Personalizované vyhľadávanie v zdrojových kódoch

Diplomová práca

SLOVENSKÁ TECHNICKÁ
UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY
A INFORMAČNÝCH TECHNOLÓGIÍ

PeWe@FIIT
personalized web group

Richard Sámela
vedúci Ing. Eduard Kuric

18. 03. 2014

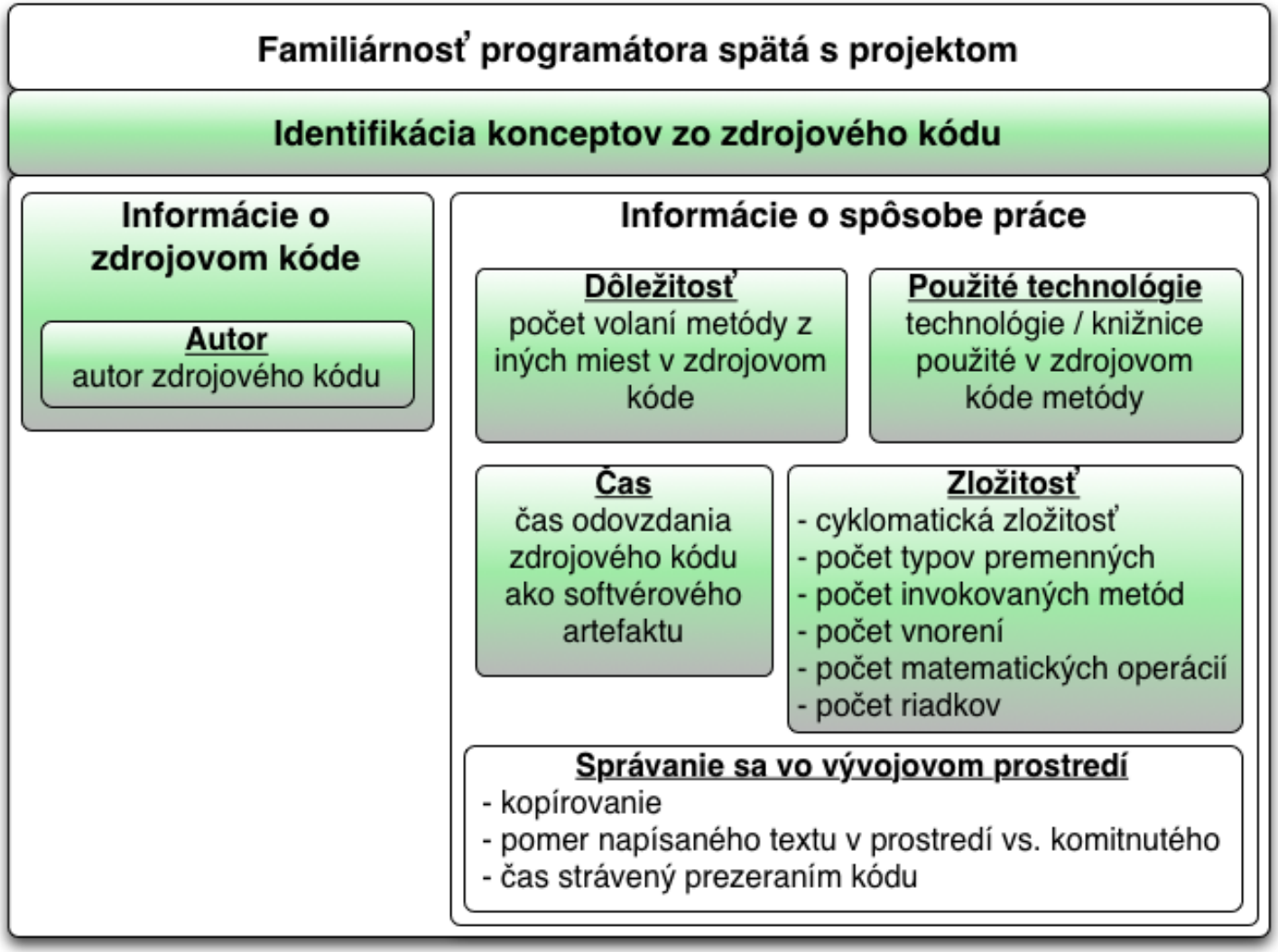
Cieľ

- navrhnúť a implementovať model skúseností programátora založený na:
  - konceptoch extrahovaných zo zdrojového kódu, ktorý vytvoril
  - renomé programátora
    - spôsob, akým pracuje
    - analýzou kódu, ktorý vytvára
- UC1: použitie modelu programátora pri vyhľadávaní za účelom zvýšenia relevancie výsledkov
- UC2: dohľadanie odborníka pre dopytovaný koncept

Riešenie

- extrahovanie konceptov zo zdrojového kódu
- metóda na identifikáciu autorov v zdrojovom kóde z verziovacieho systému
- priradenie konceptov autorom kódu z ktorého boli extrahované
- vytvorenie modelu programátora

Model programátora



Identifikácia konceptov

- algoritmus LDA (Latent Dirichlet Allocation)
- predspracovanie zdrojových súborov
- natrénovanie modelu
- predspracovanie fragmentu kódu
- vykonanie inferencie nad zdrojovým kódom

Latent Dirichlet Allocation

- JGibbLDA (Java)
- trénovacie dáta - Java projekty (neo4j, lucene, elasticsearch, minecraft...)
- veľkosť dát:
  - 4062 balíkov
  - 94220 Java tried
- obsah dokumentov: zdrojový kód, komentáre

Predspracovanie

- parsovanie Java tried
