

# Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Ilkovičova 3, 842 16 Bratislava 4

## **Odhaľovanie emocionálneho stavu používateľa**

Team EmLog

Dokumentácia k inžinierskemu dielu

**Tímový projekt 2012/2013**

**Vedúca tímu:**

Doc. Mgr. Daniela Chudá, PhD.

**Členovia tímu:**

Bc. Jozef Gajdoš

Bc. Martin Geier

Bc. Peter Greguš

Bc. Miroslav Hudák

Bc. Peter Sivák

Bc. Peter Šinský

**Kontakt:** team.10.tp@gmail.com

# 1 OBSAH

---

2	Úvod.....	4
2.1	Účel dokumentu.....	4
2.2	Štruktúra dokumentu .....	4
2.3	Zadanie projektu .....	4
3	Šprint 0 – Antónia .....	5
3.1	Zaznamenávanie aktivity používateľa .....	5
3.2	Možné smery odporúčaní .....	6
4	Šprint 1 – Frederika.....	7
4.1	Analýza programu <i>PerConIK</i> .....	7
4.1.1	Biometria.....	8
4.1.2	Architektúra.....	8
4.1.3	Zber biometrických údajov .....	9
4.2	Dátový model programu <i>PerConIK</i> .....	11
4.2.1	Klientská časť aplikácie .....	11
4.2.2	Serverová časť aplikácie .....	11
4.3	Pridanie emocionálneho stavu používateľa do logov a dátového modelu.....	14
4.3.1	Analýza.....	14
4.3.2	Návrh.....	15
4.3.3	Implementácia .....	16
4.4	Analýza emocionálneho stavu používateľa .....	16
4.4.1	Hnev .....	17
4.4.2	Únava .....	18
4.4.3	Stres.....	19
4.4.4	Radosť .....	20
5	Šprint 2 - Henrieta .....	22
5.1	Pridanie funkcionality do programu <i>PerConIK</i> .....	22
5.1.1	Logovanie emócií.....	22
5.1.2	Pridanie konvertoru pre emócie .....	22
5.1.3	Pridanie generovania mĺňníka .....	23

5.2	Analýza dátových štruktúr vstupných zariadení.....	24
5.2.1	Analýza dátových štruktúr myši.....	24
5.2.2	Analýza dátových štruktúr klávesnice.....	25
5.3	Návrh výpočtu metrík pre myš .....	26
5.3.1	Sledovanie myši pomocou programu <i>PerConIK</i> .....	26
5.3.2	Metriky pre sledovanie emočného stavu používateľa .....	26
5.4	Návrh výpočtu metrík pre klávesnicu.....	27
5.4.1	Sledovanie klávesnice pomocou programu <i>PerConIK</i> .....	27
5.4.2	Metriky pre sledovanie emočného stavu používateľa .....	28
5.5	Návrh modelu používateľa .....	30
5.6	Implementácia modelu používateľa.....	30
5.6.1	Pridané entity do dátového modelu .....	30
5.6.2	Vytváranie vektoru metrík ako model používateľa.....	31

## 2 ÚVOD

---

### 2.1 ÚČEL DOKUMENTU

Tento dokument vznikol na predmete tímový projekt ako dokumentáciou k projektu pre zisťovanie emocionálneho stavu používateľa.

### 2.2 ŠTRUKTÚRA DOKUMENTU

Štruktúra dokumentu je prispôbená vývoju metódou *Scrum*. V každej kapitole je zdokumentovaný jeden šprint. V jednotlivých podkapitolách sú opísané činnosti, ktoré sa v danom šprinte vypracovávali.

### 2.3 ZADANIE PROJEKTU

Každý používateľ počas práce s počítačom zažíva momenty, v ktorých sa cíti rôzne. Keď sa mu podarí dokončiť projekt, žiari šťastím, naopak, keď stratí neuloženú prácu v dôsledku chyby programu, počítač by najradšej rozbil. Emocionálny stav používateľa však môže poskytnúť spätnú väzbu aj k tomu, čím sa práve zaoberá, napríklad pri prezeraní časti zdrojového kódu softvérového projektu, ktorú vytvoril jeho kolega. Ak je zase používateľ unavený a frustrovaný pri vytváraní zdrojového kódu, je pravdepodobné, že tento kód bude obsahovať chyby. Vtedy mu môžeme odporučiť krátku prestávku.

Úlohou tímov bude vytvorenie modelu pre reprezentáciu emócií používateľa a následné zachytávanie emocionálneho stavu používateľa do tohto modelu. Pri tom sa jeden tím zameria na prácu používateľa s bežnými vstupnými zariadeniami, klávesnicou a myšou (ako rýchlo používateľ píše, ako často a akým spôsobom sa mýli, ...), druhý tím využije obraz používateľa získavaný webovou kamerou (výraz tváre, smer pohľadu, ...). Cieľom je vytvorenie aplikačného rozhrania umožňujúceho využitie rozpoznávaného stavu používateľa v rôznych projektoch, napríklad vo vývojom prostredí v rámci projektu podpory vývoja softvéru v prostredí firmy, ako i overenie a porovnanie vytvorených riešení v rôznych situáciách: odporúčanie prestávok, dôvera v hodnotenie poskytnuté používateľom a pod.

## 3 ŠPRINT 0 – ANTÓNIA

---

### 3.1 ZAZNAMENÁVANIE AKTIVITY POUŽÍVATEĽA

Zodpovedná osoba: Martin Geier

Táto kapitola opisuje možné alternatívy k programu *PerConIK*. Hlavnými požiadavkami boli množstvo funkcií ktoré by sme mohli uplatniť pri analýze emocionálneho stavu používateľa a licencia umožňujúca úpravu zdrojových kódov s ich následnou distribúciou.

*Auto clicker typer:*

- Poskytuje funkcionalitu na zachytávanie obrazovky s možnosťou následného spustenia ako video,
- zaznamenáva:
  - klávesnicu,
  - myš,
  - obrazovku,
- licencia – uzavretý zdrojový kód.

*KidLogger:*

- skladá sa z aplikácie na lokálnej stanici a serverovej časti,
- lokálna aplikácia odosiela nazbierané údaje na server, na ktorom je možné prehliadať aktivity a štatistiky,
- zaznamenáva:
  - klávesnicu,
  - myš,
  - obrazovku,
  - nahrávanie hlasu,
  - komunikačné nástroje – *icq, skype*,
  - spustené aplikácie,
- licencia – zverejnený zdrojový kód klientskej aplikácie pre Windows na účely štúdia či program nerobí inú činnosť ako má deklarovánú,
- programovací jazyk – C++.

*OsdHotkey:*

- jednoduchá aplikácia slúžiaca ako *keylogger*,
- zaznamenáva:
  - klávesnicu,
  - myš,
- licencia – *open source*,

- programovací jazyk – *AutoHotkey language*.

*PyKeylogger*:

- aplikácia slúžiaca prevažne ako *keylogger*,
- zaznamenáva:
  - klávesnicu,
  - pozíciu myši pri kliknutí,
  - obrazovku.
- licencia – *open source*,
- programovací jazyk – *Python*.

## 3.2 MOŽNÉ SMERY ODPORÚČANÍ

Zodpovedná osoba: Peter Greguš

Smery odporúčaní pre používateľa môžeme rozdeliť do dvoch kategórií a to okamžité a odporúčanie na základe dlhodobého pozorovania.

Do okamžitých môžeme zaradiť odporúčanie prestávok, zmena farebnej schémy na počítači, zmeniť „zameranie“ (ísť na chvíľu navštíviť sociálnu sieť a pod.), zahrať si krátku hru a iné.

Pri odporúčaní na základe dlhodobého pozorovania sa najskôr zistí štatistika používateľa napríklad akú hudbu počúva pri danej emócií a následne sa vytvorí *playlist*. Potom sa budú odporúčať tie piesne, pri ktorých mal používateľ dobrú náladu. Ďalej je možné vytvárať štatistiku v kombinácii z množstvom vykonanej práce. Na základe výstupného grafu bolo vidno, kto koľko vypracoval pri akom emočnom stave. Z toho môže šéf odporúčať svojim podriadeným dovolenku, prípadne inú formu motivácie.

## 4 ŠPRINT 1 – FREDERIKA

### 4.1 ANALÝZA PROGRAMU *PERCONIK*

Zodpovedná osoba: Miroslav Hudák

Cieľom aplikácie *PerConIK* je zaznamenanie aktivity používateľa za účelom analýzy správania sa. Výsledkom tejto analýzy je model používateľa, ktorý opisuje jeho charakteristické črty správania sa v čase.

*Udalosť* je činnosť používateľa na *GUI*, ktorá vyvolá nejaký proces. Delí sa na:

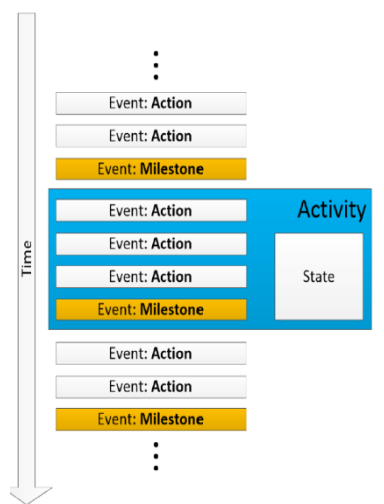
- *akciu* – predstavuje bežnú činnosť v pracovnej aktivite používateľa,
- *míľnik* – významná – spúšťa zber a odosielanie údajov o aktivite.

