Slovenská technická univerzita v Bratislave
FAKULTA ELEKTROTECHNIKY A INFORMATIKY
Katedra informatiky a výpočtovej techniky

Softvérová podpora organizovania vedeckej konferencie

Tímový projekt
(analýza, špecifikácia, návrh)

Tím 4:
Peter Kósa (kosa@decef.elf.stuba.sk)
Peter Lopeň (lopenpet@decef.elf.stuba.sk)
Richard Richter (virgo@fornax.sk)
Róbert Trebula (trebular@decef.elf.stuba.sk)
Ján Zázrivec (zazrivec@decef.elf.stuba.sk)

14. November 2001

Obsah

1 Úvod

Tento dokument popisuje naše riešenie zadania *Softvérová podpora organizácie vedeckej konferencie*, ktoré je súčasťou našej práce na predmete Tímový projekt. Momentálne obsahuje prvú etapu – etapu analýzy, špecifikácie a hrubého návrhu.

Problematika organizovania vedeckej konferencie ponúka množstvo problémov. Náročnosť organizovania konferencie v dobe globalizácie – v dobe informačnej spoločnosti – neustále stúpa. Dnes je možné zvládnuť aj veľké konferencie pomerne pružne – vyžaduje si to ale, samozrejme, iné prostriedky. Dnes organizátori konferencií disponujú možnosťami, ku ktorým ich predchodcovia nemali prístup – a to stačí ísť o jediné desaťročie dozadu. Elektronická komunikácia účastníkov pred samotnou konferenciou – najmä využitie technológie WWW – umožňuje zvládnuť prípravu konferencie za kratší čas, resp. za ten istý čas si môžu účastníci konferencie vymeniť viacej dôležitých informácií.

Ak je možné zvýšiť množstvo informácií, ktoré sa všetky zúčastnené strany dozvedia, vznikne potreba tieto informácie spracovať. Je možné, že sa o všetko postará skupina ľudí – samotní organizátori. Rovnako tak je ale možné, že časť úloh bude automatizovaná, pričom človek zasiahne, len ak to bude nutné. Je jasné, že nenastala doba, kedy stroje budú posudzovať príspevky posielané do konferencie, ale je možné napríklad priradiť príspevku kľúčové slová a na základe nich uľahčiť výber oponenta, alebo takýto výber priamo navrhnúť.

Z naznačeného vyplýva, že vhodný informačný systém by mohol byť neoceniteľným pomocníkom pri organizovaní vedeckej konferencie.

1.1 Ciele a rozsah projektu[1]

Pre úspešné zorganizovanie vedeckej konferencie treba zvládnuť niekoľko typických aktivít, ako sú:

- prezentácia konferencie;
- prihlasovanie účastníkov na konferenciu;
- zasielanie a oponovanie príspevkov;
- vytlačenie zborníka, apod.

Hoci sa vedecké konferencie líšia svojím obsahom a veľkosťou, podstata činností spojených s organizovaním konferencie je podobná.

Cieľom projektu je vytvoriť softvérový systém podporujúci organizovanie vedeckých konferencií. Organizátorom a programovému výboru má systém pomôcť prezentovať, plánovať a riadiť vedeckú konferenciu. Systém musí zohľadniť rôzne alternatívy pri organizovaní konferencie a má byť konfigurovateľný podľa špecifických požiadaviek a potrieb konferencie. Systém musí diferencovať používateľov v závislosti od role, ktorú hrajú vo vedeckej konferencii.

Systém musí umožniť základné aktivity pri organizovaní konferencie ako sú:

- prihlasovanie účastníkov na konferenciu;
- elektronické posielanie a prijímanie príspevkov;

[1] Náplňou tejto časti je samotné zadanie projektu, v ktorom sú ciele projektu dobre vystihnuté.

- evidovanie údajov o prihlásených príspevkoch a účastníkoch.

Systém musí poskytnúť podporu programovému výboru konferencie pri spoločnej komunikácii a rozhodovaniach týkajúcich sa najmä oponovania a výberu príspevkov. Treba uľahčiť najmä priraďovanie oponentov ku príspevkom posielanie príspevkov na oponovanie monitorovanie aktuálneho stavu v procese oponovania.

Vytvorený produkt by mal poslúžiť pri organizovaní medzinárodnej konferencie ADBIS 2002.

1.2 Členovia tímu

Tím **Con4U** má päť členov. Vedúcim týmu je *Richard Richter*, ktorý sa stará zároveň aj o dokumentáciu. Veľmi úzko spolupracuje s *Róbertom Trebulom*, plniacim úlohu webmastera tímu. Róbert je navyše spolu s *Jánom Zázrivcom* analytikom a návrhárom. Napokon je tu dvojica programátorov – *Peter Kósa* a *Peter Lopeň*.

1.3 Plán projektu

Pôvodný plán projektu do 8. týždňa je priložený v dokumentácii k riadeniu projektu – vznikol v piatom týždni a bolo nutné ho neskôr mierne revidovať. V piatom týždni nebola predstava o činnostiach v 9. – 12. týždni ucelená, preto plán na tieto týždne vznikol neskôr – 11. 11. 2001.

Nasledujúci plán projektu na zimný semester obsahuje revidovaný plán do 8. týždňa, takže pre týždne 4 až 8 sa medzi oboma plánmi vyskytujú rozdiely.[2] Druhou zmenou je vyradenie pravidelných činností z plánu – písanie zápisníc, aktualizácia webu, apod. Sú to úlohy, ktoré sa plnia priebežne, a sú jasné pravidlá, kto ich má plniť. Obvykle termín úlohy padá na tímové stretnutie v stredu o 15:55 – výnimky majú termín explicitne uvedený.

Úlohy do 24. 10. 2001

Kósa, Lopeň pracujú na technologickej analýze, na stretnutí by mala byť podaná ústna správa. **Richter** spojazdní mailing-list, CVS a vytvoriť plán do 8. týždňa. **Trebula** a **Zázrivec** pracujú na doménovej analýze, na stretnutí budú informovať o postupe; dohodnú voľbu techník použitých na analýzu a špecifikáciu; pripravia si otázky na zákazníka a na stretnutí nimi odštartovať špecifikáciu. **Trebula** vytvorí prvú verziu web-stránky a začleniť do nej tímové dokumenty, ktoré budú k dispozícii v HTML.

Úlohy do 31. 10. 2001

Na stretnutí 31. 10. treba prakticky ukončiť fázu analýzy. Fáza špecifikácie pokračuje otázkami na zákazníka. Na stretnutí treba vybrať jednu z technológií, ktorých prehľad podajú **Peter Kósa** a **Peter Lopeň**. **Richter** pracuje na forme dokumentov (šablóny). **Trebula, Zázrivec** majú mať písomnú správu o doménovej analýze.

[2]Pod revíziami rozumieme rozhodnutia, ktoré zmenili plán povedzme na týždeň a viac dopredu – v žiadnom prípade plán nie je zmenený spätne. Napokon o tom môže svedčiť fakt, že najmä pred odovzdávaním prvej etapy vykazoval tím sklz, čo sa týka tvorby dokumentácie.

Úlohy do 7. 11. 2001

Róbert Trebula a **Ján Zázrivec** vypracujú predbežný návrh. V tomto týždni je treba doraziť fázu špecifikácie do tej podoby, v akej sa objaví v dokumentácii k etape. **Peter Lopeň** a **Peter Kósa** sa ako implementátori intenzívne venujú štúdiu zvolených technológií. **Richard Richter** má mať v dokumentácii zapracované existujúce výstupy a pripravený plán do konca semestra.

Úlohy do 14. 11. 2001

Večer v pondelok 12. 11. treba mať k dispozícii vytlačenú dokumentáciu (**Richter**). Táto bude poskytnutá Trebulovi a Lopeňovi, ktorí ju budú pripomienkovať. Pripomienky budú dodané v utorok na RPvI opäť Richterovi. Dokumentácia bude k dispozícii všetkým členom tímu, takže akékoľvek pripomienky aj od nich budú vítané. V utorok večer budú pripomienky zapracované do dokumentácie.

Na stretnutí v stredu (14. 11.) sa odovzdáva prvá etapa (**Richter**). Tím by mal mať vyjasnené technológie a hrubý návrh. **Peter Kósa** a **Peter Lopeň** odštartujú fázu prototypovania. **Róbert Trebula** a **Ján Zázrivec** prezentujú návrh, ktorý by mal byť jemnejší, ako je uvedený v dokumentácii.

Úlohy do 21. 11. 2001

S konkurenčným tímom treba dohodnúť a zrealizovať stretnutie na výmenu dokumentov (**Richter**). Dokument druhého tímu treba prejsť a zhodnotiť (**Richter, Trebula**) – výstupom bude posudok na prácu druhého tímu. **Zázrivec, Lopeň** a **Kósa** sa budú venovať návrhu obrazoviek pre web-interface systému. Dopracujú sa nedostatky.

Úlohy do 24. 11. 2001

Vypracuje sa vyjadrenie k posudku na našu prácu (**Richter**). V prípade nutnosti sa stále analyzujú nové možnosti, ťažisko práce by malo byť v návrhu systému (**Trebula, Zázrivec, Richter**). Pokračuje sa v návrhu a postupnej implementácii obrazoviek – zatiaľ v statickej podobe (**Lopeň, Kósa, Zázrivec**). Do dokumentácie by sa mali zapracovať nové prvky návrhu a začne sa tvoriť používateľská príručka (**Richter**).

