**Реферат**

Об’єктом розробки є система електронного документообігу.

Під час вивчення предметної області, було проаналізовано основні концепції розробки адаптивних систем, проаналізовано основні фактори, що впливають на процес обміну електронними документами. Також було вивчені основні аналоги та їх можливості, по захисту даних, по зручності та надійності. На основі зібраних даних та на основі аналізу існуючих систем була розроблена і реалізована система електроного документообігу «EFOD».

У результаті роботи була реалізована система електронного документообігу, при реалізації якої використовувались технології Java Servlet, JSP, сервер бази даних MySQL і Web-сервер Apache Tomcat.

**Вступ**

В умовах ринкової економіки та зростання конкурентної боротьби помилки в діяльності будь-якої економічної системи (підприємства, фірми або корпорації) можуть істотно позначитися на фінансових результатах її діяльності і навіть призвести до банкрутства. Це викликає необхідність підвищення ефективності управління за рахунок прийняття оптимальних стратегічних та оперативних управлінських рішень та вдосконалення технології управління. Під управлінням розуміють процес завдання мети функціонування об'єкта і вироблення впливів на цей об'єкт в напрямку досягнення поставленої мети. Мета функціонування підприємства реалізується за допомогою ряду ділових процесів, що складаються з окремих компонентів, взаємодія яких задається певним регламентом їх виконання. Об'єктами управління можуть бути:

1. матеріальні;
2. грошові;
3. трудові або інформаційні ресурси, які застосовуються при виконанні ділових процесів;
4. самі ділові процеси, що виконуються на підприємстві для досягнення поставленої мети;
5. підрозділи підприємства, де протікають ділові процеси, або все підприємство в цілому.

Щоб стримувати натиск конкуруючих підприємств і організацій, бізнес повинен успішно адаптуватися в мінливому оточенні, тому відзначається прагнення керівництва компаній, з одного боку, до реструктуризації систем управління та вдосконалення процедур управління та ділових процесів, і, з іншого, до зменшення обсягів паперової інформації, зниження вартості обробки інформації та підвищенню ефективності управління нею на шляху організації спільного доступу фахівців до інформації, зберігання та пошуку документів.

Нові інформаційні технології пропонують засоби для досягнення цих цілей за допомогою двох підходів:

1. реінжинірингу бізнес - процесів;
2. переходу до безпаперової технології управління.

У даній роботі розглядається зміст другого підходу - вибору та застосування методів і засобів організації безпаперової технології управління, яка ґрунтується на використанні трьох основних концепцій: переходу до електронних документів, створення систем управління документами та системи електронного документообігу.

Концепція електронного документа з'явилася у 80-х роках з появою на ринку мікрокомп'ютерів і першого графічного інтерфейсу користувача (GUI). Як стверджують співробітники компанії Delphi Consulting Group, вже сьогодні в деяких областях діяльності електронним способом обробляється до 90% інформації.

На відміну від документів на паперових носіях з їх жорсткими рамками, статичної формою і обмеженими можливостями перехід до динамічним цифровим електронним документам забезпечує особливі переваги при створенні, спільному використанні, поширенні та зберіганні інформації. Вони можуть збільшити продуктивність безлічі програм, що використовуються в бізнесі, зменшити вимоги до розмірів накопичувачів, скоротити або повністю вирішити проблему втрат і невірного розміщення документів.

Електронні документи можуть одночасно використовуватися співробітниками в рамках однієї робочої групи, відділу або всього підприємства. Доступ до них здійснюється за кілька секунд, а не хвилин, годин, днів, а іноді й тижнів, що трапляється при використанні документів на паперових носіях. Прискорений доступ до стратегічної інформації поряд зі значною економією коштів може забезпечити і важливі конкурентні переваги.

Крім перерахованого, електронні документи дозволяють перемістити центр ваги комп'ютерної технології з традиційних структурованих алфавітно-цифрових даних на потоки даних, доповнені великими обсягами неструктурованого тексту, зображень, звуку, відео і графіки. Такі документи зможуть також включати гіпертекстові зв'язку, перероблені OLE-об'єкти, текстові об'єкти і реляційні дані. Електронний документ буде обмежений такими параметрами, як його вміст, структура даних, формати і стандарти режиму передачі і, найважливіше, характер його використання. При зміні будь-якого з цих параметрів відповідно буде змінюватися документ. Він буде відкритим, гнучким, адаптивним, багатовимірним.

За кілька років концепція електронного документа отримала свій розвиток від звичайного графічного образу документа до ідеї управління документами. Сьогодні електронний документ - це форма знайомого виду, обробка якої відбувається за допомогою послідовного застосування тісно взаємопов'язаних технологій в рамках так званих систем управління електронними документами (СУЕД) або Electronic Document Management Systems (EDMS).

Потужні системи класу EDMS, що працюють найчастіше на розподілених архітектурах, засновані на комбінації технологій збору, зберігання, пошуку і розповсюдження електронних документів. Значне підвищення продуктивності при використанні технології EDMS відзначено в багатьох галузях:

1. уряд;
2. судочинство;
3. юриспруденція;
4. фармацевтика;
5. страхування;
6. фінансові послуги;
7. охорона здоров'я;
8. виробництво;
9. система вищої освіти.

Можна виділити наступні основні завдання, які покликана вирішувати система управління документами - вміти перекладати паперові документи в електронний вигляд і вміти перехоплювати всі створювані електронні документи з усіх джерел надходження:

1. з додатків, які створюються текстовими процесорами, електронними таблицями, графічними процесорами, пакетами прикладних програм для настільних видавництв;
2. факсові повідомлення, що надійшли через факс - сервер підприємства;
3. вхідні, вихідні й внутрішні повідомлення, що надходять з електронної пошти;
4. звіти та інші документи, які готують функціональні інформаційні системи, що йдуть на друк або надіслані факсом, але їх треба перехопити і відправити у сховище електронних документів.

Багатофункціональні системи управління документами розвиваються під впливом переходу від обчислювальних середовищ на базі мейнфреймів, до систем на базі мікропроцесорів, від монолітних закритих додатків до відкритих і інтегрованих програмних компонентів.

