Contents

1 Opis problémovej oblasti

Cieľom aplikácie je zber a zaznamenávanie pracovných aktivít používateľa za účelom analýzy správania sa používateľa v kontexte SW nástrojov, ktoré používa, ale aj v kontexte údajov (dokumentov, zdrojových kódov, Web, ...), s ktorými pracuje. Výsledkom analýzy pracovných aktivít používateľa je model používateľa ako súbor charakteristických čŕt používateľa a jeho správania sa v čase a v pracovnom prostredí.

Pracovná aktivita je množina činností používateľa za určitú dobu, v kontexte údajov s ktorými pracuje, pracovného prostredia resp. nástroja a prostriedkov (HW aj SW) počítača. Aktivita je ohraničená dvoma významnými udalosťami, z ktorých pracovná aktivita nadobúda logický význam.

Udalosť je činnosť používateľa na GUI, ktorá vyvolá nejaký proces. Môže byť

- významná (míľnik), spúšťajúca zber a odoslanie balíka údajov o pracovnej aktivite
- menej významná (akcia), predstavujúca bežnú činnosť v pracovnej aktivite používateľa

Pri ukončení pracovnej aktivity sa vytvorí a balíček, ktorý sa skladá z

- míľnika aktivity
- akcií počas aktivity
- statických údajov – kontextu prace, čiže SW s ktorým robí, obsah (pri zabezpečení ochrany súkromia) s ktorým robí, biometria, stav aplikácii, využitie HW prostriedkov

Zber používateľských pracovných aktivít prebieha a je spracovaný v centrálnej aplikácii na OS pracovnej stanice používateľa. Zber rôznych typov informácií a udalostí z rôznych zdrojoch (pracovných prostredí) zabezpečujú komponenty aplikácie. Niektoré udalosti prezentujú ako míľniky v pracovnej aktivite.

Komponenty, ktoré zberajú a pripravujú stavové údaje, prípadne aj iné komponenty, budú mať možnosť reagovať na míľniky. To, pri akej udalosti dôjde k uzavretiu pracovnej aktivity a teda ktoré udalosti budú míľnikmi, bude možné konfigurovať a tým pádom riadiť granularitu odosielania kontextových údajov.

2 Analýza

2.1 Zdroje informácií

Základnú množinu identifikovaných zdrojov .NET vývojára tvoria:

- MS Visual Studio 2010 – sledovanie kontextu programovania a zostavovania programov v IDE vo forme extension. Údaje sa odosielajú do UACA cez WCF named pipes.
- MS Office 2010 OneNote – sledovanie aktivít v kontexte otvorených poznámkových blokov cez COM rozhranie
- Biometria – sledovanie biometrických údajov používateľa prostredníctvom použitia klávesnice a myši
- Application Activity and Windows – sledovanie stavu spustených aplikácií a ich okien
- HW Resources – sledovanie HW prostriedkov ako CPU a operačná pamäť a ich využitie procesmi spustených aplikácií cez systémové rozhranie Win32 API
- MS Outlook and Lync 2010 – sledovanie aktivít v kontexte vytvárania a čítania elektronickej komunikácie vo forme extensions. Údaje sa odosielajú do UACA cez WCF named pipes.
- Web Browser Extension – rozšírenie vo forme pluginu do prehliadača, kde sa sledujú aktivity spojené s browsovaním v kontexte načítaných web zdrojov. Údaje sa odosielajú do UACA cez WCF named pipes.

Tiež možno pre vytváranie používateľských modelov použiť údaje z iných systémov:

- Communication Activities and Satus – Outlook a Lync komunikácia
    - status používateľa (aktívny, neaktívny, zaneprázdnený, offline)
    - údaje odosielateľ, prijímatelia, čas, komunikačná sekvencia (ID)
- Favorites
    - bookmarknute stránky
    - navštívené stránky
    - hodnotenia, recenzie

2.1.1 MS OneNote

S obsahom OneNote dokumentov sa nepracuje priamo, ale prostredníctvom volaní rozhrania aplikácie použitím MS OneNote 14 COM API, ktoré môže byť použité priamo v komponente centrálnej klientskej aplikácii pre sledovanie aktivít. Knižnica obsahuje metódy pre volanie aplikačného rozhrania OneNote aplikácie, ktorá musí byť nainštalovaná na danom počítači.

http://msdn.microsoft.com/en-us/library/ff925947.aspx

Vieme zaznamenávať nasledovné údaje

Údaj	Funkcia	Typ	Spôsob získania
	trieda Windows		v časových intervaloch
Počet okien aplikácie	Count	ulong	
Aktívne okno aplikácie	CurrentWindow	Window	
Zoznam spustených okien	Windows[index]	Window	
	trieda Window		v časových intervaloch

Či je dané okno aktívne – posledné aktívne okno zo spustených inštancií	Active	bool	
OneNote aplikácia prislúchajúca oknu	Application	Application	
Lokalita ukotvenia okna – štandardné, vľavo, vpravo, dolu, hore	DockedLocation	DockedLocation	
Či je okno v režime plného zobrazenia (minimálne UI)	FullPageView	bool	
Či je okno v režime zobrazenia ako bočná poznámka	SideNote	bool	
Handler na okno – treba získať koreňové okno	WindowHandle	ulong	
	Window		v časových intervaloch
ID zobrazenej stránky	CurrentPageId	string	
ID zobrazenej sekcie	CurrentSectionId	string	
ID zobrazenej skupiny sekcií	CurrentSectionGroupId	string	
ID otvoreného bloku	CurrentNotebookId	string	
Na základe Id možno získať z metaúdajov notebooku meno a iné údaje o stránke, sekcii, ...			

2.1.2 Application Activities and Resource Utilization

Sledovanie stavu aplikácií resp. aktivít nad rozhraním (oknami) aplikácií možno sledovať prostredníctvom Win32 API. Pomocou tohto systémového rozhrania možno sledovať aj hardvérové prostriedky využívané aplikáciami.

Údaj	Funkcia	Typ	Spôsob získania
Stav okna – minimalizované, maximalizované, normálne, schované, ...	`GetWindowPlacement`	WINDOWPLACEMENT	v časových intervaloch
Získanie nadradeného okna aplikácie	`GetParent`	IntPtr	v časových intervaloch
Získanie textu v hlavičke okna (caption)	`GetWindowText`	string	v časových intervaloch
Či je okno minimalizované	`IsIconic`	bool	v časových intervaloch
Indikácia polohy kurzora vzhľadom na dané okno, teda či kurzor ukazuje na okno aplikácie	`GetCursorPos a WindowFromPoint`	bool	v časových intervaloch
Indikácia, či je okno na popredí resp. či má fokus	`GetForegroundWindow`	bool	v časových intervaloch
Enumerácia procesov	EnumProcesses		v časových intervaloch
Procesorový čas využívaný	GetProcessTimes		v časových

procesom			intervaloch
Operačná pamäť využívaná procesom	GetProcessMemoryInfo		v časových intervaloch

2.1.3 Biometry

Sledovanie použitia hardvérových rozhraní ako je myš a klávesnica. Údaje o používaní týchto prostriedkov na spodnej vrstve možno získať prostredníctvom Win32 API.

Údaj	Funkcia	Typ	Spôsob získania
Myš...			
Poloha -> intenzita pohybu	WM* mouse-events		event /suma za interval
Klik -> intenzita klikania	WM* mouse-events		event /suma za interval
Scroll -> intenzita pouzitia	WM* mouse-events		event /suma za interval
Klávesnica...			
Tlačidlo dole	WM,WK* keyb.events		event
Tlačidlo hore	WM,WK* keyb.events		event
• **Historia stlacenych klavesov,** • **Intenzita stlacania klaves** • **Intervaly medzi stlacenim vybranych klaves**			postupnost klavesov s casovou znackou, alebo len suma za interval

Možno filtrovať vybrané udalosti (iba myšie kliky, iba klávesnica bez modofikačných klávesov atd) a / alebo udalosti viazané na konkrétnu bežiacu aplikáciu (napríklad len aktivita v okne aplikácie Word)

2.1.4 Web Activity

Sledovanie browsovania web zdrojov v prehliadači prostredníctvom plugin do prehliadača. Využitie javascript a prostriedkov špecifických pre daný prehliadač na integráciu funkcionality do prehliadača. Zaznamenáva tieto udalosti:

- návšteva stránky
  - zadanie URL
  - otvorenie / prepnutie okna
  - otvorenie / prepnutie tabu
  - kliknutie na bookmark
  - presmerovanie cez link
- uloženie obsahu
  - uloženie stránky
  - stiahnutie súboru
- udalosti uchovávajú informácie
  - používateľ
  - URL
  - čas

2.1.5 MS Visual Studio

Aktivity používateľa a kontext, v ktorom sa tieto aktivity vykonávajú, sa sledujú pomocou rozšírenia Visual Studia. Nasledujúca kapitola je bližšie popísaná v týchto dokumentoch:

- *Analýza možností sledovania vybraných aktivít používateľa v IDE*
- *Používatelia – check out elementov kódu*

Nasledujúca tabuľka zhŕňa niektoré aktivity, ktoré je takto možné sledovať.

Aktivita	Spôsob získania	Ešte preskúmať
Písanie do kódu v okne kódu	event	
Stláčanie kláves v okne kódu	event	
Udalosti myši v okne kódu	event	
Vyberanie kódu	event	
Zmena viditeľnosti kódu – posun, približovanie...	event	
Build	event	
Refaktor a iné VS príkazy	event	
Zapnutie/vypnutie/prepnutie VS	event / v časových intervaloch	
Otvorenie/zatvorenie/uloženie súboru	event	
Otvorenie/zatvorenie solution	event	
Debug	event	X
Check-in (commit)	event	X
Vyhľadávanie	event	X
Pridanie/odstránenie súboru do/z projektu	event?	X

