Softvérové nástroje pre získavanie charakteristík používateľa

Anton ANDREJKO, Michal BARLA, Mária BIELIKOVÁ, Michal TVAROŽEK

Ústav informatiky a softvérového inžinierstva
Fakulta informatiky a informačných technológií
Slovenská technická univerzita v Bratislave
{andrejko,barla,bielik,tvarozek}@fiit.stuba.sk

Abstrakt. Prispôsobovanie sa charakteristikám používateľov sa stáva významnou vlastnosťou súčasných webových informačných systémov. Bez prispôsobovania prezentácie môže byť veľmi ťažké a dokonca často nemožné používať tieto systémy na efektívnu prácu s informáciami. Na prispôsobovanie je potrebné poznať charakteristiky jednotlivých používateľov. V príspevku opisujeme získavanie charakteristík používateľa pomocou spolupracujúcich softvérových nástrojov, kde každý nástroj vykonáva špecifickú časť tohto procesu. Charakteristiky získavame na základe monitorovania správania sa používateľa, čo minimalizuje mieru potrebného zapojenia používateľa do procesu tvorby modelu používateľa. Opisujeme štyri nástroje s dôrazom na ich vzájomné prepojenia, ktoré vedú k získavaniu charakteristík používateľa.

Kľúčové slová: model používateľa, objavovanie charakteristík používateľa, vzory správania, zaznamenávanie akcií používateľa.

1 Úvod

Dôležitou súčasťou adaptívnych webových systémov je modelovanie používateľa, ktoré zahŕňa najmä získanie a vyhodnotenie charakteristík používateľa tak, aby sa mohli použiť pri prispôsobovaní. S nárastom množstva dostupných informácií sa prispôsobovanie zamerané na konkrétneho používateľa – personalizácia stáva nevyhnutnou súčasťou súčasných webových informačných systémov. Dopyt po generických riešeniach adaptívnych systémov vyúsťuje do potreby metód a nástrojov, ktoré umožňujú automatické získavanie a vyhodnocovanie charakteristík používateľa.

Vytvorenie vhodného modelu používateľa je zložitý proces, ktorý zahŕňa viacero úloh. K ich realizácii môžeme pristúpiť vytvorením sady softvérových nástrojov, ktoré sa špecializujú na jednotlivé úlohy. Príkladom môže byť nástroj, ktorý spája informácie získané od používateľa počas registrácie z príslušných formulárov s informáciami získanými z iných nástrojov do modelu používateľa, ktorého cieľom je reprezentovať dobre odhadnuté charakteristiky používateľa.

Cieľom nášho výskumu je tvorba modelu používateľa na odporúčanie vhodných ponúk z konkrétnej domény (napr. pracovné alebo cestovné ponuky), ktorý tvorí základ pre personalizovanú navigáciu a filtrovanie obsahu pre jednotlivých používateľov. Na vyhodnotenie výsledkov nášho výskumu používame doménový model a model

používateľa vytvorený v rámci výskumného projektu „Nástroje pre získavanie, organizovanie a udržovanie znalostí v prostredí heterogénnych informačných zdrojov“ [6], ktorý sa orientuje na doménu pracovných ponúk.

V tomto príspevku prezentujeme niekoľko softvérových nástrojov, ktoré podporujú proces získavania charakteristík používateľa spolu s ich prepojeniami. V časti 2 prezentujeme proces vytvárania modelu používateľa a možné zdroje informácií, ktoré do procesu vstupujú. Definujeme sadu nástrojov, ktoré realizujú potrebné úlohy. V časti 3 sa venujeme špecifikám fázy zberu dát a opisujeme príslušné nástroje tejto fázy (*Click* – Client Side Action Recorder a *SemanticLog*). V časti 4 opisujeme spracovanie zozbieraných informácií do podoby odhadov charakteristík používateľa a ich realizáciu nástrojom *LogAnalyzer*. Nakoniec sumarizujeme príspevok a závery z našej práce prezentovanej v tomto príspevku.