Udalosti vzťahujúce sa na činnosť používateľa sa zaznamenávajú podľa možnosti chronologicky tak, ako nastali. Udalosťou môže byť spustenie *buildu* v *IDE*, prepnutie stránky vo *OneNote*, prepnutie aktívnej aplikácie a iné.

*Aktivita* používateľa je postupnosť udalostí (akcií) zakončená významnou udalosťou (míľnikom). Každá aktivita obsahuje práve jeden míľnik.

Po ukončení aktivity používateľa sa vytvorí balík údajov, ktorý sa skladá z:

- akcií,
- míľnika,
- statických údajov – biometria, stav aplikácií, využitie *HW* prostriedkov.



OBRÁZOK 1 - AKTIVITA OBSAHUJÚCA ZOZNAM AKCIÍ, MÍENIK A STAV

*Zdroje informácií:*

- *MS Visual Studio 2010* – sledovanie kontextu programovania a zostavovania programov v *IDE* vo forme rozšírenia,
- *MS Office 2010 OneNote* – aktivity v kontexte otvorených poznámkových blokov cez *COM* rozhranie,
- biometria – biometrické údaje použitím klávesnice a myši,
- *Application Activity* – stav spustených aplikácií a okien,
- *HW* prostriedky – sledovanie procesora, využitie pamäte cez rozhranie *Win32 API*,
- *MS Outlook a Lync 2010* – vytváranie a čítanie elektronickej komunikácie vo forme rozšírenia,
- rozšírenie webového prehliadača – načítané webové zdroje.

Pre účely tohto projektu sú najpodstatnejšie biometrické údaje.

## 4.1.1 BIOMETRIA

Biometrické údaje sa získavajú sledovaním použitia zariadení ako je klávesnica a myš. Údaje o používaní týchto prostriedkov sa získavajú prostredníctvom *Win32 API*.

Nasledujúci obrázok zobrazuje sledované údaje a možnosti ich získania.

Údaj	Funkcia	Typ	Spôsob získania
<b>Myš...</b>			
Poloha -> intenzita pohybu	WM* mouse-events		event /suma za interval
Klik -> intenzita klikania	WM* mouse-events		event /suma za interval
Scroll -> intenzita použitia	WM* mouse-events		event /suma za interval
<b>Klávesnica...</b>			
Tlačidlo dole	WM,WK* keyb.events		event
Tlačidlo hore	WM,WK* keyb.events		event
<ul style="list-style-type: none"> <li>• Historia stlacených klavesov,</li> <li>• Intenzita stlacania klaves</li> <li>• Intervaly medzi stlacením vybraných klaves</li> </ul>			postupnosť klavesov s časovou značkou, alebo len suma za interval

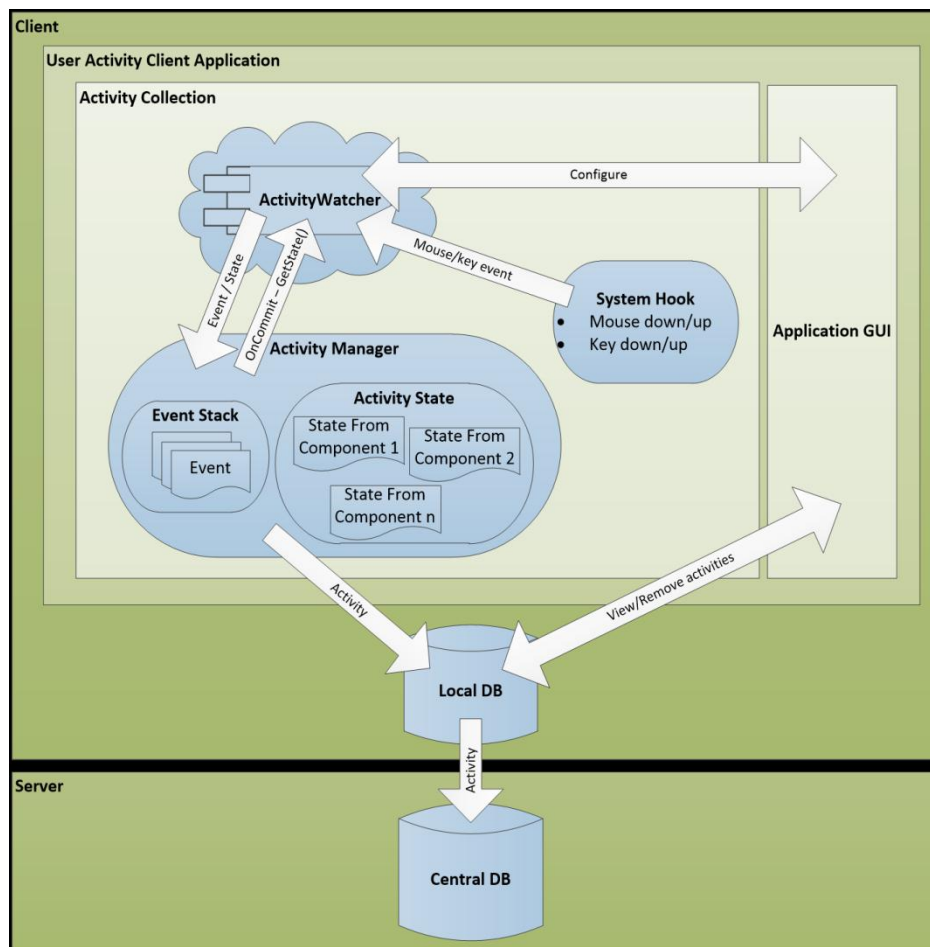
OBRÁZOK 2 - SLEDOVANÉ BIOMETRICKÉ ÚDAJE

## 4.1.2 ARCHITEKTÚRA

Zber aktivít jedného používateľa riadi na jeho klientskom počítači aplikácia *User Activity Client Application (UACA)*, v jednoduchosti *logger*, 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 *CentralDB* (pozri obrázok).



OBRÁZOK 3 - ARCHITEKTÚRA SYSTÉMU *PERCONIK*

Aplikácia *UACA* obsahuje komponenty (*ActivityWatcher*) zodpovedné za notifikáciu udalostí keď nastanú. *ActivityWatcher* notifikuje o udalosti (akcii alebo míľniku) *ActivityManager*. Ten vloží udalosť do zásobníka udalostí – *EventStack*. V zásobníku sú len udalosti pre aktuálnu aktivitu. Ak je notifikovaný míľnik, došlo k zakončeniu aktivity. V tom prípade sa od každého komponentu *ActivityWathcer* vyžiada informácia o stave, ktorých množina predstavuje celkový stav aktivity. Usporiadaný zoznam udalostí spolu s celkovým stavom aktivity sú poslané do lokálneho úložiska (*LocalDB*). Zásobník udalostí *EventStack* sa vyprázdni a začína konštrukcia novej aktivity.

#### 4.1.3 ZBER BIOMETRICKÝCH ÚDAJOV

Zber údajov z klávesnice a myši má na starosti modul *BiometryWatcher*. Je rozdelený na dva komponenty:

- *KeyboardStateWatcher* – klávesnica,
- *MouseStateWatcher* – myš.

Modul sleduje generované zmeny a tie následne posúva na konkrétny komponent. Údaje sú poskytované vo forme udalostí:

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

Modul zbiera údaje počas kompletnej aktivity používateľa. Odoslanie údajov nastane, keď je identifikovaný míľnik. Samotný modul negeneruje míľnik, preto sa spolieha na iné moduly. Zozbierané dáta sa posielajú do databázy vo formáte *BLOB*.

#### *KeyboardStateWatcher*

Tento komponent zachytáva údaje generované klávesnicou a ukladá ich do výstupnej štruktúry *KeyboardStateBlobDto*. Táto trieda reprezentuje zoznamy latencií (*List<KeyboardGraphDto>*) pre grafy (znaky), digrafy (dvojice znakov), trigrafy (trojice znakov) a klávesové skratky. Trieda *KeyboardGraphDto* obsahuje trojicu údajov:

- *Character* – hodnota stlačených znakov,
- *Latency* – latencia stlačenia,
- *FlightTime* – interval medzi stlačením rôznych kláves.

Pre každý stlačený znak na klávesnici sa vygenerujú dve udalosti, ktoré reprezentujú čas stlačenia a čas uvoľnenia klávesy. Pomocou týchto údajov sa vypočítajú latencie a čas letu.

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

#### *MouseStateWatcher*

Tento komponent zachytáva údaje generované myšou a ukladá ich do výstupnej štruktúry *MouseStateBlobDto*. Táto trieda reprezentuje zoznamy špecifických akcií myši:

- zoznam pohybov,
- zoznam akcií *Drag&Drop*,
- zoznam pohybov kolieska myši,
- zoznam všetkých typov stlačenia tlačidiel myši.

Medzi sledované údaje patria: začiatková pozícia, koncová pozícia, čas pohybu, prejdená vzdialenosť, typ stlačeného tlačidla, hodnota otočenia kolieska.

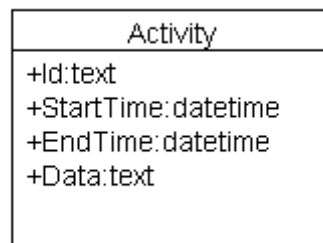
Aby sa eliminovali náhodné pohyby, odosielajú sa len pohyby, ktoré spĺňajú základné kritéria ako minimálna dĺžka pohybu a maximálne prerušenie pohybu.

