 Formát správ

Správy sú stále zasielané aplikáciám do ich hlavnej funkcie. Súčasťou správy sú informácie o ukazateľovi na okno a type správy doplnené dvoma parametrami. Hodnota týchto parametrov je závislá na type prijatej správy a nemá pevne daný typ, resp. formát. Môže teda ísť o sekvenciu bitov, nastavené príznaky, ukazateľ na štruktúru a pod. Pri spracovaní parametrov je potrebné najskôr identifikovať správu aby bolo jasné aká forma hodnôt sa dá očakávať.

4.3.2 Správy v OS Windows

Systém Windows rozlišuje primárne dva druhy správ podľa entity, ktorá ich popisuje:

- definované systémom,
- definované užívateľom.

Správy definované systémom slúžia na riadenie určitej aplikácie. Systém ich používa na delegovanie vstupov a ďalších informácií pre aplikáciu. Tento druh správ môže generovať aj aplikácia. Všetky typy správ definovaných systémom majú unikátny identifikátor a odpovedajúcu znakovú reprezentáciu. Sú definované v hlavičkových súboroch.

Správy definované užívateľom slúžia aplikáciám na vytváranie vlastných špecifických správ, na ktoré reagujú okná danej aplikácie. Ak sú takéto správy exponované prostredníctvom API aplikácie môžu byť samozrejme využívané na vzájomnú komunikáciu programov a služieb rôzneho druhu.

Pre potreby bakalárskej práce sú dôležité systémové správy nakoľko je požadované monitorovať udalosti odohrávajúce sa medzi systémom a aplikáciami a naopak.

4.4 Zachytávanie správ v .NET

Systém správ v OS Windows, je ako už bolo povedané, postavený na Windows API a nespravovanom kóde. V predchádzajúcich kapitolách bol popísaný problém používania nespravovaných funkcií vo vnútri tried spravovaných.

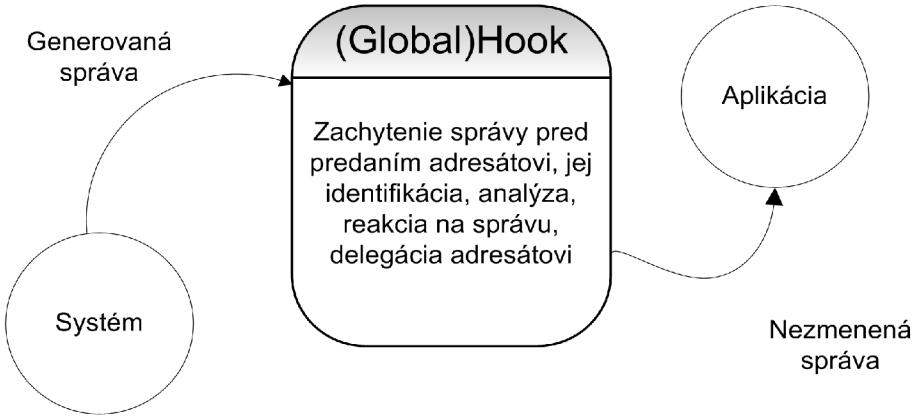
Komunikácia správami je zabezpečovaná sadou Windows API funkcií, ktoré sú volané pri získavaní správy, pri preklade virtuálneho kódu klávesy na zrozumiteľnú hodnotu, pri preposielaní správy oknu špecifikovanému ukazateľom na okno, atď. Funkcie Windows API musia byť k dispozícii v spravovanom kóde a preto je na ich vystavenie potrebné využiť export nespravovaného kódu spomínaný vyššie – technikou P/Invoke.

O technikách, ktoré v kombinácii s P/Invoke vytvárajú základ pre zachytávanie správ pojednáva nasledujúci text.

4.4.1 Techniky Hooks a GlobalHooks

Pre sledovanie komunikácie jednotlivých aplikácií so systémom je nutné získať prehľad o vymieňaných správach. V podstate je potrebné stať sa akýmsi prostredníkom, cez ktorého bude tok správ prechádzať pred tým, ako dorazí k adresátovi a bude ním spracovaný. Táto podkapitola sa zaoberá technikami, ktoré takúto činnosť umožňujú a pre potreby práce plne vyhovujú.

Zahákovanie, anglicky „hooking“, je termín v informatike označujúci techniku, ktorá umožňuje nainštalovať rutinu do systému výmeny správ a získať tak prehľad o udalostiach, ktoré v systéme nastávajú. Keďže slovenský preklad termínu pôsobí rušivo dovolím si používať v ďalšom texte pôvodný anglický názov pre techniku ako aj pre typy možných hákov – Hooks a GlobalHooks [8].



Obr. 1: *Grafické znázornenie princípu techniky hákovania*

Systém poskytuje viacero rôznych typov hákov (hooks), z ktorých každý poskytuje špecifické informácie o type správ, ktoré zachytáva. Hák, ktorého znaková konštanta je `WH_MOUSE` umožňuje napríklad odchytávať a monitorovať akcie spojené s aktivitou myši, `WH_KEYBOARD` zase akcie spojené s používaním klávesnice a pod.

Nasleduje zoznam typov hákov podporovaných systémom Windows:

- `WH_CALLWNDPROC a WH_CALLWNDPROCRET` – správy posielané oknu aplikácie,
- `WH_CBT` – aktivácia, vytváranie/uzatváranie, minimalizácia, maximalizácia, veľkosť okna,
- `WH_FOREGROUNDIDLE` – informácia o nulovej aktivite aplikácie,
- `WH_GETMESSAGE` – upozornenie pri získavaní správy z fronty,
- `WH_KEYBOARD_LL a WH_KEYBOARD` – aktivita klávesnice,
- `WH_MOUSE_LL a WH_MOUSE` – aktivita myši,
- `WH_MSGFILTER a WH_SYSMSGFILTER` – udalosti spojené s objektmi okna.

Pre prácu s hákmi a zachytávanými správami je potrebné definovať tzv. callback funkciu pre konkrétny typ háku. Táto funkcia je následne volaná zakaždým, keď sa medzi správami objaví taká, ktorá spadá do kategórie správ reprezentovaných daným typom háku – tým, pre ktorý sa callback funkcia registrovala. Registrované callback funkcie, predstavujúce háky, sú rozdelené podľa odchytávaného typu a uložené vo fronte typu FIFO. To znamená, že pokiaľ určitý typ správy zachytáva viacero hákov je správa spracovaná hákmi v takom poradí, v akom sa o ňu registrovali.

Rozdiel medzi hákmi (hooks) a globálnymi hákmi (global hooks) je ten, že lokálne háky dokážu reagovať iba na správy zasielané konkrétnej aplikácii bežiacej v konkrétnom vlákne. To, pre ktoré vlákno sa budú správy odchytávať, je určené identifikátorom vlákna predaným ako parameter do funkcie registrujúcej callback funkciu – hák.

Na rozdiel od lokálnych hákov dokážu globálne háky reagovať na všetky správy, ktoré sa vo fronte správ systému vyskytnú a ponúkajú tak možnosť sledovať komunikáciu v systéme ako celku. Táto funkcionalita je však pochopiteľne náročnejšia na systémové prostriedky, nakoľko sa zapája do spracovania akejkoľvek správy vyskytujúcej sa v systéme.

Pokiaľ sú používané lokálne háky môže byť spravovaný kód v podobe DLL, obsahujúci importované funkcie a samotnú obsluhu hákov, súčasťou adresového priestoru aplikácie. Keď však potrebujeme odchytávať všetky správy a používame globálne háky je potrebné DLL knižnicu načítať do spoločného adresového priestoru, tzv. shared memory point. Táto akcia nie je triviálna a v .NET platforme sa s ňou viažu určité problémy, ktoré tvoria druhú komplikáciu pri implementovaní aplikácie ako tá, ktorá je súčasťou bakalárskej práce. Ako sa vyrovnať s touto situáciou popisuje kapitola o implementácii finálneho riešenia.

4.5 Správa ako zdroj informácií

Okrem zachytenia správy je dôležité vedieť ako z danej správy získať informácie o procese, ktorému je určená. Systém správ je nepretržite v akcii, za zlomok sekundy sa medzi procesmi a systémom môžu vymeniť desiatky správ a nie všetky sú pre sledovanie užívateľa podstatné. Navyše, ak by sme sa snažili každú správu spracovať a analyzovať bez uváženia, mohlo by dochádzať k citeľnému spomaleniu systému. Analýza správy totiž v podstate predstavuje jej pozdržanie na ceste k cieľu a počas doby analýzy pôvodný adresát prechádza do stavu čakania.