2 Proces vytvárania modelu používateľa

Tvorba modelu používateľa pozostáva z dvoch hlavných fáz. Najprv sa získajú dáta o správaní používateľa, ktoré sa ďalej analyzujú a interpretujú s cieľom objavenia relevantných charakteristík používateľa. Takto objavené charakteristiky sa použijú pri personalizácii. Existuje viacero spôsobov získavania dát o používateľoch:

- formuláre, ktoré umožňujú používateľom priamo zadať požadované charakteristiky alebo odpovedať na špecifické otázky, ktoré charakteristiky odhaľujú nepriamo;
- explicitná spätná väzba používateľa na zobrazený obsah; spätná väzba môže byť buď pozitívna alebo negatívna;
- dokumenty dodané používateľom (napr. CV), z ktorých sa dajú extrahovať relevantné charakteristiky;
- záznam prístupov používateľa k zdrojom; používa sa na odvodenie stupňa znalosti používateľa o prezentovaných informáciách;
- záznam správania používateľa v systéme, ktorý zahŕňa implicitnú spätnú väzbu používateľa na zobrazený obsah.

Keďže je vhodné minimalizovať mieru spoluúčasti používateľa na procese vytvárania modelu používateľa, sústredíme sa na modelovanie používateľa založené na pozorovaní správania používateľa a automatickom odvodení jeho charakteristík. Identifikovali sme sadu softvérových nástrojov, ktoré pokrývajú obidve fázy procesu. V tomto príspevku opisujeme nástroje *Click* a *SemanticLog*, ktoré sú určené pre fázu získavania dát a nástroj *LogAnalyzer*, ktorý je zameraný na objavovanie charakteristík používateľa v získaných dátach.

3 Zber dát

Existujú dva hlavné prístupy monitorovania akcií používateľa: monitorovanie vykonávané na strane servera a monitorovanie na klientskej strane systému, pričom sa môže použiť aj kombinácia oboch prístupov.

Monitorovanie na strane servera sleduje prístupy používateľa ku zdrojom. Hlavná nevýhoda tohto prístupu je v tom, že neumožňuje zaznamenávanie niektorých typov interakcie používateľa so systémom (napr. interakcia s prvkami formulára) a neposkytuje presné informácie o čase zobrazenia informácií. Monitorovanie na strane servera závisí od správania webových prehliadačov, ktoré zvyčajne nežiadajú znovu od servera už navštívené stránky, ale použijú namiesto toho kópiu uloženú vo vyrovnávacej pamäti. Systém teda nemá žiadne informácie o dobe, ktorú používateľ strávil prezeraním určitej stránky. Väčšina v súčasnosti najrozšírenejších webových prehliadačov nerešpektuje direktívy HTTP protokolu, ktoré zakazujú používanie lokálnej vyrovnávacej pamäte, takže ich použitie nerieši uvedený problém. Takisto treba vziať do úvahy znížený komfort používania systému bez vyrovnávacej pamäte. Aj v [8] autori uvádzajú závery, že pre získanie presných záznamov o interakcii používateľa so systémom treba využiť monitorovanie na strane klienta. Aj napriek vyššie uvedeným problémom je prístup s využitím záznamov servera postačujúci pre veľa adaptívnych systémov. Napr. systém AHA! (Adaptive Hypermedia for All, `aha.win.tue.nl/`) tieto záznamy využíva pre sledovanie toho, aké dokumenty sa už zobrazili používateľovi [1].

Monitorovanie na strane klienta vytvára podrobný záznam akcií používateľa s presnými časovými pečiatkami. Monitorovanie sa môže vykonávať špeciálnou klientskou aplikáciou (napr. *User Action Recorder* v [10]) alebo použitím klientských webových technológií akými sú JavaScript alebo Java applety. Pretože považujeme prvý zmienený prístup za pomerne invazívny a málo flexibilný, sústreďujeme sa druhý prístup. JavaScript a Java applety v súčasnosti podporujú všetky významné webové prehliadače na hlavných platformách. Možnou nevýhodou je, že nie každý používateľ akceptuje takéto podrobné monitorovanie a niektorí môžu vo svojich prehliadačoch zablokovať vykonávanie vložených skriptov. Napokon môže nastať aj prípad, keď používateľ nemá na počítači nainštalovaný potrebný softvér.