## 4.2 DÁTOVÝ MODEL PROGRAMU *PERCONIK*

Zodpovedná osoba: Jozef Gajdoš

### 4.2.1 KLIENSKÁ ČASŤ APLIKÁCIE

Klientská časť aplikácie používa *SQLite* databázu verzia 3, v ktorej má uchovanú iba jednu tabuľku s názvom *Activity*. Jej schéma je na nasledovnom obrázku. Entita *Activity* reprezentuje aktivitu používateľa ktorá bola sledovaná medzi dvoma míľnikmi.

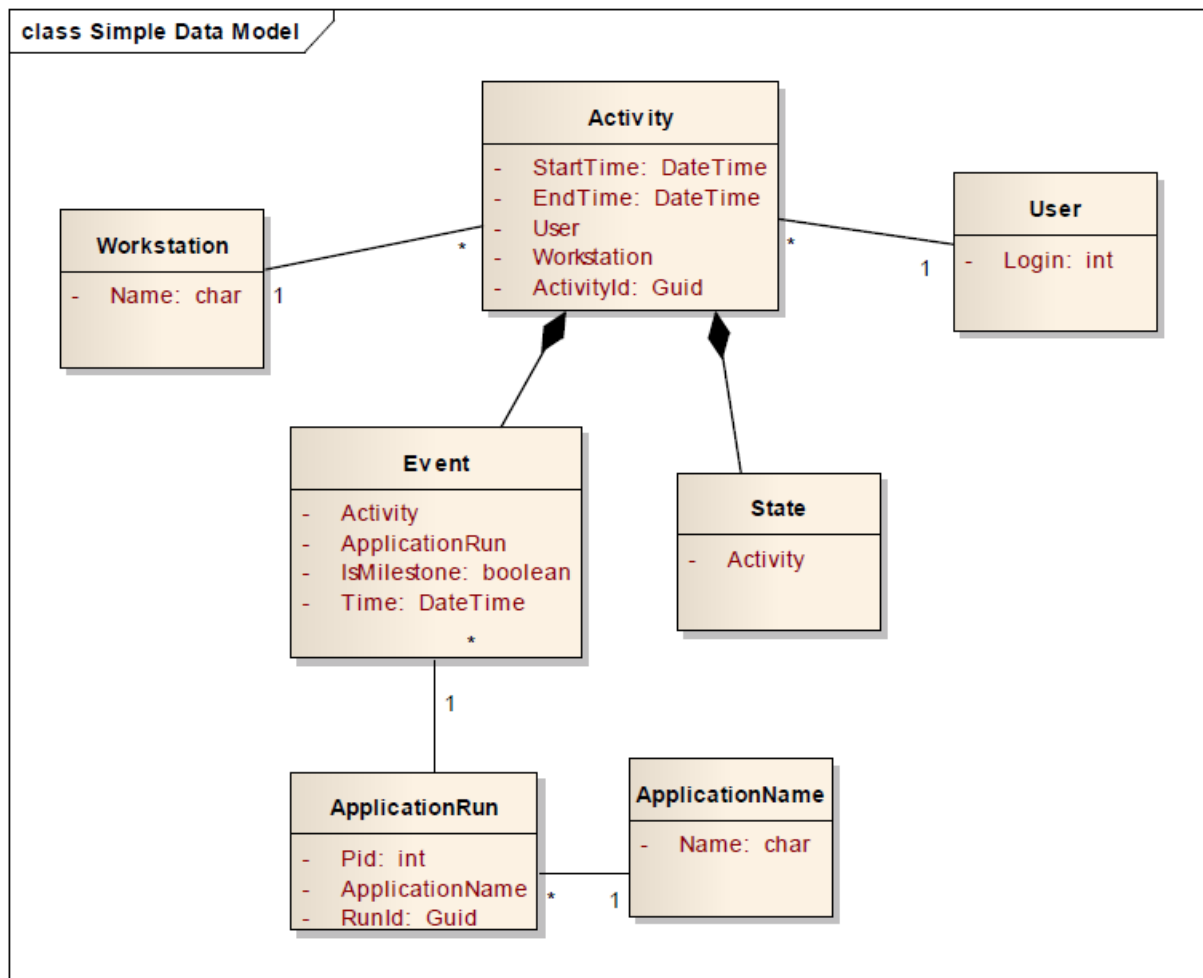


OBRÁZOK 4 - DÁTOVÝ MODEL LOKÁLNEJ DATABÁZY

*Id* aktivity je jedinečný text generovaný pre každú aktivitu zvlášť (*guid*). Atribúty *StartTime* a *EndTime* reprezentujú začiatok a koniec aktivity. Atribút *Data* obsahuje serializované objekty reprezentujúce aktivitu používateľa. Sú uložené vo forme *XML*.

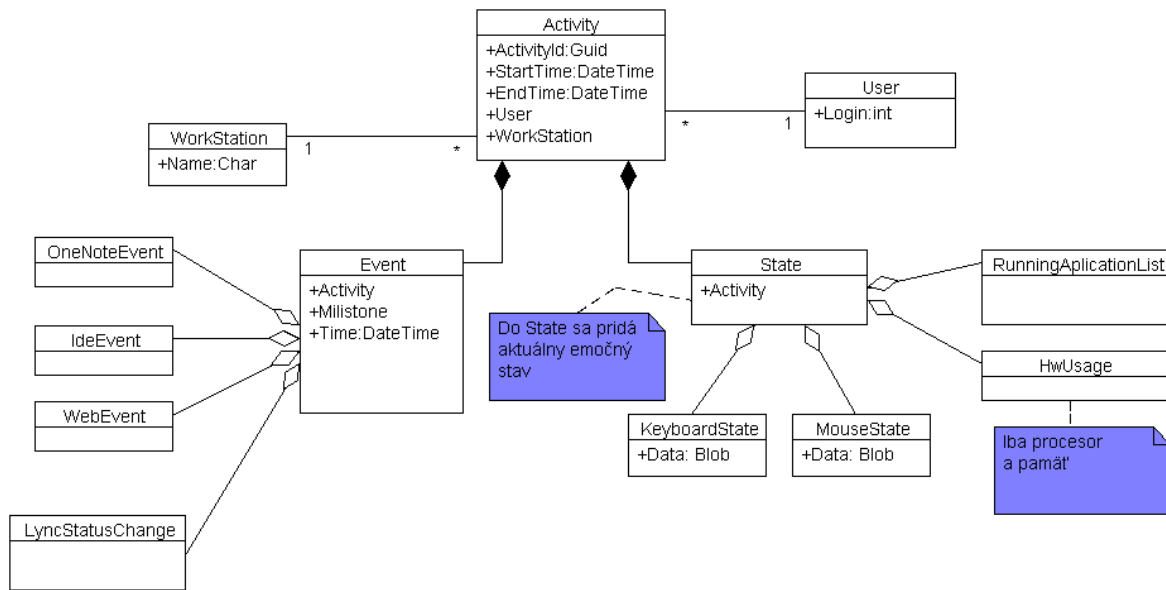
### 4.2.2 SERVEROVÁ ČASŤ APLIKÁCIE

Obrázok 5 znázorňuje logický model serverovej časti aplikácie, ktorá používa *MS SQL* server.

OBRÁZOK 5 – LOGICKÝ MODEL DATABÁZY *PERKONIK*

Ústrednou entitou je Aktivita, ktorá uchováva udalosti medzi dvoma míľníkmi a stav sledovaného systému na konci aktivity. Každá udalosť prislúcha k behu konkrétnej aplikácie.

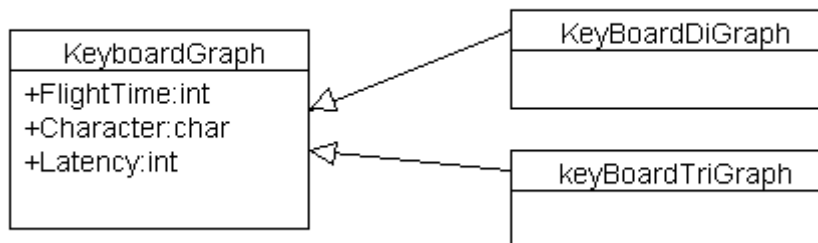
Na ďalšom obrázku je znázornený logický model serverovej časti aplikácie ktorý abstrahuje od detailov *Eventu* (stavu aplikácií) a zameriava sa na spôsob ukladania klávesnice a myši.



OBRÁZOK 6 – DETAILNÝ LOGICKÝ MODEL DATABÁZY

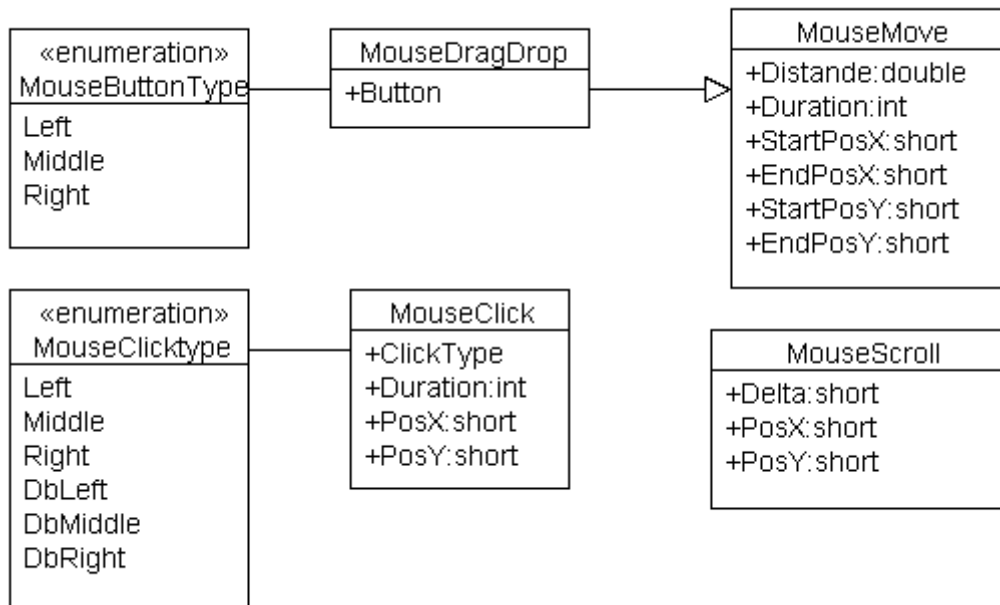
Entity uchovávajúce stav klávesnice a myši sú agregované v stave (entita *State*).