Každá správa je sprevádzaná dvojicou parametrov – `WPARAM` a `LPARAM`. Podľa typu správy táto dvojica obsahuje informácie o akcii spätej so správou v špecifickom formáte. Môže ísť o ukazateľ na štruktúru, ktorá udržiava informácie o aktuálnej polohe kurzoru myši alebo ukazateľ na štruktúru s informáciami o objekte, ktorý má aktuálne zaostrenie alebo mnoho iných informácií.

Pri analýze správy je preto potrebné identifikovať jednoznačne typ správy a parametre pretypovať na objekty, ktoré v skutočnosti predstavujú. Vo chvíli kedy je nám formát dát predaných zo správou jasný je otázkou implementácie ako s nimi naložíme.

5 Prenos informácií

Pri aplikáciách s architektúrou klient – server je dôležitou súčasťou návrhu aj spôsob prenosu informácií z klienta na server. Konkrétna implementácia tejto funkcionality by sa mala odvíjať od požiadavku na konektivitu riešenia. Použitá bude iná technika posielania dát ak bude predpoklad, že klient beží spolu so serverom lokálne, resp. v spoločnej LAN sieti a iná ak bude sever umiestnený mimo ňu. V našom prípade je cieľom implementovať server ako webovú aplikáciu, ktorá bude centrálne spravovať informácie o všetkých monitorovaných staniciach a tieto informácie bude poskytovať prostredníctvom webového rozhrania.

Táto kapitola sa preto zaoberá práve technikou, ktorá umožňuje komunikáciu aplikácie bežiacej lokálne s rozhraním serveru bežiaceho na webovom servery. Technológia popisovaná nižšie sa nazýva Webová služba – Web Service – a je súčasťou rodiny objektov .NET platformy.

5.1 Webové služby ASP .NET

Webové služby [9] sú jednou zo základných súčastí Microsoft ASP .NET technológie určenej na tvorbu a podporu moderných webových aplikácií.

Webové služby zapúzdrujú programovú logiku do formy prístupnej užívateľom, presnejšie užívateľským aplikáciám, prostredníctvom siete internet. Ide pritom o vystavovanie sady metód implementovaných v rámci služby, ktoré slúžia ako zdroj dát pre aplikácie. Na prenos dát sú využívané štandardné webové technológie ako napríklad HTTP. Komunikácia medzi aplikáciou a webovou službou je založená na výmene správ v štandardizovanom formáte XML. Používanie týchto dvoch štandardov je spojené do jedného protokolu nazývaného SOAP – Simple Object Access Protocol [10]. Pre využívanie webových služieb nie je potrebné pre konzumujúcu aplikáciu vedieť detaily ako programovací model služby alebo jazyk v akom je implementovaná. Je to možné hlavne vďaka protokolu SOAP, ktorý dokáže správy pred odoslaním transformovať do štandardizovane formy a zaslať na server a naopak.

5.1.1 Protokol SOAP

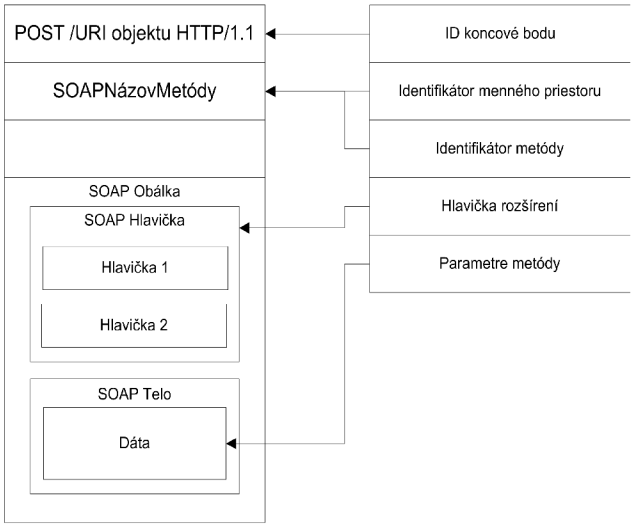
Protokol SOAP poskytuje možnosti ako sprostredkovať existujúce aplikácie ešte širšej mase ľudí, aplikácií, ktoré používajú prostredníctvom webu a webových technológií. Ako bolo spomenuté vyššie, využíva HTTP protokol na prenos správ vo formáte XML, ktoré tvoria základný komunikačný prostriedok protokolu SOAP.

SOAP neposkytuje žiadnu formu API, ktorá by ho v konečnom dôsledku zväzovala len k jednému jazyku alebo konkrétnej platforme. Princípy SOAP sú založené na atomických akciách špecifikovaných použitými technológiami. Je preto úlohou spoločností ako Microsoft, Sun alebo

DevelopMendor aby podporu SOAP implementovali do svojich platforiem a poskytli tak vývojárom ľahší prístup k funkcionalite poskytovanej SOAP protokolom.

Najčastejším použitím SOAP protokolu je jeho implementácia ako prostriedku na volanie vzdialených procedúr – RPC – Remote Procedure Call. Ak hovoríme o SOAP protokole ako o komunikačnom prostriedku webových služieb .NET platformy je SOAP využívaný práve na volanie vzdialených metód.

Práca SOAP protokolu je jednoduchá. Pri požiadavku je odosielaná HTTP GET požiadavka, na ktorú je prijímaná HTTP POST odpoveď. Súčasťou odosielanej GET požiadavky je SOAP hlavička, ktorá špecifikuje o volanie akej metódy a z akého menného priestoru máme záujem. Je nasledovaná vstupnými poprípade vstupno-výstupnými parametrami pre metódu, kódovanými vo formáte XML. Formát požiadavku je možné vidieť na obrázku nižšie.



Obr. 2: *Jednoduchá SOAP GET požiadavka*

V uvedenom grafe je možné vidieť, že SOAP obálka vo svojom najhlbšom zanorení obsahuje časť definujúcu parametre metódy a ich hodnoty. Tieto informácie sú do požiadavky vkladané v XML formáte. Je to jediná časť hlavičky reprezentovaná XML. V SOAP má preto XML jedinú úlohu a to transformovať (serializovať) dáta pre metódu do transportovateľnej podoby. V prípade dát jednoduchých typov sa môže zdať, že konverzia nie je potrebná, ale je nutné si uvedomiť, že prenášané dáta nesmú byť diskriminované na základe typu. Je tak nevyhnutné serializovať najmä dáta typu štruktúr, ukazateľov alebo komplexných objektov.

Odpoveď na SOAP požiadavku je vo formáte veľmi podobnom samotnej požiadavky. XML definícia dát však tento krát obsahuje výstupné, vstupno-výstupné hodnoty. V odpovedi chýba pôvodná SOAP hlavička, ktorá je prítomná iba v požiadavke.

5.2 Bezpečnosť webových služieb

Webové služby sú postavené na komunikácii posielanej zo súkromných sietí cez verejnú sieť internet až k adresátovi, ktorý je opäť súčasťou súkromnej siete. Ako každý takýto druh prenosu je vystavený nebezpečenstvám, ktoré na neho číhajú vo verejných sieťach kde nie je prostredie kontrolovateľné vyššou autoritou. Aj keď sa v kontexte tejto bakalárskej práce prostredníctvom webových služieb neprenášajú citlivé informácie, ide o prenos dát, a ako taký by mal byť za každých okolností zabezpečený. Aspoň aplikáciou základných bezpečnostných techník. Táto časť popisuje základné typy hrozieb vzťahujúcich sa na prenos dát pri využívaní webových služieb. Zaoberá sa tiež bezpečnosťou webových služieb ako takých.

5.2.1 Druhy potenciálnych hrozieb a webové služby

Bezpečnostné hroby spojené s používaním webových služieb by sa dali rozdeliť do troch hlavných kategórií. Ide o hrozby spojené s:

- predkladaním falošnej identity,
- predkladaním škodlivého obsahu,
- zahlcovaním služby.

Nebezpečenstvá plynúce z predkladania falošnej identity sú relevantné najmä kvôli použitiu HTTP protokolu v SOAP komunikácii. V súčasnosti existujúce štandardy spojené s autentifikáciou a podpisovaním XML dokumentov ako XML-Encryption [11] alebo XML-Signature [12] tento problém čiastočne riešia na úrovni aplikačnej vrstvy. Ako obrana proti odpozorovaniu prenášaných dát sa aplikuje SSL. Úplne zabezpečená webová služba by mala prednostne odmietať akékoľvek správy, ktorých autentifikácia, autorizácia alebo integrita vyzerá podozrivo.