Existuje viacero nástrojov, ktoré využívajú JavaScript na monitorovanie na strane klienta ako sú WebVip (Web Variable Instrumenter Program, `zing.ncsl.nist.gov/WebTools/WebVIP/overview.html`) alebo WET (Web Event-logging Tool, [3]). Obidva nástroje sú primárne určené pre účely vyhodnocovania použiteľnosti webových stránok. Nevýhodou nástroja WebVIP je vytvorenie kópie celého webového sídla, do ktorej nástroj do jednotlivých stránok pridáva identifikátory a obsluhy udalostí na HTML odkazoch. Nástroj sa preto nedá použiť pri dynamicky vytváraných webových stránkach, keďže mu nedokážeme dodať kompletné webové sídlo. Ďalšou nevýhodou nástroja WebVIP je jeho obmedzenie na štandardné HTML odkazy ignorujúce ďalšie HTML elementy stránok. Aj keď nástroj WET rieši uvedené problémy, je príliš naviazaný na doménu vyhodnocovania stránok, čo znemožňuje jeho použitie ako všeobecného nástroja na zaznamenávanie akcií používateľa. Nástroj vytvára nad stránkou ďalšiu vrstvu, ktorá obsahuje tlačidlá na ovládanie procesu zaznamenávania, pričom dáta sa posielajú na server až po stlačení tlačidla stop.

Z uvedenej analýzy vyplýva, že pri použití monitorovania na strane klienta sa vystavujeme riziku, že používateľ nebude súhlasiť s podrobným monitorovaním, čo by znamenalo, že nezískame žiadne dáta. To je silný argument proti použitiu čistého monitorovania na strane klienta. Monitorovanie na strane servera je z tohto pohľadu spoľahlivejšie, keďže sa mu vždy podarí získať nejaké dáta. Nevýhodou je, že môžeme stratiť časový aspekt zaznamenaných akcií. Náš prístup je založený na myšlienke

skombinovania oboch prístupov – naše nástroje extrahujú maximum informácií na strane servera a používajú záznam z klientskeho monitoringu ako zdroj prídavných, časovo presných informácií o aktivitách používateľa.

3.1 Nástroj na zaznamenávanie akcií používateľa na strane klienta

Click je nástroj, ktorý realizuje monitorovanie používateľa na strane klienta a zachytáva udalosti, ktoré vyvoláva webový prehliadač počas interakcie používateľa s elementmi zobrazovaných stránok:

- *Load:* zobrazenie stránky používateľovi,
- *Unload:* keď používateľ stránku opustí,
- *Click:* keď používateľ nasleduje niektorý odkaz na stránke,
- *Mouseover:* keď používateľ nasmeruje kurzor na aktívny element stránky,
- *Mouseout:* keď používateľ odstráni kurzor z aktívneho elementu stránky.

Uvažujeme aj zachytávanie ďalších udalostí, ktoré sú špecifické pre stranu klienta a nie sú dostupné na serveri. Medzi také udalosti patrí udalosť *change*, ktorá sa vyvolá po zmene obsahu prvku formulára. Sekvencia týchto udalostí informuje o poradí, v ktorom používateľ vypĺňal formulár. Iným príkladom je udalosť *scroll*, ktorá sa vyvolá, ak sa používateľ posunie po stránke.

Click zbiera nasledovné údaje o každej zachytenej udalosti:

- typ udalosti,
- čas spustenia udalosti a
- kontext zachytenej udalosti, napr. ktorý odkaz používateľ nasledoval.

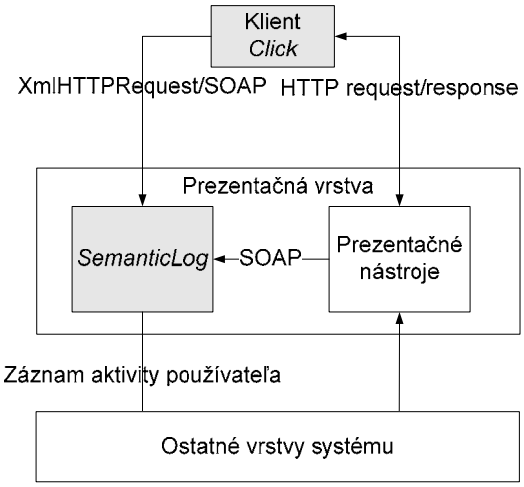
Komunikácia so serverom je realizovaná v dávkach *N* zachytených udalostí. Ak je *N* nastavené na jedna, *Click* posiela na server záznam po každej zachytenej udalosti.