*KeyboardState* má atribút *Data* v ktorom sú uložené kolekcia serializovaných objektov vyjadrujúce stlačenia kláves, digrafy a trigrafy.



OBRÁZOK 7 – MODEL TRIED SÚVISIACI S KLÁVESNICOU

Entita *MouseState*, podobne ako *KeyboardState*, obsahuje kolekciu serializovaných objektov pre kliknutia, posun myši, dvojkliky atď. Ich diagram tried je na nasledovnom obrázku.



OBRÁZOK 8 – MODEL TRIED SÚVISIACICH Z MYŠOU

Dáta klávesnice a myši sa ukladajú v binárnom formáte kvôli predpokladanému veľkému objemu a kvôli rýchlosti načítania a ukladania záznamov.

### 4.3 PRIDANIE EMOCIONÁLNEHO STAVU POUŽÍVATEĽA DO LOGOV A DÁTOVÉHO MODELU

Zodpovedné osoby: Martin Geier, Peter Greguš

V tejto časti sme sa sústredili na analýzu zdrojového kódu pri vytváraní logov používateľových akcií, ich uložení na lokálnu databázu, následnom odosielaní na server a pridania emocionálneho stavu používateľa do týchto logov.

#### 4.3.1 ANALÝZA

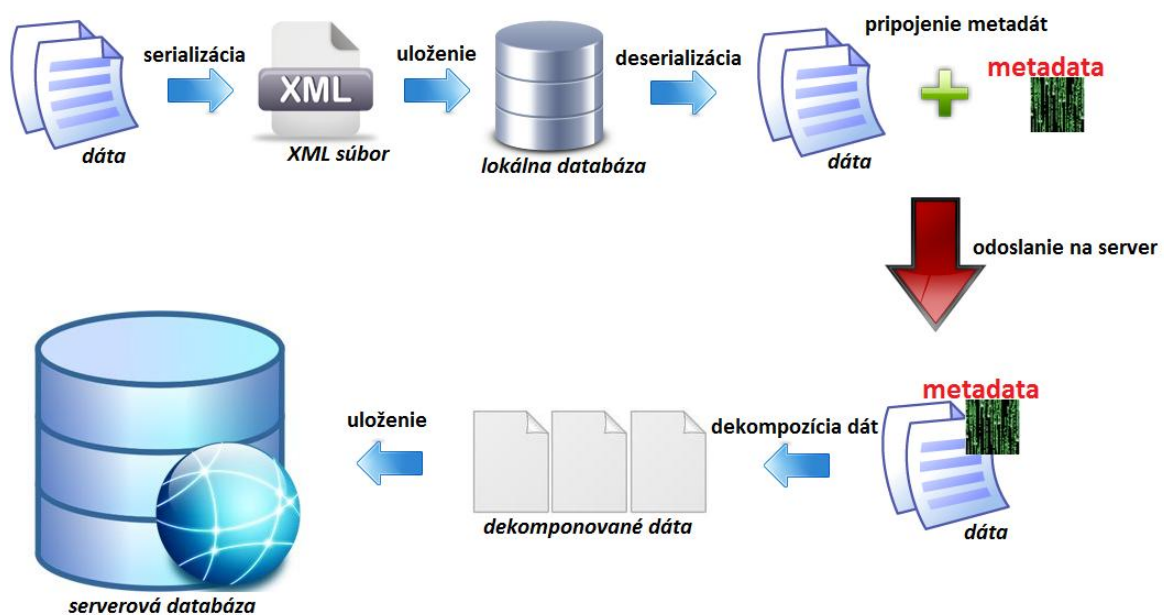
Z analýzy vyplýva, že každá sledovaná aktivita (stav klávesnice, myši) je zaznamenávaná triedou zdedenými od rozhrania *IActivityWatcher* (napr. *KeyboardStateWatcher*, *MouseStateWatcher*). Rozhranie *IActivityWatcher* predpisuje metódu *OnActivityFinishing* vracajúcu objekt typu *StateDto* nesúci konkrétny stav. Jednotlivé *Watcher*-y vytvárajú špecializované objekty zdedené od abstraktnej triedy *StateDto* (napr. *KeyboardStateDto*, *MouseStateDto*). Tieto jednotlivé objekty sú následne serializované do formátu XML uložené do lokálnej databázy. *Serializácia* vytvára XML na základe názvov atribútov dátových objektov *<Type>StateDto* a ich hodnôt.

Lokálna databáza pozostáva s jednej tabuľky, ktorej stĺpce sú nasledovné: *id* (*TEXT PRIMARY KEY*), *StartTime* (*DataTim*), *EndTime* (*DataTime*) a *Data* (*TEXT*). Začiatočný

a koncový čas nesú informáciu o trvaní konkrétneho logu a v položke *Data* sa nachádza samotný log uložený vo formáte *XML*.

Po uplynutí časového intervalu sa dáta z lokálnej databázy odošlú na server. Časový interval je nastavený v súbore *app.config* s prednastavenou hodnotou 30 minút. V tomto súbore je uložená aj adresa servera, na ktorý sa dáta odosiela. Po uplynutí stanoveného intervalu sa vyberú dáta z lokálnej databázy, deserializujú sa na objekty typu *ActivityDto*, ktorá obsahuje kolekciu objektov *<Type>StateDto* a pridá metadáta (začiatkový čas, koncový čas, názov pracovnej stanice, identifikátor používateľa). Odosielanie týchto dát je realizované prostredníctvom volania vzdialenej metódy *CommitActivity*, ktorej argumentom je objekt typu *ActivityDto*. V tejto metóde sa dáta skonvertujú na objekty, ktoré automaticky ukladajú hodnoty ich atribútov do databázy.

Autentifikácia používateľa je realizovaná pomocou atribútov triedy *ActivityDto* - názov pracovnej stanice a identifikátor používateľa.



OBRÁZOK 9 - TOK ZÍSKANÝCH DÁT O POUŽÍVATEĽOVI

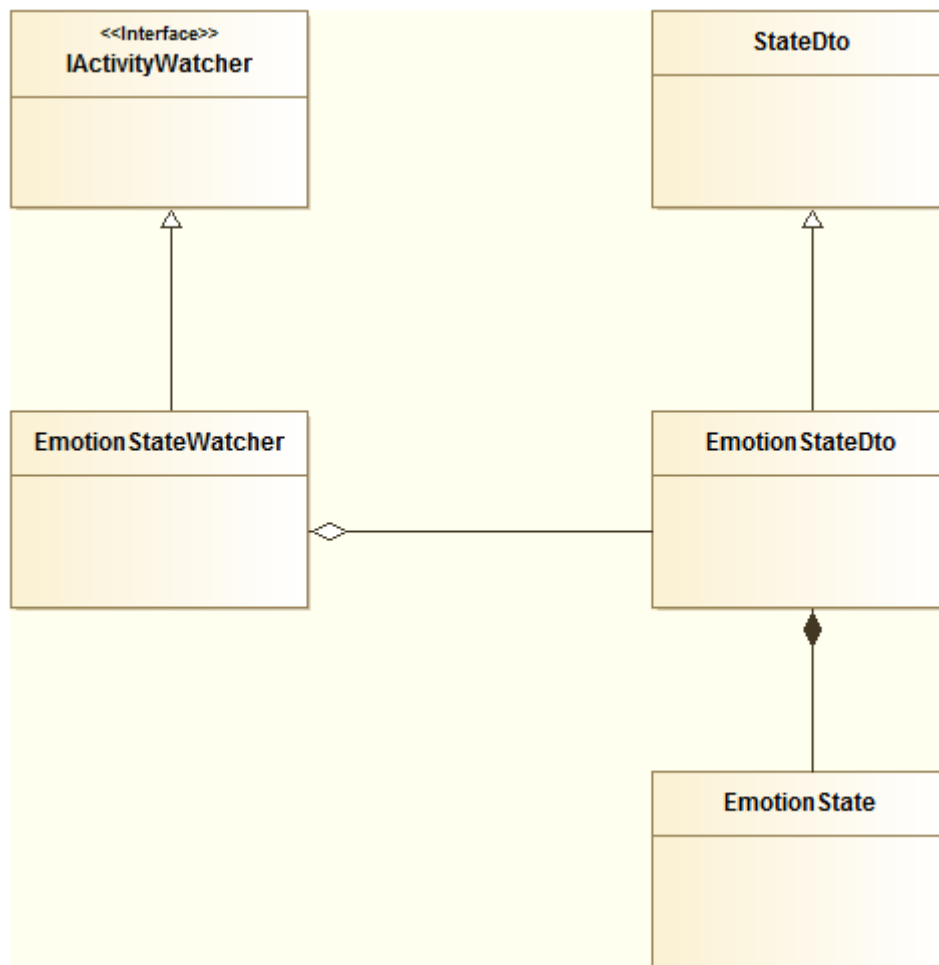
#### 4.3.2 NÁVRH

Na základe analýzy navrhujeme vytvoriť triedu *EmotionStateWatcher* zdedenú od rozhrania *IActivityWatcher*, ktorá bude uchovávať aktuálny emocionálny stav používateľa. Ten to stav sa zmení po zadaní nového stavu používateľom v *GUI* časti. Stav uvedený v tejto triede je potrebné odovzdať predpísanou metódou *OnActivityFinishing* v objekte typu *StateDto*. Preto je potrebné vytvoriť triedu *EmotionStateDto* zdedenú od *StateDto*. Táto trieda bude obsahovať jeden argument typu enumeračný typ, ktorý môže nadobúdať hodnoty (Radosť, Normálny, Stres, Únava, Hnev). Informácia o emocionálnom stave používateľa sa automaticky pridá do *XML* súboru, uloží do lokálnej databázy a odošle na server. Tam bude

v nasledujúcom kroku potrebné pridať položku emocionálny stav do databázy a pridať konvertor k týmto dátam.