Змістом третьої концепції є розробка системи електронного документообігу (СЕДО), покликаної для інтеграції всіх інформаційних додатків у єдину інформаційне середовище, що забезпечує оперативну взаємодію всіх користувачів при виконанні ними ділових процедур і функцій управління необхідною інформацією. Система електронного документообігу повинна

1. дозволяти планувати і складати маршрути пересування документів;
2. контролювати це пересування;
3. вміти керувати документообігом і регулювати його.

Ця концепція має дві стадії у своєму розвитку від втілення ідеї використання технології "docflow", пов'язаної з плануванням і управлінням документопотокамі, до технології "workflow", призначеної для управління діловими процесами і процедурами, при виконанні яких використовуються і створюються документи. Системи, засновані на застосуванні технології "workflow", отримали назву систем автоматизації ділових процесів (САДП) і призначені для моделювання ділових процесів і процедур, що виконуються на підприємстві, контролю їх виконання, оцінки та аналізу ефективності їх виконання, реструктуризації системи управління фірми.

**1. Аналіз предметної області**

В даний час в Україні спостерігається зростання інтересу до систем електронного документообігу (СЕД) серед державних і комерційних структур. Компанії-розробники готові надати СЕД як для невеликих організацій чисельністю 5-10 осіб, так і для великих підприємств чисельністю тисячі чоловік. Всього на ринку представлено більше п'ятдесяти рішень цього класу. Але на питання, за якими критеріями правильно вибирати систему, відповідь знають далеко не всі. Проведемо аналіз найбільш популярних вітчизняних розробок в області автоматизації документообігу і діловодства.

Сьогодні можна виділити наступні рішення від провідних компаній-розробників, що працюють у сфері автоматизації роботи з документами:

1. Бос-Референт;
2. ГранДок;
3. Справа;
4. Євфрат-Документообіг;
5. CompanyMedia;
6. Directum;
7. DIS-системи;
8. DocsVision;
9. LanDocs;
10. Optima-Workflow.

СЕД - це в першу чергу система, що дозволяє вирішувати усі типові завдання електронного документообігу для роботи з документами - реєстрація і введення документів, пошук документів, маршрутизація, створення звітів, ведення архіву, встановлення правами доступу в системі. Тому при виборі СЕД просто необхідно переконатися в тому, що система дійсно забезпечує виконання таких завдань. Щоб отримати уявлення про можливості систем, що дозволяють організувати електронний документообіг, була складена таблиця з необхідним набором функціонала (таблиця 1). Як показано в таблиці, всі представлені системи намагаються йти в ногу з часом і тому володіють майже всіма необхідними функціональними можливостями. Найкращим чином з типовими завданнями електронного документообігу справляються системи Дело, Євфрат-Документообіг, DocsVision і LanDocs.

Умовні позначення систем в таблицях:

1. Бос-Референт (Б-Р);
2. ГранДок (ГД);
3. "Дело" (Д);
4. Євфрат-Документообіг (Є-Д);
5. CompanyMedia (CM);
6. Directum (D);
7. DIS-системи (DIS);
8. DocsVision (DV);
9. LanDocs (LD);
10. Optima-Workflow (OW).

Пояснення до таблиць:

1. "+" - можливість є;
2. "-" - можливості немає;
3. "+/-" - опціонально або зовнішнім ПО;
4. "-/+" - неповна функціональність можливості.

Наступним моментом, на який варто звернути увагу при виборі тієї чи іншої системи, є послуги, що надаються компаніями-розробниками, і технічні вимоги до системи. Для повноцінної роботи з системою в першу чергу необхідно пройти навчання. Розробник повинен надати постійно діючу оперативну технічну підтримку. Послуги з встановлення та налаштування системи, пропоновані розробниками, є необов'язковими і можуть знадобитися тільки в тому випадку, якщо власні IT-фахівці не в силах самостійно здійснити адаптацію системи і введення її в експлуатацію. Знання технічних вимог до системи дозволяє заздалегідь оцінити витрати на придбання додаткового програмного забезпечення і нового обладнання.

Сумарні витрати на придбання системи складаються з вартості ліцензій програмного продукту, бази даних, послуг, додаткових модулів системи і стороннього програмного забезпечення. За загальної вартості серверної і користувальницьких ліцензій, а також вартості СУБД / платформи, найбільш прийнятними є системи:

1. ГранДок;
2. Євфрат-Документообіг;
3. DocsVision.

Вартість бази даних і програмних платформ варіюється від 900 $ (MS SQL Server 2005) до 4000 $ (Lotus Notes Domino). Винятком є система Євфрат-Документообіг, що має вбудовану СУБД Ніка (0 $) (таблиця 2). Порівняння повних витрат на впровадження систем практично нездійсненно в силу різного підходу компаній-розробників до надання послуг. У примітках до таблиць можна знайти інформацію про використовувані системами бази даних і платформи, а також про модулі, вартість яких врахована для систем LanDocs і CompanyMedia.

Таблиця 1. Основні можливості систем електронного документообігу

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Система  можливість | Б-Р | Г-Д | Д | Є-Д | CM | D | DIS | DV | LD | O-W |
| реєстрація документів | + | + | + | + | + | + | + | + | + | + |
| ведення реєстраційної карточки | + | + | + | + | + | + | + | + | + | + |
| ведення номенклатури справ | + | + | + | + | + | + | + | + | + | + |
| сканування | +/-1,2 | + | +/-1,5 | + | +/-1 | -/+ | + | + | + | +/-1 |
| розпізнавання документів | +/-1,2 | + | +/-1,5 | + | +/-1 | -/+ | +/-1 | +/-1 | +/-1 | +/-1 |
| зв’язані документи | + | + | + | + | + | + | + | + | + | + |
| прикріплені файли | + | + | + | + | + | + | + | + | + | + |
| Робота зі словниками та довідниками | + | + | + | + | + | + | + | + | + | + |
| час доручень | + | - | + | + | + | + | + | + | + | + |
| Пошук:   * за реквізитами; * по виду РК документу; * повнотекстовий; * з урахуванням морфології. | +  -  +  - | +  -  +  - | +  +  +  - | +  +  +  + | +  -  +  - | +  -  +  + | +  +  -  - | +  +  +  - | +  -  +  + | +  +  +  +/- |
| списання документу в архів | + | + | + | + | + | + | - | + | + | + |
| ведення архівів електронних документів | -/+ | - | +/-6 | +/-7 | -/+ | -/+ | - | -/+ | +/-9 | -/+ |
| маршрутизація | + | - | + | + | + | + | - | + | + | + |
| генерація звітів | +/-3 | +/-4 | + | + | +/-8 | + | + | + | + | +/-4 |
| розподілення прав доступу | + | + | + | + | + | + | + | + | + | + |
| ролі | + | - | -/+ | + | + | + | + | + | + | + |
| Примітка:   1. з застосуванням зовнішніх OCR 2. інтеграція з CuneiForm 3. за допомогою Report’sMan 4. за допомогою Cristal Reports 5. опціонально «Потокове сканування» 6. опціонально підсистема «Архівна справа» 7. опціонально модуль «Архіваріус» 8. опціонально за допомогою системи «Центр звітів» 9. опціонально система LanDocs: АРХІВ | | | | | | | | | | |