Click je implementovaný s využitím JavaScript technológie, ktorá nekladie žiadne špeciálne požiadavky na klienta, keďže je podporovaná väčšinou súčasných webových prehliadačov. JavaScript poskytuje natívny prístup k objektovému modelu HTML dokumentu (DOM), takže *Click* má prístup k štruktúre aj obsahu stránky.

Zachytávanie udalostí je založené na špecifikácii *DOM Level 2 Events* (`www.w3.org/TR/DOM-Level-2-Events`), ktorá definuje obsluhovače udalostí dynamicky priraďované k elementom objektového modelu. To umožňuje jednoducho integrovať nástroj do existujúcich ako aj dynamicky generovaných HTML stránok pridaním odkazu na príslušný skript v hlavičke HTML dokumentu.

Komunikácia so serverom je asynchrónna s použitím objektu *XmlHTTPRequest* (pozri obr. 1), ktorý predstavuje jadro technológie AJAX. Nástroj *Click* posiela záznamy zachytených udalostí na SOAP rozhranie nástroja *SemanticLog*. Ten agreguje získané dáta s informáciami získanými z prezentačných nástrojov, ktoré poznajú sémantiku zobrazeného obsahu. Nakoniec sa záznam obohatený o sémantiku zašle na ďalšie spracovanie vnútornými vrstvami systému.

Serverovú časť monitorovacieho systému sme realizovali ako samostatný nástroj, ktorý prijíma záznamy udalostí vytvárané nástrojom *Click* cez SOAP rozhranie webovej služby. Pre volanie metódy webovej služby z prostredia JavaScript používame knižnicu od IBM (Call SOAP Web services with Ajax).



Obr. 1. Štruktúra časti pre získavanie dát.

3.2 Nástroj na zaznamenávanie udalostí so znalosťou sémantiky

SemanticLog je generický nástroj na vytváranie serverového záznamu, ktorý vykonáva dve hlavné úlohy. Prvou je zaznamenávanie udalostí dodaných od nástrojov prezentačnej vrstvy, druhou je prijímanie a zaznamenávanie udalostí od nástrojov vykonávajúcich monitorovanie na strane klienta ako napr. nástroj *Click.*

Nástroj je implementovaný ako webová služba s verejným rozhraním, ktoré umožňuje ostatným nástrojom posielať dáta, ktoré sa spájajú do jedného spoločného záznamu. Napr., fazetový prehliadač, ktorý slúži v projekte [6] ako prezentačný nástroj, môže zaznamenávať akcie používateľa pomocou SOAP protokolu. Keďže nástroj pozná sémantiku vykonaných akcií a "rozumie" obsahu, ktorý produkuje, dokážeme v konečnom dôsledku vytvoriť podrobnejší a zároveň sémanticky obohatený záznam v porovnaní so štandardným záznamom webového servera.

Napr. ak používateľ kliknutím vyberie ohraničenie informačného priestoru založené na mieste výkonu práce zo zoznamu možných miest, nástroj nezaznamená *URL* odkazu, na ktorý používateľ klikol, ale význam vykonanej akcie spolu s *URI* zvolenej inštancie tak, ako je uložená v ontológii.

Príklad udalosti zaznamenanej nástrojom *SemanticLog:*

```
<event>
        <datetime>2006-04-23 00:54:06</datetime>
        <type>SelectRestriction</type>
        <facet>Region</facet>
        <value>http://job-offers/region#US_AZ</value>
</event>
```

Navyše, nástroje pre monitorovanie na strane klienta, ako napr. *Click*, môžu tiež využiť SOAP rozhranie a dodať udalosti z klientskej strany interakcie ako napr. použitie tlačidla „späť" alebo umiestnenie kurzora nad vybranými položkami. Tieto informácie možno asociovať s udalosťami získanými na strane servera a vytvoriť tak podrobný záznam interakcie používateľa so systémom.

Hlavnou výhodou navrhnutého prístupu je zachovanie sémantiky akcií, ktoré vykonáva používateľ ako aj sémantiky zobrazených stránok pre ďalšie spracovanie nástrojmi pre analýzu správania používateľa.

Nástroje, ktoré spracúvajú záznamy potrebujú mať informácie o zaznamenaných URL adresách a asociovaných parametroch tak, aby „pochopili“ sémantiku obsiahnutú v záznamoch, čo vedie k silnej zviazanosti jednotlivých nástrojov. Pridanie ďalšieho prezentačného nástroja alebo prípadná zmena už existujúceho nástroja vedú vždy k (väčším) zmenám nástrojov analyzujúcich záznamy.