### 4.3.3 IMPLEMENTÁCIA

Návrh bol realizovaný implementáciou uvedených tried *EmotionStateWatcher*, *EmotionStateDto* a enumeračného typu *EmotionState*.



OBRÁZOK 10 - IMPLEMENTÁCIA LOGOVANIA EMOCIONÁLNEHO STAVU POUŽÍVATEĽA

## 4.4 ANALÝZA EMOCIONÁLNEHO STAVU POUŽÍVATEĽA

Zodpovedné osoby: Martin Geier, Peter Greguš, Peter Šinský

Emócie používateľa sú kľúčovou oblasťou pre náš projekt. V tejto kapitole sa venujeme analýze štyroch emočných stavov, ktoré budeme rozpoznávať na základe používateľského vstupu. Analýza každej emócie prináša opis, príčiny a prejavy, a následne vhodné odporúčania, ktorých cieľom je zefektívniť prácu používateľa a dostať ho stavu negatívnej emócie do stavu pozitívnej.



#### 4.4.1 HNEV

Hnev je negatívny emocionálny prejav, ktorý vyjadruje istú reakciu na prekážku, ktorá sa stavia do cesty pri dosahovaní nejakého cieľa. To znamená, že ak používateľovi niečo napríklad bráni v plnení aktuálnej úlohy, má tendenciu začať sa rozčuľovať a potom sa práca iba zhoršuje. Hnev ako emócia má štyri úrovne: rozčúlenie (najslabšia), hnev, zlosť a zúrivosť (najsilnejšia).

##### *Príčiny hnevu*

Existuje mnoho príčin, ktoré u ľudí môžu vyvolávať prejavy hnevu. Ak berieme do úvahy prácu pri počítači a na pracovisku, hnev môžu vyvolať nasledovné podnety:

- problémy s počítačom (pomalý internet, vyskakovanie okien, atď.),
- neschopnosť plniť úlohu (používateľ začína byť frustrovaný, mrzutý a nahnevaný keď nezvláda plniť zdaný problém atď.),
- preťaženie z práce,
- potýčky na pracovisku (hádky z vedením, nedostatočné ohodnotenie, neférové zaobchádzanie atď.),
- a mnohé iné ...

##### *Prejavy hnevu*

Ak sa používateľ nachádza v stave hnevu, jeho prejavy môžu byť rozličné. V závislosti od úrovne rozčúlenia, sa môžu prejaviť prejavy ako plač, krik, búchanie po stole atď. V kontexte práce pri počítači sa hnev prejavuje najmä na schopnosti plniť si úlohy. Ak používateľ pracuje pod vplyvom hnevu, je veľmi pravdepodobné, že:

- nevie sa sústrediť na prácu,
- pri písaní textu robí veľa chýb (napr. stláča viacero kláves naraz),
- pohyby myšou môžu byť ostrejšie, tak isto kliknutia na tlačidlo.

##### *Odporúčania*

Existuje mnoho odporúčaní a návodov, ktoré umožňujú zvládať hnev. Pri práci s počítačom je dôležité túto emóciu odbúrať a nasledovné príklady odporúčaní by mali pomôcť:

- prestávka od aktuálnej práce,
- dychové cvičenia (+ počítanie),
- vyhľadať zábavu (prečítať si vtip, pozrieť si vtipné video, zahrať si hru),
- dať si malú prechádzku (ísť sa vyvetrať, pozrieť sa na výhľad z kancelárie).

#### 4.4.2 ÚNAVA

Prirodzeným dôsledkom dostatočne intenzívnej a dlhotrvajúcej činnosti je únava. Môže byť charakterizovaná ako vyčerpanie alebo strata energie a životného elánu. Toto môže byť dôsledkom jednak fyzického a v prípade kontextu našej práce najmä duševného vypätia.

##### *Príčiny únavy*

Únava môže byť spôsobená viacerými príčinami. Napríklad z fyziologického hľadiska, to môže byť nevhodná strava, nedostatok vitamínov alebo spánku. Tieto príčiny môžu byť ešte umocnené príčinami, ktoré vyplývajú práve z práce pri počítači. Keď používateľ pracuje priveľa, môže byť stresovaný a práve únava môže byť podvedomá obrana proti nemu. Ďalšími príčinami môže byť napríklad nedostatočne osvetlený priestor alebo zle vetraná kancelária. Únava je emocionálny stav, ktorý sa však môže prejaviť bez vysvetliteľných príčin.

##### *Prejavy únavy*

Únava je emocionálny stav, ktorý sa pri práci s počítačom môže prejaviť nasledovne:

- znížená produktivita,
- neschopnosť sústrediť sa na prácu,
- pomalé písanie na klávesnici,
- pomalé pohyby myšou a klikanie,
- robenie chýb.

##### *Odporúčania*

Únava môže byť sledovaná aj z dlhodobého hľadiska a na základe toho je možné používateľovi podať rôzne odporúčania. Pri dlhodobom prejavovaní je možné odporučiť:

- skvalitniť spánok (7 – 9 hodín),
- doplnenie vitamínov,
- výživové doplnky určené pre sústredenie.

Z krátkodobého hľadiska je možné únavu poraziť viacerými spôsobmi a to napríklad:

- prestávka od aktuálnej činnosti,
- káva alebo energetický nápoj,
- vyvetrať priestor,
- malú fyzickú aktivitu.

### 4.4.3 STRES

Podľa [Selye, 1976b, p. 64] stres je

*"A state manifested by a specific syndrome which consists of all the nonspecifically induced changes within the biological system."*

Stres je psychický stav, kedy je človek priamo alebo nepriamo ohrozovaný alebo ohrozenie očakáva. Stres vyvoláva mobilizovanie všetkých síl človeka na prichádzajúcu udalosť. Stres rozdeľujeme na:

- *eustres* – pozitívny stres, dosahujeme vyšší výkon, napr. očakávanie príchodu milovanej osoby,
- *distres* – negatívny stres, poškodzuje psychické aj fyzické zdravie,
- *prestres* – zvyšuje odolnosť k stresu - napr. adrenalínové športy.

#### *Príčiny stresu*

Stres vyvoláva tzv. stresor. Môže to byť:

- životná udalosť – smrť niekoho blízkeho,
- vnútorné prežívanie – spomienky na nejakú životnú udalosť,
- vonkajšie prostredie – hluk, uzavretý priestor, teplota,
- vnútorné prostredie:
  - životný štýl – nedostatok spánku, nabitý program,
  - negatívny postoj – sebakritika, pesimistické myslenie,
  - psychický postoj – nerealistické očakávanie od seba, branie vecí osobne.

#### *Prejavy stresu*

Stres má vonkajšie a vnútorné prejavy, čo spôsobuje že na prvý pohľad sa môže javiť ako neškodný problém. Prejavy stresu môžeme rozdeliť na:

- fyzické symptómy – únava, tráviace problémy,
- duševné symptómy – strata koncentrácie, problémy pri rozhodovaní,
- symptómy v správaní – nepokoj alebo nervozita,
- citové symptómy – smútok, netrpezlivosť, podráždenosť.

#### *Odporúčania*

Na stres neexistujú krátkodobé metódy riešenia. Je potrebné ho riešiť dlhodobým prístupom. Medzi tieto patrí:

- budovanie medziľudských vzťahov,
- psychoterapeutický výcvik,
- venovanie sa záľubám,

- zlepšiť stravovanie - vylúčiť kofeín,
- praktizovanie relaxačných techník,
- v najťažších prípadoch medikamentná liečba.

#### 4.4.4 RADOSŤ

Radosť je pozitívna emócia, ktorá je spojená so šťastím a dobrou náladou.

##### *Príčiny radosti*

Medzi hlavné príčiny radosti môžeme zaradiť:

- potešenie,
- dobré vzťahy,
- úspechy.