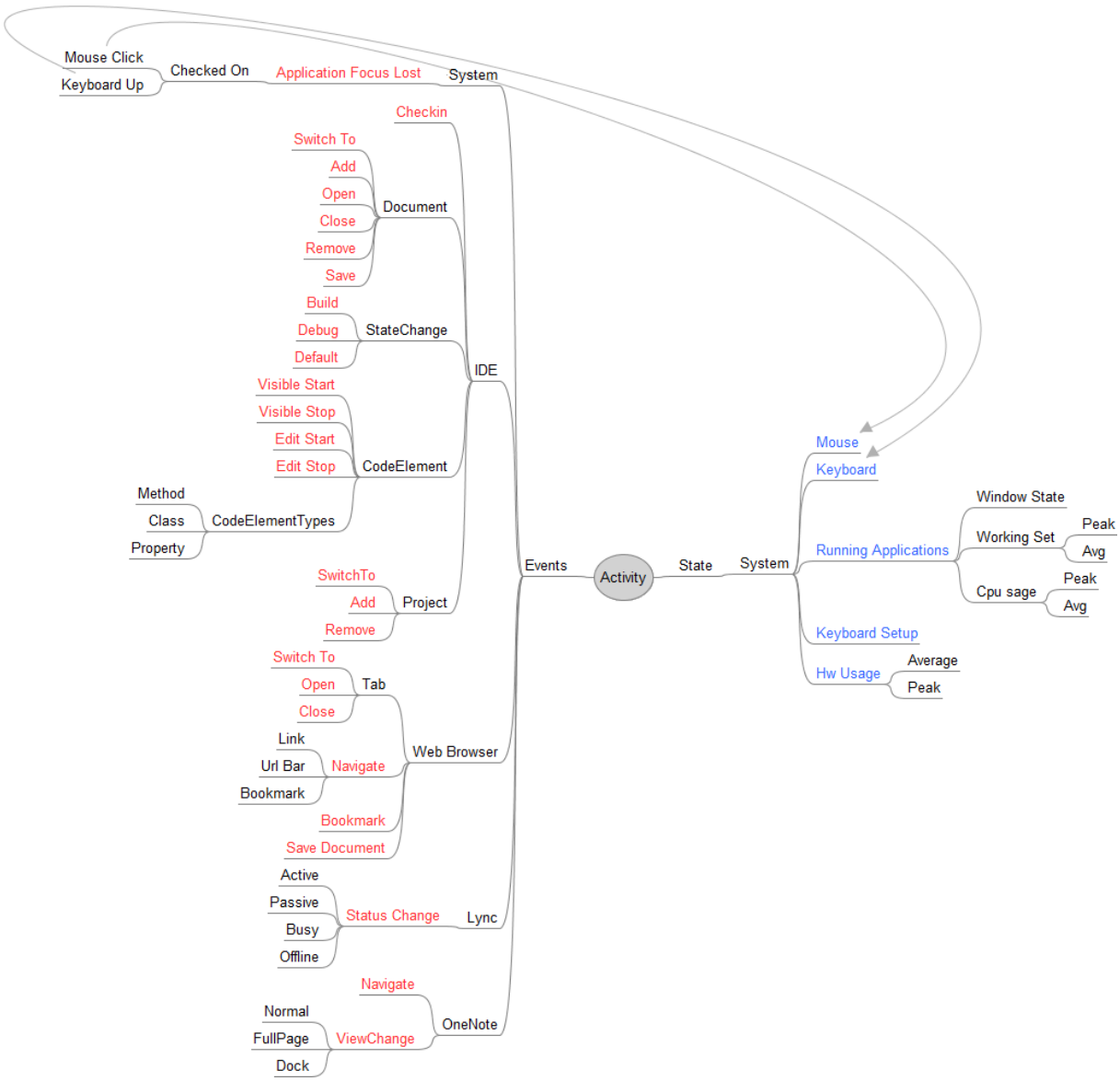
Nasledujúca tabuľka zobrazuje vybraný kontext, ktorý je možné zistiť.

Údaj	Funkcia
aktívny súbor	dte.ActiveDocument
aktívny projekt	dte.ActiveDocument.ProjectItem.ContainingProject
aktívna solution	dte.Solution
projekty v solution	dte.Solution.Projects
element kódu na konkrétnej pozícií v kóde	z AST
elementy kódu viditeľné vo výreze okna kódu	z AST
pozícia myši v kóde	
pozícia kurzoru klávesnice v kóde	
aktuálny posun (scroll)	
aktuálny zoom	
check out na elementoch kódu	TFS api

3 Špecifikácia

3.1 Sledované udalosti a stavy

Na nasledujúcom obrázku sú znázornené udalosti a stavy, ktoré sa budú sledovať. Udalosti aj stavy sú rozdelené do skupín podľa zdrojov.



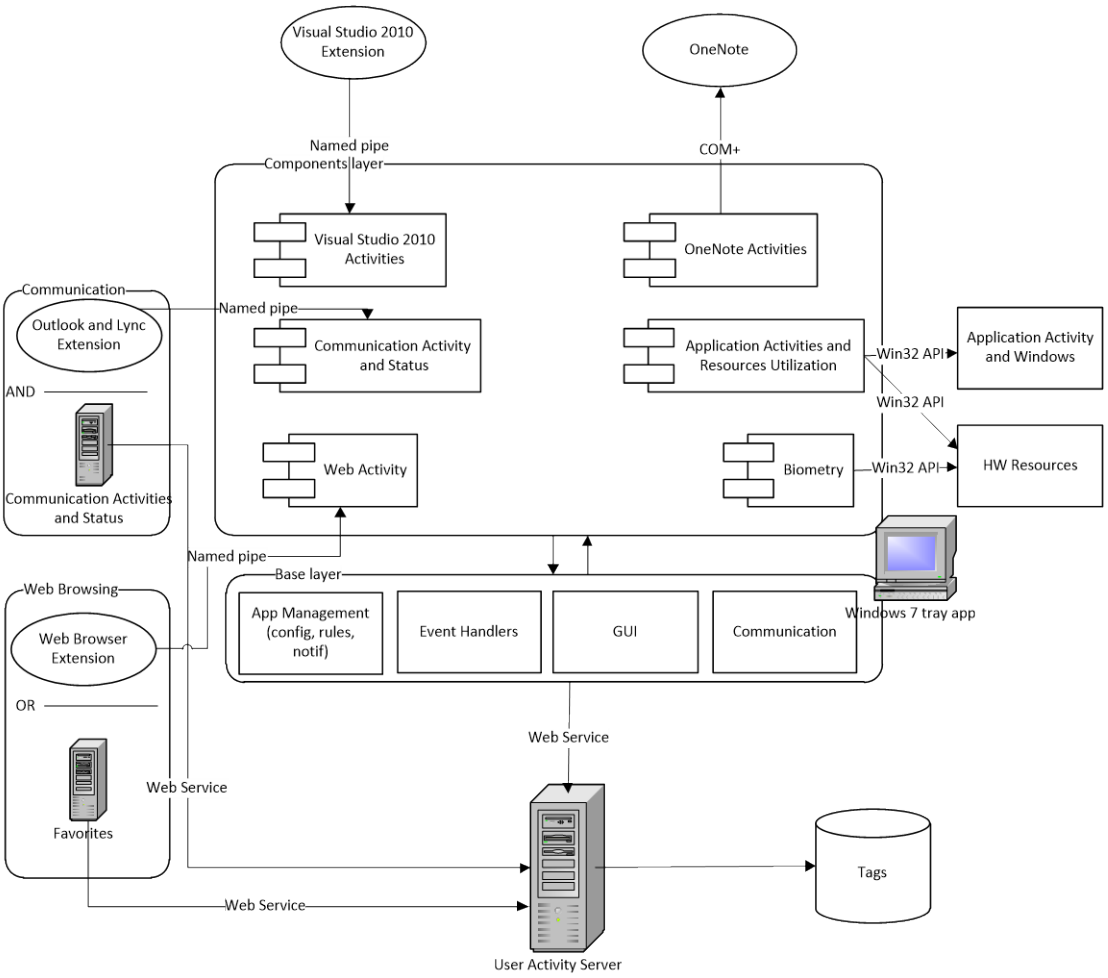
Obrázok 1 Sledované udalosti a stavy

3.2 Koncept systému

Systém pre sledovanie aktivít používateľov pozostáva z viacerých vrstiev:

- zdroje informácií – SW prostredia resp. nástroje v ktorých používateľ pracuje a zásuvné moduly do týchto SW nástrojov, pomocou ktorých sa zberajú informácie o aktivitách používateľa

- UACA (User Activity Client Application) – aplikácia bežiaca na klientskej pracovnej stanici, ktorá predstavuje centralizované miesto pre zber a odosielanie informácií o aktivitách používateľa
    - báza aplikácie – zabezpečuje komunikáciu so serverovou vrstvou, používateľské rozhranie, konfiguráciu zberu údajov, vyrovnávaciu a ochrannú medzipamäť a tiež základnú podporu pre riadenie (systémových) udalostí a ich poskytnutie časti komponentov
    - časť komponentov – jednotlivé komponenty zabezpečujú komunikáciu s vrstvou zdrojov informácií a prvotné spracovanie informácií o pracovných aktivitách (z pohľadu redukcie množstva „surových“ údajov, zabezpečenie ochrany súkromia používateľa)
- User Activity Server – poskytuje služby pre zber a uchovávanie balíkov záznamov o pracovných aktivitách jednotlivých používateľoch a pracovných staniciach. Uchováva informácie v istej miere spracovania a poskytuje ich vrstve profilovania používateľov
- profilovanie používateľov – systém, ktorý analyzuje údaje z User Activity Server a vytvára profily používateľov. Tieto profily poskytuje ako komplexné informácie pre prácu iných systémov ako inteligentné personalizované vyhľadávanie informácií.



Obrázok 2 Komponenty *ActivityWatcher*

3.3 Predpokladané komponenty UACA

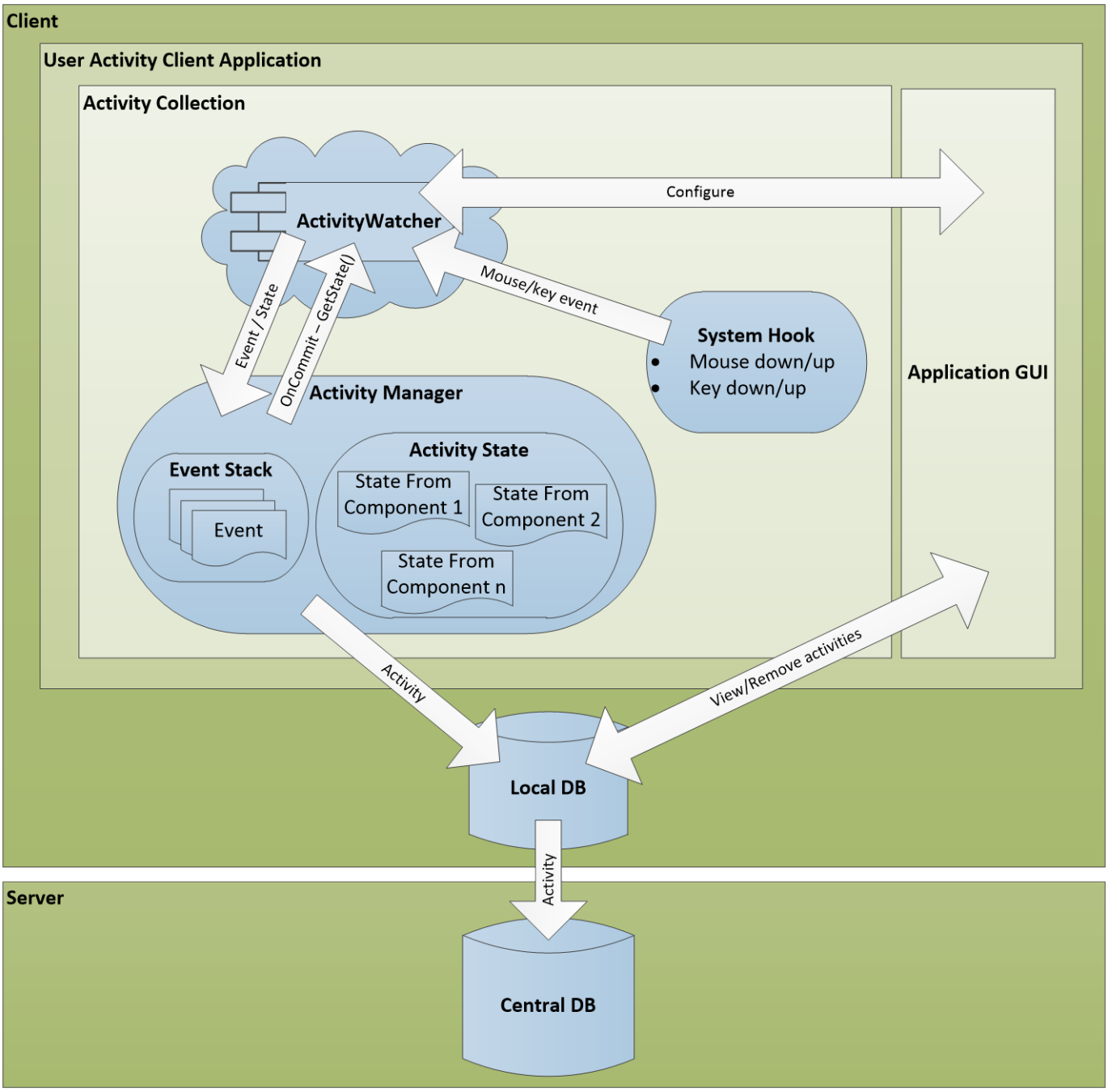
Predpokladaná množina komponentov *ActivityWatcher*, vychádzajúc zo zdrojov informácií, je nasledovná:

- Visual Studio 2010 Activities – komponent poskytuje komunikačné rozhranie prostriedkami .NET WCF Named Pipes, zabezpečujúce komunikáciu s extension vo VS v rámci OS. VS extension ako klientská časť komponentu ma za úlohu zachytávanie udalostí a informácií pri práci s VS (prepínanie v oknách, aktuálny projekt, zostavovanie, práca s TFS, činnosti pri kódovaní...) a odosielanie cez toto rozhranie komponentu v UACA. Komponent zachytáva, zaznamenáva a spracúva tieto informácie (či už nízko úrovňové alebo kontextové) a vytvára z nich informácie o aktivitách používateľa.
- OneNoteActivities – komponent pozostáva z volaní COM rozhrania na aplikáciu MS Office 2010 OneNote. Prostredníctvom tohto rozhrania sleduje aktívne inštancie aplikácie a získava informácie o aktuálne otvorenom obsahu. Vzhľadom na COM rozhranie treba získavanie informácií riadiť udalosťami alebo v časových intervaloch na strane UACA, COM rozhranie neposkytuje prostriedky pre zachytávanie aktuálnych zmien resp. udalostí.
- Biometry – komponent zbiera biometrické informácie sledovaním HW rozhraní (predovšetkým myš a klávesnica), pričom zachytáva systémové eventy týchto zariadení. Informácie spracúva do komplexnejších celkov.
- Application Activities and Resource Utilization – komponent cez systémové rozhranie Win32 API zachytáva informácie o aplikáciách ako ich spúšťanie, zmeny stavov rozhrania ale aj využitie HW prostriedkov týmito aplikáciami.
- Communication Activity and Status – prostredníctvom inštalácie pluginov do MS Office 2010 Outlook a Lync klientov možno sledovať komunikačné aktivity používateľa, nie však obsah komunikácie a iné informácie, ktorými by bolo ohrozené súkromie iných komunikujúcich a ochrana údajov firmy. Aktivity môžu byť napr. : používateľ píše nejakú dobu mail alebo správu, pozerá poštu, plánuje stretnutia, zaznamenáva si úlohy, kontakty, ... Odosielaním týchto informácií cez named pipes do komponentu a ich ďalším spracovaním možno získať informácie o významnej súčasti pracovných aktivít používateľa.
- Web Activity – pluginy vo vybraných prehliadačoch poskytujú informácie o aktivitách spojených s prehliadaním web zdrojov, ako aj informácie o samotných web zdrojoch. Tieto informácie posielajú cez named pipes do komponentu, kde sa spracujú a vytvoria sa informácie o trvaní a intenzite prehliadania web zdrojov ako aj postupnosti prehliadania web zdrojov.

4 Návrh systému

4.1 Architektúra

Obrázok 2 zobrazuje celkovú architektúru systému.



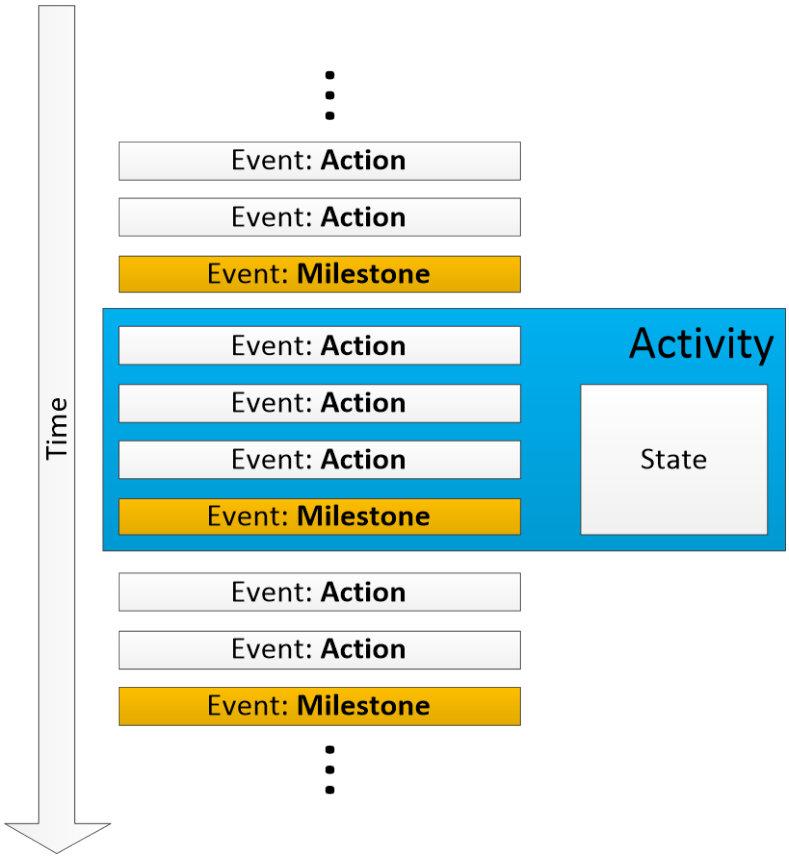
Obrázok 3 Celková architektúra systému

Zber aktivít jedného používateľa riadi na jeho klientskom počítači aplikácia *User Activity Client Application* (UACA), ktorá beží v jeho system tray. Aktivity sa ukladajú na strane klienta do lokálneho úložiska *LocalDB*, odkiaľ sú postupne prenášané do centrálneho úložiska na serveri *Central DB*.

4.1.1 Hlavné pojmy

Obrázok 3 znázorňuje vzťah hlavných pojmov. *Udalosti* vzťahujúce sa na činnosť používateľa sa zaznamenávajú podľa možností chronologicky tak, ako nastali. *Udalosťou* môže byť spustenie buildu v IDE, prepnutie stránky vo OneNote, prepnutie aktívnej aplikácie a iné. Každá postupnosť *udalostí* (*akcií*) zakončená významnou udalosťou (*míľnik*) sa nazýva *aktivita*. Každá *aktivita* obsahuje práve jeden *míľnik*. Každá *aktivita* môže obsahovať *stav* ako meta-informáciu o celej aktivite – sumárna informácia o celej aktivite, stav hardvérových prostriedkov na konci aktivity a iné.

Aktivitou sa teda rozumie postupnosť udalostí, ktoré viedli k nejakej „významnej“ *udalosti* (k *míľniku*) spolu so *stavom*, ktorý obsahuje ďalšie informácie o *aktivite*.



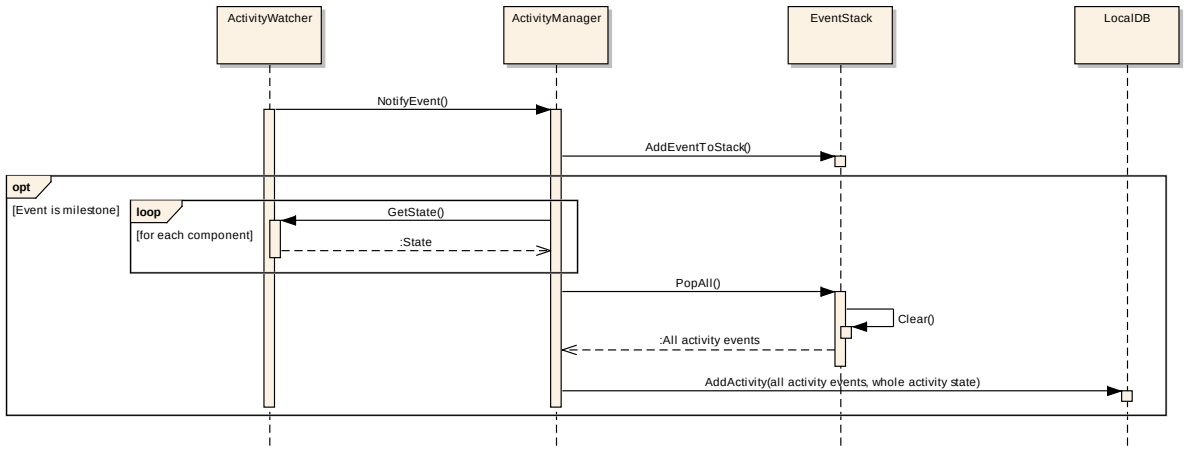
Obrázok 4 Aktivita obsahuje chronologicky usporiadaný zoznam akcií zakončený práve jedným míľnikom a stav ako sumárnu informáciu o celej aktivite (štatistické informácie, stav systému počas aktivity a iné)

4.1.2 Hlavné procesy v systéme

Obrázok 4 znázorňuje činnosť zberu aktivít. Časť UACA *Activity Collection* obsahuje IOC komponenty (*ActivityWatcher*) zodpovedné za notifikáciu o udalostiach vtedy, keď nastanú. Každý takýto komponent môže byť spojený s rozšírením aplikácie, ktorá je predmetom jeho sledovania – Visual Studio Extension, Firefox Extension a iné.