Druhú skupinu útokov tvoria útoky, kedy správa určená webovej službe je pred doručením modifikovaná a jej obsah je pozmenený tak, aby pôsobil škodlivo. Najčastejší je prípad kedy je cieľom útoku prinútiť službu aby vykonala akciu, na ktorú nebola určená. Použitím techniky SQL prieniku (SQL Injection) by sa útočník mohol snažiť získať napríklad informácie z databázy, ktoré nie sú určené k čítaniu. Ako obrana proti takýmto útokom môže slúžiť používanie vlastnej XML definičnej schémy, ktorá jasne určuje aké hodnoty, maximálne a minimálne, môže ten-ktorý parameter nadobúdať, aká je minimálna, resp. maximálna povolená dĺžka hodnoty atribútu a pod. Pred samotným spracovaním správy službou je jej validáciou voči takto presne špecifikovanej XML schéme možné odhaliť podvrhnutú správu ešte pred tým, ako začne jej spracovanie zaberať systémové prostriedky.

Poslednou skupinou hrozieb sú útoky založené na snahe zhodiť systém pomocou opakovaných požiadaviek na vykonanie určitej akcie, tzv. Denial of Service (DoS) útoky. Pri webových službách

spracovávajúcich XML dáta je typickým príklad pozmenenia SOAP správy tak, že sa objem dát niekoľko násobne navýši a dôjde k zahlteniu pamäťového priestoru aplikácie, ktorá následne „umrie“. Môže ísť však aj o modifikáciu, kedy sa štruktúra XML zmení tak aby hierarchické zanorenie elementov na relatívne malom objeme dát spôsobilo vysokú záťaž CPU s rovnakým dôsledkom ako predtým. Voči DoS útokom je účinnou obranou monitorovanie správ prijímaných službou. Štatistické spracovanie informácií o požadovanej akcii, type a počte parametrov, frekvencii prijímaných správ a pod. umožňuje dynamicky predvídať tento typ útokov.

6 Návrh, dizajn a implementácia

Výstup bakalárskej práce predstavuje systém na sledovanie aktivít programátora, pozostávajúci z dvoch navzájom nezávislých funkčných celkov – klientskej a serverovej časti. Text záverečnej kapitoly práce sa zaoberá návrhom, architektúrou a implementáciou výsledného riešenia. Pre lepšiu predstavu je popis každej časti doplnený relevantným diagramom a detailmi spojenými s implementáciou.

6.1 Klientska časť

V predchádzajúcej kapitole pojednávajúcej o off-line systémoch bolo uvedené, že takéto systémy dokážu získavať potrebné informácie o užívateľovej aktivite takmer výhradne z pozície systému, bežiaceho na monitorovanej stanici. Klientska časť riešenia bakalárskej práce teda predstavuje aplikáciu inštalovanú na klientskej stanici, spúšťanú, bežiacu a ukončovanú spoločne s operačným systémom. Aplikácia je pritom nezávislá na účte, pod ktorým je spúšťaná a dokáže monitorovať aktivitu všetkých užívateľov systému.

6.1.1 Aktivita užívateľa

Keďže úlohou aplikácie na monitorovanej stanici je zbieranie informácií o užívateľovej aktivite, je na tomto mieste vhodné definovať pojem „aktivita užívateľa“ tak, ako je chápaný v kontexte bakalárskej práce.

Pod termínom „aktivita užívateľa“ sa rozumie informácia o tom, ktorá aplikácia je aktuálne aktívna, kedy bola aktivovaná a ako dlho bola na popredí – ako dlho s ňou mohol užívateľ pracovať. Podstatná je taktiež informácia o texte okna aktívnej aplikácie, prípadne o názve modulu, ktorý aplikáciu v systéme registroval a spúšťal. Aby bolo možné čo najpresnejšie určiť či užívateľ skutočne s aktívnou aplikáciou pracoval, patria medzi sledované vlastnosti systému taktiež údaje o využívaní periférnych zariadení a vyťaženosti systémových prostriedkov (procesoru a operačnej pamäte) počas doby aktivity.

6.1.2 Implementačné detaily

Klientska časť je implementovaná ako desktop aplikácia postavená na technológii WinForms .NET Frameworku, bez primárne viditeľného rozhrania. Po spustení je minimalizovaná do systémovej lišty. Beží na pozadí a z pohľadu užívateľa mu neprekáža pri práci. Od okamihu jej spustenia až do ukončenia systému zbiera informácie o aktivite užívateľa. Jednotlivé údaje sú získavané aplikáciou techniky hákovania, vysvetlenej v kapitole o zachytávaní správ v OS Windows.

Zbierané informácie sú udržiavané v pamäti do chvíle, kedy ich množstvo nedosiahne nastavenú maximálnu hranicu. V takomto prípade sa v novom vlákne, asynchrónne, začne ukladanie informácií z pamäte do dátového úložiska na disku. Viac o formáte ukladaných dát a spôsobe ich uloženia je uvedené v nasledujúcej podkapitole. Cyklus zbierania dát, ukladania do pamäte a asynchrónneho ukladania na disk sa opakuje počas celej doby behu systému až do ukončenia aplikácie samotnej.

Dáta sú z úložiska odosielané na server v okamihu, kedy je aplikáciou detekované aktívne pripojenie na internet, a zároveň došlo k overeniu dostupnosti brány na strane spracovávajúceho serveru. Prijatie dát je serverom potvrdzované tak, aby mohla klientska aplikácia následne lokálne údaje zmazať. Takýmto spôsobom sa dbá na efektívne využívanie diskového priestoru monitorovanej stanice.

Dôležitou črtou systému je, že všetky akcie spojené s behom klientskej aplikácie sú kvôli optimalizácii, plynulosti, použiteľnosti a efektívnosti aplikácie vykonávané asynchrónne – v samostatnom vlákne. Predchádza sa tým situáciám, kedy by na seba naplánované akcie museli čakať, čím by mohlo dochádzať k citeľnému spomaleniu celého systému.

6.1.3 Správa lokálnych dát

Množstvo dát, ktoré pri sledovaní vzniká, je potrebné uchovávať bezpečne v prehľadnom a udržovateľnom formáte.

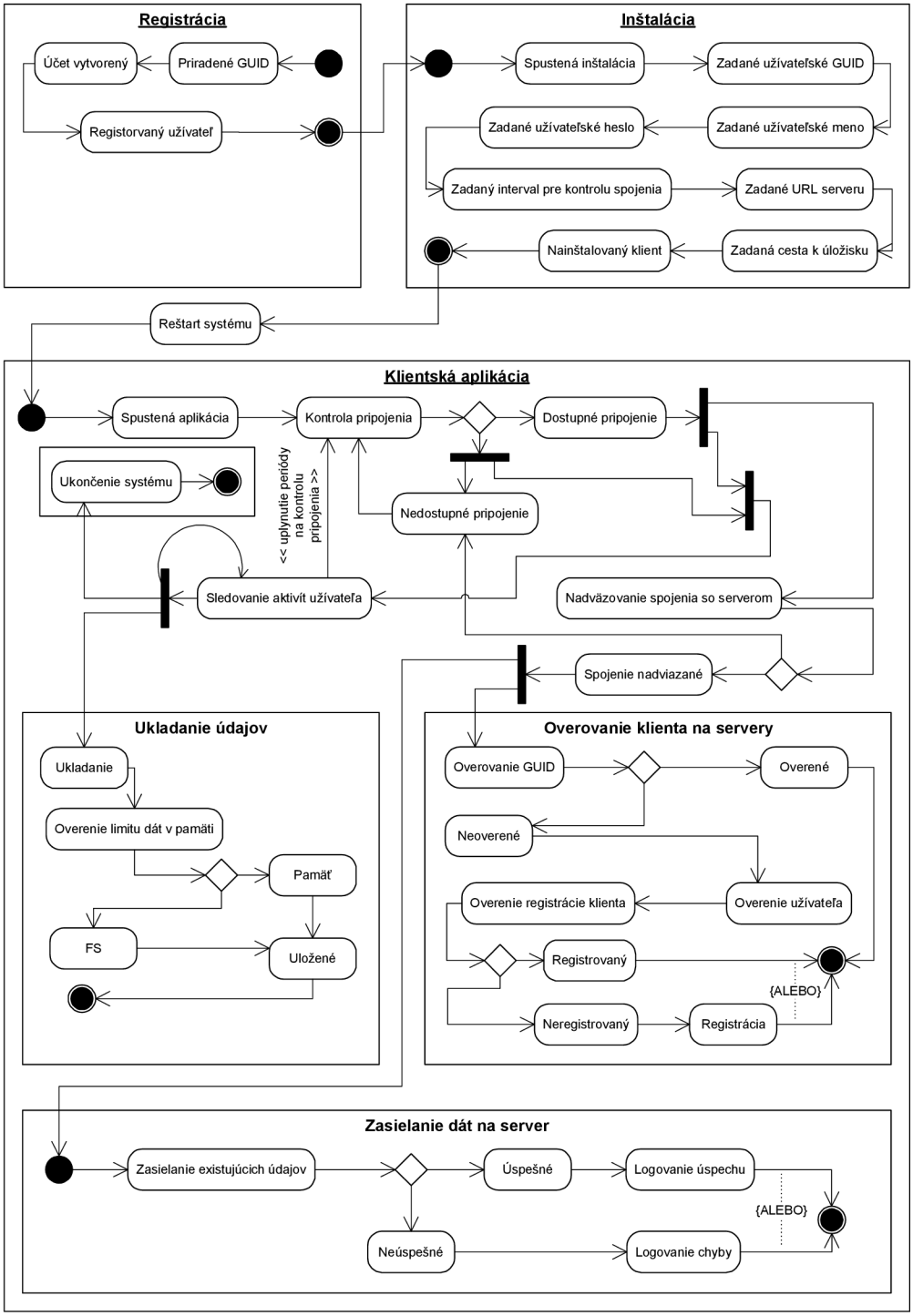
Všetky zozbierané informácie sa uchovávajú v lokálnom dátovom úložisku, ktorého lokalita na disku je špecifikovaná užívateľom pri inštalácii klientskej aplikácie. Hierarchická štruktúra adresárov v tejto lokalite priradzuje jednotlivým užívateľom systému vlastnú vetvu a napomáha tak k udržiavaniu lepšieho prehľadu v úložisku.