Potešenie zapríčiňuje strávenie človeku príjemnej aktivity (dobré jedlo, šport, hobby a pod.). Ďalším faktorom, ktorý významne vplyva na dobrú náladu človeka sú dobré vzťahy v jeho zázemí (s jeho rodinou, priateľmi, susedmi, kolegami a pod.). Nakoľko sú vzťahy väčšinou dlhodobé, majú aj väčší vplyv na samotnú emóciu radosti, ako potešenie z určitých aktivít, pretože tieto aktivity trvajú rádovo oveľa menej ako je trvanie vzťahov. Úspechy či už dosiahnuté v práci alebo osobnom živote majú taktiež dlhodobjší vplyv na náladu ľudí, keďže sa jedná o udalosť, ktorá ovplyvňuje ľudí na dlhší čas (povýšenie v práci, svadba, narodenie potomka a pod.). Emóciu radosti spôsobujú aj iné udalosti, avšak tri uvedené sa na nej podieľajú v najväčšej miere.

##### *Prejavy radosti*

Radosť sa pri počítači môže prejavovať nasledovne:

- zvýšená produktivita,
- väčšia miera kreativity,
- vyhýbanie sa chybám,
- rýchlejšia a cieľavedomejšia práca s myšou a klávesnicou.

##### *Odporúčania*

Ak sa človek dostane do stavu šťastia, je náročné v ňom zotrvať dlhodobo. Medzi dlhodobé „udržiavače“ človeka v dobrej nálade môžeme zaradiť

- ocenenia,
- rozmanitosť spúšťacích udalostí.

Pri oceneniach sa ľuďom vždy zvýši dobrá nálada nezávisle na počte ocenení. Pri zvyšných spúšťáčoch šťastia je potrebné udržiavať širokú paletu týchto spúšťáčov, aby sa predišlo

efektu rýchlej straty emócie radosti a dobrej nálady z dôvodu príliš veľkej frekvencie prežitia konkrétneho spúšťača.

## 5 ŠPRINT 2 - HENRIETA

### 5.1 PRIDANIE FUNKCIONALITY DO PROGRAMU *PERCONIK*

Zodpovedné osoby: Jozef Gajdoš, Martin Geier, Miroslav Hudák

#### 5.1.1 LOGOVANIE EMÓCIÍ

Na zadávanie emócií používateľom sa používa formulár, ktorý sa pýta používateľa na aktuálnu emóciu každých 30 minút.

The screenshot shows a web form with a light blue header containing the text "What is your emotion state?". Below the header, there is a dropdown menu with "Tired" selected, a "Submit" button to its right, and a text input field labeled "Any emotion:" below the dropdown.

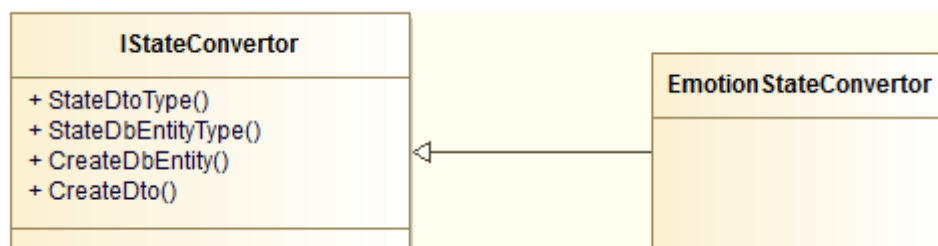
OBRÁZOK 11 - FORMULÁR PRE ZADÁVANIE EMOCIONÁLNEHO STAVU

V ďalšej fáze bude pridané kontextové menu, aby sa emócia dala zadať kedykoľvek.

Zmena zadanej emócie vygeneruje udalosť v aplikácii, ktorá vytvorí *milestone*.

#### 5.1.2 PRIDANIE KONVERTORU PRE EMÓCIE

Po pridaní emócie do tabuľky stavy bolo potrebné vytvoriť konvertor z triedy *EmotionStateDto* do *Emotion*, čo je fyzická reprezentácia dát v databáze.



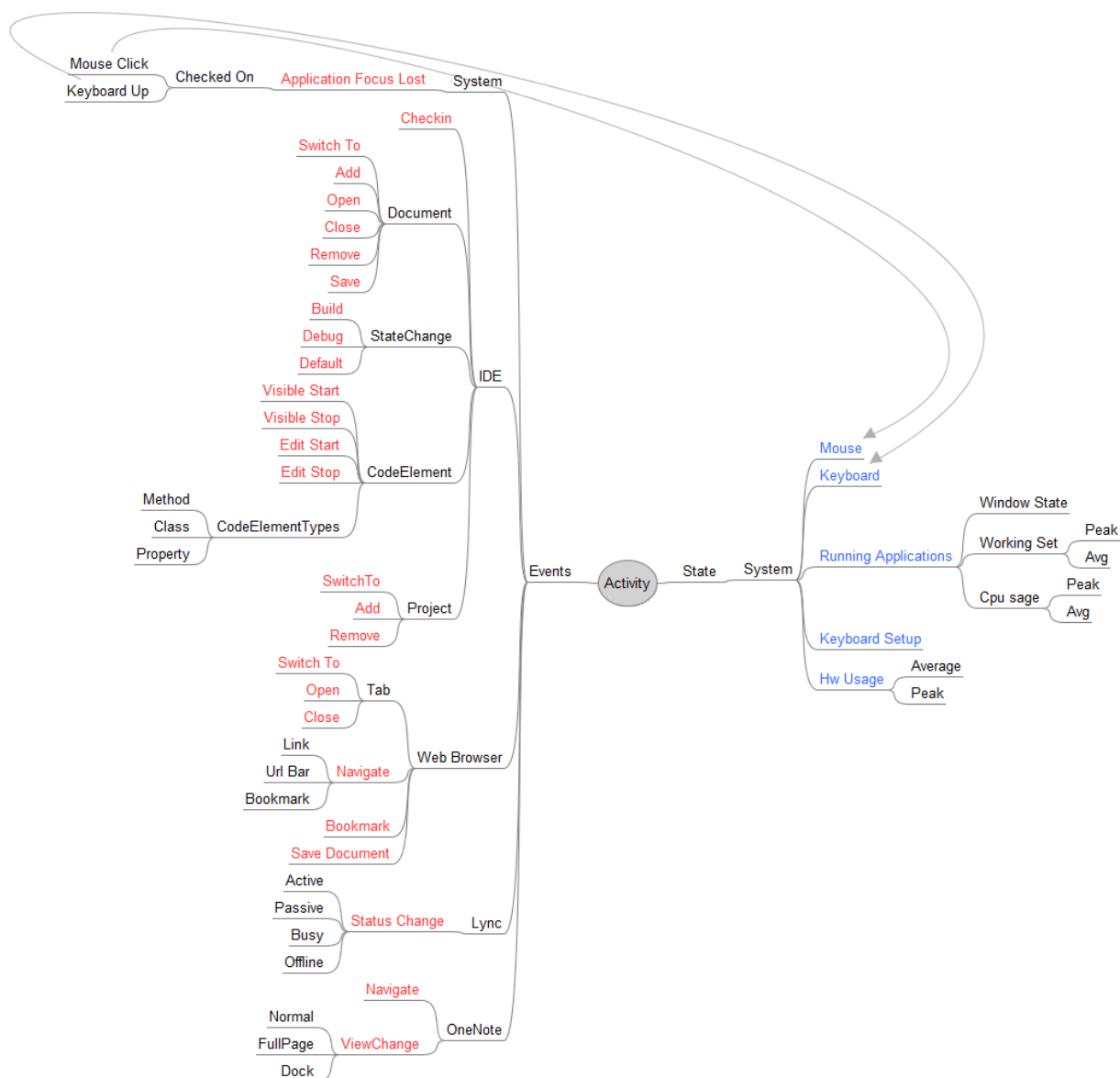
OBRÁZOK 12 - KOVERTOR PRE EMÓCIE

V triede *EmotionStateConverter* sú implementované všetky metódy definované rozhraním. Objekt je automaticky volaný podľa typu ktorý vracia metódou *StateDtoType*. Prijatá emócia je prostredníctvom objektovo relačného mapovania uložená do tabuľky *Emotion*.

### 5.1.3 PRIDANIE GENEROVANIA MÍLNIKA

Zodpovedná osoba: Miroslav Hudák

V tejto kapitole je opísaná analýza, návrh a implementácia vlastnej významnej udalosti – mílnika. Mílnik je špeciálny typ udalosti, ktorý spúšťa zber a odosielanie údajov o aktivite. Každá aktivita je zakončená práve jedným mílnikom. *PerConIK* má implementovaných viacero udalostí, ktoré sú generované jednotlivými objektami triedy *ActivityWatcher*. Nie každá udalosť je zároveň aj mílnikom. Kompletný prehľad sledovaných udalostí je znázornený na nasledujúcom obrázku.



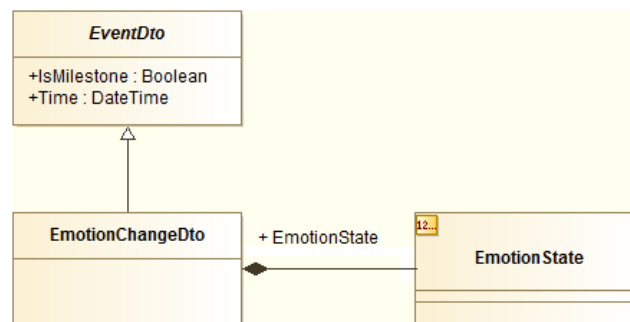
OBRÁZOK 13 - SLEDOVANÉ UDALOSTI A STAVY

Z množiny všetkých udalostí medzi míľniky patria:

- *ApplicationFocusLost*
- *LyncStatusChange*
- *IdeStateChange*
- *IdeProjectOperation*
- *IdeCheckin*

Ostatné udalosti nie sú nastavené ako míľniky, ale je možné, v prípade potreby, ich takto nakonfigurovať. Pre nás je dôležité zaznamenať udalosť vtedy, keď bola zmenená emócia. Po zmene emócie je potrebné vyvolať udalosť-míľnik, aby sa uložila predchádzajúca aktivita do databázy aj s príslušnou emóciou. Následne začne prebiehať nová aktivita s novou zmenenou emóciou.