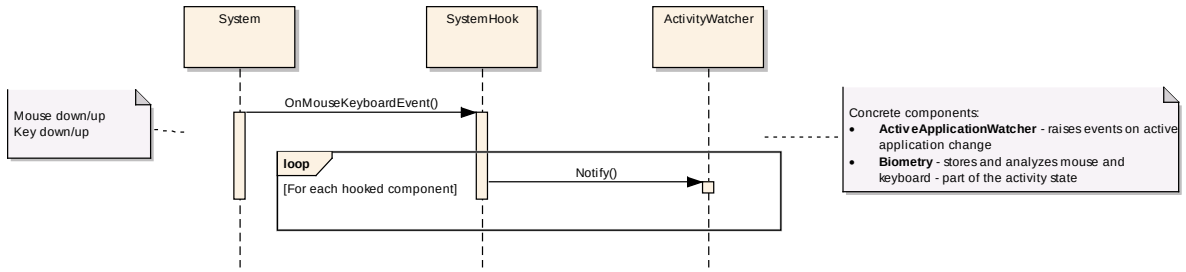
ActivityWatcher notifikuje o udalosti (akcia alebo míľnik) *ActivityManager*, ktorý túto udalosť vloží do zásobníka udalostí pre aktuálnu aktivitu – *EventStack*. Tento zásobník obsahuje udalosti len pre aktuálne zaznamenávanú aktivitu. Ak je udalosť míľnik, došlo k zakončeniu postupnosti udalostí pre túto aktivitu. V tom prípade sa od každého komponentu *ActivityWatcher* vyžiada stavová informácia, ktorých množina predstavuje celkový stav aktivity. Usporiadaný zoznam udalosti aktivity spolu so

stavom celej aktivity sú poslané do lokálneho úložiska (*LocalDB*). Zásobník udalostí *EventStack* sa vyprázdni a začína konštrukcia novej aktivity.



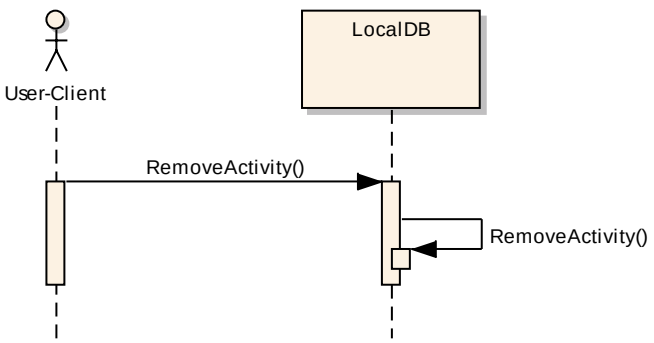
Obrázok 5 Zber aktivít

Komponenty *ActivityWatcher*, ktoré potrebujú byť notifikované o systémových udalostiach myši a/alebo klávesnice, sa môžu zaregistrovať do objektu *SystemHook*. Tento objekt sleduje globálne stláčanie tlačidiel myši a klávesnice v operačnom systéme a notifikuje o nich zaregistrované komponenty *ActivityWatcher*. (Obrázok 5)



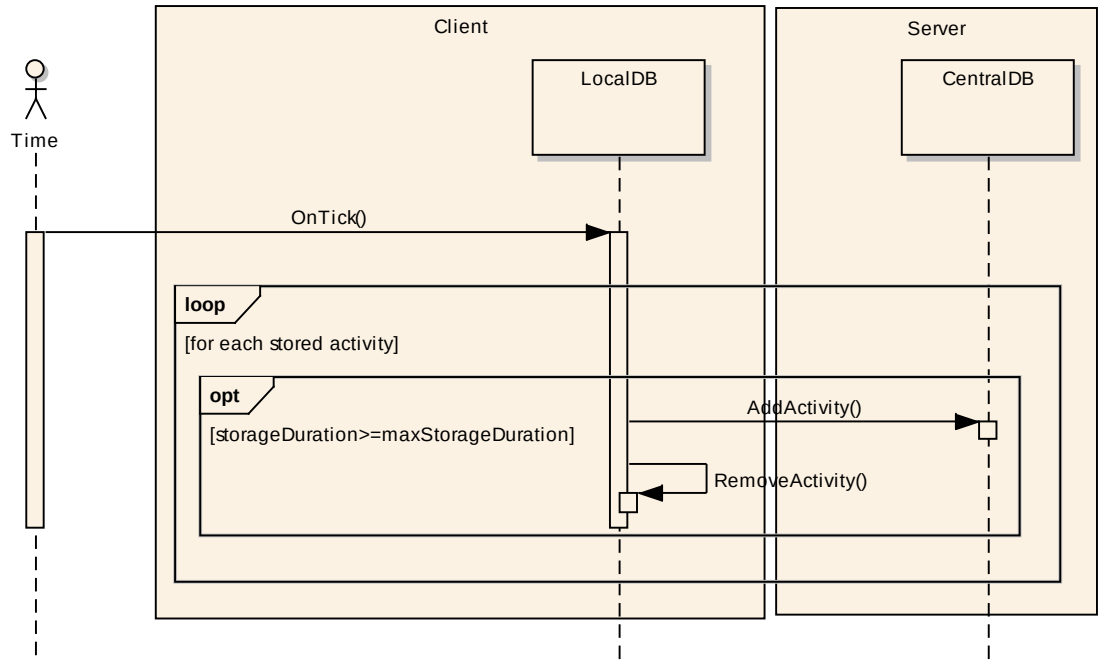
Obrázok 6 Sledovanie udalostí spojených so sledovaním myši a klávesnice

Aktivity v lokálnom úložisku (*Local DB*) si môže používateľ prezerať a mazať ešte pred ich poslaním do centrálneho úložiska na serveri (*Central DB)*.



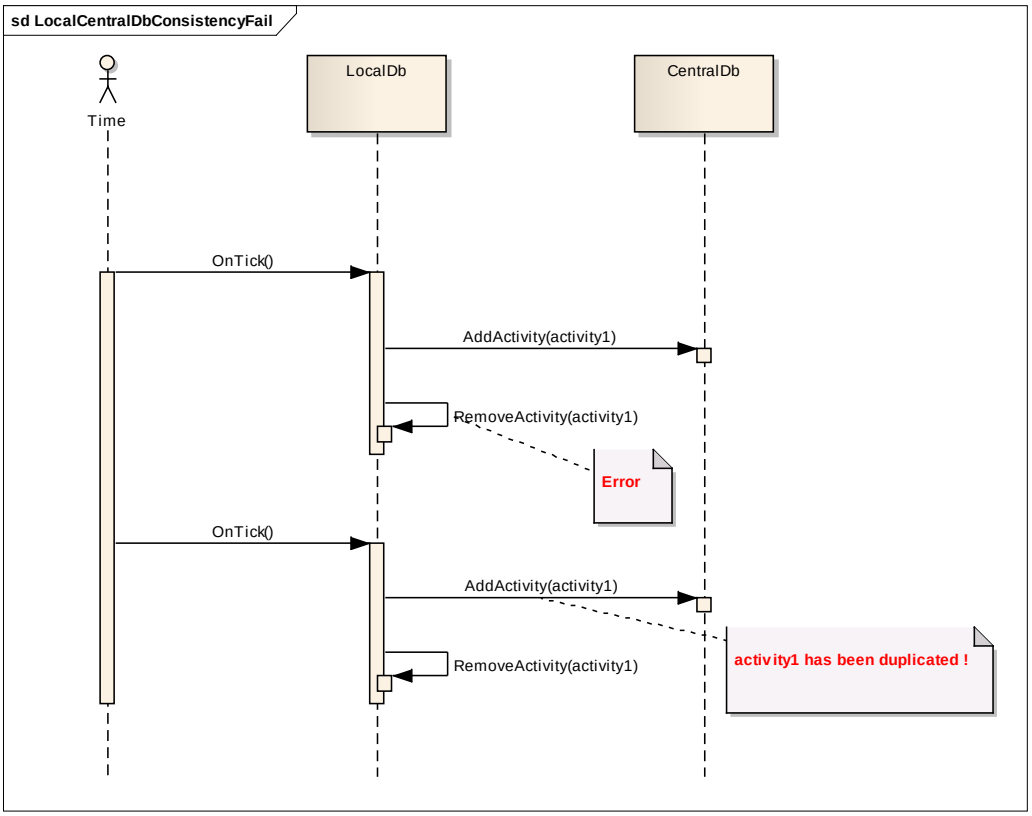
Obrázok 7 Vymazanie aktivity pred jej poslaním do centrálneho úložiska

Každá aktivita sa v lokálnom úložisku (*Local DB*) nachádza len určitú vymedzenú dobu. Po uplynutí tejto doby sa aktivita presúva do centrálneho úložiska na server (*Central DB*).



Obrázok 8 Aktivity sa po čase presunú do centrálneho úložiska

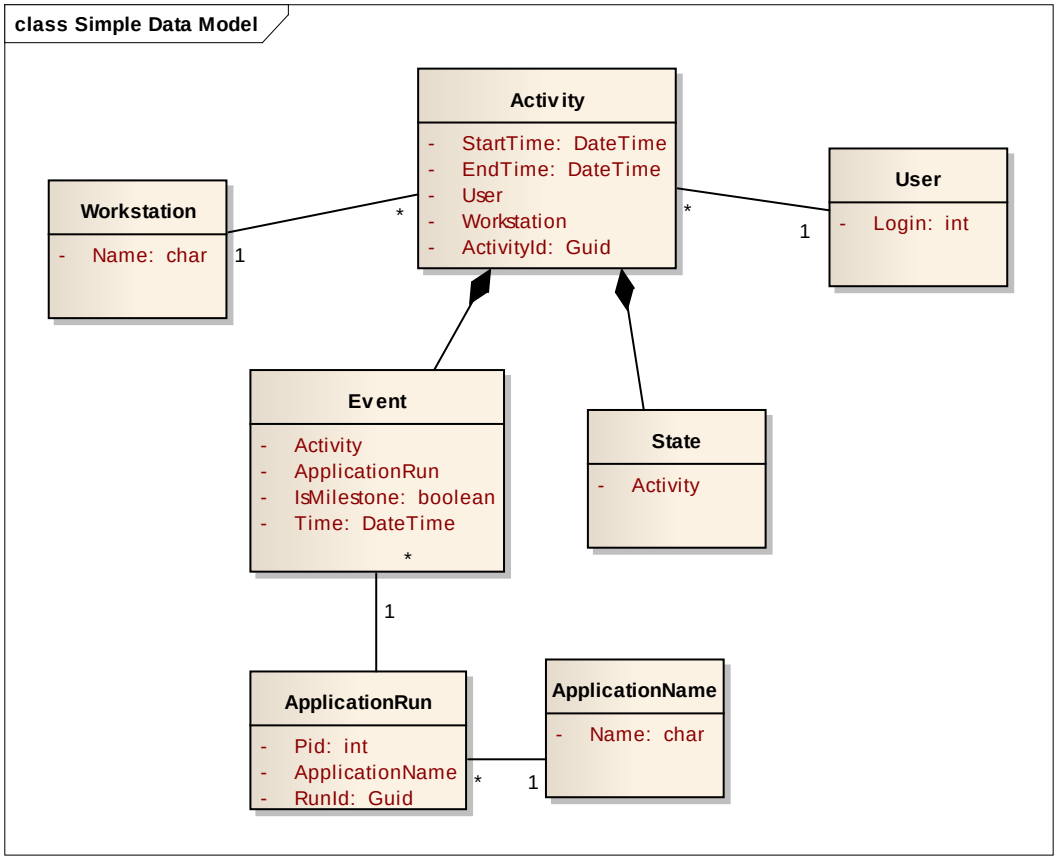
Pri odstraňovaní aktivity z lokálneho úložiska a jej ukladaní v centrálnom úložisku je nutné zabezpečiť konzistenciu medzi týmito úložiskami. T.j. nesmie sa napríklad stať, že sa aktivita uloží v centrálnom úložisku dva krát (viď nasledujúci obrázok)