Úlohy do 1. 12. 2001

Pokračuje vývoj prototypu – mali by byť jasné postupnosti obrazoviek vo web-rozhraní. Implementáciu protoypu realizujú **Kósa** a **Lopeň**. **Richter, Trebula** a **Zázrivec** sa venujú návrhu, pričom prispievajú k vývoju prototypu pripomienkami. **Richter** začína kompletovať dokumentáciu pred odovzdaním druhej etapy.

Úlohy do 8. 12. 2001

Odovzdávanie prototypu a dokumentácie – ide najmä o používateľskú príručku. Po tomto termíne je nutné sa ešte dohodnúť s konkurenčným tímom na stretnutí, predaní dokumentácie a prototypu. **Richter** a **Trebula** prejdú konkurenčný produkt a vyhotovia posudok.

2 Analýza

2.1 Jestvujúce systémy

Pred diskusiou so zákazníkom o požiadavkách na systém boli preštudované požiadavky, ktoré boli dodané ako súčasť zadania. Boli identifikované tieto základné požiadavky:

- plánovanie vedeckej konferencie;
- riadenie vedeckej konferencie;
- prezentovanie vedeckej konferencie;
- konfigurovateľnosť systému.

Vykonali sme analýzu existujúcich systémov, ktoré sa zaoberajú manažmentom vedeckých konferencií. Ako hlavný zdroj referencií poslúžila stránka - ACM Special Interest Group Governing Board: Summary of Conference Management Software[3]. Táto stránka je síce z roku 1999, takže niektoré linky na nej boli nefunkčné, poskytla však pomerne veľký počet odkazov na jednotlivé systémy na manažovanie konferencií. Pri štúdiu existujúcich produktov sme postupovali v poradí, v akom boli na tejto stránke uvedené a pri každom produkte sme si všímali najmä nové rysy systému, ktoré prinášali nové dátové, funkčné či iné prvky.

V nasledujúcich podkapitolách bude stručný popis jednotlivých systémov so zameraním na nové vlastnosti, ktorými sa tento produkt odlišuje a ktoré budú zapracované do celkového zoznamu možných vlastností systému na menežovanie vedeckej konferencie.

2.1.1 Confman

Nórsky systém Confman[4], ktorého posledná verzia je z roku 2000, sa javí ako jeden z najprepracovanejších, čo sa týka používateľnosti. Je veľmi dobre konfigurovateľný a pomerne jednoducho poňatý.

Okrem funkcií bežných pre všetky takéto systémy, má navyše tieto pozoruhodné vlastnosti:

- riadené posielanie mailov účastníkom konferencie;
- on-line oponovanie príspevkov;
- abstrakty článkov na webe;
- pri registrácii účastníkov konferencie je možné vypňovať aj položky týkajúce sa platenie poplatkov za účasť;
- posudok je tu poňatý ako formulár s prednastavenými otázkami a možnými odpoveďami (vo forme select-boxov). Tieto otázky a odpovede sú konfigurovateľné.

V systéme sa dá identifikovať niekoľko typov používateľov s príslušnými prípadmi použitia:

- **autor príspevku** pridáva svoj príspevok do systému a je informovaný o postupe pri jeho oponovaní a prijímaní;

[3] http://www.acm.org/sigs/sgb/summary.html
[4] http://confman.unik.no/~confman/ConfMan/

- **člen programovej komisie** posudzuje jednotlivé príspevky a odporúča/neodporúča ich na prijatie;
- **účastník konferencie** podáva prihlášku s vyplnením predvolených (konfigurovateľných) položiek;
- **administrátor** riadi zasielanie mailov autorom príspevkov, priraďuje články oponentom, prezerá a riadi prihlásených účastníkov na konferenciu. Navyše konfiguruje všetky konfiguroveteľné položky v rôznych častiach systému.

Systém ako taký sa môže nachádzať v jednom zo stavov:

- prijímanie príspevkov a posudkov;
- hlasovanie komisie o tom, ktoré príspevky prijať a ktoré nie;
- finálna fáza a konečné rozhodnutie.

Zmeny aktuálnej fázy vykonáva samozrejme administrátor.

2.1.2 Edas

Systém Edas[5] z newyorkskej Columbia University nemá na webe funkčnú demonštračnú verziu, takže informácie sme získavali len z dokumentácie.

Oproti už preštudovanému systému Confman sa líši v týchto smeroch:

- namiesto fáz konferencie sú tu **stavy príspevku** – podaný abstrakt, podaný príspevok, priradený na posudzovanie, posúdený, prijatý/neprijatý na konferenciu;
- člen programovej komisie si môže sám vybrať, ktoré príspevky bude oponovať;
- automatické priraďovanie príspevkov oponentom podľa miery zhody kľúčových slov príspevku a preferencií oponenta;
- v systéme je možné podávať opravy (corrections) jednotlivých príspevkov po ich odoslaní, aj ako reakciu na posudky.

2.1.3 Puma

Systém Puma – Publication Management System[6] z inej newyorkskej univerzity – Cornell University – bol pôvodne len prípadovou štúdiou použitia ich vlastného databázového systému Predator, no napriek tomu je na dobrej úrovni. Na web-stránke projektu sú odkazy na demonštračnú verziu systému, sú však nefunkčné.

V tomto systéme sú oddelené rozhrania (interfaces):

- **oponent** - posudzuje články;
- **administrátor** - priraďuje články oponentom.

[5] `http://www.cs.columbia.edu/~hgs/edas/edas.html`
[6] `http://www.cs.cornell.edu/database/predator/puma/puma.html`

Pozoruhodnou vlastnosťou systému Puma je, že sa cez tento systém nedajú posielať samotné príspevky. Všetky príspevky do systému vkladá ručne administrátor. Tento nedostatok je vysvetľovaný tým, že poslanie článku cez web-rozhranie by bolo príliš pomalé a nestihlo by sa vykonať počas jedného behu CGI skriptu. Toto je potrebné uvážiť a otestovať.

Posudky sú modifikovateľné aj po odoslaní. Sú tu zavedené tzv. **vlákna posudkov** (review threads). Jedná sa o to, že sa vytvára diskusia oponentov o jednom článku, ktorá sa zaznamenáva do stromovej štruktúry priradenej k tomuto článku.

Oponent – nazývaný rozhodca – do systému zadáva svoje preferencie, články akého typu by chcel (prípadne nechcel) oponovať. Administrátor má možnosť vytvárať si správy vo forme poradí príspevkov podľa hodnotenia oponentov a to podľa rôznych kritérií.

Systém je „mäkký“, čo sa týka dodržiavania termínov skončenia jednotlivých fáz, takže je možné aj dodatočné posielanie posudkov, hlasov a podobne, ktoré však už v závislosti na prebiehajúcej fáze neovplyvnia výsledky skončených fáz.

Systém sa zaoberá **konfliktnými posudkami** – ak sú na jeden príspevok radikálne odlišné posudky od niekoľkých oponentov, tieto konflikty sa musia vyriešiť. Práve na to sú vhodné diskusie k jednotlivým článkom. Z takejto diskusie by malo vyplynúť, ktorý z posudkov je správnejší, prípadne kto z oponentov svoj posudok zreviduje.

Puma má podrobne rozčlenené fázy – stavy systému:

1. posielanie abstraktov;
2. posielanie príspevkov;
3. posielanie preferencií oponentov;
4. priradenie oponentov príspevkom;
5. posielanie posudkov;
6. rozriešenie konfliktných posudkov;
7. príprava na stretnutie programovej komisie – generovanie reportov podľa rôznych kritérií;
8. aplikácia výsledkov stretnutia programovej komisie – poslanie oznamu o prijatí/neprijatí príspevku autorom.

2.1.4 Witan Web

Produkt Witan Web[7] z kanadskej Národnej výskumnej rady je v krajine svojho vzniku hojne používaný. Prináša niekoľko nových myšlienok, ktoré by mohli byť zapracované do návrhu nášho systému.

Pretože je nasadzovaný v multijazyčnej krajine, zaoberá sa znakovými sadami a jazykmi dokumentov.

Oponenti môžu prostredníctovm systému Witan Web žiadať o konzultáciu, pomoc od iných oponentov týkajúcu sa niektorého posudzovaného príspevku.

Skoro v každom formulári je pole „dodatočné informácie“, kam sa zapisujú všetky záznamy pre ktoré nie sú predvolené otázky.

Zaujímavosťou je položka „študentský príspevok“ vo formulári pridávania článku.

Položky formulárov v tomto systéme nie sú konfigurovateľné priamo prostredníctvom web-rozhrania, čo je asi najväčší nedostatok tohoto systému.

[7] `http://witanweb.iit.nrc.ca`

2.2 Technologická analýza

Po začiatočnej analýze zadania sa ako nevyhnutná ukázala potreba hlbšie sa oboznámiť s technologickými postupmi a metódami umožňujúcimi realizovať riešenie ako klient/server aplikáciu. Z tohto dôvodu sa v nasledujúcom texte zaoberáme jednotlivými technológiami a skúmame ich z hľadiska použitia v navrhovanom systéme.

2.2.1 Služba WWW

Základnou technológiou, ktorú použijeme v riešení, je služba World Wide Web (WWW). Táto služba vznikla na pôde CERNu v rokoch 1989–90 a je založená na troch pojmoch: **HTML**, **HTTP** a **URL**.