Dáta sú na disk ukladané vo formáte XML, ktorého schéma je určená charakterom samotných dát. Výsledné XML je produkované serializáciou objektov v pamäti do textovej podoby. Kvôli bezpečnosti a možnej citlivej povahe údajov sú pred uložením šifrované symetrickou šifrou štandardu DES – Data Encryption Standard. Ide konkrétne o jednu z jeho možných variant – TripleDES [13]. Pri použití tohto typu šifrovania sa aplikuje algoritmus na blok dát hneď tri krát. Prispieva to najmä k zvyšovaniu odolnosti šifrovaných dát voči neoprávnenému dešifrovaniu aplikovaním techník hrubej sily (brute-force attacks).

Pre každého užívateľa sa okrem samotných dát o aktivite, ukladajú navyše informácie o úspešnom nadviazaní spojenia so serverom či chybách, ktoré počas behu aplikácie nastali. Nie sú zasielané na server a slúžia výhradne na analýzu príčin prípadných problémov s transferom sledovacích dát alebo pripojenia so serverom.

6.1.4 Životný cyklus klientskej aplikácie

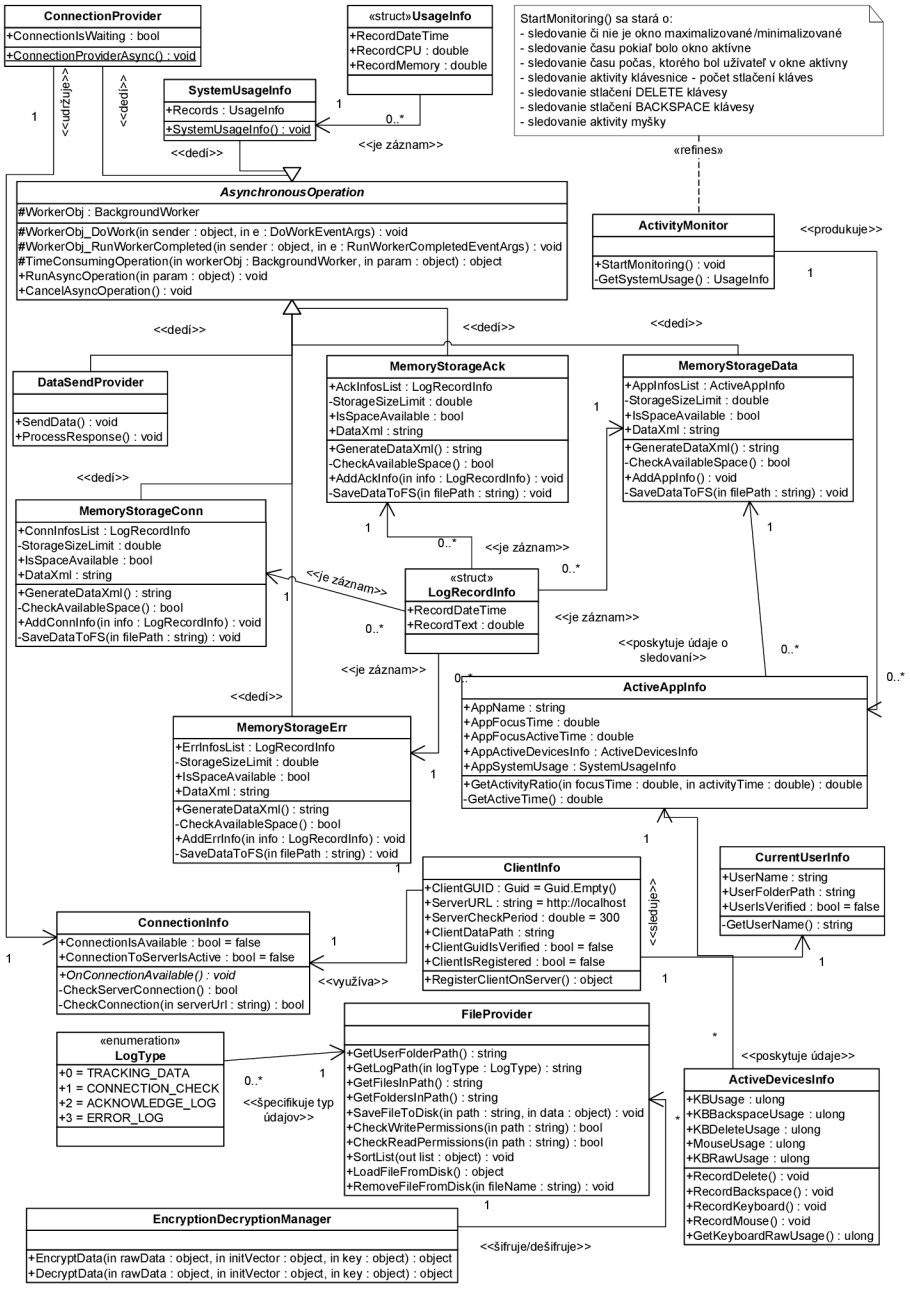
V nasledujúcom diagrame je možné vidieť životný cyklus, ktorým klientska aplikácia prechádza počas svojho behu. Cyklus začína spustením aplikácie na monitorovanej stanici. Pre prehľadnosť je v diagrame zahrnutá aj registrácia klienta prostredníctvom webového rozhrania. Užívateľ registrovaním získava unikátny identifikátor, ktorý je požadovaný počas inštalácie.



Obr. 3: *Diagram znázorňujúci prechodové stavy klientskej aplikácie*

6.1.5 Diagram tried

Diagram uvedený nižšie zachytáva prepojenie a spoluprácu jednotlivých tried tvoriacich klientsku aplikáciu. Niektoré z týchto tried, ako napríklad trieda `AsynchronousOperation`, sa využívajú taktiež na strane serveru. Pokiaľ hovoríme o tejto triede, stojí určite za zmienku, že práve ona tvorí základ všetkých akcií prebiehajúcich v systéme asynchrónne. Zdedením z tejto triedy a povinnou implementáciou jej jedinej metódy, je možné do systému zasadiť akýkoľvek ďalší prvok, ktorý bude svoje akcie vykonávať nezávisle na primárnom vlákne.



Obr. 4: *UML diagram tried klientskej aplikácie*

6.2 Serverová časť

Serverová časť implementovaného systému pozostáva z troch menších celkov. Ide o webový portál, administračné rozhranie a webovú službu – dátovú bránu.

Webový portál slúži výhradne ako registračný bod. Prostredníctvom portálu sa registrujú klienti, ktorý po registrácii získavajú unikátny identifikátor. Ten ich jednoznačne identifikuje v celom systéme. Zadáva sa pri inštalácii klientskej aplikácie na monitorovanú stanicu. Využíva sa napríklad pri nadväzovaní spojenia so serverom, pri zasielaní lokálnych dát k spracovaniu a pod.

Webové administračné rozhranie slúži na správu a prehliadanie záznamov o všetkých sledovaných staniciach, ktoré sa viažu ku klientovi – zadaním identifikátora pri inštalácii. Rozhranie poskytuje možnosť jednoduchým spôsobom kategorizovať informácie o užívateľoch. Formou grafov a štatistík zobrazuje informácie o aktivite.