- tokenizácia zdrojového kódu - INTT (Java)
- odstranenie stop slov
  - syntax Java
  - anglický jazyk (komentáre)

Predspracovanie - ukážka

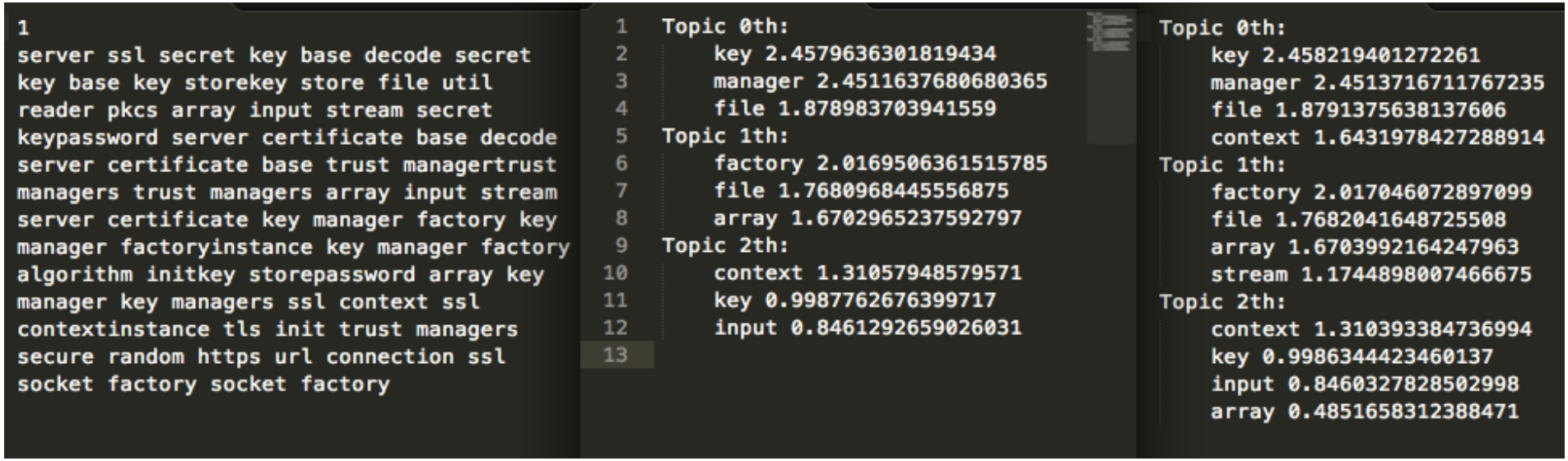
```
public static void allowServerSSL() {
    byte[] secretKey = Base64.decode(secretKeyBase64);
    KeyStore publicKeyStore = FileUtil.readerPKCS12(new ByteArrayInputStream(secretKey), password);
    byte[] serverCertificate = Base64.decode(serverCertificateBase64);
    TrustManager[] trustManagers = getTrustManagers(new ByteArrayInputStream(serverCertificate));
    KeyManagerFactory kf = KeyManagerFactory.getInstance(KeyManagerFactory.getDefaultAlgorithm());
    kf.init(publicKeyStore, password.toCharArray());
    KeyManager[] km = kf.getKeyManagers();
    SSLContext sc = SSLContext.getInstance("TLS");
    sc.init(km, trustManagers, new SecureRandom());
    HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
}

server ssl secret key base decode secret key base key storekey store file util reader pkcs array input
stream secret keypassword server certificate base decode server certificate base trust managertrust
managers trust managers array input stream server certificate key manager factory key manager
factoryinstance key manager factory algorithm initkey storepassword array key manager key managers ssl
context ssl contextinstance tls init trust managers secure random https url connection ssl socket factory
socket factory
```