HTML – Hypertext Markup Language HTML je značkovací jazyk založený na SGML na popis štruktúry webovských stránok. Je platformovo nezávislý. Prvá formálna špecifikácia je HTML 2.0, ktorá obsahuje prostriedky pre základné formátovanie a štrukturovanie dokumentu, možnosť vkladať obrázky a vytvárať formuláre. Z dôvodu neštandardizovaného rozširovania výrobcami prehliadačov vzniklo v roku 1996 konzorcium W3C, ktoré vybralo spoločnú podmnožinu rozšírení a schválilo HTML 3.2.

V súčasnosti je aktuálna verzia HTML 4.0, ktorá predstavuje veľký skok vpred. Z najdôležitejších vylepšení treba spomenúť podporu kaskádových štýlov, skripty vložené do stránky, multimediálne objekty, rozšírené možnosti tabuliek, podporu viacerých jazykov, rozšírenie formulárov a rámy. Takmer kompletnú podporu HTML 4.0 majú prehliadače Internet Explorer 4.0 a 5.0 a tiež Netscape Navigator. V roku 1999 vzniká XHTML 1.0. Ide o štandard HTML 4.0 upravený tak aby stránky v XHTML predstavovali XML dokumenty.

HTTP - Hypertext Transfer Protocol HTTP predstavuje protokol na prenos stránok medzi webovským serverom a prehliadačom. Ide o jednoduchý aplikačný a bezstavový protokol vystavaný nad protokolom TCP modelom požiadavka/odpoveď, čo prináša niektoré nevýhody pre webovské aplikácie. Existuje niekoľko verzií HTTP 0.9, HTTP 1.0, HTTP 1.1 – najnovší, podporaný vo väčšine serverov a prehliadačov.

URL - Uniform Resource Locator URL je adresa, ktorá jednoznačne identifikuje zdroj v Internete. Nie sú obmedzené len na službu WWW, ale pokrývajú celé spektrum služieb (WWW, FTP, e-mail, telnet). Počas komunikácie servera a prehliadača sa pomocou protokolu HTTP vymieňajú požiadavky používateľa (URL) a odpovede servera (HTML).

2.2.2 Platformy

Náš projekt bude klient/server aplikácia. Preto musíme uvažovať, na akom operačnom systéme bude bežať klientská a serverová časť systému. Pri ďalšej analýze budeme uvažovať, ktoré techniky sú prístupné pre ktorú platformu. Existujú dve veľké rodiny operačných systémov: Unix a Windows.

Windows Do tejto rodiny patria operačné systémy Windows 9x/ME, Windows NT/2000/ XP. Tieto operačné systémy panujú v oblasti stolných počítačov vďaka ich pohodlnému používaniu. Majú zabudovaný webovský prehliadač Microsoft Internet Explorer. Windows NT podporuje procesory x86, DEC Alpha, MIPS R4000, PowerPC.

Unix Pojem UNIX predstavuje veľkú rodinu operačných systémov. Veľmi významným predstaviteľom tejto rodiny je Linux. Je voľne šíriteľný s otvoreným zdrojovým kódom, čo podporuje jeho rozšírenie. Dnes predstavuje robustné a spoľahlivé riešenie. Podporuje obrovskú škálu procesorov. Bežne sa v jeho distribúciach dodáva prehliadač Netscape Navigator, server Apache, niektoré distribúcie obsahujú dokonca aj databázové servery MySQL alebo PostgreSQL. V oblasti serverov je veľmi populárny pre jeho dostupnosť, spoľahlivosť a rýchlosť.

2.2.3 Webovské prehliadače a servery

V predchádzajúcej kapitole sa niekoľkokrát vyskytli slová ako prehliadač alebo webovský server. Je preto vhodné, aby sme sa teraz zaoberali na trhu dostupnými prehliadačmi a servermi – a to predovšetkým z pohľadu rôznych platforiem.

WWW prehliadače

- **Microsoft Internet Explorer** patrí medzi najrozšírenejšie prehliadače na platforme Windows. V súčasnosti je aktuálna verzia 5.5.
- **Netscape Navigator** existuje vo verziách pre platformu Windows aj Unix. Vývojári ocenia XML, JavaScript, HTML4 a CSS podporu. Aktuálnou je verzia 6.0.
- Jeden z novších prehliadačov predstavuje **Opera 5.0**, ktorá ponúka podporu pre väčšinu webovských štandardov (HTML 4, XML, SSL, CSS, JavaScript a DOM). Existuje pod operačné systémy Windows aj Unix.

V ďalšom texte sa budú uvedené technológie skúmať z pohľadu týchto prehliadačov.

WWW servery

- K najrozšírenejším WWW serverom patrí bezpochyby server **Apache** (60% podiel na trhu), ktorého verzia 1.3 existuje pre platformy Windows aj Unix. Je voľne šíriteľný a jeho licencia umožňuje aj komerčné využitie. Má modulárnu architektúru a je ľahko rozšíriteľný. Je robustný a bezpečný.
- Medzi ďalšie servery, ktoré budeme v nasledujúcom texte uvažovať patria **Internet Information Server** 5.0 (IIS), určený na komerčné nasadenie a **Personal Web Server** (PWS) vhodný pre jednoduchšie a menej rozsiahle aplikácie (prácu off-line). Oba sú určené pre operačný systém Windows. IIS je primárny server pre prostredie Windows NT verzia Server aj Workstation. Dynamické stránky nie sú ponúkané pomocou CGI, ale používa sa programovacie rozhranie Internet Server Application Programing Interface (ISAPI). Aplikácie ISAPI sú v podstate dynamické knižnice. Príklad ich použitia predstavujú aplikácie ASP.

2.2.4 HTML editory

Z dôvodu uniformného výstupu a zjednodušenia tvorby HTML stránok je vhodné sa ešte pred začatím samotných prác rozhodnúť, aký spôsob tvorby použijeme. HTML kód sa dá samozrejme zapisovať pomocou obyčajných textových editorov ako je Notepad v prostredí Windows, či Vim alebo Emacs pod Unixom. Ak však HTML stránky vytvárame často je vhodné uľahčiť si prácu použitím editoru obsahujúceho podporu pre písanie HTML kódu. Z tohto pohľadu môžeme editory rozdeliť do niekoľkých skupín:

Textové editory s podporou HTML Obľúbenými predstaviteľmi tejto skupiny sú editory **HomeSite** a **HTML-Kit**, ktoré boli vytvorené špeciálne pre potreby jazyka HTML. Zvýraznenie syntaxe ponúkajú aj klasické editory ako **Vim** a **Emacs**.

Štruktúrne editory Oproti predchádzajúcej skupine ponúkajú dôležitú vlastnosť navyše – neumožňujú vytvárať syntakticky nesprávnu stránku. Do miesta ktoré editujeme môžeme vložiť iba tie tagy, ktoré sa tam môžu objaviť. Nestane sa nám teda, že stránka obsahuje skrížené elementy alebo chýbajúce ukončovacie tagy. Veľmi dobrým programom z tejto kategórie je **Wedit**. V praxi je možné použiť aj kombináciu editoru Emacs a nadstavby **PSGML**.

WYSIWYG editory Táto kategória zahrňuje editory, s ktorých podporou môžeme vytvárať HTML stránky bez znalosti HTML. WYSIWYG editor sa ovláda podobne ako textový procesor – píšeme text, môžeme meniť písmo, zarovnanie, vkladať obrázky a podobne. Výsledok sa však ukladá vo formáte HTML a počas editácie sa zobrazuje približne tak, ako bude zobrazený v prehliadači. Problémom týchto editorov je, že zaostávajú za vývojom HTML, a tiež to, že generovaný HTML kód je pre následnú ručnú editáciu neprehľadne sformátovaný. Najznámejším editorom z tejto skupiny je Microsoft FrontPage. Podobnú funkčnosť poskytuje aj Netscape Composer.

2.2.5 Kaskádové štýly (CSS)

Cascading style sheets (CSS) je deklaratívny jazyk odstraňujúci niektoré nevýhody klasického HTML. Štýly umožňujú definovať spôsob zobrazenia každého elementu na stránke. Štýl však nie je priamo súčasťou textu stránky, a tak môže byť zápis stránky prehľadnejší a dobre štruktúrovaný. Naviac štýly umožňujú definovať jednotný vzhľad určitého elementu v celom dokumente jedným spôsobom - nemusíme ho opakovať u každého elementu. Pokiaľ štýl uložíme do externého súboru, môže ho využívať viacero stránok naraz. Definícia designu všetkých stránok je potom na jednom mieste. Pri požiadavke na zmenu vzhľadu stránok stačí upraviť štýl a zmeny sa automaticky uskutočnia vo všetkých stránkach. Dobrú podporu CSS majú prehliadače Internet Explorer 4.0 a 5.0, Netscape Navigator 6.0 a Opera. Z týchto dôvodov by bolo vhodné použiť túto techniku aj v nami navrhovanom riešení.

2.2.6 Design webovských stránok

Na tomto mieste uvádzame niekoľko základných pravidiel, ktoré by sa mali dodržať pri návrhu webovského interfejsu celej aplikácie:

Grafický design

- Vhodné kombinácie farieb;
- farba pozadia a textu by mala byť zvolená tak, aby sa text dobre čítal;
- nepoužívať obrázky na pozadí;
- ako farba pozadia je vhodná biela a svetlé odtiene ďalších farieb;
- u všetkých grafických objektov uvádzať alternatívne textové popisy.

Organizácia stránok

- Hypertext umožňuje vytvoriť ľubovolnú štruktúru, ale človek je zvyknutý na konzervatívnejšie usporiadania – väčšinou na hierarchiu;
- na každej stránke by mala byť navigačná lišta;
- jedna stránka by mala obsahovať logický celok informácií;
- stránka by mala byť dobre štruktúrovaná pomocou nadpisov a zoznamov;
- obsah by sa mal oddeliť od vzhľadu a mali by sa použiť CSS.