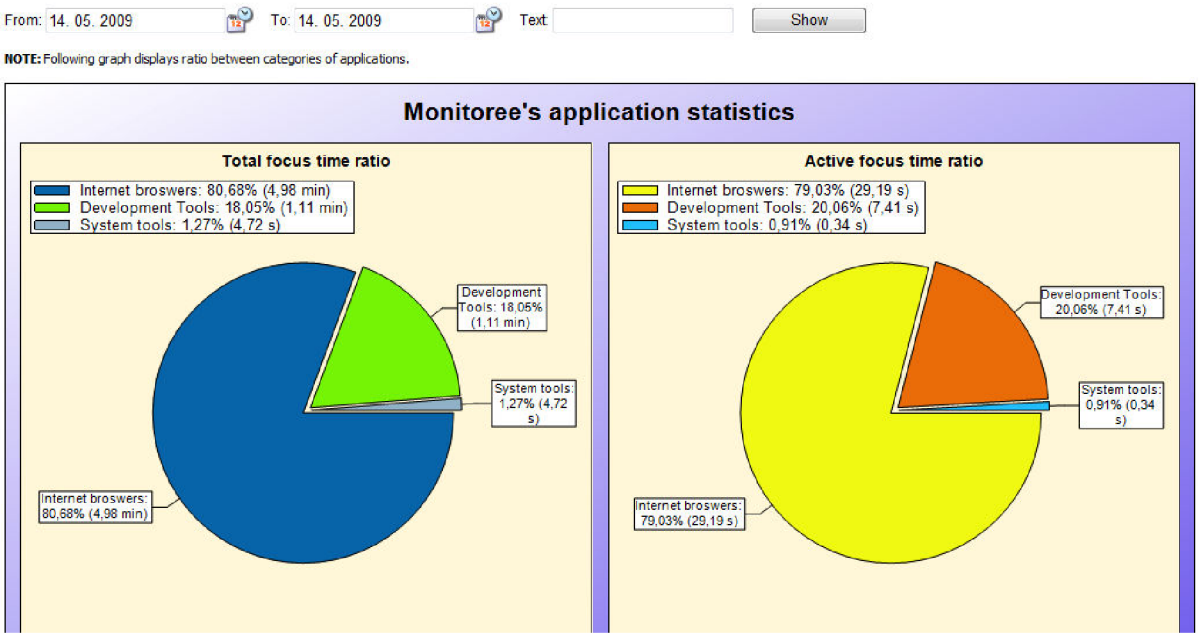
Webová služba predstavuje sadu funkcií exponovaných aplikáciám tretej strany, a teda aj našej klientskej aplikácii. Medzi funkciami, ktoré služba ponúka, je okrem iných funkcia pre overenie dostupnosti služieb serveru, overenie klienta, overenie monitorovanej stanice či funkcia na prijímanie dát.

6.2.1 Implementačné detaily

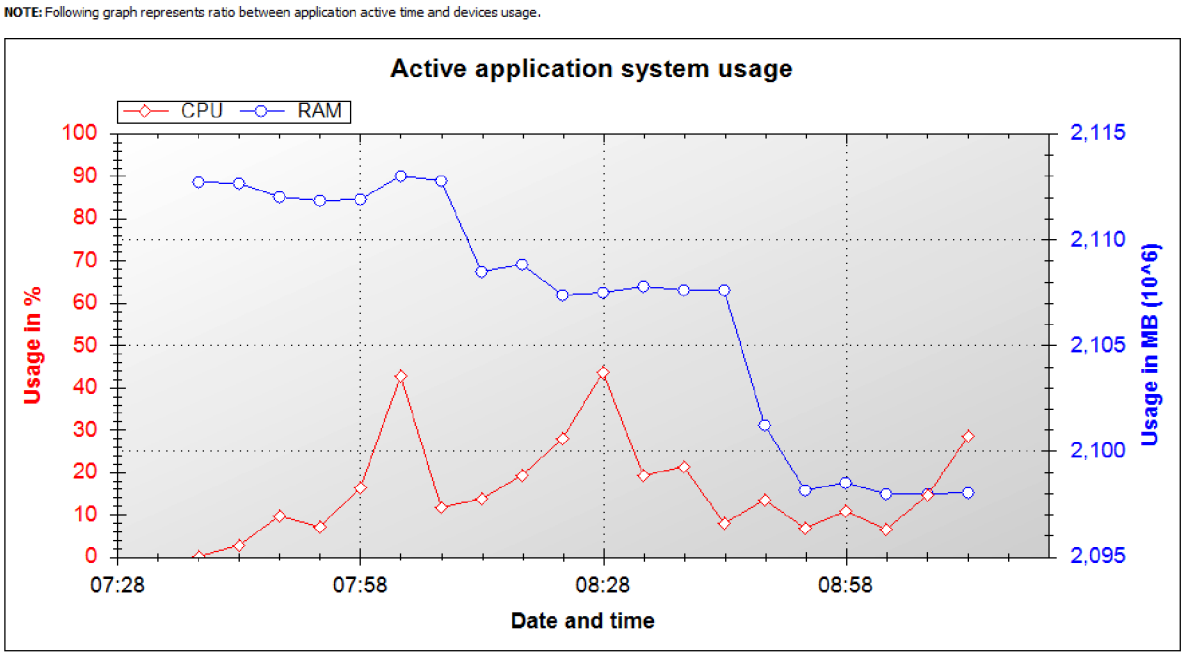
Serverová časť systému je implementovaná ako webová aplikácia využívajúca technológiu ASP .NET 2.0, v kombinácii s databázovým serverom Microsoft SQL Server. Podporované sú verzie od SQL Server 2000 vyššie.

Ako bolo spomenuté v úvode tejto podkapitoly, server pozostáva z viacerých častí. Každá časť je na sebe funkčne nezávislá, čím sa dosiahlo možnosti upravovať funkcionalitu jednotlivých častí bez toho, aby bolo nutné odstaviť pri každej úprave celý systém. Ide o dôležitú črtu, ktorá je vyžadovaná najmä preto, že implementovaný systém obsahuje rezervy v rozsahu ponúkanej funkcionality, ktoré môžu byť v budúcich verziách jednoducho doplnené.

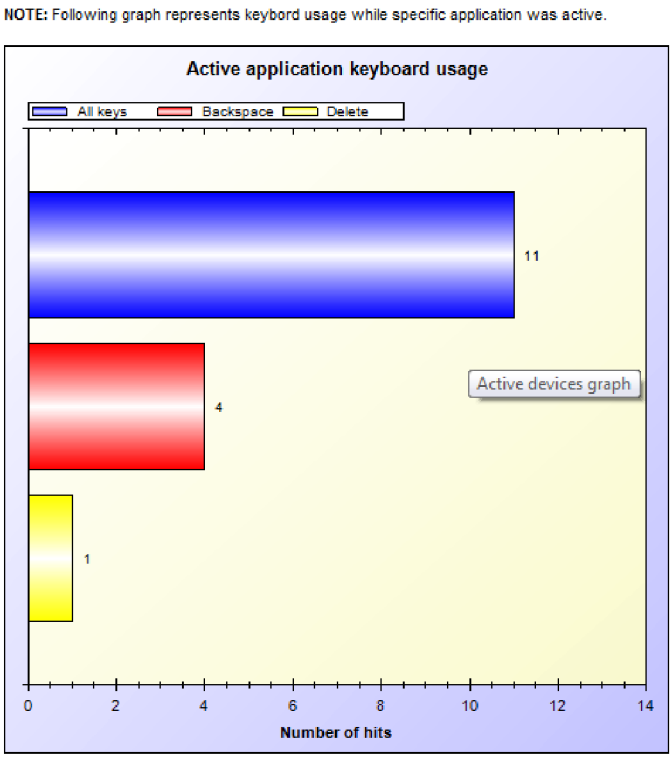
Všetky údaje zozbierané na monitorovaných staniciach sú po prenose na server dostupné k nahliadnutiu v administračnom rozhraní. To je súčasťou webovej aplikácie a je prístupné pre klientov registrovaných na servery. Slúži ako centralizovaný bod pre správu a zobrazovanie informácii o aktivite jednotlivých užívateľov. Vzhľad, ovládanie a zoznam funkcionality, ktoré toto rozhranie poskytuje je možné nájsť v užívateľskom manuály, ktorý je prílohou technickej správy. Určte je však vhodné zdôrazniť, že miesto textovej podoby je každý relevantný údaj, z kvanta informácií, reprezentovaný odpovedajúcim grafom – pokiaľ to samozrejme povaha informácie dovoľuje. Ukážka zobrazovaných grafov sa nachádza nižšie.