Trénovanie LDA

- trénovanie modelu na úrovni tried v Java
- počet topicov (3)
- počet slov spadajúcich do konceptu (3)
- veľkosť modela - 261MB

Trénovanie LDA - výstup



```
1
server ssl secret key base decode secret
key base key storekey store file util
reader pkcs array input stream secret
keypassword server certificate base decode
server certificate base trust managertrust
managers trust managers array input stream
server certificate key manager factory key
manager factoryinstance key manager factory
algorithm initkey storepassword array key
manager key managers ssl context ssl
contextinstance tls init trust managers
secure random https url connection ssl
socket factory socket factory
```

```
Topic 0th:
    key 2.4579636301819434
    manager 2.4511637680680365
    file 1.878983703941559
Topic 1th:
    factory 2.0169506361515785
    file 1.7680968445556875
    array 1.6702965237592797
Topic 2th:
    context 1.31057948579571
    key 0.9987762676399717
    input 0.8461292659026031
```

```
Topic 0th:
    key 2.458219401272261
    manager 2.4513716711767235
    file 1.8791375638137606
    context 1.6431978427288914
Topic 1th:
    factory 2.017046072897099
    file 1.7682041648725508
    array 1.6703992164247963
    stream 1.1744898007466675
Topic 2th:
    context 1.310393384736994
    key 0.9986344423460137
    input 0.8460327828502998
    array 0.4851658312388471
```

Identifikácia autorov

- riadky -> metódy
- ignorácia nepotrebných riadkov a metód
  - formátovanie a syntax ("}", deklarácia premennej)
  - jednoduché metódy (set/get)
- Problém refaktorovania
    - zmena deklarácie (názov, vstupné parametre)
    - zmena kódu v metóde
    - presun metódy do inej triedy
    - kombinácia viacerých

Identifikácia autorov - algoritmus

- snapshot projektu (úroveň riadkov kódu)
- pridelenie autora posledneho commitu
- iteracia verziami projektu
- aktualizovanie autora počiatočným riadkom kódu na základe zmien z projekte

Informácie o metóde

- počet invokácií danej metódy (Control Flow Graph)
- autor
- rôzne metriky zložitosti
  - počet typov, riadkov, invokácií iných metód, počet vnorení, cyklomatická zložitosť

Nasledujúci plán

- formulovať vzťahy a overiť závislosti medzi jednotlivými atribútmi modelu programátora