Odkazy

- Vyhnúť sa technickým pojmom;
- odkazy by mali byť iteraktívne (napr. meniť farbu);
- v rámci jedného serveru používať relatívne odkazy.

Dokument

- Každá stránka by mala obsahovať identifikačné údaje;
- malo by byť jasné, na ktorom serveri sa nachádzame;
- používať metadata na lepšiu klasifikáciu a vyhľadávanie.

2.2.7 Podpora pre webovské aplikácie v prehliadačoch

Pokiaľ chceme prostredie webovského prehliadača využiť ako platformu pre tvorbu aplikácie máme k dispozícii mnoho možností. V zásade sa však dajú rozdeliť na dve skupiny podľa toho, kde sa vykonáva kód aplikácie. Jednu z nich predstavujú aplikácie, ktoré bežia v samotnom prehliadači (klientovi). V tejto skupine sú k dispozícii nasledujúce riešenia:

1. Klientské skripty – JavaScript, VBScript a ďalšie

   Priamo do kódu HTML stránky môžeme zapisovať krátke programy v jednoduchých jazykoch ako je JavaScript, JScript alebo VBScript. Tieto sa využívajú predovšetkým na kontrolu údajov vyplnených používateľom a na zvýšenie interaktivity stránky. Jazyk JavaScript je dokonale štandardizovaný ako ECMAScript. Podpora uvažovaných prehliadačov je nasledovná:

   - **JavaScript 1.0** – Netscape 2.0 a IE 3.0;
   - **JavaScript 1.1** – Netscape 3.0 a IE 4.0;
   - **JavaScript 1.2** – Netscape 4.0 a čiastočne aj IE 4.0;
   - **JavaScript 1.3** – Netscape 4.5 a IE 5.0.
   - **VBScript** pracuje iba v Internet Exploreri.

Dynamické HTML (DHTML) Dynamické HTML stavia na využití klientských skriptov. Navyše je teraz celá stránka a ostatné komponenty prehliadača prístupná pomocou objektového modelu dokumentu (DOM). Obsah stránky sa dá interaktívne meniť aj po jej načítaní, čo umožňuje vytvárať skutočne prispôsobivé užívateľské rozhranie. Dynamické HTML je podporované prehliadačmi Internet Explorer 4.0 a 5.0 ako aj Netscape Navigatorom od verzie 4.0.

Java applety Java applet je program napísaný v jazyku Java, ktorému sa vyhradí časť HTML stránky. Na tejto časti si applet hospodári úplne vo vlastnej réžii. Preto sa applety hodia pre aplikácie kde je potrebná vysoká interaktivita pri komunikácii s používateľom a klasické klientské skripty už nestačia. Java applet má prístup k väčšine knižníc jazyka Java, a preto môže na komunikáciu s ostatnými aplikáciami používať takmer ľubovoľný protokol a umožňuje vytvoriť multimediálne aplikácie (grafika, animácia, hudba). Použitie appletov je obmedzené na novšie prehliadače, ktoré ich však vo väčšine podporujú.

ActiveX komponenty Použitie ActiveX komponentov je rovnaké ako u Java appletov. Rozdiel je v tom, že ActiveX komponenty sú spustiteľné iba na platforme Windows. Nižšia prenositeľnosť je však vynahradená väčším výkonom oproti Jave.

2. Skriptovanie na serveri

   Druhú skupinu predstavujú riešenia, pri ktorých je aplikácia vytvorená tak, že nejaký program na serveri dynamicky podľa potreby generuje HTML kód. Klientské technológie sú v tomto prípade použité prevažne na zlepšenie funkčnosti používateľského rozhrania a pri zvyšovaní interaktivity stránok. Navyše, klientské technológie nemajú prístup k databázam, ktoré sú uložené na serveri. Generovanie stránky v dobe požiadavky tak umožňuje reagovať na konkrétne podmienky a požiadavky používateľa.

Server Side Includes (SSI) SSI sú jednoduché príkazy, ktoré sa zapisujú priamo do HTML stránky ako komentár. Umožňujú vykonávanie jednoduchých činností ako je vloženie iného súboru do stránky alebo vypísanie dátumu poslednej modifikácie dokumentu. Dnes podporujú SSI všetky bežne používané servery.

CGI skripty CGI skripty majú svoj názov odvodený od rozhrania CGI – Common Gateway Interface. Toto rozhranie definuje spôsob, ktorým webovský server komunikuje s programami a umožňuje jeho prepojenie s ľubovoľným programovacím jazykom. Pokiaľ si používateľ vyžiada od servera stránku, ktorá má byť vygenerovaná CGI skriptom, predá server skriptu parametre zadané používateľom a spustí ho. Skript potom na svojom štandardnom výstupe vygeneruje odpoveď pre prehliadač v jazyku HTML. Táto odpoveď je zachytená webovským serverom a zaslaná prehliadaču. CGI skripty sú najvšeobecnejším a najuniverzálnejším riešením pre webovské aplikácie. Dajú sa písať v ľubovoľnom jazyku – najpoužívanejšie sú príkazové shelly, Perl, C/C++. Ich nevýhodou je však vysoká prevádzková réžia. Pre každú požiadavku musí byť vytvorený nový proces, v ktorom sa spúšťa CGI skript. Preto sa toto riešenie nedá použiť pre servery, na ktoré pristupuje niekoľko desiatok užívateľov súčasne. Podporu pre rozhranie

CGI má dnes každý webovský server. Pokiaľ sa na tvorbu skriptov použije jazyk nezávislý na platforme, môžeme výslednú aplikáciu použiť na webovských serveroch v Unixe i vo Windows. Pri použití CGI skriptov by sa mal klásť zvýšený dôraz na bezpečnosť.

FastCGI skripty FastCGI skripty sú snahou o zrýchlenie vykonávania CGI skriptov. Skripty upravené pre toto rozhranie sú pri prvej požiadavke zavedené do pamäti. Ďalšie požiadavky už obsluhuje program, ktorý už je zavedený v pamäti. FastCGI skripty napísané v jazyku C sú dnes asi najvýkonnejším riešením. Sú používané predovšetkým v prostredí Unixu.

ISAPI a NSAPI moduly ISAPI je špeciálne rozhranie pre komunikáciu medzi webovským serverom a aplikáciou. Umožňuje aby bol program neustále zavedený v pamäti a nemusel sa znovu načítavať a spúšťať pre každú požiadavku. ISAPI sa používa so serverom IIS aj keď toto rozhranie podporujú aj niektoré ďalšie servery. Servery firmy Netscape používajú vlastné rozhranie NSAPI.

Active Server Pages (ASP) ASP je technológia Microsoftu určená pre tvorbu webovských aplikácií. Má modulárnu štruktúru, takže umožňuje použitie takmer ľubovoľného programovacieho jazyka. Microsoft štandardne ponúka VBScript a JScript. Od ďalších firiem sa dajú získať interpreteri Perlu a Pythonu. V ASP sa dajú bez problémov používať COM objekty. Nevýhodou ASP je ich úzke zviazanie s technológiami Microsoftu. AS P beží len v prostredí IIS a PWS.

ColdFusion ColdFusion je vyvinutý Allaire. Jeho syntax je veľmi podobný syntaxi HTML, čo uľahčuje začínajúcim programátorom webovských stránok jeho používanie. Dnes je dostupný aj pre Windows NT aj pre Linux. Jeho nevýhodou je, že nie je voľne šíriteľný.

Personal Home Pages (PHP) PHP je jednoduchý, voľne šíriteľný a pritom výkonný skriptový jazyk. Zapisuje sa priamo do HTML kódu a umožňuje prirodzeným spôsobom kombinovať HTML s príkazmi. PHP sa dá prevádzať na ľubovoľnom webovskom serveri (platforma Windows alebo Unix), aj keď najlepších výsledkov sa dosiahne v spojení so serverom Apache (PHP je skompilované ako modul a je trvale uložené v pamäti). Navyše PHP obsahuje priamo podporu pre prácu s najrozšírenejšími databázovými servermi. Aktuálna je verzia 4.0.

Servlety v Jave Java applety umožňujú beh javovských aplikácií v prehliadači. Podobne servlety zase umožňujú beh Javy na serveru a tím jej využitie pre webovské aplikácie. Servlety predstavujú robustné riešenie a vďaka CORBE môžu komunikovať s ďalšími časťami informačného systému (databázy, . . .).

Java Server Pages (JSP) Pri použití JSP sa podobne ako v ASP alebo v PHP priamo do HTML kódu zapisujú príkazy. V prípade JSP sú zapisované v jazyku Java.

Špeciálny servlet zabezpečuje aby bola JSP stránka vždy po svojej modifikácii automaticky preložená do byte-code. To uľahčuje vývoj a na druhej strane je aplikácia značne výkonná, pretože skripty sú vyvolávané už v preloženom tvare. Ide o pomerne novú technológiu, ktorú zatiaľ podporujú iba servery od firmy Sun a Apache.

Všetky vyššie uvedené technológie môžeme rozdeliť do dvoch skupín podľa toho, či je pred použitím potrebné previesť program do binárnej podoby alebo či ide o interpretované jazyky. Pokiaľ vo vývoji aplikácie použijeme jazyk ako C/C++ alebo Java je potrebné pred použitím aplikáciu skompilovať. To spomaľuje vývoj, ale prináša zrýchlenie počas behu aplikácie. Pri interpretovaných jazykoch ako PHP, JavaScript, Perl je situácia opačná. Rýchlejší vývoj je kompenzovaný pomalším vykonávaním. Úbytok rýchlosti však väčšinou nie je nijak významný vzhľadom na fakt, že skripty webovských aplikácií bývajú pomerne jednoduché.