Nástroj *SemanticLog* rieši problém silnej zviazanosti prezentačných nástrojov s nástrojmi pre analýzu poskytnutím spoločnej reprezentácie udalostí získaných od všetkých prezentačných nástrojov. Ďalšou výhodou použitia nástroja *SemanticLog* oproti tradičnému zaznamenávaniu URL adries je teda aj to, že nástroje pre analýzu záznamov (napr. *LogAnalyzer*) nepotrebujú rozumieť špecifickým parametrom jednotlivých prezentačných nástrojov a sú teda len voľne zviazané.

4 Objavovanie charakteristík používateľa

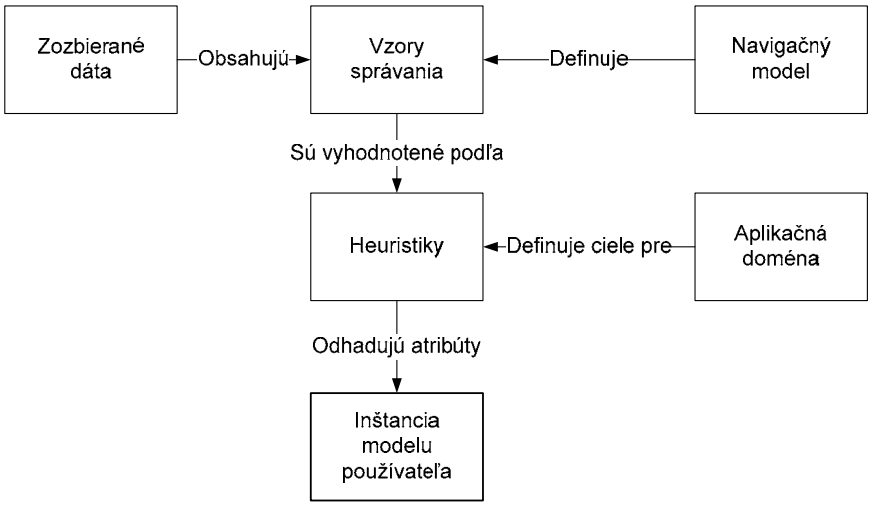
Po fáze zberu dát treba interpretovať ich význam, ktorý nepriamo súvisí so zachyteným správaním používateľa – buď zo správania vyplýva alebo je s ním asociovaný [4] (napr. identifikovať ciele alebo odhadnúť znalosti používateľa o určitých konceptoch). Niektoré charakteristiky vieme odvodiť zo spôsobu navigácie používateľa vo webovom sídle. Odvodenie predpokladá použitie takého navigačného modelu, ktorý umožňuje používateľovi relatívne slobodný pohyb medzi jednotlivými stránkami tak, aby sa v správaní prejavili charakteristiky používateľa.

Iný spôsob odhadu charakteristík používateľa predstavuje analýza reakcií používateľa na zobrazený obsah, ktorá vyhodnocuje implicitnú spätnú väzbu vyplývajúcu z akcií používateľa [7].