Keď nastane zmena emócie, potrebujeme vyvolať metódu *BeginAddEvent* objektu triedy *ActivityComposer* a ako parameter zadať náš dátový objekt popisujúci emóciu. Navrhujeme vytvoriť triedu *EmotionChangeDto* zdedenú od abstraktnej triedy *EventDto*, ktorá bude uchovávať emóciu pre ukončenú aktivitu (*EmotionState*).



OBRÁZOK 14 - IMPLEMENTÁCIA VLASTNÉHO TYPU PRE UDALOSŤ

## 5.2 ANALÝZA DÁTOVÝCH ŠTRUKTÚR VSTUPNÝCH ZARIADENÍ

Zodpovedné osoby: Jozef Gajdoš, Martin Geier

V nasledujúcej časti sú opísané dátové triedy uchovávajúce informácie o vstupných zariadeniach, ktoré sú uložené v databáze a umožňujú analýzu práce s počítačom, ktorá by neskôr mohla viesť ku odhaľovaniu emocionálnych stavov používateľa.

### 5.2.1 ANALÝZA DÁTOVÝCH ŠTRUKTÚR MYŠI

*MouseMoveDto*:

- začiatková pozícia,



- koncová pozícia,
- trvanie.

Prejdená vzdialenosť (súčasne je aj rýchlosťou  $v = \frac{\partial d}{\partial t} \xrightarrow{t=1} v_i = d_i$ ).

Zrýchlenie  $a = \frac{\partial v}{\partial t} \xrightarrow{t=1} a_i = v_i - v_{i-1} = d_i - d_{i-1}$ .

*MouseDownDropDto:*

To isté ako *MouseDownMoveDto*, dedí od nej.

*MouseDownScrollDto:*

- pozícia,
- trvanie,
- otáčanie kolieskom.

*MouseDownClickDto:*

- pozícia,
- trvanie,
- typ tlačidla myši.

*MouseDownPosDto:*

Vyjadruje pozíciu v daných štruktúrach, obsahuje súradnice x a y.

## 5.2.2 ANALÝZA DÁTOVÝCH ŠTRUKTÚR KLÁVESNICE

*KeyboardGraphDto:*

- znaky,
- latencia,
- doba letu.

Latencia je doba medzi stlačením a uvoľnením klávesy. Doba letu je doba medzi stlačením prvej a stlačením druhej klávesy.

Program *PerConIK* ukladá informácie o stlačených klávesoch, A – Z, 0 – 9 a špeciálnych znakoch. Ďalej kombinácie dvoch kláves, konkrétne: 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 a troch kláves: 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.

## 5.3 NÁVRH VÝPOČTU METRÍK PRE MYŠ

Zodpovedná osoba: Peter Šinský

Myš je vstupné počítačové zariadenie, ktoré sa používa na ovládanie polohy kurzora na obrazovke a vykonávanie operácií pomocou tlačidiel. Štandardne má myš tri tlačidlá vrátane rolovacieho kolieska. Pri práci s počítačom je myš neoddeliteľnou súčasťou a okrem klávesnice je vhodným nástrojom pre zachytávanie emocionálneho stavu používateľa.

V nasledujúcej časti sme analyzovali zachytávanie akcií vytváraných myšou v programe *PerConIK*. Na základe týchto vedomostí sme vymysleli metriky, podľa ktorých budeme vytvárať model používateľa.

### 5.3.1 SLEDOVANIE MYŠI POMOCOU PROGRAMU *PERCONIK*

Program *PerConIK* je schopný zachytávať tieto základné akcie:

- pohyb,
  - štartovná a koncová pozícia (X,Y),
  - trvanie pohybu,
  - vzdialenosť,
- „*drag and drop*“,
  - štartovná a koncová pozícia (X,Y),
  - trvanie pohybu,
  - vzdialenosť,
  - stlačené tlačidlo,
- rolovanie kolieska,
  - bod,
  - trvanie,
  - delta (otočenie kolieska)
- stlačenie tlačidla,
  - pozícia (X,Y),
  - trvanie,
  - stlačené tlačidlo.

### 5.3.2 METRIKY PRE SLEDOVANIE EMOČNÉHO STAVU POUŽÍVATEĽA

Pre vytvorenie modelu používateľa, na základe ktorého budeme schopný rozoznať používateľovo správanie, je potrebné vytvoriť metriky – sledovateľný údaj, ktorý budeme pozorovať pri práci používateľa s myšou. Tieto metriky sú vytvorené na základe vyššie spomenutých údajov, ktoré zachytáva program *PerConIK*.

*Rýchlosť kurzora*

Pohyb je najzákladnejšia akcia, ktorú je možné s myšou vykonávať. Pohybom myši nastavujeme kurzor na obrazovke na miesto, kde chceme vykonávať ďalšiu interakciu. Vzhľadom k tomu sa metrika javí ako vhodný kandidát pre rozoznávanie emócie. Rýchlosť pohybu kurzora ako metrika je vyjadrená nasledovne:

$$\text{Rýchlosť kurzora} = \frac{\text{prejdená vzdialenosť}}{\text{Čas}}$$

Rýchlosť je možné takto jednoducho sledovať. Predpokladáme, že pri zmene emočného stavu je sa bude meniť aj rýchlosť pohybu kurzora. Napr. pri rozčúlení môže používateľ reagovať agresívnejšie a rýchlosť bude vyššia, naopak pri únave môže byť rýchlosť výrazne pomalšia.

#### *Rýchlosť rolovania kolieska*

Pri prehliadaní internetu alebo rôznych dokumentov sa veľmi často používa na posúvanie koliesko. Týmto je možné posúvať prehliadaný obsah. Rovnako ako pri rýchlosti kurzora predpokladáme, že sa bude meniť vzhľadom na emočný stav. Rýchlosť rolovania kolieska je vyjadrená nasledovne:

$$\text{Rýchlosť rolovania} = \frac{\text{Delta}}{\text{Čas}}$$

#### *Trvanie stlačenia tlačidiel*

Pre vykonanie rôznych akcií je potrebné používať tlačidlá myši. Štandardne má myš tri tlačidlá. Trvanie stlačenia tlačidiel sa budeme sledovať pre všetky typy tlačidiel a ich stlačení, t.j. jednoduché stlačenie tlačidla, alebo dvojité.

## 5.4 NÁVRH VÝPOČTU METRÍK PRE KLÁVESNICU

Zodpovedná osoba: Peter Sivák

Klávesnica je vstupné počítačové zariadenie, ktoré sa používa prevažne na písanie textu, ale takisto slúži aj na vykonávanie operácií pomocou klávesových skratiek. Štandardne má klávesnica alfanumerickú, funkčnú a niekedy aj numerickú časť. Pri práci s počítačom je klávesnica neoddeliteľnou súčasťou a okrem myši je vhodným nástrojom pre zachytávanie emocionálneho stavu používateľa.

V nasledujúcej časti sme analyzovali zachytávanie akcií vytváraných klávesnicou v programe *PerConIK*. Na základe týchto vedomostí sme vymysleli metriky, podľa ktorých budeme vytvárať model používateľa.

### 5.4.1 SLEDOVANIE KLÁVESNICE POMOCOU PROGRAMU *PERCONIK*

Program *PerConIK* je schopný vytvárať nasledovné zoznamy:

- „*Graphs*“ – zoznam stlačených znakov

- „*DiGraphs*“ – zoznam stlačených dvojíc znakov
- „*TriGraphs*“ – zoznam stlačených trojíc znakov
- „*Shortcuts*“ – zoznam klávesových skratiek

Každý z vymenovaných zoznamov si v sebe uchováva nasledovné trojice údajov:

- „*Character*“ – hodnota stlačených znakov
- „*Latency*“ – latencia stlačenia
- „*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 klávesy – „*Graph*“ latencia ( $L_t$ )
- „*DiGraph*“ latencia ( $D_t$ ) – dvojica kláves
- „*TriGraph*“ latencia ( $T_t$ ) – trojica kláves

Pre každý typ latencií sa súčasne vyráta aj čas letu stlačenia ( $F_t$ ), ktorý predstavuje interval medzi stlačením rôznych kláves, takže zároveň určujú, či došlo k ich prekryvaniu.

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.

#### 5.4.2 METRIKY PRE SLEDOVANIE EMOČNÉHO STAVU POUŽÍVATEĽA

Pre vytvorenie modelu používateľa, na základe ktorého budeme schopný rozoznať používateľovo správanie, je potrebné vytvoriť metriky – sledovateľný údaj, ktorý budeme pozorovať pri práci používateľa s klávesnicou. Tieto metriky sú vytvorené na základe vyššie spomenutých údajov, ktoré zachytáva program *PerConIK*.

##### *Frekvencia stlačených kláves*

Stláčanie kláves je hlavná akcia, ktorú je možné s klávesnicou vykonávať. Stláčaním kláves píšeme text do aplikácií, alebo vykonávame rôzne operácie použitím klávesových skratiek.