3. Porovnanie jazykov pre klientské skripty

   V prípade, ak by sme sa nejakú časť aplikácie rozhodli implementovať pomocou skriptov vykonávaných u klienta v prehliadači je vhodné porovnanie dostupných skriptovacích jazykov.

   JavaScript JavaScript je kompaktný, platformovo nezávislý, na objektovom prístupe založený jazyk na vývoj aplikácií pre Internet. Pôvodne bol navrhnutý firmou Netscape Communications Corporation a po jeho úspechu prišiel aj Microsoft s podobným jazykom pod názvom JScript. Príkazy JavaScriptu sú priamo vložené v HTML stránke a sú interpretované prehliadačom.

   VBScript VBScript je skriptovací jazyk alebo presnejšie skriptovacie prostredie implementujúce podmnožinou programovacieho jazyka Visual Basic. Výsledkom zjednodušenia je produkt, ktorý je ľahko použiteľný. Je to asi aj jeho najväčšia výhoda oproti JavaScriptu. Veľká sila VBScriptu spočíva v možnosti sprístupniť množstvo ActiveX komponentov. To je súčasne dôvodom jeho najvážnejšieho nedostatku – je podporovaný iba v prehliadačoch IE 4.

4. Porovnanie jazykov pre serverovské skripty

   V prípade voľby skriptov na strane servera ako technológie, ktorú bude naše riešenie využívať je potrebné bližšie preskúmať aj jednotlivé jazyky, ktoré prichádzajú do úvahy. Hľadiská, z ktorých sa treba na tieto jazyky pozerať sú nasledujúce:

   - **Výkonnosť** – je zrejmé, že kompilované jazyky sú rýchlejšie ako interpretované. Časť z toho je spôsobená potrebou zavedenia kópie interpretera nevyhnutného na spustenie programu napísaného v skriptovacom jazyku, ako je napríklad Perl. Ďalšie zrýchlenie kompilovaných programov je aj dôsledkom použitia optimalizácií závislých na platforme, pre ktorú je program kompilovaný. Väčšinou však rozdiel vo výkonnosti nie je rozhodujúcim faktorom pre voľbu jazyka.
   - **Bezpečnosť** – vo všeobecnosti najviac bezpečnostných problémov pochádza z programovacích chýb a nedostatočného pochopenia prostredia, v ktorom skript existuje a beží.

- **Spoľahlivosť** – mal by sa použiť jazyk, ktorého stabilita sa na použitej platforme dokázala. Je vhodné, ak je jazyk stabilný na viacerých platformách.
- **Udržovateľnosť** – toto je často parameter, na ktorý sa berie najväčší ohľad pri voľbe programovacieho jazyka. Veľa programátorov používa interpretované jazyky z dôvodu, že sú ľahšie čitateľné, ľahšie na pochopenie, jednoducho udržiavateľné a testovateľné. Umožňujú inkrementálny, iteratívny vývoj a testovanie namiesto cyklu vytvorenie/modifikovanie – preloženie – testovanie – krokovanie. Tieto faktory môžu dramaticky urýchliť vývoj.
- **Prenositeľnosť** – prenositeľný kód je základom dobre navrhnutého programu. Aj v tomto prípade vychádzajú interpretované jazyky ako dobrá voľba, pretože väčšina interpreterov existuje pre viacero platforiem.

Perl Perl je interpretovaný jazyk optimalizovaný pre prehľadávanie textových súborov a získavanie informácií z týchto súborov a vytvorenie ďalšieho textového súboru na základe týchto informácií. Je tiež vhodný pre množstvo úloh súvisiacich so správou systému. Je vytvorený tak aby bol skôr praktický (ľahko požívateľný, efektívny, kompletný) ako krásny. Je dostupný a stabilný na platformách Unix aj Windows. Spolu s webovským serverom Apache vytvárajú výkonnú a populárnu kombináciu. Jedným z najznámejších projektov využívajúcich tieto dve technológie je projekt mod_perl, ktorý patrí k najstarším a najúspešnejším. Umožňuje vloženie Perl interpretera do Apachu a tým sprístupňuje interné informácie servera programom v perle.

Python Python je prenositeľný, interpretovaný, objektovo-orientovaný jazyk. Má elegantnú syntax a malé množstvo vstavaných vysoko úrovňových údajových typov. Môže byť systematicky rozširovaný pridávaním nových modulov implementovaných v kompilovanom jazyku ako je C alebo C++. Tieto rozširujúce moduly môžu definovať nové funkcie a premenné alebo tiež nové dátové typy. Obsahuje úplnú množinu operácií s reťazcami (vrátane regulárnych výrazov) a odľahčuje používateľa od správy pamäte. Tieto a ďalšie vlastnost i ho robia ideálnym pre vývoj prototypov a ostatných ad-hoc programovacích úloh. Obsahuje tiež generický spôsob, ako konvertovať objekt do prúdu bytov a opačne, čo môže byť použité pri tvorbe rôznych distribuovaných modelov. Je k dispozícii na platforme Unix, Windows, DOS, OS/2 a Macintosh a existuje ako modul do webovského servera Apache.

Tcl Tcl je výkonný skriptovací jazyk použiteľný na väčšine dostupných platforiem. Ponúka všetky typické vysoko úrovňové programovacie prostriedky ako sú: reťazce premennej dĺžky, asociatívne polia, zoznamy, štruktúry, regulárne výrazy a schopnosť definovať procedúry priamo počas vykonávania programu. Všetky dátové štruktúry sú dynamicky alokované. Tcl je programovací jazyk určený na vkladanie do iných aplikácií ako konfigur ačný a rozširujúci jazyk. Je navrhnutý tak aby bol ľahko rozšíriteľný pomocou nových primitív napísaných v jazyku C. Tcl používa rovnakú reprezentáciu pre dáta aj pre program, čo umožňuje uloženie programu do premenných, súborov alebo databázy. Taktiež podporuje udalosťami riadené programovanie a IPC medzi viacerými Tcl aplikáciami.

PHP Jednou z veľa výhod PHP je jeho rýchlosť. Ak ASP používa VBScript, spustí COM objekt. Ak píše niečo, používa iný COM objekt, ďalší, ak pristúpi k databáze, ďalší pre súborový systém, atď. V PHP všetko beží v jednom adresnom priestore, čo znamená väčšiu rýchlosť, lebo nevzniká potreba komunikovať s iným objektom v inom procese.

Oproti Perlu bol PHP vyvinutý pre generovanie dynamických webovských stránok, nie na všeobecné spracovanie textových súborov, a preto má zabudovaný veľa špeciálnych prostriedkov (napr. prístup k objektom HTML stránky cez premenné).

Pre programátorov skúsených v Java/JavaScript/C/C++/Perl je ľahko zvládnuteľná gramatika jazyka, nakoľko je blízka gramatike týchto jazykov. Ďalšou výhodou je integrácia s MySQL. PHP má veľa prostriedkov na správu a udržiavanie MySQL databázy.

2.2.8 Databázové servery

Ako jedna z častí v navrhovanom riešení bude pravdepodobne vystupovať databáza. Je preto vhodné vykonať analýzu aj tejto oblasti. Medzi najrozšírenejšie produkty v tejto oblasti patria Oracle, MS SQL Server, Sybase, Progress. Vzhľadom na fakt, že sa jedná o drahé softvérové produkty, zameriame sa skôr na analýzu voľne šíriteľných databázových serverov, medzi ktoré patria napríklad MySQL alebo PostgreSQL.

MySQL MySQL je jednoduchý databázový SQL server. Je priamo podporovaný PHP. Umožňuje na jednom počítači pracovať s viacerými databázami. Existuje pre platformu Windows aj Unix. Medzi hlavné výhody patrí:

- schopnosť obsluhovať neobmedzené množstvo používateľov súčasne;
- kapacita na uchovanie 50000000 záznamov;
- veľmi rýchle vykonávanie príkazov;
- ľahký a efektívny systém používateľských privilégií.

MySQL má navyše širokú používateľskú základňu, čo zabezpečuje, že kód je dobre testovaný.

PostgreSQL PostgreSQL je databázový server takmer úplne kompatibilný so štandardom SQL92. Podporuje teda väčšiu podmnožinu jazyka SQL ako MySQL. Server beží na Unixe, klientské programy existujú pod Unix aj Windows. Podporuje vnorené príkazy SELECT, pohľady a transakcie. Obsahuje interfejsy pre C/C++, Perl, PHP, Python, TCL, bash, ODBC. Jeho sila spočíva vo veľkej rozširovateľnosti: používateľom definované údajové typy, operátory, funkcie. Taktiež podporuje dedenie a má bohatú množinu dátových typov: boolean, časové intervaly, viac dimenzionálne polia a 2D geometrické typy. Navyše je dobre zdokumentovaný. Jeho najväčšou nevýhodou oproti MySQL je rýchlosť. Ďalšou nevýhodou je fakt, že nie je hierarchický - databázy môžu obsahovať iba tabuľky a nie ďalšie databázy. Podobne ako MySQL aj PostgreSQL je priamo podporovaný PHP.