Таблиця 2. Вартість сервера і лізенцій

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Система  характеристики | Б-Р | Г-Д | Д | Є-Д | CM | D | DIS | DV | LD | O-W |
| СУБД | Lotus  Notes/  Domino | MS SQL Server | MS SQL Server,  Oracle 9i | НІКА  MS SQL4 Oracle 9i4 | Lotus  Notes/  Domino | MS SQL Server | MS SQL Server,  Oracle 9i  Lotus  Notes/  Domino | MS SQL Server | MS SQL Server,  Oracle 9i | MS SQL Server,  Oracle 9i,  IBM DB2 Database |
| Вартість СУБД, $ | від 3964 | від 9002 | від 9002 | 05 | від 3964 | від 9002 | від 9002 | від 9002 | від 9002 | від 9002 |
| Вартість сервера + 50 ліцензій, $ | 176001 | 10200 | 121003 | 7200 | 210006 | 14800 | 197001 | 9200 | 210507 | 18200 |
| Примітка:   1. з урахуванням знижок 2. SQL Server Standart Edition 2005 Win32 English OLP NL 3. під MS SQL Server 4. проектне рішення 5. вбудована СУБД Ніка 6. СМ-справовиробництво + СМ-Workflow (р. м. користувача) 7. LanDocs: СПРАВОВИРОБНИЦТВО + LanDocs: СЕРВЕР ДОКУМЕНТІВ + LanDocs: МАРШРУТИЗАЦІЯ | | | | | | | | | | |

Купуючи систему, не слід забувати про «підводні камені» її впровадження та промислової експлуатації. Може виявитися, що всі достоїнства функціоналу придбаної системи зійдуть «нанівець», коли справа дійде до впровадження і використання системи. Наприклад, якщо система, яка купується не містить інструментарію необхідного для настройки під потреби організації власними силами, то процес впровадження вимагатиме участі розробника, що істотно вплине на підвищення вартості впровадження. Адаптувати систему під потреби своєї організації доступними засобами без участі розробника і без додаткових витрат зможуть власники систем Євфрат-Документообіг і DocsVision. Замовникам інших систем пропонується набір стандартних установок, які можуть бути доопрацьовані розробниками або їх партнерами, що приведе до збільшення витрат на впровадження (таблиця 3).

Вартість і терміни впровадження є одними з факторів, що впливають на поширеність системи. У кожному конкретному впровадженні вони залежать від можливостей обраної системи та структури компанії-замовника (масштаб, специфіку діяльності і т. д.), що набуває систему. За наявними оцінками (таблиця 4), на сьогоднішній день найбільша кількість впроваджень мають системи Євфрат-Документообіг, Дело і CompanyMedia. Система Дело популярна в державних установах завдяки тому, що вона орієнтована на автоматизацію процедур традиційного діловодства. Поширеність системи Євфрат-Документообіг пояснюється універсальністю її використання і гнучкою ціновою політикою. Велика кількість впроваджень CompanyMedia забезпечує модульність системи та реалізація на платформі Lotus Notes / Domino.

Таблиця 3. Доступний інструментарій для налаштування системи

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Система інструментарій | Б-Р | Г-Д | Д | Є-Д | CM | D | DIS | DV | LD | O-W |
| дизайнер форм карточок документів | + | - | -/+3 | + | +/-4 | -/+7 | - | + | -/+8 | + |
| дизайнер маршрутів/процесів | + | - | -/+ | + | +/-5 | -/+7 | - | + | + | + |
| редактор звітів | +/-1 | +/-2 | - | + | +/-6 | -/+7 | -/+ | + | + | +/-2 |
| створювання та зміна словників та довідників | + | + | - | + | + | -/+7 | - | + | + | + |
| Примітка:   1. за допомогою Report’sMan 2. за допомогою Cristal Reports 3. можливість додавання нових реквизитів 4. додатково, за допомогою інструментів для розробки на ActiveFrame 5. додаткова система CM-Workflow (р. м. розробника) 6. додатково, система «Центр звітів» 7. за допомогою інструменту розробки IS-Builder 8. настройки реквізитного складу реєстраційної карточки | | | | | | | | | | |

Таблиця 4. Досвід впровадження систем

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Система питання | Б-Р | Г-Д | Д | Є-Д | CM | D | DIS | DV | LD | O-W |
| кількість впроваджень | >100 | >100 | >5001 | >1000 | >5002 | >100 | >100 | >100 | >100 | >100 |
| чи можливе впровадження власними силами\* | ні | ні | ні/так | Так | Ні | ні/так | ні/так | так/ні | ні | ні |
| впровадження проектні (П) та власними (С)силами? \*\* | П | П | П<С | П<<С | П | П>>С | П>С | П>С | П | П |
| Примітка: \* - «ні/так» - жорсткі настройки, «так/ні» - необхідний спеціаліст  \*\* - П > (<) С – проектних впроваджень більше (менше) ніж власними силами; «>> (<<)» - значно більше (менше);  П – проектні впровадження  1 – враховуються тільки впровадження продукту Дело  2 – з урахуванням впроваджень продукту OfficeMedia | | | | | | | | | | |

Припускаючи, що впровадження пройшло успішно, слід розглянути, що ж ще є істотним для автоматизації електронного документообігу. Крім необхідного набору можливостей (таблиця 1) наявність у системі додаткового функціоналу (таблиця 5) може значно спростити роботу з системою всім співробітникам організації (наприклад, настройка повідомлень і нагадувань). У деяких випадках цей функціонал просто необхідний (наприклад, WEB-доступ). Потреби замовника в автоматизації процесів обробки документів з часом можуть мінятися (наприклад, у разі зміни організаційної структури підприємства), що може призвести до необхідності в розвитку системи. Наприклад, виникає ситуація, коли замовник, успішно експлуатує систему в головному офісі організації, згодом приймає рішення про впровадження її у філії. Як видно з таблиці 5, найбільш повноцінний додатковий функціонал за рахунок гнучких настроювань користувальницького інтерфейсу надають системи Бос-Референт та Optima-Workflow. За іншими параметрами впевнені позиції займають системи Directum, DocsVision і Євфрат-Документообіг.