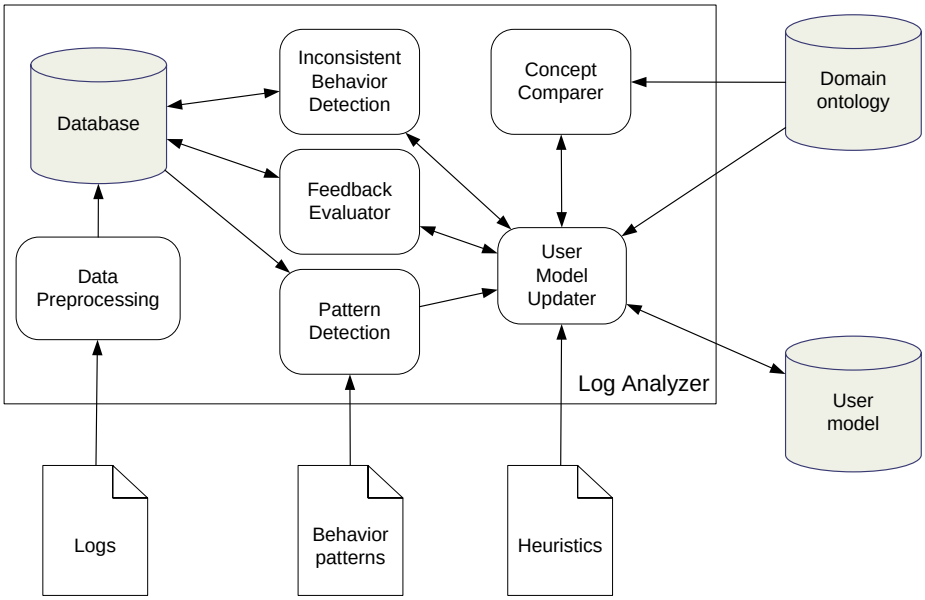
V obidvoch prípadoch hľadáme preddefinované vzory správania v postupnosti akcií používateľa vo webovom sídle. Následne využijeme heuristiku asociovanú s identifikovaným vzorom a vykonáme príslušnú zmenu v modeli používateľa. Heuristiky sú definované s ohľadom na typické ciele používateľa v príslušnej aplikačnej doméne. Obr. 2 zobrazuje zdroje informácií na vytvorenie alebo aktualizovanie inštancie modelu používateľa.

Na analýzu zozbieraných dát s využitím vyššie uvedených princípov sme navrhli softvérový nástroj *LogAnalyzer*, ktorý na základe analýzy zozbieraných dát upravuje model používateľa.

Činnosť nástroja je rozdelená do troch samostatných fáz: (i) predspracovanie získaných dát, (ii) objavovanie charakteristík používateľa a (iii) aktualizácia modelu používateľa. Architektúru nástroja *LogAnalyzer* sme navrhli s ohľadom na tieto tri fázy (pozri Obr. 3). Prvú fázu realizuje modul *DataPreprocessing*. (predspracovanie dát) Druhú fázu, objavovanie charakteristík používateľa, je zabezpečujú moduly *PatternDetection* (odhaľovanie vzorov), *FeedbackEvaluator* (vyhodnocovanie spätnej väzby) a *IncosistentBehaviorDetection* (odhaľovanie nekonzistentného správania). Poslednú fázu – aktualizáciu modelu používateľa, realizuje modul *UserModelUpdater* (aktualizácia modelu používateľa), ktorý používa modul *ConceptComparer* (porovnávanie konceptov).



Obr. 2. Zdroje informácií pre aktualizovanie inštancie modelu používateľa.



Obr. 3. Architektúra nástroja *LogAnalyzer*.

Predspracovanie dát

Modul *DataPreprocessing* identifikuje v získaných záznamoch jednotlivé sedenia používateľa a uloží dáta v podobe vhodnej pre ďalšiu analýzu. V tejto fáze sa vykonajú aj ďalšie transformácie dát, ktoré odstránia identické, po sebe idúce akcie (napr. keď používateľ kvôli pomalej odozve systému zopakuje vykonanú akciu).

Dôležitou úlohou tohto modulu je úprava hodnôt, ktoré predstavujú dobu čítania stránky. Zvyčajne je táto doba vo vzťahu so záujmom používateľa o obsah danej stránky, avšak ak je extrémne dlhá, môžeme predpokladať, že používateľ prestal pracovať so systémom a nevenuje sa zobrazenému obsahu [9]. Preto čas upravujeme do primeraných intervalov.

Odhaľovanie nekonzistentného správania

Modul *InconsistentBehaviorDetection* realizuje metódu dolovania sekvenčných vzorov nad predchádzajúcimi sedeniami používateľa. Získané vzory pokladáme za typické vzory správania používateľa. Aktuálne sedenie porovnáme so získanými vzormi a zisťujeme tak, či je konzistentné s predchádzajúcimi sedeniami. Podobný prístup je použitý aj v [5], kde sa dolovanie sekvenčných vzorov používa na objavovanie zaujímavých navigačných ciest v adaptívnych výučbových systémoch.

Odhaľovanie vzorov

Modul *PatternDetection* zisťuje výskyt preddefinovaných vzorov v spracovaných záznamoch. Odhaľovanie vzorov je založené na použitých dátových štruktúrach, v našom prípade od dátovej štruktúry *suffix tree* (komprimovaný *trie* [2]).