2.2.9 Závery technologickej analýzy

Pre náš projekt sme si vybrali kombináciu:

webowský server	**Apache**
skriptovanie na strane servera	**PHP**
skriptovanie na strane klienta	**JavaScript**
databáza	**MySQL**

Hlavné kritériá nášho výberu v poradí podľa dôležitosti:

- Platformová nezávislosť, prenositeľnosť – aby ani klient, ani server nebol určený pre konkrétny operačný systém, konkrétny prehliadač;
- efektívnosť;
- nízka cena tvorby a používania;
- jednoduchá implementácia.

Webovský server Apache je dostupný na obidvoch platformách (dokonca bezplatne) a má veľmi dobré ukazovatele efektívnosti.

Jazyk PHP je priamo podporovaný serverom Apache ako jeho modul pre obidve platformy a je najrýchlejším jazykom pre interpretované skripty. Programátori nášho tímu už majú skúsenosti s jeho používaním.

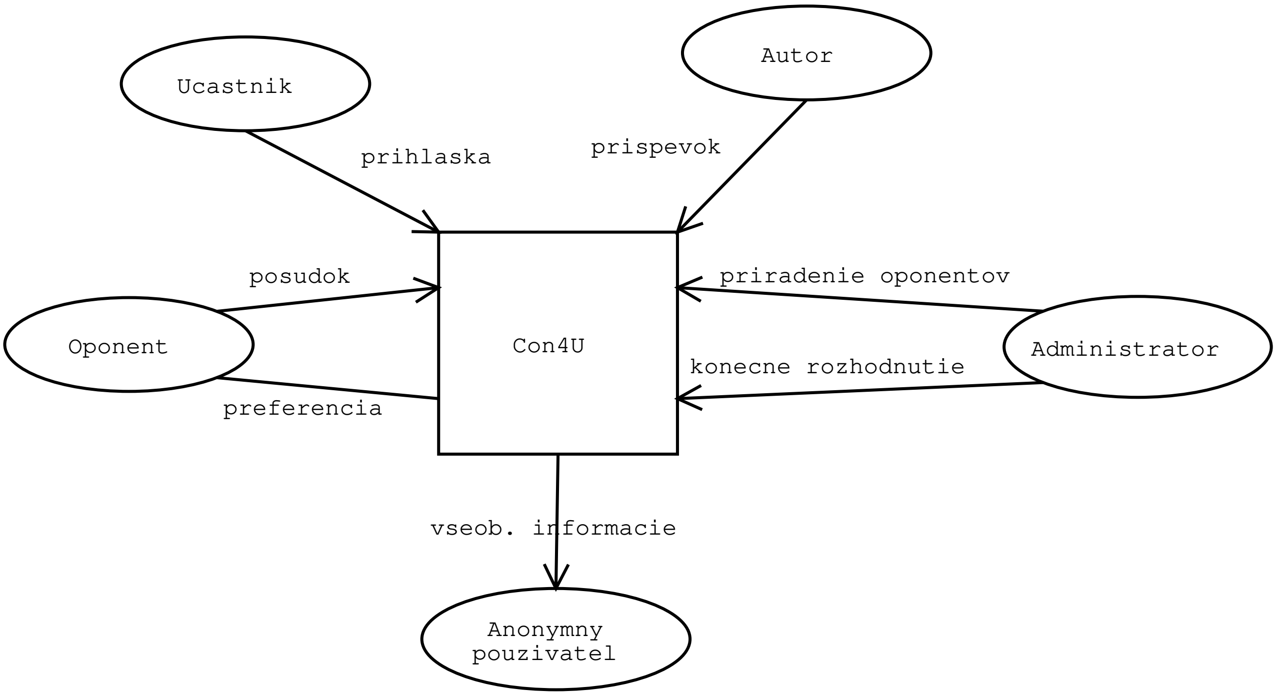
Databázový systém MySQL, tak isto ako PHP a Apache, je platformovo nezávislý. Je veľmi rýchly aj pri väčšom množstve dát. Všetky hore uvedené prostriedky možno získať bezplatne.

JavaScript je jediný skript na strane klienta, ktorý je podporovaný všetkými analyzovanými prehliadačmi.

3 Špecifikácia

3.1 Kontext systému

Obrázok 1 zobrazuje kontext systému – jeho okolie a ľudí, ktorý s ním prichádzajú do kontaktu. Systém je znázornený obdĺžnikom uprostred, okolné entity oválmi, vstupy a výstupy šípkami.



Obrázok 1: Kontext systému

Kontext vyplýva v mnohom zo zadania – máme tu typických používateľov systému (**účastník**, **oponent**, **anonymný používateľ**, **administrátor** a **autor** príspevku), ako aj údaje, ktoré jasne vyplývajú z požadovaných funkcií systému. Ak má vedieť systém prijímať príspevky od jednotlivých autorov, objaví sa tu vstup nazvaný **príspevok**. Podobne od účastníka konferencie sa prijíma **prihláška**. Oponent zadáva svoje **preferencie**, čo sa týka kľúčových slov (tie popisujú jednotlivé témy) a zasiela **posudok** na pridelený príspevok. Anonymný používateľ sa dostane iba k **všeobecným informáciam**. Napokon administrátor systému – čo je vlastne predseda komisie – má rozhodujúce slovo pri **pridelení oponentov** na príspevky a vykonáva **konečné rozhodnutia** vo väčšine riešených otázok.

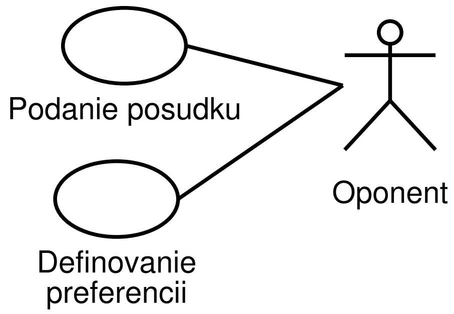
Funkcie, cez ktoré jednotlivý používatelia pristupujú k systému, sú popísané v nasledujúcej časti, potom nasleduje popis modelu údajov a napokon sú špecifikovaný používatelia – tým by sa mali vyjasniť pojmy, ktoré sú v kontextovom diagrame použité.

3.2 Funkcie systému

3.2.1 Funkcie systému z pohľadu oponenta

Systém bude realizovať rozhranie, pomocou ktorého sa budú môcť oponenti:

- Prihlásiť a definovať svoje preferencie. Preferencie oponenta vyjadrujú znalosti v danej

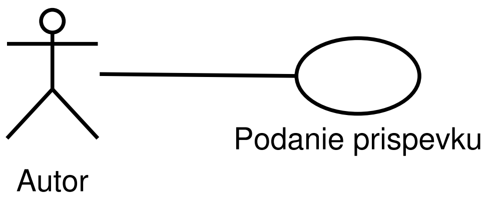


Obrázok 2: Akcie oponenta

oblasti, ktorá je definovaná kľúčovými slovami. Tieto preferencie bude možné vybrať z preddefinovanej množiny preferencií.

- Podanie vlastného posudku na pridelený článok. Pri tejto akcii oponent zároveň vyplní hodnotiace kritéria, ktoré budú stanovené pri štarte systému.

3.2.2 Funkcie systému z pohľadu autora

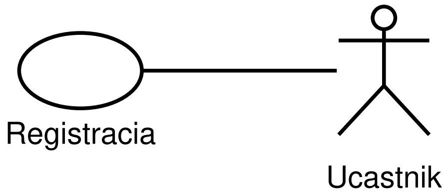


Obrázok 3: Akcie autora

Rozhranie v prípade prístupu autora bude umožnovať:

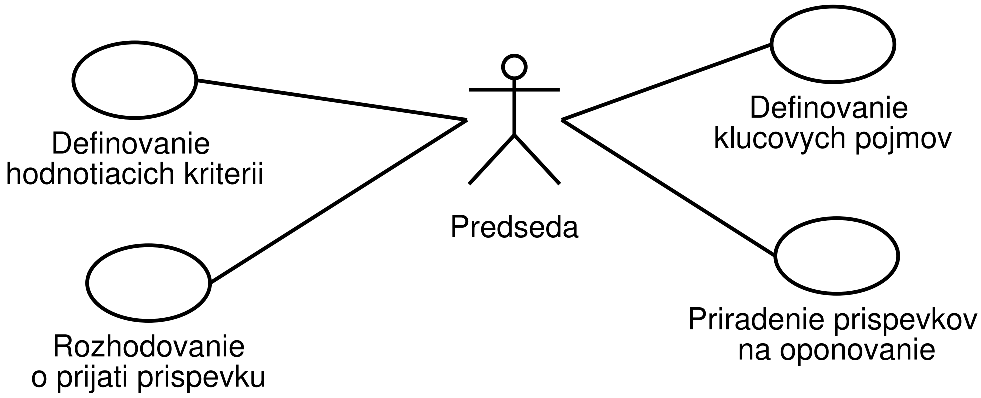
- Podanie príspevku do systému. Pri podávaní sa zároveň zvolí téma príspevku – bude možné vybrať si z definovaných kľúčových slov, ktoré vhodne popisujú obsah príspevku.

3.2.3 Funkcie systému z pohľadu účastníka



Obrázok 4: Akcie učastníka

Systém bude poskytovať možnosť podávania prihlášky na konferenciu prostredníctvom internetu. Toto bude umožnené vytvorením rozhrania pre vyplnenie registrácie učastníka konferencie.



