Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Systémové autentizační mechanismy v Javě

Plzeň, 2007 Jan Boháč

5 Stávající implementace přihlašování uživatelů do OpenCms

Přihlášení do systému OpenCms je řešeno standardním způsobem jako u většiny webových aplikací - tedy ověřením identity uživatele zadaným jménem a heslem proti interní databázi systému. Pokud jsou tato data správně vložena, systém uživatele přihlásí.

Celý proces začíná na úvodní obrazovce s formulářem pro vložení identifikačních údajů a končí porovnáním těchto informací s tabulkou `cms_users`, kde jsou uloženy záznamy o všech uživatelích systému. Není zde použit žádný sofistikovaný autentizační mechanizmus, jenž by zabezpečoval přihlašování.

5.1 Třídy zapojené do procesu přihlašování

UML schéma s třídami, které se zapojují do procesu přihlašování, je zobrazeno jako příloha 1. Dynamický model vztahů mezi těmito třídami je zobrazen jako: příloha 2: diagram spolupráce tříd při přihlašování.

Dále jsou popsány třídy, které se do tohoto procesu zapojují:

`CmsLogin`

- metoda `displayDialog()`
- balík `org.opencms.workspace`

Tato třída zajišťuje zobrazení `html` kódu s přihlašovací stránkou a zobrazení oken s informacemi o případné chybě při přihlašování nebo zprávou od administrátora systému.

Metoda `displayDialog()` zobrazuje úvodní přihlašovací stánku s formulářem pro zadání uživatelského jména a hesla. Po jejich vložení a stisku tlačítka *Login* se zahájí proces přihlašování uživatele.

CmsJspLoginBean

- metoda `loginUser(userName, password)`
- balík `org.opencms.jsp`

Obsahuje metody potřebné pro zajištění funkce přihlašovací stránky. Je jakýmsi prostředníkem mezi jádrem systému OpenCms a formulářem, který zprostředkovává vkládání identifikačních dat. Díky tomu je možné napsat vlastní přihlašovací stránku, která bude zapadat do celkového designu vytvářeného projektu a splňovat požadovanou funkčnost.

Metoda `loginUser()` zajišťuje předání uživatelských dat dále do třídy `CmsObject` a zpracovává případné chyby, které mohou při autentizaci vzniknout. Jedinou výjimkou, která ve stávající implementaci reprezentuje chybu při autentizaci, je `CmsAuthentificationException`. Ta je v této metodě odchycena a jako reakce na ni je vytvořena zpráva, která je pomocí metody `displayDialog()` třídy `CmsLogin` zobrazena uživateli a popis vzniklé chyby je zaznamenán do log souboru. Pokud autentizace proběhla v pořádku, jsou informace o uživateli a o čase přihlášení také zapsány do log souboru.

CmsObject

- metoda `loginUser(userName,password,remoteAddress,type)`
- balík `org.opencms.file`

Toto je základní třída zabezpečující autorizovaný přístup do VFS OpenCms. Zapouzdřuje uživatelovu identifikaci a jeho oprávnění k využívání zdrojů systému. Při přístupu do VFS kontroluje, zda uživatel má na danou akci povolení (autorizaci).

Metoda `loginUser()` má na starost přihlášení do systému – pokud jsou identifikační údaje správné, inicializuje spuštění workplace sytému a přepnutí na přihlášeného uživatele. Třídě `CmsJspLoginBean` poté vrací uživatelské jméno. Pokud se během ověřování uživatelem zadaného jména a hesla vyskytla chyba, propaguje tato metoda výjimku `CmsException`.

CmsSecurityManager

- metoda `loginUser(CmsRequestContext, userName, password, remoteAddress, userType)`
- balík org.opencms.db

Tato třída kontroluje povolení, potřebná pro uživatelovy akce vyvolané objektem `CmsObject`. Pokud k nim uživatel má právo, vyvolá `CmsSecurityManager` odpovídající akci ve třídě `CmsDriverManager`, která zajišťuje přístup k databázi.

Metoda `loginUser()` se pokouší o autentizaci uživatele pomocí předaného hesla. Pokud uspěje, vrací instanci třídy `CmsUser`, která odpovídá správně přihlášenému uživateli. Stejně jako `CmsObject` propaguje výjimku `CmsException`.