Vzhľadom k tomu sa metrika javí ako vhodný kandidát pre rozpoznávanie emócie. Frekvencia stlačených kláves ako metrika je vyjadrená nasledovne:

$$\text{Frekvencia stlačených kláves} = \frac{\text{počet stlačených kláves}}{\text{Čas}}$$

Frekvenciu je možné takto jednoducho sledovať. Predpokladáme, že pri zmene emočného stavu sa bude meniť aj frekvencia stlačených kláves. Napr. pri rozčúlení môže používateľ reagovať agresívnejšie a frekvencia bude vyššia, naopak pri únave môže byť frekvencia výrazne menšia.

#### *Doba letu*

Ďalšou metrikou je doba letu. Reprezentuje trvanie medzi stlačením kláves. Hodnoty získame priamo zo získaných dát logovača.

#### *Latencia*

Ďalšou metrikou je latencia. Reprezentuje dobu stlačenia klávesy. Hodnoty získame priamo zo získaných dát logovača.

#### *Vyrovnanosť pri písaní*

Ďalšou zaujímavou metrikou je vyrovnanosť pri písaní textu. Keď používateľ píše na klávesnici nevyrovnane, môžeme predpokladať, že je nervóznejší ako obvykle. Naopak keď je jeho štýl písania vyrovnanejší, t.j. čas letu medzi jednotlivými klávesmi je približne rovnaký, používateľ sa javí byť pokojnejší a sústredenejší. Vyrovnanosť pri písaní je vyjadrená nasledovne:

$$\text{Vyrovnanosť pri písaní} = \sum_{i=1}^n (\text{FlightTime}_i - \text{PriemernýFlightTime})^2$$

Výsledok 0 vyjadruje maximálnu vyrovnanosť pri písaní a čím je daný výsledok vyšší, tým je používateľ menej vyrovnaný.

#### *Frekvencia výskytu chýb*

Pri častom písaní textu na klávesnici sa používateľ dopúšťa chýb. Niekedy urobí viac chýb pri písaní a niekedy menej. Túto skutočnosť taktiež považujeme za dôležitú metriku, pretože čím viac chýb používateľ robí, tým je vzhľadom na emočný stav nervóznejší, ale takisto môže u neho prevládať hnev alebo aj únava, kedy už nie je plne koncentrovaný. Naopak, keď sa používateľ dopustí menšieho počtu chýb, môžeme predpokladať, že je pokojnejší a sústredenejší. Frekvencia výskytu chýb je vyjadrená nasledovne:

$$\text{Frekvencia výskytu chýb} = \frac{\text{Počet stlačení backspace klávesy}}{\text{Čas}}$$

## 5.5 NÁVRH MODELU POUŽÍVATEĽA

Zodpovedná osoba: Peter Greguš

Model používateľa z predchádzajúcej časti môžeme reprezentovať vektorom hodnôt metrick pre myš a klávesnicu. Môžeme si zhrnúť uvedené metriky.

Metriky pre myš:

- rýchlosť kurzora,
- rýchlosť rolovania kolieska,
- trvanie stlačenia tlačidiel.

Metriky pre klávesnicu:

- latencia *graph-ov*, *digraph-ov*, *trigraph-ov*,
- doba letu *digraph-ov*, *trigraph-ov*,
- frekvencia stlačených kláves,
- vyrovnanosť pri písaní,
- frekvencia výskytu chýb.

Tieto metriky budú reprezentované ako grafy početností jednotlivých hodnôt. Pomocou nich dokážeme vidieť správanie používateľa v jednotlivých emocionálnych rozpoloženiach. Po ich normalizácii ku množstvu získaných dát pomocou metódy logovania môžeme porovnávať konkrétne emócie, čo nám môže pomôcť pri rozpoznávaní modelu používateľa.

## 5.6 IMPLEMENTÁCIA MODELU POUŽÍVATEĽA

Zodpovedné osoby: Jozef Gajdoš, Martin Geier, Peter Greguš

### 5.6.1 PRIDANÉ ENTITY DO DÁTOVÉHO MODELU

Do modelu databázy boli pridané tri nové tabuľky:

- *UserModel*,
- *EmotionVector*,
- *Emotion*.

Teraz si ich podobnejšie popíšeme:

*UserModel*

Entita predstavuje model používateľa, ktorý v sebe agreguje modeli jednotlivých emócií. Tie sú uložené ako binárne dáta do typu *BLOB*. Modeli jednotlivých emócií sa ukladajú pomocou serializácie. Ďalej obsahuje svoje *Id* a je asociovaný ku konkrétnemu používateľovi.

*EmotionVector*

Entita *EmotionVector* slúži na dočasné ukladanie modelu emócie. Priamo v sebe má uvedenú emóciu a čas uloženia. Takisto sú jednotlivé záznamy asociované ku konkrétnym používateľom.

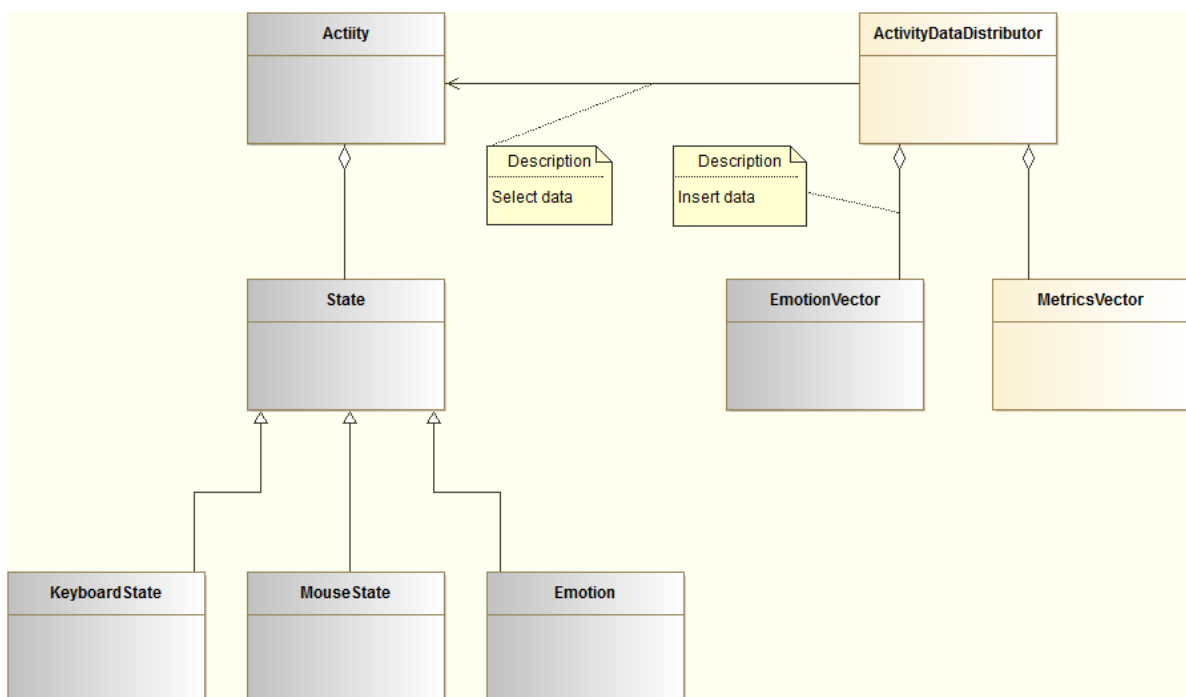
### *Emotion*

Entita v sebe ukladá emóciu zadanú používateľom a aj vypočítanú hodnotu, obe hodnoty v čase vzniku nemusia byť uvedené.

Entita *Emotion* dedí od entity *State*.

## 5.6.2 VYTVÁRANIE VEKTORU METRÍK AKO MODEL POUŽÍVATEĽA

Vektor emócie vytvoríme naplnením vektora metrík pre jednotlivé emócie. Z databázy vyberieme dáta získané od používateľa, ktoré následne spracujeme a pomocou vzorcov uvedených v časti návrh výpočtu metrík a uložíme do príslušného vektora. Po príchode ďalších dát sa bude tento vektor prepočítavať a meniť podľa správania používateľa.



OBRÁZOK 15 - VYTVORENIE MODELU POUŽÍVATEĽA

Na predošlom obrázku je šedou farbou znázornená databázová časť a hnedou časť implementačná.

Implementované metriky:

- rýchlosť kurzora,
- akcelerácia kurzora,
- rýchlosť rolovania kolieska,

- trvanie stlačenia tlačidiel.
- latencia *graph-ov*, *digraph-ov*, *trigraph-ov*,
- doba letu *digraph-ov*, *trigraph-ov*.

Trieda *MetricsVector* reprezentuje model používateľa v ktorom sú údaje zapísané nasledovnou formou:

```
/*KEYBOARD*/
public Dictionary<string,int[]> graphLatency = new Dictionary<string, int[]>();
public Dictionary<string,int[]> diGraphLatency = new Dictionary<string, int[]>();
public Dictionary<string,int[]> triGraphLatency = new Dictionary<string, int[]>();
public Dictionary<string,int[]> diGraphFlightTime = new Dictionary<string, int[]>();
public Dictionary<string,int[]> triGraphFlightTime = new Dictionary<string, int[]>();
/*MOUSE*/
public int[] mouseSpeed = new int[vectorSize];
public int[] mouseAcceleration = new int[vectorSize];
public int[] scrollSpeed = new int[vectorSize];
public Dictionary<MouseClickedTypeEnum,int[]> clickDuration = new
Dictionary<MouseClickedTypeEnum, int[]>();
```

Tieto dáta sú následne uložené do databázy ako serializované dáta v *BLOB* forme.