Obrázok 5: Akcie predsedu

3.2.4 Funkcie systému z pohľadu predsedu

Funkcie systému pre predsedu konferencie budú umožňovať:

- Definovanie klúčových pojmov konferencie. Pri tejto akcii bude možné definovať klúčové slová, ktoré popisujú jednotlivé tématické okruhy konferencie a zároveň preferencie oponentov.
- Definovanie hodnotiacich kriterií umožnuje definovať klúčové slova, ktoré budú vhodne slovne pomenovávať jednotlivé hodnotiace kritéria.
- Priradenie príspevkov na oponovanie. Pri tomto úkone predseda bude môcť priradiť prijaté články prihlaseným oponentom, pričom mu môže pomôcť systém na základe definovaných tém článku a preferencií oponenta.
- Rozhodovanie o prijatí príspevku. V tejto akcií bude predseda rozhodovať o sporných článkoch, ktoré neboli automaticky prijaté, prípadne zamietnuté.

3.3 Model údajov

3.3.1 Popis jednotlivých dátových entít:

Clovek abstraktná dátová entita, ktorá bude slúžiť iba ako entita, ktorá definuje základné atribúty pre identifikáciu používateľa systému.

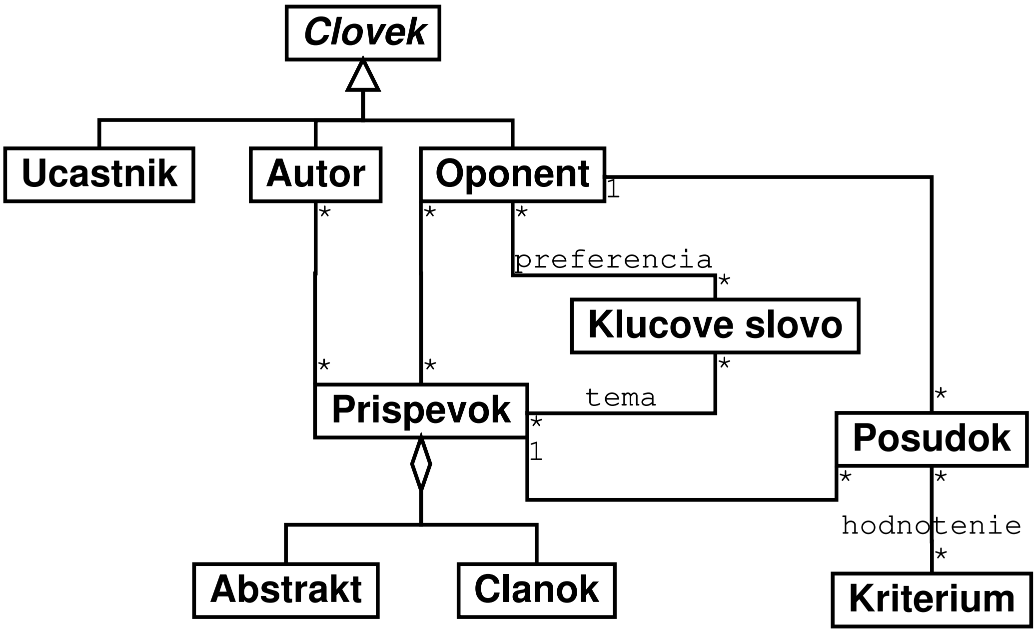
Ucastnik rozšírujúca entita entity *Clovek*. Táto entita je iba obohatená o niektoré atribúty nutné pre evidenciu účastníkov konferencie.

Autor dátova entita rozširujúca entitu *Clovek*. Bude obsahovať údaje o autoroch článkov.

Oponent entita rozširujúca entitu *Clovek*, podobne ako *Autor*. Jej atribúty budú presnejšie definovať oponenta.

Klucove slovo ide o dátovú entitu, ktorá pomenuváva tématické okruhy konferencie a zároveň preferencie oponentov. Tieto dve navzájom rôzne entity bolo nutné vsunúť do jednej, kvôli automatickému priraďovaniu príspevkov k oponentom.

Prispevok dátová entita, ktorá obsahuje príspevok autora, pre systém bude viditeľný aspoň abstrakt a samotný text článku.



Obrázok 6: Model údajov v systéme vedeckej konferencie

Posudok dátová entita obsahujúca text posudku na daný článok.

Kriterium dátová entita osahujúca textový popis kritéria, ktoré by mal spĺňať príspevok.

3.3.2 Popis relácií:

Autor – Prispevok Definuje vzťah, že autor napísal niekoľko príspevkov, prípadne ešte ani jeden. V opačnom smere vyjadruje táto relácia možnosť napísania príspevku aspoň jedným autorom, prípadne kolektívom autorov.

Príspevok – klucove slovo: tema Definuje priradenie jednému príspevku niekoľko možných tématických okruhov – klúčových slov. Pre príspevok nie je nutné definovať tému (priradiť mu kľúčové slová), no je to vhodné. Jedno klúčové slovo môže byť použité na identifikáciu niekoľkých príspevkov, z čoho vyplýva možnosť mať v systéme niekoľko článkov na tú istú tému. Samozrejme nie je nutné priradiť príspevkom všetky kľúčové slová.

Prispevok – oponent Definuje vzťah medzi oponentom a zaslaným príspevkom. Vyjadruje priradenie príspevku aspoň jednému, prípadne viacerým oponentom. Oponent nemusí mať priradený ani jeden príspevok.

Oponent – klucove slovo: preferencie Vyjadruje vzťah medzi oponentom a jeho preferenciami. Svoje preferencie si oponent zvolí podľa toho, aké tématické okruhy ovláda a je schopný ich oponovať.

Oponent – posudok Popisuje, že daný oponent napísal posudok. Je možné, aby oponent napísal niekoľko rôznych posudkov, prípadne žiaden. Každý posudok má definovaného iba jedného oponenta, nemôže existovať jeden a ten istý posudok od viacerých oponentov.

Posudok – prispevok Opisuje vzťah medzi posudkom a príspevkom. Jeden príspevok môže mať niekoľko rôznych posudkov, prípadne ani jeden. Posudok môže byť priradený nanajvýš jednému príspevku.

Posudok – kriterium: hodnotenie Táto relácia vyjadruje vzťah medzi posudkom a dátovou entitou kritéria – ide vlastne o hodnotenie – pričom táto relácia definuje aj výšku samotného hodnotenia. Nie je striktne povedané, že je nutné používať hodnotenie – vtedy posudok nebude mať priradené ani jedno kritérium. V opačnom prípade je možné priradiť všetky kritéria k posudku.

3.4 Používatelia systému

Systém Con4U budú cez webovské rozhranie používať používatelia, ktorý budú vystupovať v rôznych rolách. V ďalších častiach budeme opisovať možné roly používateľov tohto systému.

Každý používateľ (okrem anonymného používateľa) bude mať svoje prihlasovacie údaje a pre prácu so systémom sa bude pomocou nich autentifikovať. Jedna osoba môže disponovať viacerými rolami – ak bude nejaký používateľ súčasne administrátorom aj oponentom, bude mať priradené len jedno prihlasovacie meno a heslo.

3.4.1 Anonymný používateľ

Anonymný používateľ má možnosť si prezerať len statické web-stránky o konferencii, ktoré v skutočnosti ani nebudú pod správou Con4U. Tieto môžu zahŕňať informácie o samotnej konferencii (program, niektorí účastníci, sponzori, organizátori a podobne) a aj informácie konkrétnejšieho charakteru, ktoré môžu mať podobný dizajn ako stránky spravované Con4U systémom a môžu s nimi byť aj hypertextovo zviazané, ale nie sú ním generované.

Okrem toho môže byť systém nastavený tak, že si anonymní používatelia môžu prezerať aj niektoré informácie o prijatých článkoch – napríklad mená autorov a názvy článkov, alebo aj abstrakty. Táto možnosť môže byť zapnutá pri nastavovaní konfiguračných parametrov systému administrátorom.

O anonymných používateľoch systém nebude evidovať žiadne informácie. Anonymní používatelia nemôžu meniť žiadne dáta v systéme a nemôžu si prezerať žiadne iné dáta v systéme ako tie, ktoré boli vopred povolené administrátorom.

3.4.2 Účastník konferencie

Účastník konferencie zadáva osobné údaje prípadne iné údaje do formulára. Tieto údaje použijú organizátori pri plánovaní organizačných detailov konferencie. Niektoré položky (napríklad týkajúce sa platenia poplatkov za účasť) sa vzťahujú síce k účastníkovi konferencie, vypĺňať ich však bude administrátor.

O účastníkoch konferencie bude systém evidovať údaje (ako je popísané v 3.3). Časť týchto údajov bude do systému vkladať sám účastník, časť bude zapisovať administrátor. Účastník si môže prezerať a modifikovať len údaje o svojej prihláške na konferenciu – žiadne iné.

3.4.3 Autor príspevku

Autor príspevku do vedeckej konferencie bude pri podávaní článku zadávať svoje osobné údaje, údaje o článku a článok samotný. Tento článok bude posudzovaný oponentami a prijatý alebo neprijatý na konferenciu. Ako údaje o článku bude autor vyznačovať kategórie – kľúčové slová, ktorých sa jeho článok týka. Zoznam týchto kľúčovýh slov bude konfigurovateľný a bude ich zadávať administrátor.

Autor si môže prezerať a modifikovať okrem svojich osobných údajov ešte údaje o svojich článkoch. Nebude môcť prezerať žiadne iné údaje okrem tých, ktoré si bude prezerať ako anonymný používateľ.