Vyhodnocovanie spätnej väzby

Modul *FeedbackEvaluator* vyhľadáva vzory implicitnej spätnej väzby [7] v získaných dátach a odhaduje používateľove hodnotenie zobrazeného obsahu.

Porovnávanie konceptov

Informácia o ohodnotení zobrazených informácií reprezentovaných konceptmi na odhad charakteristík používateľa nepostačuje. Dôležité sú dôvody, prečo je konkrétne ohodnotenie nízke alebo vysoké. Keďže sa toto ohodnotenie pri rôznych konceptoch líši, je vhodné poznať vzťahy medzi jednotlivými konceptmi. Túto úlohu realizuje modul *ConceptComparer*, ktorý porovnáva koncepty reprezentované ontologickými inštanciami a hľadá ich spoločné a rozdielne aspekty (mieru podobnosti).

V prípade domény pracovných ponúk môžeme uviesť ako príklad dve pracovné ponuky, ktoré majú veľmi rozdielne ohodnotenie. Ak sú tieto dve ponuky takmer identické a líšia sa len v mieste výkonu práce, vieme z rozdielneho ohodnotenia odvodiť, že je miesto výkonu práce pre používateľa dôležité. Navyše vieme povedať, ktorá hodnota je pre používateľa vhodná a ktorá nevyhovuje.

Modul *ConceptComparer* počas porovnávania skúma vzťahy porovnávaných inštancií na základe použitej taxonómie tried. Hľadá spoločnú nadtriedu týchto inštancií – čím viac krokov treba na nájdenie tejto nadtriedy, tým menej sú si porovnávané inštancie podobné. Tento modul tiež analyzuje dátové a objektové atribúty porovnávaných inštancií. Pri porovnávaní dátových atribútov (najmä reťazcov) nepostačuje jednoduché porovnanie zhody – treba vyhodnotiť zhodnosť textov na sémantickej úrovni. Metóda takéhoto vyhodnocovania je navrhnutá v [11].

Pri objektových atribútoch sa algoritmus vykonáva rekurzívne na príslušné asociované inštancie. Čiastkové výsledky sú agregované s použitím váh jednotlivých atribútov, ktoré môžu byť preddefinované alebo uložené v modeli používateľa.

Aktualizácia modelu používateľa

Modul *UserModelUpdater* agreguje výsledky ostatných modulov a vykonáva aktualizáciu inštancie modelu používateľa podľa preddefinovaných heuristík.

Nižšie uvádzame príklad jednoduchej heuristiky v doméne pracovných ponúk:

> „Ak si používateľ zobrazil detaily o aspoň ‘dostatočnom počte‘ ponúk z odvetvia *A* (napr. zdravotníctvo alebo IT), zvýš relevantnosť tohto odvetvia v modeli ideálnej pracovnej ponuky uloženej v modeli používateľa.“

Iná heuristika, ktorá predpokladá navigáciu v informačnom priestore pomocou sémantického fazetového prehliadača [12] je:

> „Ak si používateľ hneď po začiatku sedenia vybral ohraničenie *X*, zvýš relevantnosť charakteristiky spojenej s týmto ohraničením.“

5 Záver

V príspevku sme opísali výskum v oblasti automatizovaného získavania charakteristík používateľa, ktoré je založené na sledovaní používateľa. Definovali sme dvojfázový proces, ktorý postupuje od správania používateľa v rámci webového informačného systému k odhadu charakteristík používateľa v jeho modeli. Každá fáza procesu je pokrytá softvérovými nástrojmi, ktoré vykonávajú jednotlivé definované úlohy.

Prvá fáza – *zber dát* produkuje detailný záznam aktivít používateľa. Navrhli sme nástroje *Click* a *SemanticLog* spolu s ich vzájomným prepojením a napojením na zvyšok webového informačného systému. Nástroje vykonávajú monitorovanie používateľa na strane servera a klienta a vytvárajú spoľahlivý záznam aktivít používateľa v časových súvislostiach so zachovaním sémantiky jednotlivých udalostí.