Таблиця 5. Додаткові можливості роботи з системою

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Система можливість | Б-Р | Г-Д | Д | Є-Д | CM | D | DIS | DV | LD | O-W |
| настройка інтерфейсу користувача | + | -/+ | -/+ | +/-2 | -/+ | -/+ | -/+ | -/+ | + | + |
| настройка повідомлень та нагадувань | + | -/+ | -/+ | + | + | -/+ | + | + | + | + |
| інтеграція електронної пошти | + | - | -/+ | + | + | + | -/+ | + | +/-8 | + |
| WEB-доступ | + | - | +/-1 | + | +/-3 | -/+5 | + | + | +/-9 | + |
| територіально-розподілена робота\*\* | + | - | - | + | +4 | +/-6 | + | +/-7 | +/-10 | + |
| наявність API | + | - | + | + | + | + | - | + | + | + |
| Примітка:  \* - настройка кнопок на панелі  \*\* - наявність у системі стандартних засобів  1 – опціонально Дело-Web  2 – додатково, за допомогою інструментів для розробки на ActiveFrame  3 – опціально СМ-Портал  4 – СМ-Представництво/Філія  5 – опціально Сервер web-доступа  6 – опціально Вторинний сервер реплікації  7 – опціонально Модуль реплікації  8 – опціально LanDocs: ПОШТОВИЙ КЛІЄНТ  9 - опціально LanDocs: ІНТЕРНЕТ ДОСТУП  10 - опціально LanDocs: XML-ПІДСИСТЕМА ОБМІНУ | | | | | | | | | | |

автоматизація документообіг програмний продукт

Важливими характеристиками СЕД є її безпека, надійність і продуктивність (таблиця 6). З таблиці 6 слід, що системи Directum, DocsVision, Optima-Workflow і Євфрат-Документообіг дозволяють забезпечити необхідний рівень надійності і безпеки електронного документообігу. У свою чергу продуктивності розглянутих в огляді рішень достатньо, щоб автоматизувати як невеликі, так і великі структури, тому що продуктивність в більшості випадках залежить від можливостей серверного апаратного забезпечення. У разі створення розподіленої системи електронного документообігу (окремі сервери бази даних для кожного підрозділу) кількість одночасно працюючих користувачів може вимірюватися тисячами. Хочеться відзначити, що системи Бос-Референт і CompanyMedia, реалізовані на платформі Lotus Notes / Domino, пропонують найбільш розвинені засоби адміністрування при розподіленій роботі за рахунок можливостей платформи.

Таблиця 6. Надійність, безпека

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Система  можливість | Б-Р | Г-Д | Д | Є-Д | CM | D | DIS | DV | LD | O-W |
| авторизація користувачів з паролем | + | + | + | + | + | + | + | + | + | + |
| шифрування документів | +/-1 | - | +/-2 | + | +/-3 | + | + | + | +/-4 | + |
| підтримка ЕЦП | +/-1 | + | +/-2 | + | +/-3 | + | + | + | +/-4 | + |
| резервне копіювання БД за розкладом | + | - | + | + | + | + | - | + | + | + |
| протокол подій | + | -/+ | + | + | + | + | + | + | + | + |
| Примітка:  1 - замовлена розробка з використанням БОСС-Крипто  2 – опціонально «ЕЦП та Шифрування»  3 – опціально за допомогою інфраструктури Locker  4 – опціально LanDocs: ПІДСИСТЕМА БЕЗПЕКИ | | | | | | | | | | |

При виборі системи електронного документообігу слід врахувати безліч чинників, і остаточне рішення переважно приймати на основі комплексного аналізу можливостей СЕД залежно від вимог і специфіки замовника. У таблицях огляду всебічно розглянуті найбільш важливі аспекти вибору, впровадження та подальшого використання СЕД, що враховують крім обов'язкових функціональних можливостей, вартості і досвіду їх використання, можливості їх адаптації, модифікації та розвитку, необхідність яких не так очевидна на початковому етапі впровадження. Критерії, за якими були розглянуті системи - досить універсальні і відповідають завданням автоматизації документообігу як в комерційних, так і в державних установах.

Якщо говорити про вибір СЕД для організації з передбачуваною кількістю одночасно працюючих користувачів близько 50, то найбільш оптимальним рішенням бачиться система Євфрат-Документообіг, розробка компанії Cognitive Technologies. При помірній вартості система, з одного боку, містить весь необхідний функціонал для автоматизації роботи з документами, а з іншого боку, пропонує широкі можливості налаштування та модифікації при впровадженні розробником, партнерами і безпосередньо замовниками. Цікавою пропозицією також є система DocsVision, яка також включає в себе весь необхідний функціонал для автоматизації документообігу і тісно інтегрована з продуктами компанії Microsoft. Система надає гнучкі можливості при адаптації до вимог замовника. В даний час впровадження цієї системи здійснюється через партнерську мережу компанії DocsVision, що говорить про достатню незалежність системи від розробника. Крім цього за окремими критеріями добре показали себе системи Дело, Directum та Optima-Workflow.

Таким чином, використання аналітичних викладок, запропонованих в огляді, допоможе організаціям, зацікавленим в автоматизації документообігу, зробити найбільш прийнятний вибір рішення і тим самим уникнути цілого ряду "прихованих" на початковому етапі впровадження проблем, пов'язаних з використанням систем електронного документообігу.