CmsDriverManager

- metoda `loginUser(CmsDbContext,userName,password,Address,type)`
- balík `org.opencms.db`

Pomocí metod rozhranní `I_CmsUserDriver` zajišťuje nepřímý přístup k databázi systému. Díky tomu je tato třída odstíněna od toho, na jakém databázovém stroji systém OpenCms běží. Neboť všechny třídy, které jsou odpovědné za přímé čtení z databáze, toho rozhraní implementují.

Metoda `loginUser()` již zastupuje přístup do databáze a porovnávání uživatelem zadaného jména a hesla s daty z tabulky `cms_users`. Pokud se shodují, vrací instanci třídy `CmsUser`. Vyskytne-li se při čtení nějaká chyba, například nesprávně zadané jméno nebo špatné heslo, je vyvolána výjimka `CmsAthenticationException`, která se propaguje až do třídy `CmsJspLoginBean`. Jestliže autentizace uspěje, provádí se kontrola dalších údajů, které souvisejí s právě autentifikovaným uživatelem. Prověří se, zda uživatel nepřekročil limit pokusů o přístup s chybným heslem nebo zda je v daný okamžik povoleno přihlašování.

Popis činnosti této metody je zobrazen diagramem aktivit a pseudokódem v příloze 3.

CmsLoginManager

- balík `org.opencms.db`

Zajišťuje metody pro kontrolu přihlašování uživatelů. Ukládá si záznamy o neúspěšných pokusech o přihlášení a při překročení jejich limitu umožňuje dočasně znepřístupnit provinilému uživateli vstup do systému. Tyto záznamy si uchovává ve formě: jméno, IP adresa, typ uživatele. To znamená, že uživateli může být zakázáno přihlašování z jedné IP adresy, ale z jiné adresy se stále může přihlásit. Dovoluje na krátkou dobu úplně znepřístupnit přihlašování (například po dobu údržby systému).

CmsUser

- balík `org.opencms.file`

Tato třída reprezentuje uživatele v OpenCms. Sdružuje o něm všechna potřebná data (jméno, email, login, heslo, čas posledního přihlášení do systému…) a metody pro jejich nastavování a vracení.

interface I_CmsUserDriver

- balík org.opencms.db

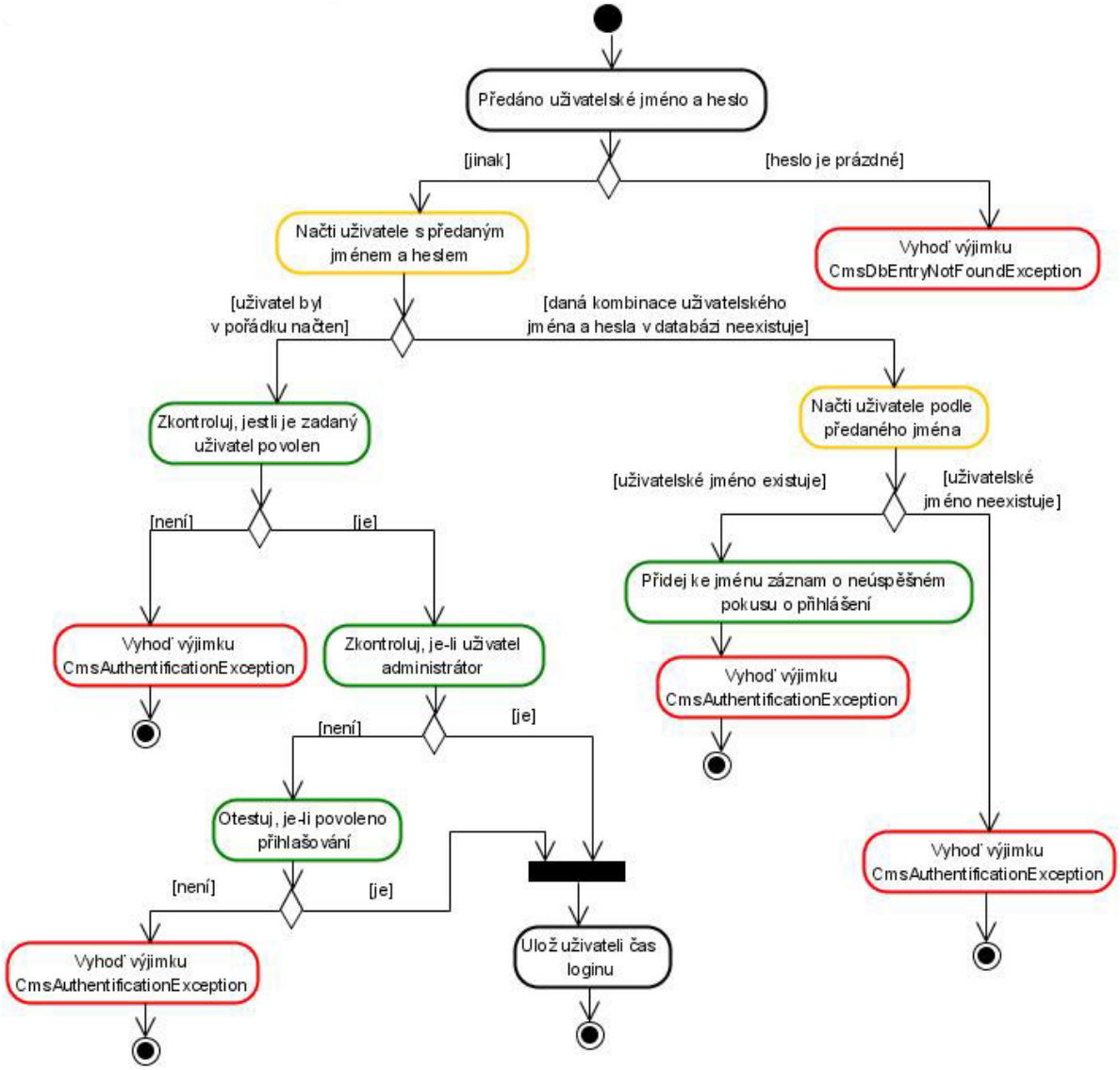
Toto rozhraní definuje metody pro správu uživatelů (vytváření, mazání a upravování uživatelů a skupin), které musí být implementovány třídou, jenž zajišťuje přímý přístup do databáze. Její metody využívá třída `CmsDriverManager`. Ta díky tomu zůstává odstíněna od tříd, které zajišťují komunikaci s databázovým serverem.