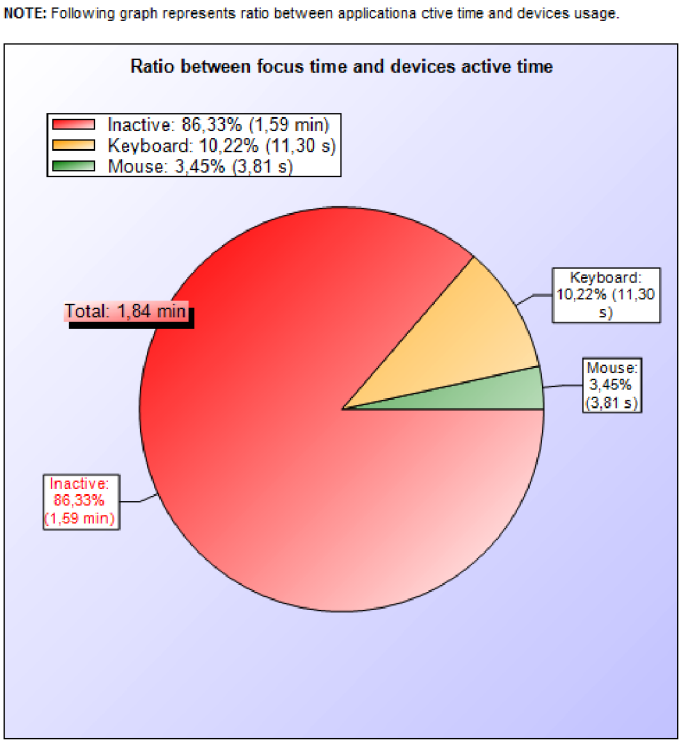
Obr. 5: *Graf zobrazujúci celkový čas, počas ktorého boli aktivované aplikácie jednotlivých kategórií (vľavo) a graf zobrazujúci čas užívateľa strávený prácou s aplikáciami jednotlivých kategórií (vpravo)*



Obr. 6: *Graf zachytávajúci využitie systémových prostriedkov (CPU, RAM) počas doby kedy bola aplikácia aktívna*



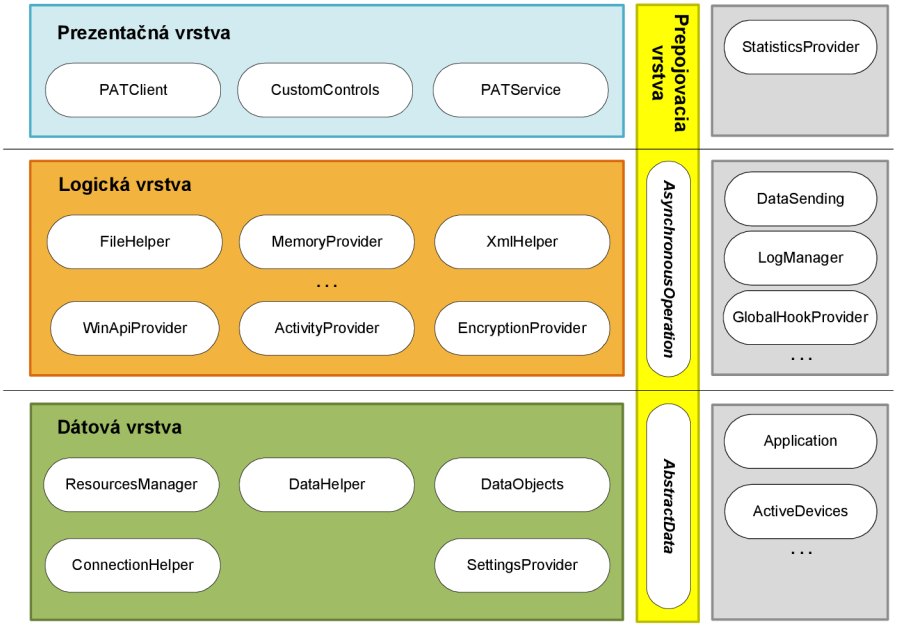
Obr. 7: *Graf zobrazujúci detailné využitie klávesnice počas doby kedy bola aplikácia aktívna*



Obr. 8: *Graf zobrazujúci podiel času aktivity jednotlivých*

Z toho čo bolo doposiaľ k implementácii serverovej časti systému uvedené je zjavné, že server používa klasickú trojvrstvú architektúru. Pozostáva teda z vrstvy zabezpečujúcej kooperáciu s databázovým systémom – získavanie a predávanie dát, transformácia dát do podoby objektov a pod. Výstupy dátovej vrstvy slúžia ako vstupy pre nadradenú, logickú vrstvu. Tá prostredníctvom nadefinovanej sady metód a objektov, a dopredu definovaného správania transformuje vstupy z dátovej vrstvy na vstupy pre najvyššie položenú vrstvu. Je ňou vrstva prezentačná, ktorá je zodpovedná za prezentovanie informácií, výsledkov spracovania, užívateľovi. Na zobrazovanie informácií pritom opäť využíva sadu metód poskytovaných API implementovaného systému.

Nasledujúci diagram zobrazuje postavenie jednotlivých vrstiev v architektúre systému. Ako je možné vidieť, systém poskytuje na každej vrstve triedu, ktorá tvorí prepojenie medzi systémom a modulom zodpovedným za určitú funkcionalitu systému. Zdedením z prepojovacej triedy je možné systém jednoducho rozšíriť bez toho, aby bolo potrebné vlastniť jeho zdrojový kód. Systém tak môžeme označiť za značne otvorený voči prispôsobovaniu samotným užívateľom.

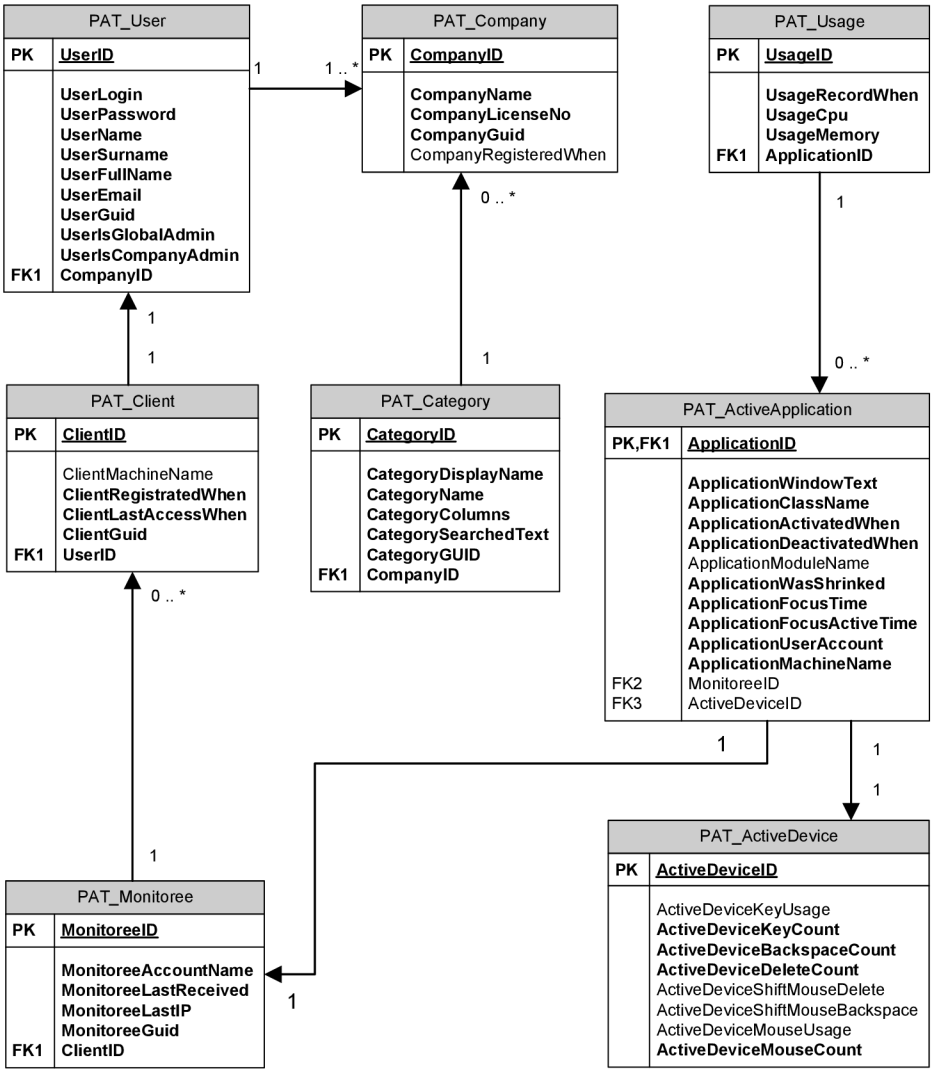


Obr. 9: *Architektúra systému*

6.2.2 Dáta na strane serveru

Na monitorovanej stanici sú informácie ukladané v lokálnom dátovom úložisku. Na strane serveru tieto informácie predstavujú omnoho väčší objem dát, a preto je nevyhnutné uchovávať jednotlivé údaje v relačnej databáze.