Obrázok 9 Porušenie konzistencie centrálneho úložiska

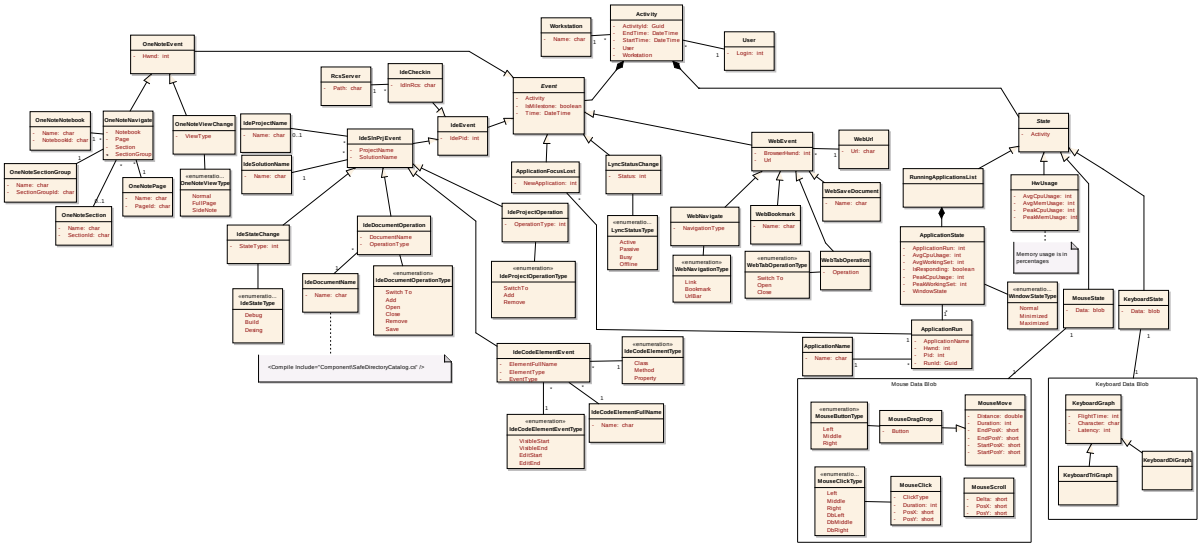
Tento typ problémov sa vo všeobecnosti rieši vykonaním operácií v rámci jednej distribuovanej transakcie. Problémom je ale skutočnosť, že v prípade realizácie lokálneho úložiska ako súborového systému, alebo neserverových DB riešení (napr. SQLite) nie sú distribuované transakcie podporované. Ak by sa vykonala operácia uloženia pred operáciou mazania, tak v prípade zlyhania operácie mazania by sa zmena centrálneho úložiska v rámci distribuovanej DB zrušila. Takto by sa vyriešil problém zduplikovanej aktivity. Môže ale nastať nový problém v prípade, že sa distribuovaná transakcia odvolá (angl. rollback) až po zmazaní aktivity z lokálneho úložiska (napríklad v prípade prekročenia max. času). V takomto prípade sa daná aktivita stratí. Druhom možnosťou riešenia problému je zabezpečiť idempotentnosť uloženia aktivity v centrálnom úložisku. V takomto prípade sa druhý pokus o uloženie aktivity vyhodnotí ako duplicitný a odignoruje sa. Za týmto účelom je nutné pridať do aktivity identifikátor, podľa ktorého je možné globálne jednoznačne identifikovať aktivitu.

4.2 Db Model



Obrázok 10 Základný Db model

Na Obrázku 9 je znázornené jadro Db modelu. Ústrednou entitou je aktivita, ktorá zoskupuje udalosti vykonané medzi dvoma míľnikmi, ako aj stav sledovaného systému ku koncu aktivity. Každá udalosť prislúcha k behu aplikácie. Na nasledujúcom obrázku sa nachádza kompletný model, kde sú rozkreslené jednotlivé udalosti a stavy zdedené zo základných entít.



Obrázok 11 Kompletný Db model

Za zmienku stoja entity MouseState a KeyboradState, ktorých dáta sú uložené ako blob. Dôvodom je predpokladané príliš veľké množstvo údajov.

5 Návrh modulov UACA

5.1 Biometria

Modul pre zachytávanie udalostí generovaných vstupnými zariadeniami (klávesnica, myš), pričom udalosti sú vyvolané aktivitou používateľa a jeho prácou v OS Windows. Prostredníctvom knižnice Win32 sa sleduje množina nízkoúrovňových udalostí vstupných zariadení ako stlačenie/uvoľnenie kláves a taktiež aj pohyb, klik a otočenie kolieska myši. Takéto údaje je možné následne spracovať na komplexnejšie udalosti obsahujúce časovú informáciu, v špeciálnych prípadoch (pohyb myši) aj informácie o prejdenej vzdialenosti. Modul monitorujúci aktivitu používateľa pri používaní klávesnice a myši získava informácie z OS Windows s využitím funkcií Win32a teda nepotrebuje zásuvný modul, týmto modulom je samotná aplikácia.

5.1.1 Zber údajov - BiometryWatcher

Údaje o klávesnici a myši sú v tomto prípade získavané s použitím systémových prostriedkov Win32 reprezentovaných triedou SystemInputListener. Modul sleduje zmeny generované klávesnicou a myšou, pričom tieto zmeny následne posúva na konkrétny komponent.

Modul BiometryWatcher je logicky rozdelený na dva komponenty (KeyboardStateWatcher, MouseStateWatcher), pričom pre každý z nich sleduje zmeny generované klávesnicou, resp. myšou.

Tieto údaje sú pre komponenty poskytované vo forme udalostí:

- MouseChanged(Point, Message, Delta);
    - Point – pozícia myši (x,y)
    - Message – typ akcie (Win32.MouseMessages)
    - Delta – hodnota otočenia kolieska myši
- KeyboardChanged(Message, Key);
    - Message – typ akcie (Win32.KeyboardMessages)
    - Key – hodnota stlačenej klávesy (System.Windows.Input.Key)

Modul zbiera údaje počas kompletnej aktivity používateľa, pričom tieto dáta zasiela postupne v balíkoch. Požiadavka na odoslanie dát z modulu je vytvorená v prípade, že počas aktivity používateľa bol identifikovaní mílnik. Samotný modul negeneruje žiadne miľníky, takže sa spolieha na ostatné komponenty, ktoré rozdeľujú aktivitu používateľa na menšie sekvencie. Zozbierané dáta počas jednej sekvencie sa odosielajú do databázy vo formáte BLOB.

Šablóny a modely

Šablóny predstavujú charakteristický profil v rámci krátkodobých činností v kontexte pracovnej aktivity na nejakej aplikácii (vektory udalostí). Šablóny tvoriť v rámci UACA aplikácie a odosielať vždy pri uzatvorení pracovnej aktivity.

Model je množina šablón, z ktorých sa analyzujú nejaké ukazovatele a charakteristiky, napríklad priemerné hodnoty a štandardné odchýlky pre konkrétne udalosti. Modely sa tvoria na strane servera z jednotlivých šablón.

Úlohou komponentu je zber a spracovanie údajov o biometrii klávesnice a myši používateľa. Výstupom komponentu pre každú pracovnú aktivitu používateľa je stavová informácia vo forme biometrickej šablóny.

5.1.2 Komponent KeyboardStateWatcher

Komponent zachytáva a spracúva údaje generované klávesnicou do výstupnej štruktúry KeyboardStateBlobDto.

Trieda KeyboardStateBlobDto reprezentuje zoznamy latencií:

- Graphs – zoznam stlačených znakov
- DiGraphs - zoznam stlačených dvojíc znakov
- TriGraphs - zoznam stlačených trojíc znakov
- Shortcuts – zoznam klávesových skratiek

Každý zoznam v skutočnosti predstavuje **List<KeyboardGraphDto>(),** pričom KeyboardGraphDto obsahuje trojicu údajov:

- Character - hodnota stlačených znakov
- Latency -latencia stlčenia
- FlightTime - čas letu

Pre každý stlačený znak (s) na klávesnici sa vygenerujú dve udalosti, pričom prvá predstavuje čas stlačenia (T_p) a druhá, čas uvoľnenia klávesy (T_u). Pomocou týchto údajov sa pre znak vypočítajú 3 typy **latencií**:

- Doba stlačenia klávesu – Graf latencia (L_T)
- Digraf latencia (D_T) – dvojica kláves
- Trigraf latencia (T_T) – trojica kláves

Pre každý typ latencií sa súčasne vyráta aj čas letu stlačenia (F_T), ktorú predstavuje interval medzi stlačením rôznych kláves, takže zároveň určuj, či došlo k ich prekrývaniu.

Prostredníctvom nasledovných vzorcov budú vyjadrené časové hodnoty pre každý znak, takže celkový počet vyrátaných latencií.

$$L_T(s) = T_u(s) - T_p(s)$$

$$F_T(s) = T_p(s+1) - T_p(s)$$

$$D_T(s) = T_u(s) - T_p(s-1) = F_T(s-1) + L_T(s)$$

$$T_T(s) = T_u(s) - T_p(s-2) = F_T(s-2) + D_T(s)$$

Je nám známe, ktoré latencie boli vytvorené na základe odchytených udalostí z jedného procesu. Na základe procesov vieme potom povedať o akú aplikáciu išlo, čo nám poskytuje možnosť odlišovať latencie (štýl písania používateľa) v rôznych aplikáciách.

5.1.3 Komponent MouseStateWatcher

Komponent zachytáva a spracúva údaje generované myšou do výstupnej štruktúry MouseStateBlobDto.

Trieda MouseStateBlobDto reprezentuje zoznamy špecifických akcií myši:

- MouseMoves = new List<MouseMoveDto>();
    - zoznam pohybov
- MouseDragDrops = new List<MouseDragDropDto>();
    - zoznam akcií Drag&Drop
- MouseScrolls = new List<MouseScrollDto>();
    - zoznam pohybov kolieska myši
- MouseClicks = new List<MouseClickDto>();
    - zoznam všetkých typov stlačenia tlačidiel myši

MouseMoveDto:

- StartPos – počiatočná pozícia pohybu
- EndPos – koncová pozícia pohybu
- Duration – celkový čas pohybu
- Distance – skutočná prejdená vzdialenosť

MouseDragDropDto:

- StartPos – počiatočná pozícia pohybu
- EndPos – koncová pozícia pohybu
- Duration – celkový čas pohybu
- Distance – skutočná prejdená vzdialenosť
- MouseButton – typ stlačeného tlačidla { Left, Middle, Right }

MouseClickDto:

- Pos – pozícia myši
- Duration – dĺžka stlačenia tlačidla
- MouseClickType – typ akcie {Left, Middle, Right, DbLeft, DbMiddle, DbRight}

MouseScrollDto:

- Pos – pozícia myši
- Duration – dĺžka stlačenia tlačidla
- Delta – hodnota otočenia kolieska myši

Akcie typu **DragAndDrop a MouseMove** (cielený pohyb) sú základom úspechu priebežného overovania pre myš. V prípade oboch typov akcií sú zaznamenané udalosti predstavujúce pohyb kurzora v konkrétnom smere zakončený stlačením, resp. uvoľnením tlačidla myši.