**2. Постановка задачі**

Згідно з метою дипломного проекту необхідно розробити прикладну веб базовану систему для автоматизації документообігу деякого підприємства. Дана система повинна реалізувати функції реєстрації, авторизації, завантаження електронних документів будь-яких форматів, зберігання їх та видачу їх користувачу для редагування та перегляду. Вимогами до системи: є кросплатформеність, невисока продуктивність, невисока надійність, безкоштовність.

В даний час системи автоматизації документообігу є гарячою областю досліджень і розробок. Користь від використання таких систем для автоматизації роботи підприємства очевидна: незалежність розташування філій, пришвидшення роботи з документами і незалежність від платформи. Встановлена програма, з підтримкою в одному місці, може використовуватися тисячами тих, хто користується нею, головною умовою є тільки можливість доступу до Internet. Така програмна система повинна забезпечувати: просту і зручну реєстрацію в системі, авторизацію користувачів в системі, можливість додавання документу до системи, можливість надання прав перегляду та корегування документів.

Робота буде присвячена:

1. детальному опису підсистеми додавання документу до реєстру;
2. детальному опису підсистеми надання прав на редагування документів;
3. детальному опису підсистеми ведення статистики редагування документів;

Система повинна бути простою в установці та налаштування, надійна у роботі, невимогливою до апаратного забезпечення. Також важливим є завдання забезпечення потрібної і необхідної функціональності системи без високої вартості.

**3. Обґрунтування вибору технології розробки, програмного середовища та мови програмування**

**3.1 Вибір програмного забезпечення для створення програмного продукту**

Останнім часом у зв'язку з розвитком мережі Інтернет у програмуванні починає все більш різко виділятися окрема область. Спочатку вона не могла зрівнятися по своїй складності з іншими областями програмістського ремесла, не "дотягуючи" не тільки до системного, але навіть і до прикладного програмування.

Мова йде про програмування сценаріїв для Web, або, як часто говорять, Web-програмуванні. У наші дні, роль цієї галузі у структурі Інтернету все більше зростає, відповідно зростає і середня оцінка складності сценаріїв. Багато систем за обсягом коду наближаються до розміру вихідних кодів серйозних пакетів прикладних програм.

Дана робота представляє собою систему, написану на мові Java з використанням технологій JSP, Java Servlet і бази даних MySQL. Обгрунтування вибору зазначених засобів програмування для реалізації роботи наведено нижче.

Мова програмування Java надає наступні основні можливості:

а) автоматичне керування пам'яттю;

б) розширені можливості обробки виняткових ситуацій;

в) багатий набір засобів фільтрації вводу / виводу;

г) набір стандартних колекцій, таких як масив, список, стек і т. п.;

д) наявність простих засобів створення мережевих програмних продуктів (у тому числі з використанням протоколу RMI);

е) наявність класів, що дозволяють виконувати HTTP-запити й обробляти відповіді;

ж) вбудовані в мову засоби створення багатопоточних додатків;

з) уніфікований доступ до баз даних на основі JDBC і SQLJ;

і) підтримка шаблонів.

**3.2 Вибір засобу зберігання даних**

Вибір системи управління баз даних (СУБД) являє собою складне багатопараметричное завдання і є одним з важливих етапів при розробці програмних продуктів баз даних. Обраний програмний продукт повинен задовольняти як поточними, так і майбутнім потребам програми. Для даного програмного продукту в якості СУБД була обрана СУБД MySQL.

MySQL є рішенням для малих і середніх програмних продуктів. Входить в LAMP. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або видалені клієнти, проте в дистрибутив входить бібліотека внутрішнього сервера, що дозволяє включати MySQL в автономні програми.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Завдяки відкритій архітектурі і GPL-ліцензуванню, в СУБД MySQL постійно з'являються нові типи таблиць.

До основних переваг СУБД MySQL можна віднести:

а) багатопоточність, підтримка декількох одночасних запитів;

б) оптимізація зв'язків з приєднанням багатьох даних за один прохід;

в) записи фіксованої та змінної довжини;

г) гнучка система привілеїв і паролів;

д) до 16 ключів в таблиці. Кожен ключ може мати до 15 полів;

е) підтримка ключових полів і спеціальних полів в операторі CREATE;

ж) підтримка чисел довжиною від 1 до 4 байт (ints, float, double, fixed), рядків змінної довжини і міток часу;

з) заснована на потоках, швидка система пам'яті;

і) всі дані зберігаються у форматі ISO8859\_1;

к) всі операції роботи з рядками не звертають уваги на регістр символів у оброблюваних рядках;

л) легкість управління таблицею, включаючи додавання та видалення ключів і полів.

**3.3 Вибір web-сервера**

Apache Tomcat (у старих версіях - Catalina) - програма-контейнер сервлетів, написана на мові Java і реалізовує специфікацію сервлетів, специфікацію Java Server Pages (JSP) та Java Server Faces (JSF), які є стандартами для розробки веб-програм на мові Java. Tomcat дозволяє запускати веб-додатки, містить ряд програм для само-конфігурації. Tomcat використовується в якості самостійного веб-сервера, в якості сервера контенту в поєднанні з веб-сервером Apache HTTP Server, а також як контейнер сервлетів в сервері програм JBoss.

**3.4 Вибір IDE**

Eclipse - в першу чергу повноцінна Java IDE, націлена на групову розробку, має засоби роботи з системами контролю версій (підтримка CVS входить у поставку Eclipse, активно розвиваються кілька варіантів SVN модулів, існує підтримка VSS та інших). З огляду на безкоштовність, у багатьох організаціях Eclipse - корпоративний стандарт для розробки програмного забезпечення.

Друге призначення Eclipse - служити платформою для розробки нових розширень (чим і завоював популярність - будь-який розробник може розширити Eclipse своїми модулями). Такими стали C / C + + Development Tools (CDT), розроблювані інженерами QNX разом із IBM, COBOL, FORTRAN, PHP кошти від різних розробників. Безліч розширень доповнює Eclipse менеджерами для роботи з базами даних, серверами додатків і ін.

Eclipse написана на Java, тому є платформо-незалежним продуктом, за винятком бібліотеки SWT, яка розробляється для всіх поширених платформ. Бібліотека SWT використовується замість стандартної для Java бібліотеки Swing. Вона повністю залежить від нижче перерахованої платформи (операційної системи), що забезпечує швидкість і натуральний зовнішній вигляд для користувача інтерфейсу, але іноді викликає проблеми сумісності та стійкості програм на різних платформах.