V druhej fáze procesu, fáze objavovania charakteristík používateľa, sme navrhli nástroj *LogAnalyzer*, ktorý spracúva záznam z predchádzajúcej fázy a analyzuje ho z viacerých hľadísk (navigácia, implicitná spätná väzba, konzistencia správania). Výsledky čiastočných analýz kombinuje s heuristikami a upravuje model používateľa. Modul *ConceptComparer* nástroja *LogAnalyzer* by mohol byť definovaný aj ako samostatný nástroj, ktorý poskytuje službu porovnávania ontologických inštancií pre rôzne časti adaptívneho systému.

Jednotlivé nástroje realizujeme a experimentálne overujeme v rámci výskumného projektu v doméne pracovných príležitostí. V budúcnosti plánujeme tiež ich overenie v doméne vedeckých publikácií.

Poďakovanie

Táto práca vznikla za čiastočnej podpory Agentúry na podporu vedy a techniky (APVT-20-007104), vedeckej grantovej agentúry VEGA (VG1/3102/06) a štátnym programom pre výskum a vývoj „Budovanie informačnej spoločnosti“ (1025/04).

Literatúra

1. De Bra, P., Calvi, L.: AHA: A generic adaptive hypermedia system. In: Proc. of *2nd Workshop on Adaptive Hypertext*. Pittsburgh, USA (1998) 5–12.

2. Dunham, M.: *Data Mining: Introductory and Advanced Topics*. Prentice Hall PTR, Upper Saddle River, NJ, USA (2002).
3. Etgen, M., Cantor, J.: What does getting WET (Web Event-logging Tool) mean for web usability? In: *Proc. 5th Conf. on Human Factors & the Web.* (1999).
4. Judd, T., Kennedy, G.: *Making sense of audit trail data. Australasian Journal of Educational Technology* 20(1) (2004) 18–32.
5. Krištofič, A., Bieliková, M.: Improving Adaptation in Web-Based Educational Hypermedia by means of Knowledge Discovery. In *Proc. of 16th ACM Conf. on Hypertext and Hypermedia, HT'05,* ACM Press (2005) 184–192.
6. Návrat, P., Bieliková, M., Rozinajová, V.: Methods and tools for acquiring and presenting information and knowledge in the web. In: *Proc. of Int. Conf. on Computer Systems and Technologies, CompSysTech'05*, Varna, Bulgaria (2005).
7. Oard, D.W., Kim, J.: Implicit feedback for recommender systems. In *Proc. of AAAI Workshop on Recommender Systems*, Madison, WI, USA (1998) 80-82.
8. Paganelli, L., Paterno, F.: Intelligent analysis of user interactions with web applications. In *Proc. of the 7th Int. Conf. on Intelligent User Interfaces*, *IUI'02,* New York, NY, USA, ACM Press (2002) 111–118.
9. Rafter, R., Smyth, B.: Passive profiling from server logs in an online recruitment environment. In *Proc. of the IJCAI Workshop on Intelligent Techniques for Web Personalisation, ITWP'01*, Seattle, Washington, USA (2001) 35-41.
10. Thomas, R., Kennedy, G., Draper, S., Mancy, R., Crease, M.,Evans, H., Gray, P..: Generic usage monitoring of programming students. In Crisp, G., Thiele, D., Scholten, I., Barker, S., Baron, J., eds.: *Proc. of 20th Annual Conf. of the Australasian Society for Computers in Learning in Tertiary Education, ASCILITE'03,* Adelaide, Australia (2003) 715-719.
11. Tury, M., Bieliková, M.: An Approach to Detection Ontology Changes. In *Proc. of 1st Int. Workshop on Adaptation and Evolution in Web Systems Engineering, AEWSE'06 at ICWE 2006*, ACM, Palo Alto, CA, USA (2006).
12. Tvarožek, M.: Personalized navigation in the semantic web. In V. Wade, H. Ashman, and B. Smyth, editors, *Proc. of 4th Int. Conf. on Adaptive Hypermedia and Adaptive Web-Based Systems*, *AH'06*, Springer, LNCS 4018, Dublin, Ireland, (2006) 467–471.

Annotation

Software Tools for User Characteristics Acquisition

Nowadays, many web-based information systems need the ability to adapt to user characteristics. Otherwise it would be difficult or even impossible to use them efficiently. This paper describes an approach to user characteristics acquisition performed by a set of cooperating software tools where each tool performs an individual task of the characteristics acquisition process. Characteristics are acquired based on user behavior, what minimizes the amount of necessary user involvement. Four tools are described with stress on their interconnections in order to provide effective user characteristics acquisition.