Pomocou nameraných údajov je možné vypočítať aj ďalšie hodnoty, ktoré sa možu použiť vo finálnej fáze vyhodnocovania. Medzi tie najpodstatnejšie údaje, ktoré je možné vyextrahovať patrí rýchlosť a smer pohybu. Taktiež je možné porovnať priamu (kalkulovanú) vzdalenosť a skutočnú prejdenú vzdialenosť. Taktiež ako aj pri latenciách môžeme rozlišovať aj zmeny charakteristík myši v rôznych aplikáciách.

5.1.4 Ohraničenia

Ohraničenia komponentov súvisia s vysokým množstvom generovaných údajov a preto vznikli obmedzenia na počet spracúvaných/odosielaných/ukladaných charakteristík. Obmedzenia z pohľadu ochrany súkromia a úniku citlivých informácií boli aplikované na údaje zachytené klávesnicou, takže pri odosielaní údajov sú dáta usporiadné podľa abecedy a neobsahujú časové údaj, pomocou, ktorých by blo možné vyskladať pôvodnú postupnosť znakov. Všetky konfigurovateľné nastavenia obmedzení modulu BiometryWatcher sú preddefinované v súbore **BiometrySettings.settings.**

Obmedzenia pre KeyboardStateWatcher

Komponent sleduje len tie najfrekventovanejšie kombinácie kláves, identifikované na základe frekvenčnej analýzy anglického a slovenského jazyka, respektíve najpoužívanejších klávesových skratiek.

Grafy: A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,R,Q,S,T,U,V,W,X,Y,Z,0,1,2,3,4,5,6,7,8,9

Digrafy:
TH,IN,EH,RE,AN,ES,ER,ON,AT,TO,ST,NT,OR,TI,EN,EA,ND,LE,NG,ED,OF,CO,IS,AR,ET,PR,OV,PO,NA,NE,KO,VA,RO,RA,OU,OM,OS,NO,LI,LA,DO,VE,VO,HO,VI,LO,TE,SA

Trigrafy:
THE,ING,AND,ION,ENT,ATI,TIO,ESS,THA,FOR,ERE,TTH,EAR,SIN,STH,HAT,LEA,ILL,HER,ERS,COM,RES,INT,MEN,ETO,OVA,PRE,ALI,STA,TOR,STI,OVE,EHO,PRI,NOS,KTO,AKO,PRA,SPO,LOV,POD,OLO,CAS,KOU

Klávesové skratky:
Tab,Back,Delete,Down,End,Return,Enter,Escape,F5,Home,Left,Right,Up,Space,Next,LControlKeyV,LControlKeyC,LControlKeyV,LControlKeyX,LControlKeyA,LControlKeyV,LControlKeyS,RControlKeyC,RControlKeyV,RControlKeyX,RControlKeyA,RControlKeyS,LControlKeyLShiftKeyB,RControlKeyRShiftKeyB,LShiftKeyLControlKeyB,RshiftKeyRControlKeyB

Obmedzenia pre KeyboardStateWatcher

Obmedzenia tohto komponentu spočívajú v eliminovaní náhodných pohybov. Odosielajú sa len cielené pohyby, ktoré spĺňajú základné kritéria ako minimálna dĺžka pohybu a maximálne prerušenie pohybu. Pre akcie typu *MouseMoveDto* a *DragAndDropDto* je minimálna dĺžka pohybu nastavená na 0,03 metra, t.j. skutočná prejdená vzdialenosť pohybu musí byť minimálne 3 centimetre. Maximálne prerušenie plynulosti akcie je pre *MouseMoveDto* nastavené na 1000 milisekúnd a pre *MouseScrollDto* táto hodnota predstavuje 2000 milisekúnd. Pohyby, ktoré neboli zakončené klikom a ich prerušenie bolo príliš veľké sa neukladajú(neplatí pre DragAndDropDto). Akcie typu *MouseScrollDto* sa ukladajú po každom prerušení, ale aj pri zmene smeru otáčania.

5.2 Web Browser

Sledovanie aktivít v internetovom prehliadači Mozilla Firefox.

5.2.1 Zásuvný modul / Rozšírenie pre zber údajov

Zber informácií je realizovaný prostredníctvom rozšírenia (add-in) prehliadača Mozilla Firefox. Ako je uvedené v kompletnom dátovom modeli (Obrázok 11), máme jednu základnú skupinu udalostí odchytávaných v prehliadači. Hlavnou abstraktnou triedou je WebEvent od ktorej sú odvodené nasledujúce konkrétne typy udalostí :

- WebNavigate
- WebBookmark
- WebTabOperation
- WebSaveDocument

5.2.1.1 WebNavigate

Udalosť nastane vždy pri zmene aktuálnej url adresy v prehliadači, kde táto udalosť obsahuje novú url adresu a typ url adresy.

Typy url adresy (enum WebNavigationType):

- Link – načítaná url adresa bola vyvolaná kliknutím na link v rámci poslednej načítanej stránky
- Bookmark – načítaná url adresa patrí medzi používateľove bookmark
- UrlBar – pokrýva zvyšné prípady zadanie url adresu, t.j. urlBar, navigácia pomocou spať a dopredu

5.2.1.2 WebBookmark

Udalosť nastane vždy pri modifikácii ľubovoľného bookmarku, t.j. zmena, pridanie, odstránenie. Daná udalosť obsahuje názov bookmarku a príslušnú url adresu pre daný bookmark.

5.2.1.3 WebTabOperation

Udalosť nastane pri práci s tabmi (prepnutie, pridanie, odstránenie), udalosť obsahuje url adresu a typ uskutočnenej operácie nad tabmi (enum WebTabOperationType):

- Switch To – vyvolá sa pri prepnutí na iný tab (v rámci inštancie jedného prehliadača), v danom prípade atribút url adresa zodpovedá url adrese tabu, na ktorý sa preplo
- Open – vyvolá sa pri otvorení nového tabu, v danom prípade atribút url adresa zodpovedá url adrese nového tabu
- Close – vyvolá sa pri zatvorení tabu, v danom prípade atribút url adresa zodpovedá url adrese zatvoreného tabu

5.2.1.4 WebSaveDocument

Udalosť nastane pri ukončení sťahovania dokumentu, daná udalosť obsahuje názov stiahnutého súboru (pod akým bol uložený na disk) a url adresu, z ktorej bol daný súbor stiahnutý.

5.2.2 Komunikačné rozhranie

Komunikačné rozhranie je realizované pomocou WCF http s kontraktom, obsahujúcim operáciu pre každý typ udalostí:

- ```
  [WebGet(UriTemplate =
  "WebSaveDocument?sessionId={sessionId}&url={url}&id={id}&name={name}")]
  void WebSaveDocument(int sessionId, string url, int id, string name);
  ```

- ```
  [WebGet(UriTemplate =
  "WebTabOperation?sessionId={sessionId}&url={url}&operation={operation}")]
  void WebTabOperation(int sessionId, string url, string operation);
  ```

- ```
  [WebGet(UriTemplate =
  "WebBookmark?sessionId={sessionId}&url={url}&name={name}")]
  void WebBookmark(int sessionId, string url, string name);
  ```

- ```
  [WebGet(UriTemplate =
  "WebNavigate?sessionId={sessionId}&url={url}&navigationType={navigationType}")]
  void WebNavigate(int sessionId, string url, string navigationType);
  ```

Firefox Add-in posiela activity watcher všetky potrebné parametre pre každý typ udalosti na základe, ktorých následne activity watcher vytvorí kompletné udalosti.

5.2.3 Komponent WebBrowserWatcher

Úlohou tohto komponentu je hosťovať spomínaný servis a sledovať volanie jeho operácií. Keď sa servisová operácia zavolá, tento komponent spracuje zadané parametre a vytvorí konkrétnu typ objekt na reprezentáciu udalosti. Nasledovne tento komponent vloží udalosť do lokálneho úložiska. Komponent negeneruje žiaden stav.

5.3 MS Visual Studio

Sledovanie aktivít vo Visual Studio IDE.

5.3.1 Zásuvný modul / Rozšírenie pre zber údajov

Zber informácií je realizovaný prostredníctvom rozšírenia (add-in) prostredia Visual Studio. Ako je uvedené v kompletnom dátovom modeli (Obrázok 11), sledované udalosti sú rozdelené do nasledovných skupín:

- Zmena stavu IDE
- Operácia nad dokumentom
- Operácia nad projektom
- Check-in
- Udalosť nad elementom kódu

5.3.1.1 Zmena stavu IDE

Sú využité nasledovné udalosti:

- BuildEvents.OnBuildBegin – zmena stavu na *build*, ak sa jedná o *build* alebo *rebuild all*
- BuildEvents.OnBuildDone – zmena stavu na *design*, ak sa jedná o *build* alebo *rebuild all*
- DebuggerEvents.OnEnterDesignMode – zmena stavu na *design*
- DebuggerEvents.OnEnterRunMode - zmena stavu na *debug*

Možnosti naviazania sa na tieto udalosti sú bližšie popísané v dokumente *IDEActivityAnalyze*.

5.3.1.2 Operácia nad dokumentom

Sú využité nasledovné udalosti:

- ProjectItemsEvents.ItemAdded – typ operácie *add*
- ProjectItemsEvents.ItemRemoved – typ operácie *remove*
- DocumentEvents.DocumentOpening – typ operácie *open*
- DocumentEvents.DocumentClosing – typ operácie *close*
- DocumentEvents.DocumentSaved – typ operácie *save*
- WindowEvents.WindowActivated – typ operácie *switch to*; ak sa aktivovalo iné okno obsahujúce dokument

Možnosti naviazania sa na tieto udalosti sú bližšie popísané v dokumente *IDEActivityAnalyze*.

5.3.1.3 Operácia nad projektom

Sú využité nasledovné udalosti:

- SolutionEvents.ProjectAdded - typ operácie *add*
- SolutionEvents.ProjectRemoved - typ operácie *remove*

Možnosti naviazania sa na tieto udalosti sú bližšie popísané v dokumente *IDEActivityAnalyze*.

5.3.1.4 Check-in

Naviaže sa na udalosť VersionControl.CommitCheckin každej registrovanej kolekcie projektov. Hodnota atribútu *IdInRcs* tejto udalosti predstavuje *changeset id*. Hodnota atribútu *RcsServer tejto* udalosti predstavuje uri servera.
Možnosti naviazania sa na túto udalosť sú bližšie popísané v dokumente *IDEActivityAnalyze*.

5.3.1.5 Udalosť nad elementom kódu

Tieto udalosti zatiaľ nie sú predmetom riešenia – pozri kapitolu 5.3.4 Obmedzenia.

5.3.2 Komunikačné rozhranie

Komunikačné rozhranie je realizované pomocou WCF Named Pipes s kontraktom, obsahujúcim operáciu pre každú skupinu udalostí:

- void NotifyIdeStateChange(IdeStateChangeDto eventDto);
- void NotifyIdeDocumentOperation(IdeDocumentOperationDto eventDto);
- void NotifyIdeProjectOperation(IdeProjectOperationDto eventDto);
- void NotifyIdeCheckin(IdeCheckinDto eventDto);

Add-in posiela activity watcher komponentu kompletné udalosti. Activity watcher komponent ich už nijako nemodifikuje.