Architektúra databázy použitej pre uchovanie získaných údajov je zobrazená na diagrame uvedenom nižšie.



Obr. 10: *E-R diagram použitého databázového riešenia*

Z diagramu plynie, že pre každého klienta zaregistrovaného v systéme existuje spoločnosť, ktorú reprezentuje. Objekt spoločnosť zastrešuje všetky monitorované stanice, ktorých klientska aplikácia bola inštalovaná s použitím registračných údajov a identifikátoru konkrétneho klienta. Definované kategórie sú spoločné pre všetky sledované objekty patriace pod jednu spoločnosť.

6.2.3 Webová služba

Webová služba, ktorá je súčasťou serverovej časti sledovacieho systému, plní funkciu brány. Prostredníctvom nej do systému prúdia údaje o sledovaných užívateľoch, zozbierané klientskymi aplikáciami.

Webová služba v prvom rade slúži na prijímanie informácií a neposkytuje žiadnu logickú funkcionalitu, ktorá by bola klientom využívaná. Ide o jednosmernú komunikáciu v smere od klienta k serveru, a preto nie je možné prostredníctvom služby získavať informácie o sledovaní ostatných užívateľov v systéme.

6.2.4 Zabezpečenie webovej služby

Vystavenie funkcionality tretím stranám prostredníctvom webovej služby, vytvára priestor pre potenciálne útoky. Môže ísť najmä o snahy zo strany skupín, ktoré sa prostredníctvom webovej služby chcú dostať k údajom uloženým na servery alebo majú záujem na poškodení systému ako takého.

Pri zakomponovaní webovej služby do architektúry systému je potrebné zakaždým uvažovať aj bezpečnostný aspekt takéhoto rozhodnutia. V prípade implementovaného riešenia je ochrana voči útokom, spomenutým v kapitole o bezpečnosti webových služieb, dvojstupňová.

Prvým stupňom ochrany je rozhodnutie neposielať autorizačné údaje ako súčasť dát odosielaných na server. Všetky dáta sú odosielané v šifrovanej podobe tak, ako boli uložené v lokálnom úložisku. Spolu s nimi sa odosiela navyše iba identifikátor klienta, ku ktorému sa dáta viažu. Dešifrovanie prijímaných dát prebieha na strane serveru aplikovaním dešifrovacieho algoritmu, za pomoci užívateľského mena a hesla zadaného pri registrácii. Tieto údaje by mali byť známe iba poverenej osobe na strane klienta. Zabezpečiť, aby sa tieto údaje nedostali do rúk nepovolanej osoby, už nie je teda v réžii systému. Ak sa spracovávané dáta nepodarí dešifrovať, je jasné, že neboli šifrované klientom, ktorý sa prostredníctvom zaslaného identifikátora prezentoval.

Druhý stupeň v ochrane pred zneužitím, resp. zneprístupnením funkcionality webovej služby, je validácia XML schémy prijímaných správ. Pred ďalším spracovaním sa po dešifrovaní deserializujú dáta z textovej podoby, pričom dochádza k validácii schémy. Pokiaľ deserializácia prebehne v poriadku je na mieste vyhlásiť dáta za validné. V opačnom prípade je pravdepodobné, že s dátami bolo manipulované, a preto sú systémom ignorované.

Vyššie uvedenými technikami sa podarilo významne znížiť náchylnosť systému k jeho zneužitiu neautorizovaným prístupom či aplikovaním v ľubovoľnom smere škodlivých dát.

7 Záver

Momentálna situácia v spoločnosti poskytuje s rastúcim napätím na finančných trhoch, väčší priestor pre sledovacie systémy najrôznejšieho charakteru. Systémy, ktoré informácie zozbierané na sledovanej stanici zasielajú k spracovaniu na server, sa radia k najvyužívanejším. K takýmto systémom patrí aj implementované riešenie.

Práca na vytváraní systému dokázala, že sa rozhodne nejedná o triviálne zadanie s jednoduchým riešením. Hovoríme pritom o obmedzeniach kladených legislatívou krajín s vyspelým právnym systémom, limitovaných možnostiach čo sa týka povahy sledovateľných údajov či technologických obmedzeniach daných v súčasnosti dostupnými technológiami.

Výsledkom bakalárskej práce je systém, ktorý spĺňa všetky body zadania práce. Navyše, nad rámec zadania, bolo navrhnutou a implementovanou architektúrou, umožnené schopnému užívateľovi jednoducho rozširovať stávajúcu funkcionalitu. V žiadnom prípade sa však nejedná o systém, v ktorom by nebolo miesto pre vylepšenia a rozšírenia priamo na úrovni systémového API. Vývoj na systéme bude určite naďalej pokračovať. Nasledujúce verzie systému by mohli napríklad rozširovať záber zbieraných údajov, o detailné informácie o stlačených klávesoch - nie z pohľadu sémantického významu sekvencie znakov. Ďalším možným rozšírením by mohla byť aj multiplatformná podpora.

Skúsenosti z pohľadu autora, ako programátora, sa vyvíjali hlavne v rovine spolupráce nízkoúrovňového kódu s kódom spravovaným behovým prostredím. Skúsenosti pri návrhu a hľadaní komunikačného kanálu, medzi klientom a serverom a implementácia serverovej časti, prehĺbili znalosti a skúsenosti s návrhom systému s použitím jazyka UML.

Literatúra

[1] *Zákon č. 262/2006 Sb. [on-line]*, [cit. 2009-01-05],
URL: <http://business.center.cz/business/pravo/zakony/zakonik-prace/>

[2] *Listina základných práv a slobôd [on-line]*, [cit. 2009-01-06],
URL: <http://www.psp.cz/docs/laws/listina.html>

[3] *Európsky dohovor o ochrane ľudských práv a základných slobôd [on-line]*, [cit. 2009-01-07],
URL: <http://www.lexisnexis-online.cz/umluva-o-ochrane-lidskych-prav-a-svobod.html>

[4] *Windows API [on-line]*, [cit. 2009-01-25],
URL: <http://msdn.microsoft.com/en-us/library/aa383723(VS.85).aspx>

[5] ROBINSON, Simon , et al. *C# Programujeme profesionálně* . Brno : Computer Press, 2003. 1160 s. ISBN 80-251-0085-5.

[6] KRUGLINSKI, David J., SHEPHERD, George, WINGO, Scot. *Programujeme v Microsoft Visual C++* . Brno : Computer Press, 2000. 1046 s. ISBN 8072263625.

[7] *Platform Invoke Tutorial [on-line]*, [cit. 2009-01-25],
URL: <http://msdn.microsoft.com/en-us/library/aa288468.aspx>

[8] *Hooks [on-line]*, [cit. 2009-01-25],
URL: <http://msdn.microsoft.com/en-us/library/ms632589(VS.85).aspx>

[9] KAČMÁŘ, Dalibor. *Programujeme .NET aplikace ve Visual Studiu .NET*. Brno : Computer Press, 2004. 344 s. ISBN 8072265695.

[10] *SOAP Version 1.2 (Second Edition) [on-line]*, [cit. 2009-01-28],
URL: <http://www.w3.org/TR/soap12-part1/>

[11] *XML Encryption Syntax and Processing [on-line]*, [cit. 2009-01-28],
URL: <http://www.w3.org/TR/xmlenc-core/>

[12] *XML Signature Syntax and Processing (Second Edition) [on-line]*, [cit. 2009-01-28],
URL: <http://www.w3.org/TR/xmldsig-core/>

[13] *Triple DES [on-line]*, [cit. 2009-01-28],
URL: < http://en.wikipedia.org/wiki/Triple_DES>

Zoznam príloh

Príloha 1: Inštalačný manuál,

Príloha 2: Práca s webovým rozhraním,

Príloha 3: CD obsahujúce manuály v elektronickej podobe, inštalačný program a zdrojové kódy.