**4. Опис програмної системи**

**4.1 Підсистема реєстрації**

Підсистема реєстрації потрібна для того, щоб у систему можна було додавати користувачів. На даному етапі розробки системи, користувача можна зареєструвати двома способами:

1. реєстрація користувачем;
2. реєстрація системним адміністратором.

При першому варіанті реєстрації користувач заходить на форму реєстрації за посиланням, вводить свої особисті дані та дані для майбутньої авторизації. Після цього йому присвоюється роль користувача і він може приступати до користування системою.

У другому випадку, системний адміністратор самостійно вводить дані користувача, задає йому роль у системі(на даному етапі присутні лише дві ролі – юзер та адміністратор, причому один користувач може об’єднувати у собі обидві ці ролі). Після додавання всієї інформації про користувача та успішної реєстрації користувач, після авторизації, може приступати до роботи у системі.

Програмний код реєстрації користувача приведено нижче:

String name = (String) req.getParameter("name");

UserManagerImpl um = new UserManagerImpl();

User user = um.load(name);

if (user == null) {

req.getSession().setAttribute("show\_pass\_error", "");

user = new User();

user.setName(name);

Date d = new Date();

boolean er = false;

List errors\_date = new ArrayList();

user.setBirth(d);

user.setEmail(req.getParameter("eMail"));

user.setFirstName(req.getParameter("firstName"));

user.setLastName(req.getParameter("lastName"));

user.setPassword(req.getParameter("password"));

try {

StringTokenizer st = new StringTokenizer(req

.getParameter("birthDay"), "-");

d.setDate(Integer.parseInt(st.nextToken()));

d.setMonth(Integer.parseInt(st.nextToken()) - 1);

d.setYear(Integer.parseInt(st.nextToken()) - 1900);

} catch (Exception e) {

errors\_date.add("error in date");

er = true;

}

if (!er) {

UserValidator valid = new UserValidator();

List errors = valid.vilidateFields(user);

if (errors.size() > 0) {

System.out.println("have errors");

req.getSession().setAttribute("errors", errors);

resp.sendRedirect("regestry.jsp");

} else {

RoleManagerImpl rm = new RoleManagerImpl();

Role role = rm.getRole("user");

user.addRole(role);

um.create(user);

role = (Role) req.getSession().getAttribute("role");

if (role != null) {

try {

if (role.getRole().equals("admin")) {

resp.sendRedirect("getallusers.jsp");

} else { resp.sendRedirect("registergood.jsp");

}

} catch (Exception e) { resp.sendRedirect("registergood.jsp");

}

} else {

resp.sendRedirect("registergood.jsp");

}

}

} else {

user.setBirth(null);

req.getSession().setAttribute("erruser", user);

req.getSession().setAttribute("errors", errors\_date);

resp.sendRedirect("regestry.jsp");

}

} else {

String error = "Error, same login has exist already";

req.getSession().setAttribute("show\_pass\_error", error);

resp.sendRedirect("regestry.jsp"); }

Адміністратор та користувач, під час реєстрації повинні ввести вісі дані коректно, тому варіація та зберігання даних про користувача однакове. Приведений вище код буде виконаний при будь-якому з видів реєстрації.

**4.2 Підсистема додавання електронного документу**

В основу концепцій роботи підсистеми документообігу покладено принцип, який можна описати наступними кроками:

1. користувач завантажує за допомогою форми на сервер деякий документ;
2. документ зберігається на жорсткому диску сервера;
3. документ реєструється у реєстрі документів, що було завантажено;
4. користувачу, що завантажив документ надаються права на редагування документу;
5. користувачу, що завантажив документ надаються права на надання прав доступу до цього документу.

Завдяки тому, що данні про електронний документ зберігаються у БД, а сам документ на жорсткому диску, дана система виграє час на пошуку документів(оскільки БД не засмічується великими за обсягами документами), і водночас виграє час на доступі до файлів, за рахунок майже прямих посилань(детальніше це описано в описі підсистеми ведення статистики редагування документів).

Програмний код завантаження документу має вигляд:

protected void doPost(HttpServletRequest request,

HttpServletResponse response) throws ServletException, IOException {

DiskFileItemFactory fileItemFactory = new DiskFileItemFactory();

fileItemFactory.setSizeThreshold(1 \* 1024); // 1 KB

fileItemFactory.setRepository(tmpDir);

ServletFileUpload uploadHandler = new ServletFileUpload(fileItemFactory);

try {

List items = uploadHandler.parseRequest(request);

Iterator itr = items.iterator();

User user = (User)request.getSession().getAttribute("user");

request.getSession().setAttribute("user", user);

DocumentManagerImpl documentManagerImpl = new DocumentManagerImpl();

while (itr.hasNext()) {

FileItem item = (FileItem) itr.next();

if (!item.isFormField()) {

File file = new File(destinationDir, item.getName());

item.write(file);

Document document = new Document();

document.setName(item.getName());

document.setPath(DESTINATION\_DIR\_PATH+"//"+item.getName());

Set<User> userSet = new HashSet<User>();

userSet.add(user);

document.setUser(userSet);

documentManagerImpl.create(document);

}

}

response.sendRedirect("index.jsp");

} catch (FileUploadException ex) {

log("Error encountered while parsing the request", ex);

} catch (Exception ex) {

log("Error encountered while uploading file", ex);

}}

Менеджер роботи з документами DocumentManagerImpl під час додавання до системи документу присвоює йому версію номер один, та задає ім’я влсника.

**4.3 Підсистема статистки редагування документів**

Підсистема статистики редагування документів діє наступним чином: коли користувач хоче відредагувати файл, він заходить на форму відображення документів, що він може корегувати. Після цього він може скачати документ та відредагувати його. Відредагувавши документ, його знову завантажують у систему, за допомогою форми, після чого файл помічається, як остання версія.

Останні 10 версій документу зберігаються на жорсткому диску. В любий момент часу користувач має можливість отримати доступ до них. Завдяки такому методу зберігання даних, користувач завжди може перевірити, як саме редагувався документ.