5.3.3 Komponent VsActivityWatcher

Úlohou tohto komponentu je hosťovať spomínaný servis a sledovať volanie jeho operácií. Keď sa servisová operácia zavolá, tento komponent vloží udalosť bez akejkoľvek transformácie do lokálneho úložiska. Komponent negeneruje žiaden stav.

5.3.4 Ohraničenia

5.3.4.1 Zásuvný modul / Rozšírenie pre zber údajov

V prípade, ak meno solution nemôže byť determinované, je uvedené ako *<UNKNOWN>*. Známy je zatiaľ prípad, kedy sa vo Visual Studiu otvorí projekt namiesto solution. Visual Studio automaticky vytvorí pracovnú solution, ktorej meno ale nie je možné cez add-in priamo zistiť.

Meno solution nezahŕňa celú cestu k súboru na disku – je to len krátke meno súboru. Nie je preto možné jednoducho rozlišovať medzi dvoma solution súbormi s rovnakým menom. Hlavný problém pri uvedení celej cesty je možný výskyt mena používateľa v nejakej podobe. Podobe aj meno projektu predstavuje len krátke meno súboru.

Meno dokumentu tiež predstavuje len krátke meno súboru. V rámci jedného projektu sa tak môže nachádzať viacero dokumentov s rovnakým menom. Dalo by sa to vyriešiť uvádzaním aj relatívnej cesty dokumentu od projektu. Nie vždy je však možné z add-inu zistiť projekt, v ktorom sa dokument nachádza.

5.3.4.1.1 Zmena stavu IDE

Stav *debug* znamená, že v rámci Visual Studia je spustená vyvíjaná aplikácia. Ak by sa rozlišovalo medzi tým, kedy používateľ skutočne debuguje (krokuje, skáče medzi breakpointami a iné) a kedy je aplikácia len spustená, dochádzalo by ku generovaniu veľkého množstva udalostí. V prípade, že sa to bude nakoniec riešiť, bude treba tieto udalosti redukovať.

5.3.4.1.2 Operácia nad dokumentom

Pri niektorých úkonoch používateľa sa vygenerujú aj na prvý pohľad neočakávané udalosti. Napríklad pri pridaní dokumentu do projektu sa vygeneruje postupnosť: *open, close, add, swhitch to...* Táto postupnosť by sa mohla dať redukovať.

Pri niektorých udalostiach nie je uvedené meno projektu (*close, open*).

5.3.4.1.3 Operácia nad elementom kódu

Tieto udalosti sa zatiaľ neriešia pre potrebu častej konštrukcie abstraktného syntaktického stromu kódu, v ktorom sa pracuje. Mohlo by dochádzať k spomaleniu Visual Studia.

5.4 MS Lync 2010 status

Modul pre zisťovanie stavu prítomnosti používateľa pracovnej stanice využíva funkcionalitu aplikácie MS Lync 2010, ktorá je nainštalovaná na OS pracovnej stanice. MS Lync 2010 sleduje aktivitu na vstupných zariadeniach (klávesnica, myš) pracovnej stanice, dĺžku nečinnosti, stav naplánovaných úloh a stretnutí v kalendároch produktov MS Office, na základe ktorých určuje stav aktivít používateľa. Takisto poskytuje možnosť nastaviť stav používateľom cez rozhranie aplikácie. Modul zisťujúci stav prítomnosti používateľa získava informácie z Lync klienta volaním funkcií MS Lync 2010 API a teda nepotrebuje zásuvný modul alebo rozšírenie pre Lync aplikáciu, týmto modulom je samotná aplikácia.

5.4.1 Microsoft Lync 2010 API

Lync API je manažovaná .NET knižnica, ktorá na pozadí využíva COM objekty komunikujúce s Lync klientom. Pre získavanie informácií týkajúcich sa zisťovania statusu je postačujúce využitie

Microsoft.Lync.Model, v rámci ktorého sa nachádza trieda *LyncClient*, predstavujúca klientsky objekt pre komunikáciu s Lync aplikáciou. Využiju sa nasledovné prostriedky:

LyncClient.StateChanged	Manipulátor udalosti reprezentujúci zmenu stavu aplikácie
ClientState	Číselník reprezentujúci zmenu stavu Lync klienta (inicializácia, prihlasovanie, odhlasovanie, vypínanie, ...)
LyncClient.ClientDisconected	Manipulátor udalosti prerušenia spojenia s Lync klientom
LyncClient.Self	Reprezentuje používateľa prihláseného na lokálnej pracovnej stanici
Self.Contact	Reprezentuje kontakt v rámci Lync, osoba alebo číslo
Contact.ContactInformationChanged	Manipulátor udalosti zmeny informácií o kontakte, obsahujúci aj informácie o zmene statusu
Contact.GetContactInformation	Získanie informácií o kontakte
ContactAvailability	Číselník reprezentujúci status kontaktu

Výstupom pre ďalšie spracovanie sú hodnoty ContactAvailability:

- None
- Free
- FreeIdle
- Busy
- BusyIdle
- DoNotDisturb
- TemporarilyAway
- Away
- Offline
- Invalid

Scenáre, ktoré treba ošetriť, sú nasledovné:

1. Lync klient nie je nainštalovaný
2. Lync klient je vypnutý
    a. pred spustením UACA
    b. po spustení UACA
3. Lync klient je spustený
4. Lync klient neočakávane ukončil činnosť (spadol)
5. Prihlásenie používateľa
6. Odhlásenie používateľa
7. Zmena statusu (dostuponosti, aktivity) prihláseného používateľa

5.4.2 Komponent LyncStatusWatcher

Komponent LyncStatusWatcher sleduje zmeny stavov Lync klienta a generuje udalosti vychádzajúce z týchto zmien. Komponent stavové informácie (State) neposkytuje.

Zmeny stavov sú sledované vytvorením manipulátorov udalostí a ich zavesením na jednotlivé typy udalostí. Pri spustení aplikácie a v určitých stavoch možno priamo volať metódy pre zistenie statusu.

Status sa zisťuje kombináciou zisťovania stavu aplikácie a statusu kontaktu (status kontaktu možno zisťovať iba v prihlásenom stave aplikácie).

Výstup obsahuje nasledovné:

- nový status
- čas zmeny statusu
- či je zmena statusu míľnikom v rámci udalostí, pričom všetky zmeny statusu sú míľnikom

Status môže nadobúdať nasledovné hodnoty:

- Active
- Passive
- Bussy
- Offline

pričom mapovanie hodnôt získaných z ContactAvailability na status je takéto:

None	Offline
Free	Active
FreeIdle	Passive
Busy	Busy
BusyIdle	Passive
DoNotDisturb	Busy
TemporarilyAway	Passive
Away	Passive
Offline	Offline
Invalid	Offline

5.4.3 Ohraničenia

Obmedzením pri volaní prostriedkov Lync API sú situácie, kedy *LyncClient* neplatné informácie o stave Lync aplikácie, ako napríklad náhle ukončenie alebo vypnutie Lync klienta. *LyncClient* si priebežne pamätá stav prostriedkov Lync klienta a komunikačných kanálov s ním, preto je v príslušných scenároch potrebné jeho stav uviesť do neplatného (volaním niektorého z prostriedkov, ktorých volanie v neaktuálnom stave vyvolá výnimku a tým sa stav pamäte LyncClinet zaktualizuje). Vzhľadom na to, že v neplatnom stave objektu nemožno zachytávať zmeny stavov klienta, nemožno využívať manipulátory na zistenie stavov zapnutia, reštartu alebo prihlásenia sa do klienta a teda nemožno udalosťou zachytiť stav, kedy je potrebné zneplatniť vyrovnávaciu pamäť *LyncClient* a následne ju znovu uviesť do platného stavu. Toto možno zabezpečiť prostredníctvom volania zneplatnenia v časových intervaloch. Obmedzením je fakt, že zmena stavu sa zistí až pri vyvolaní zneplatnenia a teda aktualizácia závisí od intervalu časovača. Zneplatnenie a obnovenie prostriedkvo LyncClient treba volať v hlavnom vlákne (v časovači cez Dispatch), ináč sa prostriedky nesprávajú korektne.

5.5 Stav aplikácií

Modul zabezpečuje sledovanie prepínania aktívnych aplikácií. Okrem toho poskytuje v rámci akcie zoznam aktuálne bežiacich aplikácií.

5.5.1 Zmena aktívnej aplikácie

Za účelom sledovania zmeny aktívnej aplikácie bude využitý systémový hák (angl. hook), ktorý umožňuje sledovať udalosti generované v systéme. Pre naviazanie sa na danú udalosť slúži funkcia operačného systému (http://msdn.microsoft.com/en-us/library/windows/desktop/dd373640%28v=vs.85%29.aspx)

```
HWINEVENTHOOK WINAPI SetWinEventHook(
  __in  UINT eventMin,
  __in  UINT eventMax,
  __in  HMODULE hmodWinEventProc,
  __in  WINEVENTPROC lpfnWinEventProc,
  __in  DWORD idProcess,
  __in  DWORD idThread,
  __in  UINT dwflags
);
```

Podstatné parametre sú eventMin a eventMax, definujúce rozsah udalostí, ktoré majú byť sledované. V našom prípade sa jedná len o EVENT_SYSTEM_FOREGROUND. Parametrom dwflags je možné okrem iného určiť v akom kontexte sa budú udalosti sledovať .

- WINEVENT_INCONTEXT – Zavesenie sa realizuje priamo v procese sledovanej aplikácie. Udalosti je teda možné sledovať hneď ako nastanú. Tento spôsob však nie je možné využívať v .NET, pretože funkcia pre ošetrenie udalosti musí byť implementovaná v nemanažovanej dll.
- WINEVENT_OUTOFCONTEXT – Zavesenie sa realizuje mimo procesu sledovanej aplikácie. Notifikácie sú do aplikácie, ktorá hák vytvorila, prenášané cez frontu správ. Spracovanie je teda asynchrónne, pričom je ale stále zabezpečené poradie notifikácií.

5.5.2 Zoznam bežiacich aplikácií

Pre získanie zoznamu bežiacich aplikácií bude použitá metóda GetProcesses triedy System.Diagnostics.Process (http://msdn.microsoft.com/en-us/library/1f3ys1f9.aspx). Výstupom metódy je zoznam aktuálne bežiacich procesov. Pre naše účely budeme sledovať len tie procesy, ktoré majú vytvorené okno a používateľ s nimi môže teda interagovať (ďalej len aplikácie).
Zoznam sa získa vždy pri poskytovaní stavu v rámci aktivity, pričom sa porovná s predchádzajúcim zoznamom. Ukončené aplikácie sa odstránia so zoznamu a novým sa priradí jednoznačný globálny identifikátor behu RunId.