Príloha 1

Inštalačný manuál

***Poznámka**: V nasledujúcom texte bude spomínaný demonštračný webový portál. Jeho URL adresa je v dobe odovzdávania bakalárskej práce* http://patsystem.aspone.cz/*. Portál slúži pre jednoduchšie testovanie hodnotiacou osobou. Autor práce si preto vyhradzuje právo odstaviť portál po vypršaní hodnotiaceho obdobia.*

Minimálne požiadavky

- *Klient* – OS Windows XP a vyššie, .NET Framework 2.0 (je súčasť inštalačného balíka), internetové pripojenie (potrebné na zasielanie údajov na server),
- *Server* – OS Windows XP a vyššie, IIS, .NET Framework 2.0, Microsoft SQL Server 2000 a vyššie (skript na vytvorenie databázy dostupný vo verzii pre SQL Server 2005 a 2008).

Postup inštalácie

Inštalácia serverovej časti

Najjednoduchší spôsob ako systém odskúšať, je využiť pripravený demonštračný web na adrese http://patsystem.aspone.cz.

V prípade, že nechceme pripravený portál používať alebo nie je momentálne dostupný, máme na priloženom nosiči k dispozícii samorozbaľovací archív obsahujúci súbory portálu. Obsah archívu musí byť rozbalený do koreňového adresára, ktorý IIS server používa ako zdroj webovej stránky.

V zložke sa spoločne s archívom nachádza aj priečinok, obsahujúci skript pre vytvorenie databázovej štruktúry a naplnenie prvotnými dátami. Skript je dostupný vo verzii pre SQL Server 2005 a 2008. Po nainštalovaní serveru a vytvorení databázy, je nakoniec potrebné upraviť konfiguračný súbor webovej aplikácie tak, aby používala vytvorenú databázu.

Registrácia nového klienta

Po prístupe na webový portál sa cez položku menu '*Register*' dostávame k registračnému formuláru. Úlohou registrácie je vytvorenie nového účtu, cez ktorý budú spravované informácie o sledovaných staniciach. Výsledkom registrácie je vygenerovaný unikátny identifikátor, ktorý sa zadáva pri inštalácii klientskej aplikácie na monitorované stanice.

Registračný formulár vyzerá nasledovne:

Obr. 1: *On-line registračný formulár*

Po úspešnej registrácii sú zobrazené nasledujúce údaje:

- *GUID* – identifikátor registrovaného klienta,
- *Service URL* – URL koncového bodu brány.

! Údaje sú potrebné pre úspešne nainštalovanie klientskej aplikácie, preto by mali byť starostlivo uchované. Nie je možné dostať sa k údajom neskôr, bez administrátorského prístupu.

Inštalácia klientskej aplikácie

Aplikáciu je možné nainštalovať dvoma spôsobmi:

1. Spustiť inštalačný balíček z priloženého CD nosiča. Inštalátor klientskej aplikácie je dostupný ako klasický EXE súbor poprípade MSI inštalátor.
2. Inštalačný balík je možné získať na webovom portály http://patsystem.aspone.cz. Odkaz pre stiahnutie inštalátoru sa zobrazí po registrácii nového klienta.

Sprievodca inštaláciou klientskej aplikácie prechádza niekoľkými krokmi:

1. Potvrdenie súhlasu s licenčnými podmienkami,
2. Zadávanie:
    a. *GUID* – unikátneho identifikátora klienta, získaného pri registrácii na portály,
    b. *Login* – užívateľské meno zvolené pri registrácii,
    c. *Password* – heslo zvolené pri registrácii,
    d. *Monitoree GUID* –identifikátor existujúceho sledovaného užívateľa.
3. Zadávanie:
    a. *Connection check period* – časový interval medzi kontrolami aktívneho spojenia so serverom,
    b. *Service URL* – adresa brány bežiacej na strane serveru, po zaregistrovaní je aktuálna adresa zobrazená v súhrne registrácie (v prípade použitia demonštračného webu je to http://patsystem.aspone.cz/PATServiceGate.asmx)
    c. *Local data store* – cesta k lokálnemu úložisku dát – koreňový adresár.
4. Výber lokality pre inštaláciu samotného klienta.
5. Dokončenie inštalácie.
6. Reštartovanie systému. Pokiaľ sa systém po ukončení inštalácie nereštartuje, je potrebné klienta spustiť ručne z cieľového adresára.

Od chvíle, kedy sa v systémovej lište vedľa ukazovateľa hodín objaví ikonka klientskej aplikácie (stĺpcový graf), je aktivita užívateľa logovaná a v prípade aktívneho spojenia odosielaná na server.

Príloha 2

Práca s webovým rozhraním

Webové rozhranie slúži ako administračný bod na prehliadanie informácií zozbieraných pri sledovaní užívateľov. Slúži zároveň na správu týchto údajov. Administrácia funguje v dvoch režimoch – užívateľskom a administrátorskom.

Administrátorsky mód

V prípade, že sa do systému prihlási užívateľ ako administrátor (užívateľské meno 'administrator' a heslo 'admin' pre demonštračný web a web inštalovaný zo samorozbaľovacieho archívu), rozhranie sprístupní administrátorské funkcie.

Rozšírená funkcionalita predstavuje rozhranie pre správu spoločností (*Administration → Companies*), užívateľov (*Administration → Users*), SQL dotazov používaných pre získavanie údajov z databázy (*Administration → Queries*) či rozhranie pre správu lokalizačných reťazcov, umožňujúcich preklad rozhrania do ľubovoľného jazyka (*Administration → Localization*).

Samozrejmosťou je prítomnosť všetkých funkcií dostupných pre bežného užívateľa. Viac o nich v ďalšej časti tohto manuálu.

Užívateľský mód

Ide o základný mód administračného rozhrania. Každý klient zaregistrovaný do systému získava registráciou právo do rozhrania pristupovať. Po prihlásení do systému sú pre užívateľa viditeľné záložky '*Records*', '*Administration*' a '*About*'. Na záložke '*About*', ako už názov napovedá, je možné nájsť základné informácie o práci s rozhraním, v podaní tohto krátkeho manuálu.

Záložka 'Administration'

Bežný užívateľ môže v tejto sekcii spravovať kategórie, do ktorých sa aktívne aplikácie jeho sledovaných staníc rozdeľujú.

Každá kategória sa definuje tak, že sa určia stĺpce v databáze, ktoré sa majú prehliadať na výskyt hľadaného textu. Text môže predstavovať názov aplikácie, text okna, ktorý je zobrazený ak je aplikácia aktívna, cestu k modulu spúšťajúcemu aplikáciu a pod. Vyhľadávať sa môže aj na základe viacerých textových hodnôt, oddelených znakom '*;*'. Pri zobrazovaní aplikácií určitej kategórie sú potom zobrazené všetky také aplikácie, ktoré vo vybraných stĺpcoch obsahujú hľadaný text.

Pre ukážku špecifikácie kategórie prejdite prosím do sekcie '*Administration → Categories*' a zeditujte niektorú z kategórií pomocou tlačidla s ikonou ceruzky ( ).

Sekcia 'Records'

V tejto časti administračného rozhrania užívateľ, ako správca, môže nájsť kompletný prehľad o aktivite jednotlivých užívateľov. V zozname užívateľov, '*Monitorees*', sa pochopiteľne zobrazujú iba užívatelia, ktorí sú sledovaní klientskou aplikáciou inštalovanou s použitím registračných údajov aktuálneho užívateľa – klienta.

Detail sledovaného užívateľa je možné zobraziť pomocou tlačidla s ikonou lupy (). Ikonou krížika () sa zmažú všetky údaje konkrétneho užívateľa. V zozname je okrem názvu účtu sledovaného užívateľa uvedený tiež dátum, kedy boli naposledy serverom od sledovanej stanice prijaté dáta.

Detail sledovaného užívateľa obsahuje tri záložky. Na záložke '*General*' je v tejto úvodnej verzii uvedená stručná informácia o dátume a čase, IP adrese a GUID identifikátore poslednej komunikácie užívateľa so serverom.

Záložka '*Activity*' obsahuje všetky aplikácie aktivované počas doby sledovania. Nad samotným zoznamom sa nachádza filter, ktorým je možné vyfiltrovať zobrazované aplikácie podľa kategórie, dátumu aktivácie, a iných vlastností. V zozname sa dajú položky opäť mazať ako v predchádzajúcom príklade. Pri zobrazení detailu aplikácie () je možné prehliadať si štatistické údaje o danej aplikácii. Dostupné sú informácie o využití klávesnice, polohovacieho zariadenia či systémových prostriedkov v podobe grafov.

Poslednou v detailoch užívateľa je záložka '*Statistics*'. V podobe koláčových grafov sa tu zobrazujú informácie o pomere aktivity aplikácií patriacich do definovaných kategórií. Podobne ako v zozname všetkých aplikácií, aj tu je možnosť pomocou filtru určiť záber zobrazovaných údajov.