Принцип зберігання даних було взято з принципу роботи Subversion – централізованої системи зберігання версій даних. Проте, оскільки повністю реалізація даного принципу є дуже складною та об’ємною частиною, повністю вона не реалізована. У майбутньому планується розробка системи Subversion, спеціально для даної системи.

**4.4 Підсистема надання прав на редагування документів**

Підсистема надання прав на редагування документів діє по наступному принципу:

Той, хто вперше створив документ являється його власником, та має повні права на редагування та видалення. У випадку, коли необхідно, щоб хтось інший міг також редагувати цей документ, власник, переходячи на спеціальну форму, додає цю людину у список власників. Після цього, людина, яка отримала права на редагуваня, має такі самі права, як і власник.

Такий підхід у електронному документообігу дуже важливий, оскільки він виключає доступ до файлів тих людей, які не мають на це повноважень. Тим самим підтримується конфіденційність.

Програмний код сервлету редагування прав користувачів приведено нижче:

public class EditDocumentRool extends HttpServlet {

@SuppressWarnings("unchecked")

public void doGet(HttpServletRequest req, HttpServletResponse resp)

throws IOException {

String name = req.getParameter("name");

DocumentManagerImpl documentManagerImpl = new DocumentManagerImpl();

Document document = documentManagerImpl.load(name);

if(document == null){

document = new Document();

}

if(document.getUser() == null){

document.setUser(new HashSet<User>());

}

UserManagerImpl userManager = new UserManagerImpl();

List<User> users = userManager.getAll();

for(int i = 0; i < users.size()-1; i++){

for(int j = i+1; j < users.size(); j++){

if(users.get(i).getName().equals(users.get(j).getName())){

users.remove(j);

j--;

}}}

for(int i = 0 ; i < users.size(); i++){

Iterator iter = document.getUser().iterator();

while(iter.hasNext()){

if(users.get(i).getName().equals(((User)iter.next()).getName())){

users.remove(i);

i--;

break;

}}}

users.removeAll(document.getUser());

req.getSession().setAttribute("users", users);

req.getSession().setAttribute("document", document);

resp.sendRedirect("editDocumentUsers.jsp");

}

public void doPost(HttpServletRequest req, HttpServletResponse resp)

throws IOException {

String[] users = req.getParameterValues("names");

Document document = (Document) req.getSession()

.getAttribute("document");

boolean result = (new DocumentManagerImpl()).addUsersToDocument(

document, users);

if (result) {

resp.sendRedirect("myDocumentsListVisualisation");

} else {

doGet(req, resp);

}}}

На даному етапі розробки, під час надання прав доступу іншому користувачу, власник документу бачить всіх користувачів системи у вигляді списку. Такий підхід дає можливість надавати права користування файлом напряму людині, а не деякій групі людей. Це робить систему більш гнучкою та безпечною. Наприклад, якщо документ відноситься до відділу розробки, то до нього потрібен доступ певної особи з відділу контролю якості. При такому підході, доступ надається саме цій особі, а не усьому відділу контролю якості. Це зменшує ризик витоку інформації, а також покращує контроль за документами, оскільки до них мають доступ лише конкретні люди, список яких можна отримати в любий момент часу.

В майбутньому планується розширення системи в напрямку юзабіліті інтерфейсу, а саме, створення додаткових груп користувачів. Проте групи користувачів будуть потрібні лише для більш сприйнятливого відображення списків користувачів. Надання прав користування все рівно буде надаватися окремо для кожної людини. Після створення таких груп, окрім покращення юзабіліті, ця система отримає нову, перспективну форму, при якій однією системою зможе користуватися не одне підприємство, а декілька, при цьому, використовуючи апаратні можливості тільки одного сервера. Це зекономить гроші на закупівлю та обслуговування обладнання, а також, на заробітну плату обслуговуючому персоналу.

**5. Керівництво з використання програми**

Для того, щоб повноцінно користуватися всіма функціями системи «EFOD», необхідно пройти реєстрацію. Для цього необхідно виконати наступні дії:

1. перейти за посиланням «реєстрація» (див. рис.5.1);
2. заповнити поля, які запропонує система (див. рис.5.2);
3. після заповнення полів натиснути кнопку «Регистрация».

У разі успішної реєстрації, користувач переходить на головну сторінку. Для того, щоб почати користуватися системою, йому необхідно авторизуватися. Для цього необхідно набрати свій пароль і логін і натиснути кнопку «Login».

Після цього користувач перейде на стартову сторінку, і справа вгорі на екрані відобразиться його привітання (див. рис.5.3)

Для того, щоб загрузити документ, необхідно:

1. вибрати в меню «upload document», і перейти на сторінку, як показано на рис. 5.4;
2. за допомогою кнопки «Выбрать…» вибрати(або написати вручну) шлях до файлу;
3. натиснути кнопку «upload file» і дочекатися переходу на головну сторінку, все файл завантажено.

Для завантаження сторінки довідки необхідно натиснути на посилання «FAQ»

Для редагування прав користувачів необхідно:

1. натиснути посилання «My documents»;
2. вибрати файл та натиснути на кнопку «add new user to correct» (див. рис. 5.5);
3. поставити галочки навпроти тих, хто повинен надалі мати право на доступ до файлу(див. рис. 5.5);
4. натиснути кнопку «Confirm».

Для того, щоб видалити файл, необхідно:

1. натиснути посилання «My documents»;
2. вибрати файл та натиснути на кнопку «delete document»(див. рис. 5.5);
3. погодитися на видалення документу.

У випадку, коли користувач являється ще й адміністратором, він може переглянути розгорнутий список всіх користувачів системи. Для цього необхідно натиснути на посиланні «Users list». Після цього з’явиться сторінка зі списком сих користувачів системи. Причому на даній сторінці можна редагувати ролу користувачів, а також видаляти акаунти користувачів.

**Висновки**

В результаті виконання дипломного проекту була розроблена система електронного документообігу, що дозволяє скоротити час на обробку даних на паперових носіях, та автоматизувати процес документообігу. Дана система дозволяє отримати швидкий доступ до інформації, а також захистити її від несанкціонованого доступу.