5.5.3 Ohraničenia

Keďže sú notifikácie prepnutia aplikácie prenášané asynchrónne, je možné, že sa udalosť prepnutia zachytí až po udalostiach vyvolaných prácou v danej aplikácií. Napríklad sa môže stať, že sa používateľ prepne do internetového prehliadača s tým, že rovno klikne na záložku, čo vyvolá udalosť zmeny URL. Udalosť prepnutia do prehliadača sa však môže spracovať až po udalosti zmeny URL a teda nemusí byť jasné v ktorej aplikácií daná udalosť nastala. Tento problém je do určitej miery minimalizovaný skutočnosťou, že jednotlivé udalosti obyčajne obsahujú určitý identifikátor aplikácie z ktorej pochádzajú. Napr. Vs aktivity obsahujú okrem iného identifikátor procesu (pid).

Ďalším problémom je práca na vzdialenom PC cez remote desktop connection. Pri takejto práci sa nezachytávajú prepnutia aplikácií v rámci vzdialeného PC a nie je ani získavaný zoznam procesov, ktoré na danom PC bežia. Ďalší problém pri remote dektop je spôsobený funkciou SetWinEventHook, ktorá v prípade prepnutia z remote desktop nezachytí prepnutie do lokálnej aplikácie.

5.6 Vyťaženie HW prostriedkov

Úlohou modulu je snímanie hodnôt vyťaženia CPU a operačnej pamäte počas aktivity.

5.6.1 Počítadlá výkonu

Za účelom získavania potrebných údajov budú využité systémové počítadlá výkonu (angl. performance counter). Presnejšie sa jedná o nasledujúce počítadla:

- *Vyťaženie CPU v %* - bude využitý pre získanie priemerného a maximálneho vyťaženia CPU počas trvania aktivity.
- *Voľná operačná pamäť v MB* –bude využitý pre získanie priemerného a maximálneho vyťaženia operačnej pamäte v percentách. Pre prepočet voľnej pamäte na vyťaženie v percentách je nutné získať celkovú veľkosť operačnej pamäte. Za týmto účelom nám poslúži hodnota TotalVisibleMemorySize vo WMI (Windows Management Instrumentation) triede Win32_OperatingSystem.

Jednotlivé hodnoty budú snímané v pravidelných intervaloch pomocou časovača (angl. timer).

5.7 MS OneNote 2010 activity

Modul je určený pre zisťovanie aktivít v rámci aplikácie OneNote nainštalovanej na pracovnej stanici používateľa. Informácie získava priamo z aplikácie volaním funkcií MS OneNote 2010 API a teda nepotrebuje zásuvný modul alebo rozšírenie pre OneNote aplikáciu, týmto modulom je samotná aplikácia.

5.7.1 Microsoft OneNote 2010 API

OneNote API je manažovaná .NET COM knižnica, ktorá implementuje prostriedky pre komunikáciu s OneNote klientom. Pre získavanie informácií o inštanciách, presnejšie oknách inštancie aplikácie, a aktuálneho obsahu treba využiť väčšinu funkcií z priestoru *Microsoft.Office.Interop.OneNote*, ktorý však nie je príliš rozsiahly. Využiju sa nasledovné prostriedky:

Application	Trieda predstavujúca rozhranie na OneNote aplikáciu
Application.GetHierarchy	Načítanie metadát o aktuálne otvorených poznámkových blokoch, z ktorých možno získať informácie o celej štruktúre poznámkových blokov.
Window	Bežiace okno aplikácie predstavujúce „inštanciu aplikácie“ získané z Application.Windows
Window.WindowHandle	Handler na okno predstavujúce samostatný editor poznámkového bloku, javiaci sa ako samostatná inštancia aplikácie. Treba zistiť handler na koreňové okno.
Window.IsSideNote	Hovorí o tom, či je okno otvorená ako poznámka
Window.FullPageView	Hovorí o tom, či je pohľad na obsah roztvorený na celé okno UI
Window.DockedLocation	Lokalita ukotvenia okna pri zobrazení UI v móde ukotvenia na

	pracovnej ploche
Window.CurrentNotebookId	Identifikátor práve otvoreného poznámkového bloku v danom okne
Window.CurrentSectionGroupId	Identifikátor práve otvorenej sekčnej skupine v danom okne, ak nejaká skupina existuje
Window.CurrentSectionId	Identifikátor práve otvorenej sekcie v danom okne
Window.CurrentPageId	Identifikátor práve otvorenej stránky v danom okne

Inštancia aplikácie OneNote je len jedna. Ak spustím OneNote viackrát, inštancia je zdieľaná, vytvárajú sa iba viacerá inštancie okien predstavujúce samostatné UI. Dôsledkom existencie iba jednej inštancie aplikácie je fakt, že všetky okná majú otvorenú rovnakú množinu poznámkových blokov, každé UI však môže zobrazovať iný blok z tejto množiny.

Pre zistenie koreňového okna UI je potrebné využitie funkcie Win32 API

```
[DllImport("user32.dll")]
static extern IntPtr GetParent(IntPtr hWnd);
```

Výstupom pre ďalšie spracovanie sú teda hodnoty:

- Stav UI
  - roztvorenie okna
  - umiestnenie okna
- Aktuálny obsah
  - Poznámkový blok
  - Sekčná skupina
  - Sekcia
  - Stránka
- Metadáta o obsahu
  - štruktúra
  - umiestnenie
  - identifikátory
  - názvy

Scenáre, ktoré treba ošetriť, sú nasledovné:

1. Spustenie inštancie
2. Zatvorenie inštancie
3. Zmena v aktuálne zobrazenom obsahu (navigácia v obsahu)
4. Neotvorené žiadne poznámkové bloky v aplikácii
5. Zmena spôsobu zobrazenia UI

5.7.2 Komponent OneNoteActivityWatcher

Komponent OneNoteActivityWatcher sleduje zmeny v navigácií na obsah a zmeny v zobrazení UI OneNote aplikácie a generuje udalosti vychádzajúce z týchto zmien. Komponent stavové informácie (State) neposkytuje.

OneNote COM API neumožňuje zachytávať udalosti o zmenách otvoreného obsahu alebo .zobrazenia UI, umožňuje iba zistenie stavu zobrazenia obsahu a UI v danom momente. Preto je potrebné zisťovať aktuálne zobrazený obsah aj stav UI v pravidelných časových intervaloch. Ak chceme generovať udalosti o zmenách, treba následne porovnať aktuálne zistená hodnoty s hodnotami zistenými v predchádzajúcom intervale.

Udalosti a ich obsah bude nasledovný:

- Navigácia
    - aktuálny poznámkový blok
    - aktuálna sekčná skupina
    - aktuálna sekcia
    - aktuálna stránka
- Zmena pohľadu
    - či je okno otvorené ako bočná poznámka
    - typ zobrazenia (normálne, plné roztvorenie, ukotvenie)
- Oboje budú obsahovať
    - handler na okno
    - čas zistenia zmeny
    - či je zmena míľnikom v rámci udalostí, pričom žiadne zmeny nie sú míľnikom

5.7.3 Ohraničenia

Vzhľadom na získavanie informácií o otvorenom obsahu a stave UI v časových intervaloch, presnosť získaných údajov závisí od dĺžky intervalu. To znamená, že ak sa používateľ navigoval medzi dvoma získaniami stavu na viaceré stránky v poznámkovom bloku, alebo menil pohľady UI, informácie o týchto zmenách nebudú zaznamenané, udalosť bude spojená iba so stavom aktuálnym v čase zisťovania stavu. Vzhľadom na presnosť zaznamenávania zmien, vyťaženia prostriedkov počítača, ale aj z pohľadu toho, či chceme zaznamenávať aj stavy s veľmi krátkym trvaním (napr. že sa používateľ navigoval na stránku, na ktorej bol iba 10s, alebo že v rýchlom slede prehľadával stránky), treba zvoliť vhodný časový interval.

Ďalším ohraničením je scenár, kedy používateľ zatvorí všetky poznámkové bloky. V tomto prípade sú otvorené tzv. Unfilled notes. Ak však používateľ zatvorí aplikáciu a potom ju otvorí, nie je v inštancii otvorené nič. V takomto prípade funkcia *Window.CurrentNotebookId* vyvolá chybu, čo je pravdepodobne chyba v implementácii COM API, pretože aj v tomto prípade možno v metadátach tento identifikátor nájsť.

6 Nasadenie

6.1 Požiadavky

6.1.1 Minimálne

6.1.1.1 Server

- SQL2005/2008
- .NET 4.0
- MS Windows Server2008/2008R2
- IIS
- Zaregistrovaný ASP.Net 4.0 v IIS (aspnet_regiis **–ir**) (http://msdn.microsoft.com/en-us/library/k6h9cz8h%28v=VS.100%29.aspx)
- Registrácia WCF a WF v IIS (`ServiceModelReg.exe -ia`) (http://msdn.microsoft.com/en-us/library/ms732012.aspx)

6.1.1.2 Klient

- .NET 4.0

6.1.2 Odporúčané

Pre fungovanie všetkých modulov musí mať klient nainštalované:

- Mozilla Firefox
- MS Visual Studio 2010
- MS Office 2010 OneNote
- MS Lync 2010

6.2 Aplikácia

Aplikácia sa inštaluje prostredníctvom UserActivity.AppSetup.msi. V rámci inštalácie sa inštalujú zásuvné moduly do Visual Studio a Mozilla Firefox. Po skončení inštalácie treba skontrolovať, či sú aktivované.

6.3 Web služba

Servis je nutné sprístupniť cez IIS štandardným spôsobom, čiže zabezpečiť IIS S*ite*, IIS A*ppPool* a IIS *application,* pričom je nutné aby *AppPool* bežal pod .NET 4.0. Do lokálneho adresára IIS aplikácie je nutné prekopírovať adresár svc.

Vo web.config je potrebné primerane upraviť nasledujúce nastavenia:

1. *ConnectionString* pre User Activity DB
```
<add name="UserActivityContainer"
connectionString="metadata=res://*/UserActivity.csdl|res://*/UserActivity.ssdl|
res://*/UserActivity.msl;provider=System.Data.SqlClient;provider connection
string=&quot;data source=perconikdb1;initial
catalog=PerConIK_UserActivity;integrated
security=True;multipleactiveresultsets=True;App=EntityFramework&quot;"
providerName="System.Data.EntityClient" />
```
2. *ConnectionString* pre Anonymization DB
```
<add name="AnonymizationModelContainer"
connectionString="metadata=res://*/AnonymizationModel.csdl|res://*/Anonymizatio
nModel.ssdl|res://*/AnonymizationModel.msl;provider=System.Data.SqlClient;provi
```

```
der connection string=&quot;data source=perconikdb1;initial
catalog=PerConIK_Anonymization;integrated
security=True;multipleactiveresultsets=True;App=EntityFramework&quot;"
providerName="System.Data.EntityClient" />
```

6.4 Databáza

Po vytvorení databáz pre uchovávanie UserActivity údajov ako aj údajov o používateľoch Anonymization, s názvami zhodnými v konfigurácii web služby, je nutné spustiť priložené skripty „UserActivity.edmx.sql“ a „AnonymizationModel.edmx.sql“. Používateľ pod ktorým beží príslušný IIS app pool musí mať právo na čítanie a zápis v databázach.