3.4.4 Oponent – člen programovej komisie

Člen programovej komisie si na začiatku práce so systémom vyberie z preddefinovaných kategórií (kľúčových slov) svoje preferencie. Podľa týchto preferencií mu bude systém prideľovať príspevky na oponovanie. Toto prideľovanie príspevkov na oponovanie bude vykonávať administrátor, Con4U mu v tom však bude radiť vytváraním zoznamov možných oponentov podľa poradia najväčšej zhody oponentových preferencií a kľúčových slov článku.

Na samotné recenzovanie pridelených článkov bude mať možnosť si všetky tieto články zo systému stiahnuť. Po ich preštudovaní oponent vyplní formulár posudku s preddefinovanými otázkami, týkajúcimi sa rôznych kritérií kvality článku, napr. originalita, správnosť, forma, zhoda s predmetom konferencie a podobne. Nakoniec určí mieru svojho odporúčania/neodporúčania prijatia tohoto článku do programu konferencie.

Oponent nebude poznať ostatných oponentov priradených k článku a nebude vidieť ani hodnotenie článku ostatnými oponentmi. Bude si môcť prezerať, prípadne modifikovať len svoje vlastné posudky.

3.4.5 Administrátor – predseda programovej komisie

Administrátor na začiatku nakonfiguruje systém pre danú konferenciu – určí možné kľúčové slová (oblasti) článkov, nastaví možnosti prezerania dát pre anonymných používateľov, priradí používateľov do rolí členov programovej komisie.

Počas etapy prijímania článkov a preferencií oponentov bude sledovať pridávané články, zjavne nevhodné príspevky (žarty, omyly, viacnásobne sa opakujúce články a podobne) bude rušiť.

Ďalšou úlohou predsedu programovej komisie je prideľovanie článkov jednotlivým oponentom na posudzovanie. Túto činnosť mu bude Con4U uľahčovať generovaním zoznamov vhodných oponentov k článkom (prípadne aj zoznamov vhodných článkov k oponentom) podľa miery zhody kľúčových slov článku a preferencií oponentov.

Počas bežiacej etapy posudzovania článkov bude sledovať progres jednotlivých oponentov pri posudzovaní vo forme tabuliek s počtom ukončaných a chýbajúcich posudkov a jeho úlohou bude pripomenúť zaostávajúcim oponentom, že by mali dôslednejšie plniť svoju úlohu.

Po skončení etapy posudzovania bude mať predseda spolu s členmi programovej komisie k dispozícii generované poradia úspešností článkov podľa rôznych kritérií. Ich spoločnou úlohou bude určiť, ktoré články budú prijaté, a ktoré zamietnuté.

Po tom, ako sa programová komisia dohodne na zozname prijatých článkov, predseda tieto články vyznačí v systéme Con4U a systém to oznámi mailami autorom článkov. Generovaním zoznamov prijatých článkov sa úloha administrátora a aj celého systému Con4U končí.

4 Návrh

4.1 Hrubý návrh dátového modelu

Hrubý návrh dátového modelu je oproti modelu z kapitoly 3.3 obohatený o významné atribúty jednotlivých dátových entít. Zároveň boli rozbité niektoré asociačné väzby, nakoľko aj tie obsahujú isté atribúty potrebné pre presnejší opis vzťahu. Zároveň sa tým eliminovali nežiadúce vzťahy M ku N. Vzťahy M ku N, ktoré nenesú v asociácii žiadne atribúty okrem cudzích kľúčov, boli ponechané bez zmeny.

4.1.1 Popis atribútov neväzobných dátových entít

Dátové entity, ktoré boli opísané vo fáze analýzy a špecifikácie, sa v tejto časti už opisovať nebudú a dátové entity, ktoré vznikli až vo fáze hrubého návrhu, budú stručne opísane, aj keď budú mať rovnaký význam ako nimi nahradzované asociácie.

Clovek

meno, priezvisko skutočné meno osoby, ktorá chce používať systém ako účastík, autor, prípadne oponent;

e-mail kontaktnú e-mailovú adresu danej osoby;

heslo prístupové heslo do systému.

Prispevok

abstrakt abstrakt príspevku – ten je nutný vyplniť extra, aby mohol systém vytvárať program konferencie;

clanok celý článok autora;

dátum prijatia dátum skutočného prijatia prispevku;

stav atribút identifikujúci stav príspevku (prijatý na konferenciu alebo zamietnutý, pripravená alebo nepripravená prezentácia, odprezentovaný príspevok, atď...).

Klucove slovo

meno jednoduchý názov kľúčového slova;

popis presnejší opis významu kľúčového slova.

Posudok

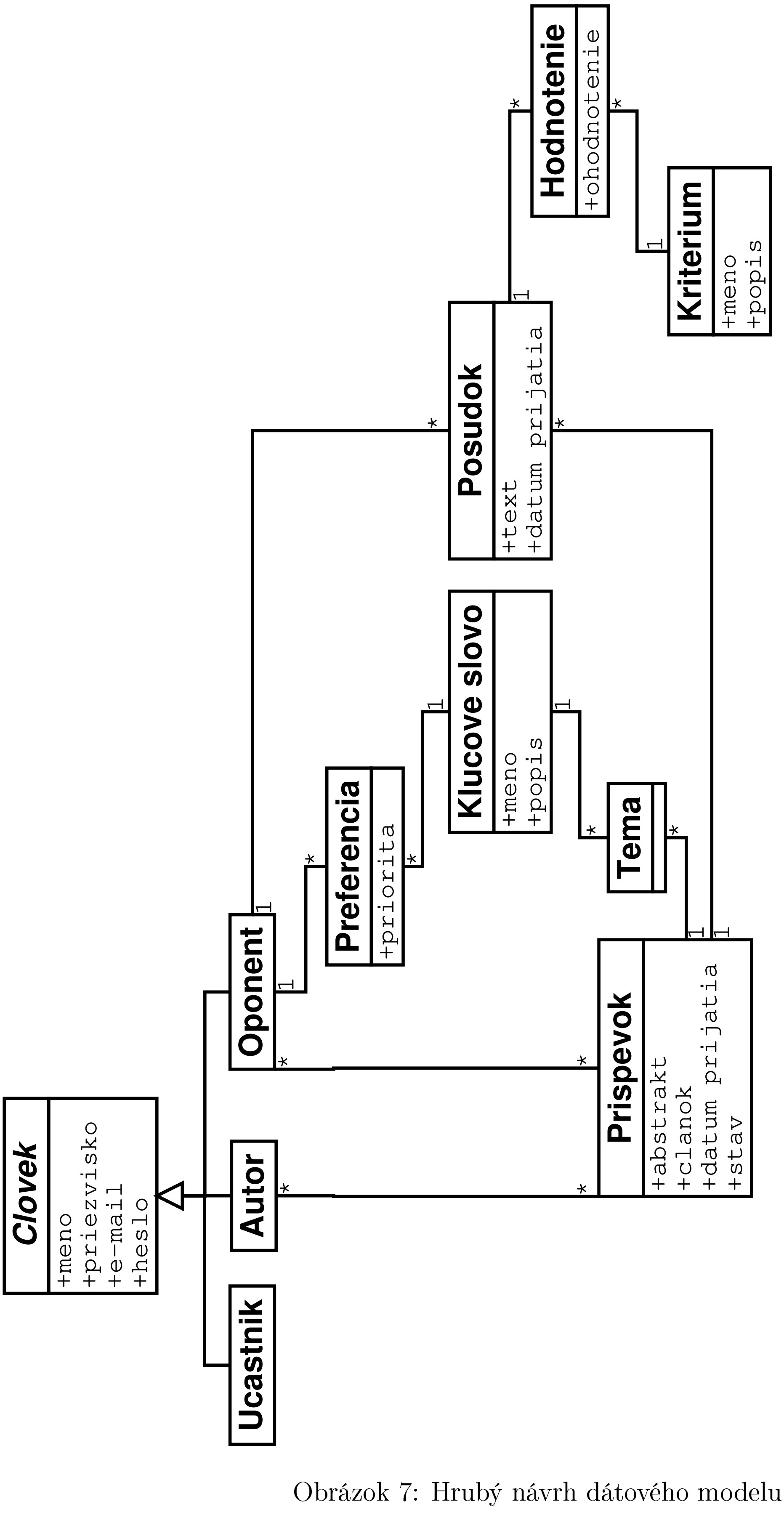
text obsah posudku, ktorý napísal oponent na článok;

dátum prijatia dátum skutočného prijatia posudku.

Kriterium

meno jednoduchý názov kritéria;

popis presnejší opis významu daného kritéria.



Obrázok 7: Hrubý návrh dátového modelu

4.1.2 Popis atribútov väzobných dátových entít

Preferencia dátová entita vyjadrujúca priradenia jednotlivých klúčových slov – preferencií jednotlivým oponentom.

priorita určuje číselne vyjadrenie znalosti danej preferencie oponenta.[8]

Tema dátová entita určujúca vzťah medzi článkom a kľúčovým slovom – témou. Každý článok spadá pod jednu alebo niekoľko tém konferencie, autor nemusí túto väzobnú entitu nutne vytvárať (cez web-rozhranie aplikácie). Toto bude mať za následok, že programová logika nebude vedieť automaticky priradiť článok konkrétnemu oponentovi podľa preferencií.

Hodnotenie dátová entita určujúca vzťah medzi posudkom a jednotlivými hodnotiacimi kritériami, zadaných predsedom konferencie.

ohodnotenie číselne vyjadrenie miery splnenia daného kritéria, na ktoré odkazuje hodnotenie.

[8]V systéme sa bude používať automatické priradovanie číselných hodnôt na základe zotriedeného vstupu preferencií od oponenta.