Výše popsané třídy lze rozdělit do tří kategorií:

- Zajištění přístup k databázi uživatelů – `CmsUserDriver` a rozhraní `I_CmsUserDriver`.
- Kontrola a řízení přístupu k VFS a databázi systému – `CmsObject`, `CmsSecurityDriver`, `CmsLoginManager` a částečně i `CmsUser`.
- Reakce na vzniklé výjimky, zobrazování úvodní přihlašovací obrazovky a informačních oken - `CmsJspLoginBean` a `CmsLogin`.

Příloha 3

Diagram aktivit v metodě `CmsDriverManager.loginUser()` – originální verze.



Z důvodu větší přehlednosti tištěné verze jsem pro aktivity, které jsou zajišťovány metodami jiných tříd, použil barevné oddělení:

- černá – aktivity prováděné přímo v metodě `loginUser()`
- zelená – činnosti zajišťované třídou `CmsLoginManager`
- žlutá – aktivity prováděné metodami rozhraní `I_CmsUserDriver`

Pseudokód metody `CmsDriverManager.loginUser()`

```
public CmsUser loginUser(CmsDbContext, userName, password,
  remoteAddress, userType) {
  if heslo jsou jen prázdné znaky {
      vyhoď výjimku CmsDbEntryNotFoundException
  }
  Vytvoř novou instanci CmsUser;
  try{
      pokus se do CmsUser načíst uživatele z databáze pomocí zadaného hesla a jména
  } catch(CmsDbEntryNotFoundException){
      // v databázi nebyl nalezen záznam odpovídající kombinaci hesla a jména
      boolean userExists  = true
      try{
          pokus se načíst do CmsUser uživatele se zadaným jménem
      }catch(CmsDbEntryNotFoundException){
          // v databázi se nenachází uživatel se zadaným jménem
          userExists = false
      }
      if (userExists) {
          přidá do databáze záznam o neúspěšném loginu
          vyhodí výjimku CmsAuthentificationException
      }else{
          vyhodí výjimku CmsAuthentificationException
      }
  }
  if uživatel existuje, ale je zakázán {
      otestuje, zda nebyl u daného uživatele překročen maximální počet neúspěšných
      pokusů o přihlášení případně odstraní všechny předchozí neúspěšné pokusy o
      přihlášení
  }
  if přihlašující se uživatel nemá roli administrátora {
      zkontroluje, zda je aktuálně povoleno přihlašování uživatelů
  }
  nastaví přihlášenému uživateli čas, kdy by přihlášen
  vyčisti potřebné cashe paměti
  }
}
```