На стадії проектування системи був проведений аналіз систем такого класу. Оцінивши всі достоїнства і недоліки вже запропонованих систем, стало можливим зрозуміти, якою саме має бути система, щоб вона задовольняла основному завданню – максимально пришвидшити процес документообігу та захистити данні від втрати та несанкціонованого доступу. Крім того, було сформульовано ряд вимог, які повинна задовольняти система. З огляду на динамічність системи, були побудовані вимоги до платформи і методам розробки системи.

У процесі розробки система була протестована на різних платформах і в різних умовах навантаження. Система показала себе добре, а помилки, які були допущені в процесі розробки були усунені, а також внести додаткові елементи захисту інформації.

В результаті розробки отримана система, яка може працювати в реальних умовах з сильним навантаженням, і при цьому залишатися стабільною. В майбутньому планується розширення даної системи, що дозволить збільшити коло її користувачів. Окрім того, планується покращити підсистему зберігання інформації та застосувати більш надійні алгоритми шифрування даних.

**Перелік посилань**

1. web-сайт посвященный програмуванню [Електронний ресурс] – Режим доступа : www/URL: http://tools.ietf.org/html/rfc3174 - 10.12.2003 г. - Загл. з екрану.
2. web-сайт, присвячений програмуванню [Електронний ресурс] – Режим доступа: www/ URL: http://www.glossary.ru – 30.05.2009г. - Загл. з екрану.
3. Бондарев, В.М. Учебное пособие по программированию на Java [Текст] / Бондарев В.М.. – Х.: Диалектика, 2003. – 296 с.
4. Хансен Г.Д. Базы данных: разработка и управление [Текст] / Хансен Г.Д., Хансен Д.А. – М.: БИНОМ, 1999. – 704 с.
5. M. Brandis The Oberon System Family [Текст] / M. Brandis R. Crelier M. Franz J. Templ – Software Practice and Experience, 1995, p. 1331-1366.
6. Planning the content of instruction / Brecht B.J., McCalla G.I., Greer J.E. 1989, p. 32-41.
7. Methods and techniques of adaptive hypermedia. / Brusilovsky P., 1996, p. 87-129.
8. Brusilovsky P., Eklund J., Schwarz E., 1997, p. 278-285.
9. Knowledge tracing in the ACT programming tutor. / Corbett A.T., Anderson J.A., 1992.

**Додаток А**

Програмний код класу взаємодії з БД сутності «користувач»(User)

package diploma.hibernateDAO;

import java.util.ArrayList;

import java.util.List;

import diploma.dao.DAO;

import org.apache.commons.logging.Log;

import org.apache.commons.logging.LogFactory;

import org.hibernate.Session;

import org.hibernate.Transaction;

import org.hibernate.annotations.Cascade;

import diploma.Role;

public abstract class HibernateDAO<T> implements DAO<T> {

private static final Log log = LogFactory.getLog(HibernateDAO.class);

public boolean create(T newInstance) {

Session session = HibernateUtils.openSession();

Transaction tx = session.beginTransaction();

try {

session.save(newInstance);

tx.commit();

return true;

} catch (Exception e) {

System.out.println(e);

log.error(e);

tx.rollback();

return false;

} finally {

HibernateUtils.closeSession(session);

}}

public boolean delete(T inctance) {

Session session = HibernateUtils.openSession();

Transaction tx = session.beginTransaction();

try {

session.delete(inctance);

tx.commit();

return true;

} catch (Exception e) {

System.out.println(e);

log.error(e, e);

tx.rollback();

return false;

} finally {

HibernateUtils.closeSession(session);

}}

@SuppressWarnings("unchecked")

public List<T> getAll() {

Session session = HibernateUtils.openSession();

Transaction tx = session.beginTransaction();

try {

Class clazz = (Class) ((java.lang.reflect.ParameterizedType) getClass()

.getGenericSuperclass()).getActualTypeArguments()[0];

List list = session.createCriteria(clazz).list();

return list;

} catch (Exception e) {

log.error(e, e);

return new ArrayList<T>();

} finally {

HibernateUtils.closeSession(session);

}}

public boolean update(T newInstance) {

Session session = HibernateUtils.openSession();

Transaction tx = session.beginTransaction();

try {

session.update(newInstance);

tx.commit();

} catch (Exception e) {

log.error(e, e);

tx.rollback();

return false;

} finally {

HibernateUtils.closeSession(session);

}

return true;

}}

**Додаток Б**

Програмний код основного CRUD менеджеру

package diploma.hibernateDAO;

import java.util.ArrayList;

import java.util.List;

import diploma.dao.DAO;

import org.apache.commons.logging.Log;

import org.apache.commons.logging.LogFactory;

import org.hibernate.Session;

import org.hibernate.Transaction;

import org.hibernate.annotations.Cascade;

import diploma.Role;

public abstract class HibernateDAO<T> implements DAO<T> {

private static final Log log = LogFactory.getLog(HibernateDAO.class);

public boolean create(T newInstance) {

Session session = HibernateUtils.openSession();

Transaction tx = session.beginTransaction();

try {

session.save(newInstance);

tx.commit();

return true;

} catch (Exception e) {

System.out.println(e);

log.error(e);

tx.rollback();

return false;

} finally {

HibernateUtils.closeSession(session);

}

}

public boolean delete(T inctance) {

Session session = HibernateUtils.openSession();

Transaction tx = session.beginTransaction();

try {

session.delete(inctance);

tx.commit();

return true;

} catch (Exception e) {

System.out.println(e);

log.error(e, e);

tx.rollback();

return false;

} finally {

HibernateUtils.closeSession(session);

}

}

@SuppressWarnings("unchecked")

public List<T> getAll() {

Session session = HibernateUtils.openSession();

Transaction tx = session.beginTransaction();

try {

Class clazz = (Class) ((java.lang.reflect.ParameterizedType) getClass().getGenericSuperclass()).getActualTypeArguments()[0];

List list = session.createCriteria(clazz).list();

return list;

} catch (Exception e) {

log.error(e, e);

return new ArrayList<T>();

} finally {

HibernateUtils.closeSession(session);

}

}

public boolean update(T newInstance) {

Session session = HibernateUtils.openSession();

Transaction tx = session.beginTransaction();

try {

session.update(newInstance);

tx.commit();

} catch (Exception e) {

log.error(e, e);

tx.rollback();

return false;

} finally {

HibernateUtils.closeSession(session);

}

return true;

}

}